IBM Spectrum Scale
Version 5.0.0

Concepts, Planning, and Installation Guide

GC27-9219-03

IBM Spectrum Scale
Version 5.0.0

Concepts, Planning, and Installation Guide

GC27-9219-03

Note

Before using this information and the product it supports, read the information in “Notices” on page 429.

This edition applies to version 5 release 0 modification 0 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 (product number 5725-S28)
- IBM Spectrum Scale for IBM ESS (product number 5765-ESS)

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 xxv. 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

Chapter 3. Steps for establishing and starting your IBM Spectrum Scale cluster 221

Chapter 4. Installing IBM Spectrum Scale on Linux nodes and deploying protocols 223

Chapter 5. Installing IBM Spectrum Scale on AIX nodes. 319

Chapter 6. Installing IBM Spectrum Scale on Windows nodes 323

Accessibility features for IBM Spectrum Scale 427

Notices 429

Glossary 433

Index 439

Figures

Tables

© Copyright IBM Corp. 2014, 2018

About this information

This edition applies to IBM Spectrum Scale™ version 5.0.0 for AIX®, Linux, and Windows.

IBM Spectrum Scale is a file management infrastructure, based on IBM® General Parallel File System (GPFS™) technology, which 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     (for SLES and Red Hat Enterprise Linux)
dpkg -l | grep gpfs     (for Ubuntu Linux)
```

To find out which version of IBM Spectrum Scale is running on a particular Windows node, open **Programs and Features** in the 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 on page xii.

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: Concepts, Planning, and Installation Guide	This guide provides the following information: **Product overview** • Overview of IBM Spectrum Scale • GPFS architecture • Protocols support overview: Integration of protocol access methods with GPFS • Active File Management • AFM-based Asynchronous Disaster Recovery (AFM DR) • Data protection and disaster recovery in IBM Spectrum Scale • Introduction to IBM Spectrum Scale GUI • IBM Spectrum Scale management API • Introduction to Cloud services • Introduction to file audit logging • IBM Spectrum Scale in an OpenStack cloud deployment • IBM Spectrum Scale product editions • IBM Spectrum Scale license designation • Capacity based licensing • IBM Spectrum Storage™ Suite **Planning** • Planning for GPFS • Planning for protocols • Planning for Cloud services • Firewall recommendations • Considerations for GPFS applications	System administrators, analysts, installers, planners, and programmers of IBM Spectrum Scale clusters who are very experienced with the operating systems on which each IBM Spectrum Scale cluster is based

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

Information unit	Type of information	Intended users
IBM Spectrum Scale: Concepts, Planning, and Installation Guide	**Installing** • Steps for establishing and starting your IBM Spectrum Scale cluster • Installing IBM Spectrum Scale on Linux nodes and deploying protocols • Installing IBM Spectrum Scale on AIX nodes • Installing IBM Spectrum Scale on Windows nodes • Installing Cloud services on IBM Spectrum Scale nodes • Setting up a Cloud services cluster • Installing and configuring IBM Spectrum Scale management API • Installing Active File Management • Installing and upgrading AFM-based Disaster Recovery • Installing call home • Installing file audit logging • Steps to permanently uninstall GPFS and/or Protocols	System administrators, analysts, installers, planners, and programmers of IBM Spectrum Scale clusters who are very experienced with the operating systems on which each IBM Spectrum Scale cluster is based

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

Information unit	Type of information	Intended users
IBM Spectrum Scale: Concepts, Planning, and Installation Guide	**Upgrading** • IBM Spectrum Scale supported upgrade paths • Upgrading to IBM Spectrum Scale 5.0.x from IBM Spectrum Scale 4.2.y • Upgrading to IBM Spectrum Scale 4.2.y from IBM Spectrum Scale 4.1.x • Upgrading to IBM Spectrum Scale 4.1.1.x from GPFS V4.1.0.x • Upgrading from GPFS 3.5 • Online upgrade support for protocols and performance monitoring • Upgrading object packages • Upgrading NFS packages • Upgrading SMB packages • Manually upgrading pmswift • Manually upgrading the performance monitoring tool • Manually upgrading the IBM Spectrum Scale management GUI • Upgrading Cloud services • Upgrading to IBM Cloud Object Storage software level 3.7.2 and above • Upgrading IBM Spectrum Scale components with the installation toolkit • Migrating from Express Edition to Standard Edition • Completing the upgrade to a new level of IBM Spectrum Scale • Reverting to the previous level of IBM Spectrum Scale • Coexistence considerations • Compatibility considerations • Considerations for IBM Spectrum Protect™ for Space Management • GUI user role considerations • Applying maintenance to your GPFS system	System administrators, analysts, installers, planners, and programmers of IBM Spectrum Scale clusters who are very experienced with the operating systems on which each IBM Spectrum Scale cluster is based

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

Information unit	Type of information	Intended users
IBM Spectrum Scale: Administration Guide	This guide provides the following information: **Configuring** • Configuring the GPFS cluster • Configuring the CES and protocol configuration • Configuring and tuning your system for GPFS • Parameters for performance tuning and optimization • Ensuring high availability of the GUI service • Configuring and tuning your system for Cloud services • Configuring file audit logging • Configuring Active File Management • Configuring AFM-based DR • Tuning for Kernel NFS backend on AFM and AFM DR **Administering** • Performing GPFS administration tasks • Verifying network operation with the mmnetverify command • Managing file systems • File system format changes between versions of IBM Spectrum Scale • Managing disks • Managing protocol services • Managing protocol user authentication • Managing protocol data exports • Managing object storage • Managing GPFS quotas • Managing GUI users • Managing GPFS access control lists • Considerations for GPFS applications • Accessing a remote GPFS file system	System administrators or programmers of IBM Spectrum Scale systems

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

Information unit	Type of information	Intended users
IBM Spectrum Scale: Administration Guide	• Information lifecycle management for IBM Spectrum Scale • Creating and maintaining snapshots of file systems • Creating and managing file clones • Scale Out Backup and Restore (SOBAR) • Data Mirroring and Replication • Implementing a clustered NFS environment on Linux • Implementing Cluster Export Services • Identity management on Windows • Protocols cluster disaster recovery • File Placement Optimizer • Encryption • Managing certificates to secure communications between GUI web server and web browsers • Securing protocol data • Cloud services: Transparent cloud tiering and Cloud data sharing • Managing file audit logging • Highly-available write cache (HAWC) • Local read-only cache • Miscellaneous advanced administration • GUI limitations	System administrators or programmers of IBM Spectrum Scale systems

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

Information unit	**Type of information**	**Intended users**
IBM Spectrum Scale: Problem Determination Guide	This guide provides the following information: **Monitoring** • Performance monitoring • Monitoring system health through the IBM Spectrum Scale GUI • Monitoring system health by using the mmhealth command • Monitoring events through callbacks • Monitoring capacity through GUI • Monitoring AFM and AFM DR • GPFS SNMP support • Monitoring the IBM Spectrum Scale system by using call home • Monitoring remote cluster through GUI • Monitoring file audit logging **Troubleshooting** • Best practices for troubleshooting • Understanding the system limitations • Collecting details of the issues • Managing deadlocks • Installation and configuration issues • Upgrade issues • Network issues • File system issues • Disk issues • Security issues • Protocol issues • Disaster recovery issues • Performance issues • GUI issues • AFM issues • AFM DR issues • Transparent cloud tiering issues • File audit logging issues • Maintenance procedures • Recovery procedures • Support for troubleshooting • References	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*

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

Information unit	**Type of information**	**Intended users**
IBM Spectrum Scale: Command and Programming Reference	This guide provides the following information: **Command reference** • gpfs.snap command • mmaddcallback command • mmadddisk command • mmaddnode command • mmadquery command • mmafmconfig command • mmafmctl command • mmafmlocal command • mmapplypolicy command • mmaudit command • mmauth command • mmbackup command • mmbackupconfig command • mmblock command • mmbuildgpl command • mmcallhome command • mmces command • mmcesdr command • mmchattr command • mmchcluster command • mmchconfig command • mmchdisk command • mmcheckquota command • mmchfileset command • mmchfs command • mmchlicense command • mmchmgr command • mmchnode command • mmchnodeclass command • mmchnsd command • mmchpolicy command • mmchpool command • mmchqos command • mmclidecode command • mmclone command • mmcloudgateway command • mmcrcluster command • mmcrfileset command • mmcrfs command • mmcrnodeclass command • mmcrnsd command • mmcrsnapshot command	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

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

Information unit	Type of information	Intended users
IBM Spectrum Scale: Command and Programming Reference	• mmdefedquota command • mmdefquotaoff command • mmdefquotaon command • mmdefragfs command • mmdelacl command • mmdelcallback command • mmdeldisk command • mmdelfileset command • mmdelfs command • mmdelnode command • mmdelnodeclass command • mmdelnsd command • mmdelsnapshot command • mmdf command • mmdiag command • mmdsh command • mmeditacl command • mmedquota command • mmexportfs command • mmfsck command • mmfsctl command • mmgetacl command • mmgetstate command • mmhadoopctl command • mmhealth command • mmimgbackup command • mmimgrestore command • mmimportfs command • mmkeyserv command • mmlinkfileset command • mmlsattr command • mmlscallback command • mmlscluster command • mmlsconfig command • mmlsdisk command	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

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

Information unit	Type of information	Intended users
IBM Spectrum Scale: Command and Programming Reference	• mmlsfileset command • mmlsfs command • mmlslicense command • mmlsmgr command • mmlsmount command • mmlsnodeclass command • mmlsnsd command • mmlspolicy command • mmlspool command • mmlsqos command • mmlsquota command • mmlssnapshot command • mmmigratefs command • mmmount command • mmmsgqueue command • mmnetverify command • mmnfs command • mmnsddiscover command • mmobj command • mmperfmon command • mmpmon command • mmprotocoltrace command • mmpsnap command • mmputacl command • mmquotaoff command • mmquotaon command • mmremotecluster command • mmremotefs command • mmrepquota command • mmrestoreconfig command • mmrestorefs command • mmrestripefile command • mmrestripefs command • mmrpldisk command • mmsdrrestore command • mmsetquota command • mmshutdown command • mmsmb command • mmsnapdir command • mmstartup command • mmtracectl command • mmumount command • mmunlinkfileset command • mmuserauth command • mmwinservctl command • spectrumscale command	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

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

Information unit	Type of information	Intended users
IBM Spectrum Scale: Command and Programming Reference	**Programming reference** • IBM Spectrum Scale Data Management API for GPFS information • GPFS programming interfaces • GPFS user exits • IBM Spectrum Scale management API commands	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

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

Information unit	Type of information	Intended users
IBM Spectrum Scale: Big Data and Analytics Guide	This guide provides the following information: **IBM Spectrum Scale support for Hadoop** • HDFS transparency • Supported IBM Spectrum Scale storage modes • Hadoop cluster planning • Installation and configuration of HDFS transparency • Application interaction with HDFS transparency • Upgrading the HDFS Transparency cluster • Rolling upgrade for HDFS Transparency • Security • Advanced features • Hadoop distribution support • Limitations and differences from native HDFS • Problem determination **BigInsights® 4.2.5 and Hortonworks Data Platform 2.6** • **Planning** – Hardware requirements – Preparing the environment – Preparing a stanza file • **Installation** – Set up – Installation of software stack – BigInsights value-add services on IBM Spectrum Scale • **Upgrading software stack** – Migrating from BI IOP to HDP – Upgrading IBM Spectrum Scale service MPack – Upgrading HDFS Transparency – Upgrading IBM Spectrum Scale file system – Upgrading to BI IOP 4.2.5	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

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

Information unit	Type of information	Intended users
IBM Spectrum Scale: Big Data and Analytics Guide	• **Configuration** – Setting up High Availability [HA] – IBM Spectrum Scale configuration parameter checklist – Dual-network deployment – Manually starting services in Ambari – Setting up local repository – Configuring LogSearch – Setting IBM Spectrum Scale configuration for BigSQL – Hadoop Kafka/Zookeeper and IBM Spectrum Scale Kafka/Zookeeper • **Administration** – IBM Spectrum Scale-FPO deployment – Ranger – Kerberos – Short-circuit read (SSR) – Disabling short circuit write – IBM Spectrum Scale service management – Ambari node management – Restricting root access – IBM Spectrum Scale management GUI – IBM Spectrum Scale versus Native HDFS • **Troubleshooting** – Snap data collection • **Limitations** – Limitations and information • **FAQ** – General – Service fails to start – Service check failures	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

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 describes the typographic conventions used in this information. UNIX file name conventions are used throughout this information.

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

Table 2. Conventions

Convention	Usage
bold	**Bold** words or characters represent system elements that you must use literally, such as commands, flags, values, and selected menu options. Depending on the context, **bold** typeface sometimes represents path names, directories, or file names.
bold underlined	**bold underlined** keywords are defaults. These take effect if you do not specify a different keyword.
`constant width`	Examples and information that the system displays appear in `constant-width` typeface. Depending on the context, `constant-width` typeface sometimes represents path names, directories, or file names.
italic	*Italic* words or characters represent variable values that you must supply. *Italics* are also used for information unit titles, for the first use of a glossary term, and for general emphasis in text.
<*key*>	Angle brackets (less-than and greater-than) enclose the name of a key on the keyboard. For example, **<Enter>** refers to the key on your terminal or workstation that is labeled with the word *Enter*.

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

{*item*}	Braces enclose a list from which you must choose an item in format and syntax descriptions.
[*item*]	Brackets enclose optional items in format and syntax descriptions.
<**Ctrl-***x*>	The notation <**Ctrl-***x*> indicates a control character sequence. For example, <**Ctrl-c**> means that you hold down the control key while pressing <**c**>.
item...	Ellipses indicate that you can repeat the preceding item one or more times.
|	In *synopsis* statements, vertical lines separate a list of choices. In other words, a vertical line means *Or*. In the left margin of the document, vertical lines indicate technical changes to the information.

Note: CLI options that accept a list of option values delimit with a comma and no space between values. As an example, to display the state on three nodes use **mmgetstate -N** *NodeA*,*NodeB*,*NodeC*. Exceptions to this syntax are listed specifically within the command.

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 that are made to the previous edition of the information.

Summary of changes
for IBM Spectrum Scale version 5.0.0
as updated, November 2018

This release of the IBM Spectrum Scale licensed program and the IBM Spectrum Scale library includes the following improvements. All improvements are available after an upgrade, unless otherwise specified.

AFM and AFM DR-related changes

- Compression policy is supported. For more information, see *File compression* in *IBM Spectrum Scale: Administration Guide*.
- ILM for snapshots is supported. For more information, see *ILM for snapshots* in *IBM Spectrum Scale: Administration Guide*.
- Throttling the number of filesets to perform recoveries on the gateway node. For more information, see *Gateway node failure and recovery* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- Improved hashing algorithm to load balance filesets across gateway nodes. For more information, see *Primary gateway* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- Configuration parameters have been added - **afmMaxParallelRecoveries**, **afmAsyncOpWaitTimeout**, **afmSyncOpWaitTimeout**, and **afmRevalOpWaitTimeout**. For more information, see *Configuration parameters for AFM* and *Configuration parameters for AFM-based DR* in *IBM Spectrum Scale: Administration Guide*.
- Configuration parameters have been modified - **afmRPO** and **afmHashVersion**. For more information, see *Configuration parameters for AFM* and *Configuration parameters for AFM-based DR* in *IBM Spectrum Scale: Administration Guide*.

Authentication: Primary group selection configurable for AD + RFC2307 based authentication

Ability to choose primary group as set in "UNIX attributes" of a user on Active Directory introduced with AD + RFC2307 based authentication scheme. Earlier, Windows primary group was by default selected as the primary group.

Big data and analytics changes

For information on changes in IBM Spectrum Scale Big Data and Analytics support, see Big data and analytics - summary of changes.

IBM Spectrum Scale management API changes

Added the following new commands:

- GET /perfmon/data
- GET /filesystems/{filesystemName}/afm/state
- DELETE /nodes/{name}
- POST /nodes
- GET /nodeclasses
- POST /nodeclasses
- DELETE /nodeclasses/{nodeclassName}
- GET /nodeclasses/{nodeclassName}

- PUT /nodeclasses/{nodeclassName}
- DELETE /jobs/jobId
- POST /filesystems/{filesystemName}/filesets/{filesetName}/psnaps
- DELETE /filesystems/{filesystemName}/filesets/{filesetName}/psnaps/{snapshotName}
- GET /thresholds
- GET /thresholds/{name}
- POST /thresholds
- DELETE /thresholds/{name}

For more information on the API commands, see *IBM Spectrum Scale management API commands* in *IBM Spectrum Scale: Command and Programming Reference*. You can also access the documentation corresponding to each API command from the GUI itself. The API documentation is available in the GUI at: *https://<IP address or host name of API server>:<port>/ibm/api/explorer/*. For example: *https://scalegui.ibm.com:443/ibm/api/explorer*.

The JSON object structure is modified for the following API commands:

/cluster

rshSudoWrapper, rcpSudoWrapper: Changed type from String to Boolean

/filesets

Moved *filesetName* and *filesystemName* out of the *config* object.

/nfs/exports

Removed the *config* object and moved the attributes one level up.

/nsds and /disks

Removed the *config* object and moved the attributes one level up.

quorumDisk: Changed type from String to Boolean.

size, availableBlocks, availableFragments: Changed type from String to Long. Values are now in bytes.

/smb/shares

Removed the *config* object and moved the attributes one level up.

/snapshots

Removed the *config* object and moved the attributes one level up.

IBM Spectrum Scale GUI changes

The following changes are made to the GUI:

- Added new **Networks** page to monitor the performance, configuration, and adapters of network configurations in the cluster. You can monitor the network performance with respect to the IP and RDMA interfaces used in the configuration. For more information, see *Monitoring networks* in *IBM Spectrum Scale: Problem Determination Guide*.
- Added new **Monitoring** > **Thresholds** page to create and monitor the threshold rules that are defined in the system. For more information, see *Monitoring thresholds* in *IBM Spectrum Scale: Problem Determination Guide*.
- Added **Access** > **Remote Connections** page to enable the GUI node of the local cluster to monitor the remote cluster by establishing a connection with the GUI node of the remote cluster. For more information, see *Monitoring remote cluster through GUI* in *IBM Spectrum Scale: Problem Determination Guide*.
- Added **Settings** > **Call Home** page to configure call home. Configuring the call home feature helps the IBM Support to monitor the system. Configuring call home also helps to reduce the response time of the IBM Support to resolve any issues.

  The diagnostic data that is downloaded through the **Settings** > **Diagnostic Data** can be uploaded to a problem management record (PMR) by using the call home feature in the backend. To upload the diagnostic data, right-click the relevant data set from the Previously Collected Diagnostic Data, and select **Upload to PMR**.For more information, see *Configuring call home using GUI* in *IBM Spectrum Scale: Problem Determination Guide*.

- Added the file system creation capabilities in GUI. Use the **Files** > **File Systems** > **Create File System** option to launch the Create File System wizard. In the Create File System wizard, you can specify the following details of the file system:
  - File system name
  - Storage pools
  - NSDs for the file systems
  - Failure groups
  - NSD order for data writes
  - Maximum number of Spectrum Scale clients
  - Maximum number of inodes of the root fileset
  - Whether to enable quota and scope for the quota definition
  - Whether to enable DMAPI
  - Mount point and automatic mount mode

  For more information, see *Creating and managing file systems using GUI* in *IBM Spectrum Scale: Administration Guide*.
- Added the aggregation levels *Access Point* and *Filesets* and removed *Account* for the resource type Transparent Cloud Tiering in the **Monitoring** > **Statistics** page.
- The **Files** > **Transparent Cloud Tiering** page now displays the file systems and filesets that are mapped with the TCT services. It also shows the connection of such a container pair configuration to a cloud account and the corresponding CSAP that is configured for the cloud account.
- Changed and updated information in the **Added DMPs for TIP events** page. The DMPs help the users resolve issues that are associated with the tip events. For more information, see the *Directed maintenance procedures for tip events* topic in the *IBM Spectrum Scale: Problem Determination Guide*.
- **Changes to capacity monitoring in the GUI**

  The following changes are made to the capacity monitoring options in the GUI:
  - Moved the **Capacity** page from **Monitoring** to **Files** menu in the navigation and renamed the GUI page to **User Capacity**.
  - Only the file data user capacity can be viewed from the **Files** > **User Capacity** page. Removed the pools, filesets, file system capacity monitoring options from the **Files** > **User Capacity** page. You can monitor the capacity of these components from the respective GUI pages.
  - Replaced the GPFSPoolCap sensor with the GPFSPool sensor and separate data and metadata level capacity monitoring are introduced in the performance charts available in the **Files** > **File Systems** and **Storage** > **Pools** pages.
  - New GPFSPool-based data and metadata performance monitoring metrics are available for selection in the **Files** > **Statistics** > **Edit** > **Capacity** section. You need to select the aggregation level as Pool to view these metrics.

  For more information, see *Monitoring capacity through GUI* in *IBM Spectrum Scale: Problem Determination Guide*.
- **AFM monitoring changes in the GUI**

  The following AFM monitoring features are introduced:
  - Provides the number of AFM filesets and the corresponding export server maps. Each export map establishes a mapping between the gateway node and the NFS host name to allow parallel data transfers from cache to home.

  - By using the **Request Access** option available in the **Files** > **Active File Management** or **Access** > **Remote Connection** page in the GUI, you can now establish connection with remote clusters. After establishing the connection, you can monitor the following AFM and AFM DR configuration details across clusters:
    - On home and secondary, you can see the AFM relationships configuration, health status, and performance values of the Cache and Disaster Recovery grids.
    - On the Overview tab of the detailed view, the available home and secondary inodes are available.
    - On the Overview tab of the detailed view, the details such as NFS throughput, IOPs, and latency details are available, if the protocol is NFS.

  For more information, see *Monitoring AFM and AFM DR using GUI* in *IBM Spectrum Scale: Problem Determination Guide*.

- New option to create AFM peer snapshots through GUI. Use the **Create Peer Snapshot** option in the **Files** > **Snapshots** page to create peer snapshots. You can view and delete these peer snapshots from the Snapshots page and also from the detailed view of the **Files** > **Active File Management** page.

IBM Spectrum Scale functionality to support GDPR requirements

To understand the requirements of EU General Data Protection Regulation (GDPR) compliance that are applicable to unstructured data storage and how IBM Spectrum Scale helps to address them, see the IBM Spectrum Scale functionality to support GDPR requirements technote.

Encryption: GSKit V8 improves cryptographic performance on IBM POWER8®

The IBM Global Security Kit (GSKit) Version 8 and later improves cryptographic performance on IBM POWER8 hardware. For more information, see the IBM article *GSKIT V8 Support for PKCS#11 device integration and on-CPU acceleration* (https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/W746177d414b9_4c5f_9095_5b8657ff8e9d/page/PKCS%2311%20device%20integration). The version of GSKit that is shipped with IBM Spectrum Scale v5.0.0 offers better performance on POWER8, compared with the versions shipped with earlier releases.

File audit logging

File audit logging is a new capability in IBM Spectrum Scale v5.0.0. File audit logging takes locally generated file system events and puts them on a multinode message queue from which they are consumed and written to a retention enabled fileset. These events, called lightweight events, occur at the file system level. They are protocol agnostic, which means that they capture all access to a monitored file system from protocol exports to even root access that occurs directly on nodes. File audit logging is integrated into the system health infrastructure. Alerts are generated for elements of the message queue and the processes that consume the events and create the audit logs. For more information, see *Introduction to file audit logging* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

File compression: The lz4 library provides fast access to compressed data

File compression supports the lz4 compression library. Lz4 is intended primarily for active data and favors read-access speed over maximized space saving. For more information, see the topic *File compression* in the *IBM Spectrum Scale: Administration Guide*.

File data: Smaller subblock sizes improve I/O performance and reduce fragmentation

Subblock sizes are smaller relative to block sizes; for example, a 4 MiB block uses 8 KiB subblocks. The same choice of data block sizes is available, but the default data block size is larger, 4 MiB instead of 256 KiB. A larger block size improves the file system performance and a smaller subblock size reduces the amount of unused space. For many business applications, the default value of 4 MiB provides the best balance of improved performance and reduced fragmentation. For more information, see *mmcrfs command* in the *IBM Spectrum Scale: Command and Programming Reference*. This feature is supported only for file systems that are created at file system format number 18.00 (IBM Spectrum Scale 5.0.0.0) or later.

File encryption: AES-XTS encryption is faster on x86 in non-FIPS mode

On x86 architecture in non-FIPS mode, file encryption with the AES algorithm in XTS mode is faster than it was. For more information, see the topics *Encryption policy rules* and *Encryption and FIPS-140-2 certification* in the *IBM Spectrum Scale: Administration Guide*.

File systems: File system rebalancing is faster

Rebalancing is implemented by a lenient round-robin method that typically runs faster than the previously used method of strict round robin. The strict round robin method is available as an option. For more information, see the topics *mmrestripefs command*, *mmrestripefile command*, *mmadddisk command*, and *mmdeldisk command* in the *IBM Spectrum Scale: Command and Programming Reference*.

Installation toolkit changes

- The installation toolkit has added support for the installation and the deployment of protocol and client nodes in a cluster that contains Elastic Storage Server (ESS). For more information, see *ESS awareness with the installation toolkit* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- The installation toolkit has added support for enabling and configuring call home. For more information, see *Enabling and configuring call home using the installation toolkit* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- The installation toolkit has added support for enabling and configuring file audit logging. For more information, see *Enabling and configuring file audit logging using the installation toolkit* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- The installation toolkit has added support for the installation and the deployment of IBM Spectrum Scale on Ubuntu 16.04 LTS nodes.
- The installation toolkit has added support for the installation and the deployment of IBM Spectrum Scale on Red Hat Enterprise Linux 7.4 nodes.
- The installation toolkit has added verification of passwordless SSH during prechecks before installation, deployment, or upgrade.
- The installation toolkit has added support for cumulative object upgrade.

mmafmctl command

The **--outband** parameter is deprecated.

mmcallhome command: Enhancements

Addition of -Y option

The **-Y** displays the command output in a parseable format with a colon (:) as a field delimiter.

Addition of --pmr option

The **--pmr** option allows you to upload data to existing PMRs using the **mmcallhome run SendFile** command. For more information, see *mmcallhome command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmchconfig command: Enhancements

Encrypted files can be copied into an LROC device

With the **lrocEnableStoringClearText** attribute, you can control whether file data from encrypted files, which is held in memory as cleartext, is copied into a local read-only cache (LROC) device.

InfiniBand addresses can be specified for RDMA transfers

In the **verbsPorts** attribute, you can specify InfiniBand addresses for RDMA transfers between an NSD client and server.

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

mmchnsd command: Change NSDs without unmounting the file system

When you add or remove NSDs or do other operations with **mmchnsd**, you do not need to unmount the file system. For more information, see the topic *mmchnsd command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmcrfs command: Enhancements

The default data block size is 4 MiB with an 8 KiB subblock size

If no block size is specified, a file system is created with a 4 MiB block size and an 8 KiB subblock size. The minimum release level (**minReleaseLevel**) of the cluster must be 5.0.0 or greater when the file system is created. For more information, see the entry **File data: Smaller subblock sizes improve I/O performance and reduce fragmentation** earlier in this topic.

The default log file size depends on block size and metadata size

If the block size is 512 KiB or larger and the metadata block size 256 KiB or larger, then the default log file size is 32 MiB. Otherwise, the default log file size is 4 MiB or the metadata block size, whichever is larger. For more information, see the topic *mmcrfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

The default method for updating atime is relatime

If the minimum release level (**minReleaseLevel**) of the cluster is 5.0.0 or greater when the file system is created, the default method for updating **atime** is **relatime**. For more information, see the topics *mmcrfs command* and *mmchfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmdsh command: Several options are no longer supported

The **--ignoreSignal**, **-I**, and **-d** options are no longer supported. Do not use these options unless instructed to by IBM support personnel. For more information, see the topic *mmdsh command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmfsck command: Display an interim status report at any time

While a long-running instance of **mmfsck** is in progress, you can start another instance of **mmfsck** with the **--status-report** parameter to display current status information from all the nodes that are participating in the **mmfsck** run. For more information, see the topic *mmfsck command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmgetstate command: Display the unresponsive state

The command returns the **unresponsive** state when the GPFS daemon is running but is not responding. For more information, see the topic *mmgetstate command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmhealth command: Addition to measurement options

Measurement options for filesystem, SMB node, and NFS node has been added to the **mmhealth** command. For more information, see the topic *mmhealth command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmkeyserv command: The simplified setup method supports certificate chains from a certificate authority.

In the simplified setup, with the **--kmip-cert** parameter, you can set up encryption with IBM® Security Key Lifecycle Manager (SKLM) as the key management server and with a certificate signed by a certificate authority (CA) on the KMIP port of the SKLM server. For more information, see the topic *mmkeyserv command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmnetverify command: Enhancements

Verify the network operation of nodes in a subnet

With the **--subnets** parameters, you can specify the subnet addresses of the nodes that you want to verify.

Verify that nodes can handle a new MTU size
With the **ping-packet-size** parameter, you can specify the size of the ICMP echo request packets that are sent between local node and the target node during the **ping** test.

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

mmpmon command: Enhancements
A new option **loc_io_s** is added to **mmpmon** command. It stands for locality I/O statistics.

mmtracectl command: Display the tracing status
The **--status** parameter displays the tracing status of the specified nodes. For more information, see *mmtracectl command* in the *IBM Spectrum Scale: Command and Programming Reference*.

Threshold monitoring enhancements
If multiple thresholds rules have overlapping entities for the same metrics, only one of the concurrent rules is made actively eligible. For more information, see the *New features for threshold monitoring* section in *Threshold monitoring for system health* topic in the *IBM Spectrum Scale: Problem Determination Guide*.

NFS: Dynamic export changes
You can dynamically change the export configuration without restarting the NFS service. For more information, see the *Changing NFS export configuration* topic in the *IBM Spectrum Scale: Administration Guide*.

Object changes
The following changes have been done.

- Object protocol support is extended to Ubuntu 16.04 LTS.
- The sudo wrapper for object is added on Ubuntu 16.04 LTS.
- Cumulative object upgrades from older versions are supported.
- Object snap enhancement are done to contain Keystone logs.

Protocol support: Enhanced
NFS, SMB, and object support is extended to Ubuntu 16.04 LTS.

Setting up a system for storing crash files for Ubuntu
Support is added to set up a system for storing crash files for Ubuntu. For more information, see the *Setting up an Ubuntu system to capture crash files* topic in the *IBM Spectrum Scale: Problem Determination Guide*.

SMB: DFS redirects for SMB shares
New option to configure DFS redirects for SMB shares. For more information, see *mmsmb command* in the *IBM Spectrum Scale: Command and Programming Reference*.

SMB: SMB server upgrade changes
Two events on CTDB version match/mismatch are added to the RAS events. For more information, see the *RAS events* topic in the *IBM Spectrum Scale: Problem Determination Guide*.

Sudo wrappers: Root-level processes can call administration commands directly
Root-level background processes, such as **cron** and callback programs, can successfully call administration commands directly rather than through **sudo** when sudo wrappers are enabled. For more information, see *Root-level processes that call administration commands directly* in the *IBM Spectrum Scale: Administration Guide*.

Cloud services changes
The following enhancements are made:

- A new robust set of CLI functions, which allow for much greater configuration of the system.
- Support for multiple cloud storage accounts
- Support for multiple file systems or filesets per node group

- Enhanced support for large file systems provided by container spillover
- Support for associating file sets with containers for enhanced granularity
- Multiple URL and region support at the node level
- Support for creating a Cloud services separately for tiering and sharing operations. For more information, see the *Creating Cloud services* topic in the *IBM Spectrum Scale: Administration Guide*.
- Unique encryption key per cloud container
- Support for remotely mounted clients. For more information, see the *Setting up Transparent cloud tiering service on a remotely mounted client* topic in the *IBM Spectrum Scale: Administration Guide*.
- Support for Amazon S3 regions requiring Sigv4 security support, including the US government cloud region.
- Ability to enable or disable transparent recall for files for a given file system instantly, without having to rewrite a policy.
- Support for backing up and restoring the Cloud services configuration in case of any disaster. For more information, see the *Backing up the Cloud services configuration* topic in the *IBM Spectrum Scale: Administration Guide*.
- Support for backing up the Cloud services database to the cloud. For more information, see the *Backing up the Cloud services database to the cloud* topic in the *IBM Spectrum Scale: Administration Guide*.
- Support for restoring Transparent cloud tiering service on an identical backup cluster. For more information, see the *Restoring Transparent Cloud services on a backup cluster* topic in the *IBM Spectrum Scale: Administration Guide*.
- Support for checking the integrity of the Cloud services database after any system crash or outage. For more information, see the *Checking the Cloud services database integrity* topic in the *IBM Spectrum Scale: Administration Guide*.
- Support for auditing events relating to each operation performed in Cloud services. For more information, see the *Cloud services audit events* topic in the *IBM Spectrum Scale: Problem Determination Guide*.

Documented commands, structures, and subroutines

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

New commands

These commands are new in this release.

- **mmaudit**
- **mmmsgqueue**

New structures

There are no new structures.

New subroutines

There are no new subroutines.

Changed commands

The following commands were changed:

- **mmafmctl**
- **mmadddisk**
- **mmcallhome**
- **mmchattr**
- **mmchcluster**
- **mmchconfig**
- **mmchfs**
- **mmchnsd**

- **mmcloudgateway**
- **mmcrcluster**
- **mmcrfs**
- **mmdeldisk**
- **mmdsh**
- **mmfsck**
- **mmgetstate**
- **mmkeyserv**
- **mmnetverify**
- **mmnfs**
- **mmpmon**
- **mmrestripefile**
- **mmrestripefs**
- **mmsmb**
- **mmtracectl**
- **mmuserauth**
- **spectrumscale**

Changed structures
There are no changed structures.

Changed subroutines
There are no changed subroutines.

Deleted commands
mmrest

Deleted structures
There are no deleted structures.

Deleted subroutines
There are no deleted subroutines.

Messages
The following are the new, changed, and deleted messages:

New messages
6027-1264, 6027-1757, 6027-2394, 6027-2395, 6027-2396, 6027-2397, 6027-2398, 6027-2399, 6027-2400, 6027-2401, 6027-3259, 6027-3408, 6027-3597, 6027-3598, 6027-3599, 6027-3600, 6027-3601, 6027-3602, 6027-3603, 6027-3604, 6027-3730, 6027-3921, 6027-3922, 6027-3923, 6027-3924, 6027-3925, 6027-3926, 6027-3927, 6027-3928, 6027-3929, 6027-3930, 6027-3931, and 6027-4019.

Changed messages
6027-928

Deleted messages
None.

Changes in documentation

Separate sections for installing and upgrading documentation
The *Installing and upgrading* section is split into two separate sections:

- *Installing*
- *Upgrading*

Terminology change in upgrade documentation

In the upgrade documentation, the instances of "migration" are replaced with "upgrade".

Product name changes

The following product name changes are done:

- z System is renamed to IBM Z
- Linux on IBM Z is renamed to Linux on Z

List of documentation changes in product guides and respective Knowledge Center sections

The following is a list of documentation changes including changes in topic titles, changes in placement of topics, and deleted topics:

Table 3. List of changes in documentation

Guide	Knowledge center section	List of changes
Concepts, Planning, and Installation Guide	Product overview	The following sub-sections are moved from the *Introduction to Cloud services* topic to *Library and related publications* → *Applying IBM Spectrum Scale*: • *Transparent cloud tiering use cases* • *Supported Cloud data sharing use cases* • *Supported WORM storage use cases*
	Installing	• *Creating a user-defined node class for Transparent cloud tiering or Cloud data sharing* topic is moved from the *Configuring* section to the *Installing* section • The *IBM Spectrum Scale packaging overview* topic is moved one level up. • The following topics and their sub-topics are removed. The content in these topics is now covered in *Manually installing IBM Spectrum Scale and deploying protocols on Linux nodes* and its sub-topics. – *Manually installing IBM Spectrum Scale on Red Hat Enterprise Linux 7.x systems* – *Manually installing IBM Spectrum Scale on SLES 12 systems*
	Upgrading	The upgrade topics have been reordered to bring out the upgrade flow more clearly and separate manual and installation toolkit topics clearly.
Administration Guide	Configuring	The following topics are removed from the *Configuring and tuning your system for Cloud services* section: • *Pre-validating the cloud account settings* • *Associating a file system with Cloud services nodes*
	Administering	Moved the *Manual recovery of Transparent cloud tiering database* topic under *Administering files for Transparent cloud tiering* section

Changes in the Library and related publications section

A *Video resources* topic was added in this section. This topic contains links to IBM Spectrum Scale videos.

Chapter 1. Introducing IBM Spectrum Scale

Overview of IBM Spectrum Scale

IBM Spectrum Scale is a cluster file system that provides concurrent access to a single file system or set of file systems from multiple nodes. The nodes can be SAN attached, network attached, a mixture of SAN attached and network attached, or in a shared nothing cluster configuration. This enables high performance access to this common set of data to support a scale-out solution or to provide a high availability platform.

IBM Spectrum Scale has many features beyond common data access including data replication, policy based storage management, and multi-site operations. You can create a cluster of AIX nodes, Linux nodes, Windows server nodes, or a mix of all three. IBM Spectrum Scale can run on virtualized instances providing common data access in environments, leverage logical partitioning, or other hypervisors. Multiple IBM Spectrum Scale clusters can share data within a location or across wide area network (WAN) connections.

The strengths of IBM Spectrum Scale

IBM Spectrum Scale provides a global namespace, shared file system access among IBM Spectrum Scale clusters, simultaneous file access from multiple nodes, high recoverability and data availability through replication, the ability to make changes while a file system is mounted, and simplified administration even in large environments.

For more information, see the following:

- “Improved system performance” on page 2
- “File consistency” on page 2
- “Increased data availability” on page 2
- “Enhanced system flexibility” on page 3
- “Shared file system access among IBM Spectrum Scale clusters”
- “Simplified storage management” on page 4
- “Simplified administration” on page 4

Shared file system access among IBM Spectrum Scale clusters

IBM Spectrum Scale allows you to share data between separate clusters within a location or across a WAN.

IBM Spectrum Scale clusters are independently managed but IBM Spectrum Scale also shares data access through remote cluster mounts. This is known as a multicluster environment. When multiple clusters are configured to access the same IBM Spectrum Scale file system, IBM Global Security Kit (GSKit) can be used to authenticate and check authorization for all network connections.

GSKit can be used for authentication and to encrypt data passed between the clusters. If you use a GSKit cipher, the data is encrypted for transmissions.

Note: If you have a version 4.1, or later, cluster and a version 3.5, or earlier, cluster, you can use GSKit on the 4.1, or later, cluster and Open Secure Socket Layer (OpenSSL) on the 3.5, or earlier, cluster.

The multicluster environment has the following features:

- The cluster that is hosting the file system can specify different security levels for each cluster authorized to mount a particular file system.

- The local cluster can remain active while changing security keys. Periodic changing of keys is necessary for a variety of reasons:
  - The number of keys should remain small to facilitate good performance.
  - Key changes prevent use or continued use of compromised keys.
  - As a matter of policy, some institutions require security keys to be changed periodically.

IBM Spectrum Scale uses public key authentication in a manner similar to the host-based authentication mechanism of OpenSSH. Each cluster has a pair of these keys that identify the cluster. In addition, each cluster also has an `authorized_keys` list. Each line in the `authorized_keys` list contains the public key of one remote cluster and a list of the file systems that that cluster is authorized to mount. For details about multicluster (remote mount) file system access, see *Accessing a remote GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

For related information, see *Active file management* in the *IBM Spectrum Scale: Administration Guide*.

Improved system performance

Using GPFS file systems can improve system performance in a number of ways.

- Allowing multiple processes or applications on all nodes in the cluster simultaneous access to the same files. That is, it allows concurrent reads and writes from multiple nodes.
- Increasing aggregate bandwidth of your file system by spreading reads and writes across multiple disks.
- Balancing the load evenly across all disks to maximize their combined throughput, eliminating storage hotspots.
- Supporting very large file and file system sizes.
- Allowing concurrent reads and writes from multiple nodes.
- Provides sophisticated token management that handles fast and fine grained access to cluster, file system, and file resources.
- Allowing for the specification of multiple networks for GPFS daemon communication and for GPFS administration command usage within your cluster.

Achieving high throughput to a single, large file requires striping data across multiple disks and multiple disk controllers. Rather than relying on striping in a separate volume manager layer, GPFS implements striping in the file system. Managing its own striping affords GPFS the control it needs to achieve fault tolerance and to balance load across adapters, storage controllers, and disks. Large files in GPFS are divided into equal sized blocks, and consecutive blocks are placed on different disks in a round-robin fashion.

GPFS automatically detects common data access patterns and automatically begins pre-fetching data accordingly. This pre-fetching and caching provides high throughput and fast response times. Some of the recognized I/O patterns include sequential, reverse sequential, and various forms of strided access patterns.

File consistency

IBM Spectrum Scale provides concurrent access to clients across the cluster by utilizing sophisticated token management. This provides concurrent and detailed access to IBM Spectrum Scale features, file systems, and file resources.

For more information, see "GPFS architecture" on page 9.

Increased data availability

GPFS provides multiple features that improve the reliability of your file system. This includes automatic features like file system logging and configurable features like intelligently mounting file systems on startup to providing tools for flexible synchronous replication.

GPFS allows you to organize your storage hardware into *failure groups*. A failure group is defined as a set of disks that share a common point of failure that could cause them all to become simultaneously unavailable. Failure groups are defined by the system administrator, so care needs to be taken when defining disks to ensure proper failure group isolation. When used in conjunction with the *replication* feature of GPFS, the creation of multiple failure groups provides for increased file availability should a group of disks fail. Replication in GPFS ensures that there is a copy of each block of replicated data and metadata on disks in different failure groups. In this case, should a set of disks become unavailable, GPFS fails over to the replicated copies in another failure group.

During configuration, you assign a replication factor to indicate the total number of copies of data and metadata you wish to store. Currently, the maximum replication factor is 3. Replication allows you to set different levels of protection for each file or one level for an entire file system. Since replication uses additional disk space and requires extra write time, you should consider the impact of replication on your application, especially if the replication is occurring over a WAN. To reduce the overhead involved with the replication of data, you may also choose to replicate only metadata as a means of providing additional file system protection. For further information on GPFS replication, see "File system replication parameters" on page 160.

GPFS is a logging file system. It creates separate logs for each file system. GPFS automatically replicates recovery logs if multiple failure groups are available. When used in conjunction with geographic based replication, disaster recovery abilities are provided. For further information on failure groups, see "Network Shared Disk (NSD) creation considerations" on page 147. For further information on disaster recovery with GPFS, see *Data Mirroring and Replication* in *IBM Spectrum Scale: Administration Guide*.

Once your file system is created, it can be configured to mount whenever the GPFS daemon is started. This feature assures that whenever the system and disks are up, the file system will be available. When utilizing shared file system access among GPFS clusters, to reduce overall GPFS control traffic you may decide to mount the file system when it is first accessed. This is done through either the **mmremotefs** command or the **mmchfs** command using the **-A automount** option. GPFS mount traffic may be reduced by using automatic mounts instead of mounting at GPFS startup. Automatic mounts only produce additional control traffic at the point that the file system is first used by an application or user. Mounting at GPFS startup on the other hand produces additional control traffic at every GPFS startup. Thus startup of hundreds of nodes at once may be better served by using automatic mounts. However, when exporting the file system through Network File System (NFS) mounts, it might be useful to mount the file system when GPFS is started.

Enhanced system flexibility

With GPFS, your system resources are not frozen. You can add or delete disks while the file system is mounted.

When the time is favorable and system demand is low, you can rebalance the file system across all currently configured disks.

With the QoS capability, you can prevent I/O-intensive, long-running administration commands from dominating file system performance and significantly delaying other tasks.

You can also add or delete nodes without having to stop and restart the GPFS daemon on all nodes.

Note: GPFS allows a large number of quorum nodes to facilitate maintaining quorum and continued cluster operation. GPFS also allows a tiebreaker disk configuration to further enhance cluster availability. For additional information, refer to "Quorum" on page 136.

If the physical connection to the disk is broken, GPFS dynamically switches disk access to the servers nodes and continues to provide data through NSD server nodes. GPFS falls back to local disk access when it discovers that the path has been repaired.

After GPFS has been configured for your system, depending on your applications, hardware, and workload, you can re-configure GPFS to increase throughput. You can set up your GPFS environment for your current applications and users, secure in the knowledge that you can expand in the future without jeopardizing your data. GPFS capacity can grow as your hardware expands.

Simplified storage management

IBM Spectrum Scale can help you achieve information lifecycle management (ILM) through powerful policy-driven, automated tiered storage management.

IBM Spectrum Scale provides storage management based on the definition and use of:

- Storage pools
- Policies
- Filesets

Storage pools

A *storage pool* is a collection of disks or RAIDs with similar properties that are managed together as a group. Storage pools provide a method to partition storage within a file system. While you plan how to configure your storage, consider factors 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
- Improved reliability by providing for:
  - Replication based on need
  - Better failure containment

Policies

Files are assigned to a storage pool based on defined *policies*. Policies provide for:

Placement policies

Placing files in a specific storage pool when the files are created

File management policies

- Migrating files from one storage pool to another
- Deleting files based on file characteristics
- Changing the replication status of files
- Snapshot metadata scans and file list creation
- Compressing static files

Filesets

Filesets provide a method for partitioning a file system and allow administrative operations at a finer granularity than the entire file system. For example, filesets allow you to:

- Define data block and inode quotas at the fileset level
- Apply policy rules to specific filesets
- Create snapshots at the fileset level

For further information on storage pools, filesets, and policies, see *Information lifecycle management for IBM Spectrum Scale* in *IBM Spectrum Scale: Administration Guide*.

Simplified administration

GPFS offers many of the standard file system interfaces allowing most applications to execute without modification.

Operating system utilities complement the GPFS utilities. That is, you can continue to also use the commands you have always used for ordinary file operations. For more information, see "Considerations for GPFS applications" on page 219.

GPFS administration commands are similar in name and function to UNIX and Linux file system commands with one important difference: *the GPFS commands operate on multiple nodes*. A single GPFS command can perform an administration function across the entire cluster. See the individual commands as documented in the *IBM Spectrum Scale: Command and Programming Reference*.

GPFS commands save configuration and file system information in one or more files. These are collectively known as GPFS cluster configuration data files. GPFS maintains the consistency of its configuration files across the cluster and this provides accurate and consistent confirmation information. See "Cluster configuration data files" on page 26.

The basic IBM Spectrum Scale structure

IBM Spectrum Scale is a clustered file system defined over one or more nodes. On each node in the cluster, IBM Spectrum Scale consists of three basic components: administration commands, a kernel extension, and a multithreaded daemon.

For more information, see the following topics:

1. "GPFS administration commands"
2. "The GPFS kernel extension"
3. "The GPFS daemon"
4. For nodes in your cluster operating with the Linux operating system, "The GPFS open source portability layer" on page 6

For a detailed discussion of IBM Spectrum Scale, see "GPFS architecture" on page 9.

GPFS administration commands

GPFS administration commands are scripts and programs that control the operation and configuration of GPFS.

By default, GPFS commands can be executed from any node in the cluster. If tasks need to be done on another node in the cluster, the command automatically redirects the request to the appropriate node for execution. For administration commands to be able to operate, passwordless remote shell communications between the nodes is required.

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

The GPFS kernel extension

The GPFS kernel extension provides the interfaces to the operating system vnode and virtual file system (VFS) layer to register GPFS as a native file system.

Structurally, applications make file system calls to the operating system, which presents them to the GPFS file system kernel extension. GPFS uses the standard mechanism for the operating system. In this way, GPFS appears to applications as just another file system. The GPFS kernel extension will either satisfy these requests using resources which are already available in the system, or send a message to the GPFS daemon to complete the request.

The GPFS daemon

The GPFS daemon performs all I/O operations and buffer management for GPFS. This includes read-ahead for sequential reads and write-behind for all writes not specified as synchronous. I/O operations are protected by GPFS token management, which ensures consistency of data across all nodes in the cluster.

The daemon is a multithreaded process with some threads dedicated to specific functions. Dedicated threads for services requiring priority attention are not used for or blocked by routine work. In addition to managing local I/O, the daemon also communicates with instances of the daemon on other nodes to coordinate configuration changes, recovery, and parallel updates of the same data structures. Specific functions that execute in the daemon include:

1. Allocation of disk space to new files and newly extended files. This is done in coordination with the file system manager.
2. Management of directories including creation of new directories, insertion and removal of entries into existing directories, and searching of directories that require I/O.
3. Allocation of appropriate locks to protect the integrity of data and metadata. Locks affecting data that may be accessed from multiple nodes require interaction with the token management function.
4. Initiation of actual disk I/O on threads of the daemon.
5. Management of user security and quotas in conjunction with the file system manager.

The GPFS Network Shared Disk (NSD) component provides a method for cluster-wide disk naming and high-speed access to data for applications running on nodes that do not have direct access to the disks.

The NSDs in your cluster may be physically attached to all nodes or serve their data through an NSD server that provides a virtual connection. You are allowed to specify up to eight NSD servers for each NSD. If one server fails, the next server on the list takes control from the failed node.

For a given NSD, each of its NSD servers must have physical access to the same NSD. However, different servers can serve I/O to different non-intersecting sets of clients. The existing subnet functions in GPFS determine which NSD server should serve a particular GPFS client.

Note: GPFS assumes that nodes within a subnet are connected using high-speed networks. For additional information on subnet configuration, refer to “Using public and private IP addresses for GPFS nodes” on page 17.

GPFS determines if a node has physical or virtual connectivity to an underlying NSD through a sequence of commands that are invoked from the GPFS daemon. This determination, which is called *NSD discovery*, occurs at initial GPFS startup and whenever a file system is mounted.

Note: To manually cause this discovery action, use the **mmnsddiscover** command. For more information, see *mmnsddiscover command* in *IBM Spectrum Scale: Command and Programming Reference*.

This is the default order of access used during NSD discovery:

1. Local block device interfaces for SAN, SCSI, IDE, or DASD disks
2. NSD servers

This order can be changed with the **useNSDserver** mount option.

It is suggested that you always define NSD servers for the disks. In a SAN configuration where NSD servers have also been defined, if the physical connection is broken, GPFS dynamically switches to the server nodes and continues to provide data. GPFS falls back to local disk access when it has discovered the path has been repaired. This is the default behavior, and it can be changed with the useNSDserver file system mount option.

For further information, see “Disk considerations” on page 146 and “NSD disk discovery” on page 25.

The GPFS open source portability layer

On Linux platforms, GPFS uses a loadable kernel module that enables the GPFS daemon to interact with the Linux kernel. Source code is provided for the portability layer so that the GPFS portability can be built and installed on a wide variety of Linux kernel versions and configuration.

When installing GPFS on Linux, you build a portability module based on your particular hardware platform and Linux distribution to enable communication between the Linux kernel and GPFS. For more information, see "Building the GPFS portability layer on Linux nodes" on page 237.

IBM Spectrum Scale cluster configurations

An IBM Spectrum Scale cluster can be configured in a variety of ways. The cluster can be a heterogeneous mix of hardware platforms and operating systems.

IBM Spectrum Scale clusters can contain a mix of all supported node types including Linux, AIX, and Windows Server and these operating system can run on various hardware platforms such as IBM POWER®, x86-based servers, and IBM Z. These nodes can all be attached to a common set of SAN storage or through a mix of SAN and network attached nodes. Nodes can all be in a single cluster, or data can be shared across multiple clusters. A cluster can be contained in a single data center or spread across geographical locations. To determine which cluster configuration is best for your application start by determining the following:

- Application I/O performance and reliability requirements
- Properties of the underlying storage hardware
- Administration, security, and ownership considerations

Understanding these requirements helps you determine which nodes require direct access to the disks and which nodes should access the disks over a network connection through an NSD server.

There are four basic IBM Spectrum Scale Configurations:

- All nodes attached to a common set of LUNS
- Some nodes are NSD clients
- A cluster is spread across multiple sites
- Data is shared between clusters

All nodes attached to a common set of LUNS

In this type of configuration, all of the nodes in the cluster are connected to a common set of LUNS (for example, over a SAN). The following are areas to consider with this configuration:

- The maximum number of nodes accessing a LUN you want to support
- The fact that you cannot mix different operating systems with IBM Spectrum Scale to directly access the same set of LUNs on SAN.

For an example, see Figure 1 on page 8.

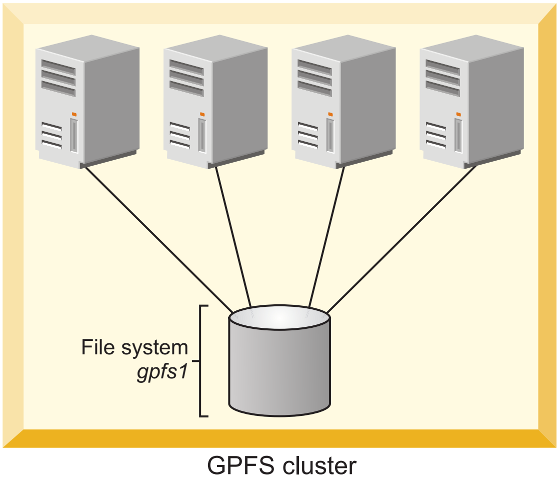


Figure 1. A cluster with disks that are SAN-attached to all nodes

Some nodes are NSD clients

In this type of configuration, only some nodes are connected to disks. Other nodes access the disks using the NSD path.

For an example, see Figure 2.

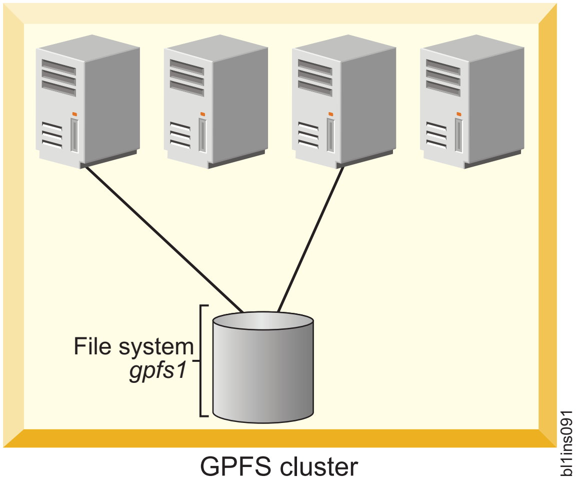


Figure 2. A cluster with some nodes connected to disks

IBM Spectrum Scale Servers and clients

You can configure an IBM Spectrum Scale cluster in which some nodes have a direct attachment to the disks and others access the disks through other IBM Spectrum Scale nodes. This configuration is often used in large clusters or to provide a cost-effective, high-performance solution.

When an IBM Spectrum Scale node is providing access to a disk for another IBM Spectrum Scale node, the node that provides access is called an NSD Server. The node that accesses the data through an NSD server is called an IBM Spectrum Scale client.

Sharing data across multiple IBM Spectrum Scale clusters

IBM Spectrum Scale allows you to share data across multiple IBM Spectrum Scale clusters. After a file system is mounted in another IBM Spectrum Scale cluster, all access to the data is the same as if you were in the host cluster. You can connect multiple clusters within the same data center or across long distances over a WAN. In a multicluster configuration, each cluster can be placed in a separate administrative group simplifying administration or provide a common view of data across multiple organizations.

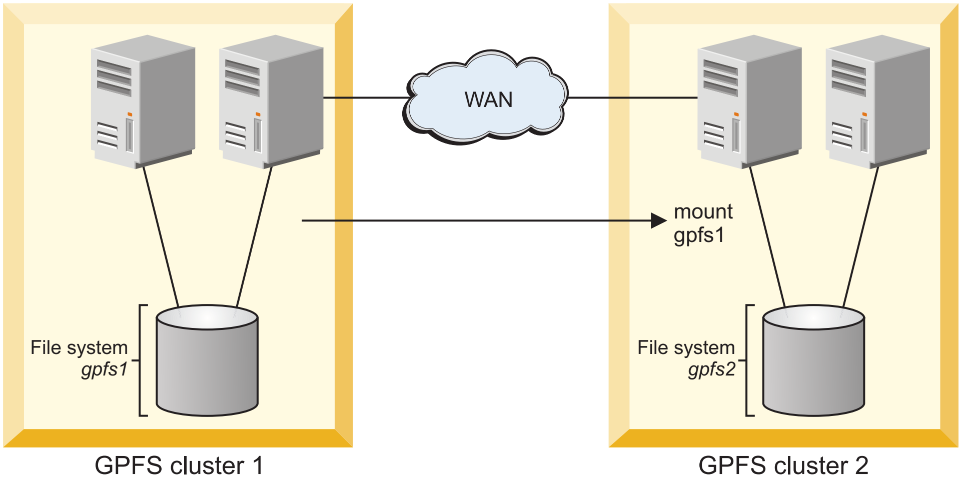


Figure 3. A multicluster configuration

Note: For more information, see *Accessing a remote GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

GPFS architecture

Use this information to understand the architecture of GPFS.

Interaction between nodes at the file system level is limited to the locks and control flows required to maintain data and metadata integrity in the parallel environment.

A discussion of GPFS architecture includes:
- "Special management functions"
- "Use of disk storage and file structure within a GPFS file system" on page 12
- "GPFS and memory" on page 15
- "GPFS and network communication" on page 17
- "Application and user interaction with GPFS" on page 19
- "NSD disk discovery" on page 25
- "Failure recovery processing" on page 25
- "Cluster configuration data files" on page 26
- "GPFS backup data" on page 27

Special management functions

In general, GPFS performs the same functions on all nodes. It handles application requests on the node where the application exists. This provides maximum affinity of the data to the application.

In the following cases, one node provides a more global function affecting the operation of multiple nodes. These are nodes acting as:

1. "The GPFS cluster manager"
2. "The file system manager"
3. "The metanode" on page 12
4. "CES node (protocol node)" on page 12
5. "AFM Gateway node" on page 12

The GPFS cluster manager

There is one GPFS cluster manager per cluster. The cluster manager is chosen through an election held among the set of quorum nodes designated for the cluster.

See "Quorum" on page 136 for more information.

The cluster manager performs the following tasks:

- Monitors disk leases.
- Detects failures and manages recovery from node failure within the cluster.

  The cluster manager determines if a quorum of nodes exists to allow the GPFS daemon to start and for file system usage to continue.
- Distributes certain configuration changes that must be known to nodes in remote clusters.
- Selects the *file system manager* node.

  The cluster manager prevents multiple nodes from assuming the role of file system manager to avoid data corruption. The token management service resides on the file system manager node and any other nodes you specify. For more information, see *Using multiple token servers* in *IBM Spectrum Scale: Administration Guide*.
- Handles UID mapping requests from remote cluster nodes.
- Aggregates health information from all nodes in the cluster.

To identify the cluster manager, issue the **mmlsmgr -c** command. For more information, see *mmlsmgr command* in *IBM Spectrum Scale: Command and Programming Reference*.

To change the cluster manager, issue the **mmchmgr -c** command. For more information, see *mmchmgr command* in *IBM Spectrum Scale: Command and Programming Reference*.

The file system manager

There is one file system manager per file system, which handles all of the nodes using the file system.

The services provided by the file system manager include:

1. File system configuration

   Processes the following file system changes:

   - Adding disks
   - Changing disk availability
   - Repairing the file system

   Mount and unmount processing is performed on both the file system manager and the node requesting the service.
2. Management of disk space allocation

   Controls which regions of disks are allocated to each node, allowing effective parallel allocation of space.
3. Token management

The file system manager node may also perform the duties of the token manager server. If you have explicitly designated some of the nodes in your cluster as file system manager nodes, then the token server load will be distributed among all of the designated manager nodes. For more information, see *Using multiple token servers* in *IBM Spectrum Scale: Administration Guide*.

The token management server coordinates access to files on shared disks by granting tokens that convey the right to read or write the data or metadata of a file. This service ensures the consistency of the file system data and metadata when different nodes access the same file. The status of each token is held in two places:

a. On the token management server
b. On the token management client holding the token

The first time a node accesses a file it must send a request to the token management server to obtain a corresponding **read** or **write** token. After having been granted the token, a node may continue to read or write to the file without requiring additional interaction with the token management server. This continues until an application on another node attempts to read or write to the same region in the file.

The normal flow for a token is:

- A message to the token management server.

  The token management server then either returns a granted token or a list of the nodes that are holding conflicting tokens.
- The token management function at the requesting node then has the responsibility to communicate with all nodes holding a conflicting token and get them to relinquish the token.

  This relieves the token server of having to deal with all nodes holding conflicting tokens. In order for a node to relinquish a token, the daemon must give it up. First, the daemon must release any locks that are held using this token. This may involve waiting for I/O to complete.

4. Quota management

   In a quota-enabled file system, the file system manager node automatically assumes quota management responsibilities whenever the GPFS file system is mounted. Quota management involves:

   - Allocating disk blocks to nodes that are writing to the file system
   - Comparing the allocated space to the quota limits at regular intervals

   Notes:

   a. To reduce the number of space requests from nodes writing to the file system, the quota manager allocates more disk blocks than requested (see "Enabling quotas" on page 161). That allows nodes to write to the file system without having to go to the quota manager and check quota limits each time they write to the file system.
   b. File systems that have quotas enabled and more than 100,000 users or groups should avoid designating nodes as manager where memory is low or that are otherwise heavily loaded because of the high memory demands for quota manager operations.

The file system manager is selected by the cluster manager. If a file system manager should fail for any reason, a new file system manager is selected by the cluster manager and all functions continue without disruption, except for the time required to accomplish the takeover.

Depending on the application workload, the memory and CPU requirements for the services provided by the file system manager may make it undesirable to run a resource intensive application on the same node as the file system manager. GPFS allows you to control the pool of nodes from which the file system manager is chosen through:

- The **mmcrcluster** command, when creating your cluster
- The **mmaddnode** command, when adding nodes to your cluster
- The **mmchnode** command, to change a node's designation at any time

These preferences are honored except in certain failure situations where multiple failures occur. For more information, see *Multiple file system manager failures* in *IBM Spectrum Scale: Problem Determination Guide*. You may list which node is currently assigned as the file system manager by issuing the **mmlsmgr** command or change which node has been assigned to this task through the **mmchmgr** command.

The metanode

There is one metanode per open file. The metanode is responsible for maintaining file metadata integrity.

In almost all cases, the node that has had the file open for the longest continuous period of time is the metanode. All nodes accessing a file can read and write data directly, but updates to metadata are written only by the metanode. The metanode for each file is independent of that for any other file and can move to any node to meet application requirements.

CES node (protocol node)

Only nodes that are designated as CES nodes can serve integrated protocol function.

Nodes in the cluster can be designated to be CES nodes using **mmchnode --ces-enable** *Node*. Each CES node will serve each of the protocols (NFS, SMB, Object) that are enabled. CES IP addresses assigned for protocol serving can failover to any of CES nodes that are up based on the failback policy configured. CES functionality can be designated only on nodes running on supported operating systems, and all the CES nodes must have the same platform (either all Intel or all POWER Big Endian).

For information about supported operating systems and their required minimum kernel levels, see IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html)

For more information about Cluster Export Services, see *Implementing Cluster Export Services* and *Configuring the CES and protocol configuration* in *IBM Spectrum Scale: Administration Guide*.

Related concepts:

"Protocols support overview: Integration of protocol access methods with GPFS" on page 27
IBM Spectrum Scale provides additional protocol access methods. Providing these additional file and object access methods and integrating them with GPFS offers several benefits. It enables users to consolidate various sources of data efficiently in one global namespace. It provides a unified data management solution and enables not only efficient space utilization but also avoids having to make unnecessary data moves just because access methods might be different.

AFM Gateway node

Each cache fileset in a cluster is served by one of the nodes designated as the gateway in the cluster.

The gateway node that is mapped to a fileset is called the primary gateway of the fileset. The primary gateway acts as the owner of the fileset and communicates with the home cluster. While using parallel data transfers, the primary gateway might also communicate with other participating gateway nodes for data transfer to home clusters.

All other nodes in the cluster, including other gateways, become the application nodes of the fileset. Therefore any node in the cache cluster can function as a gateway node and an application node for different filesets based on configuration of the node.

Use of disk storage and file structure within a GPFS file system

A file system (or stripe group) consists of a set of disks that store file data, file metadata, and supporting entities such as quota files and recovery logs.

When a disk is assigned to a file system, a file system descriptor is written on each disk. The *file system descriptor* is written at a fixed position on each of the disks in the file system and is used by GPFS to identify this disk and its place in a file system. The file system descriptor contains file system specifications and information about the state of the file system.

Within each file system, files are written to disk as in other UNIX file systems, using inodes, indirect blocks, and data blocks. Inodes and indirect blocks are considered *metadata*, as distinguished from data, or actual file content. You can control which disks GPFS uses for storing metadata when you create the file system with the **mmcrfs** command or when you modify the file system with the **mmchdisk** command.

The metadata for each file is stored in the inode and contains information such as file size and time of last modification. The inode also sets aside space to track the location of the data of the file. On file systems that are created in IBM Spectrum Scale v3.5 or later, if the file is small enough that its data can fit within this space, the data can be stored in the inode itself. This method is called data-in-inode and improves the performance and space utilization of workloads that use many small files. Otherwise, the data of the file must be placed in data blocks, and the inode is used to find the location of these blocks. The location-tracking space of the inode is then used to store the addresses of these data blocks. If the file is large enough, the addresses of all of its data blocks cannot be stored in the inode itself, and the inode points instead to one or more levels of indirect blocks. These trees of additional metadata space for a file can hold all of the data block addresses for very large files. The number of levels that are required to store the addresses of the data block is referred to as the *indirection level* of the file.

To summarize, on file systems that are created in v3.5 or later, a file typically starts out with data-in-inode. When it outgrows this stage, or if the file system was created before v3.5, the inode stores direct pointers to data blocks; this arrangement is considered a zero level of indirection. When more data blocks are needed, the indirection level is increased by adding an indirect block and moving the direct pointers there; the inode then points to this indirect block. Subsequent levels of indirect blocks are added as the file grows. The dynamic nature of the indirect block structure allows file sizes to grow up to the file system size.

For security reasons, encrypted files skip the data-in-inode stage. They always begin at indirection level zero.

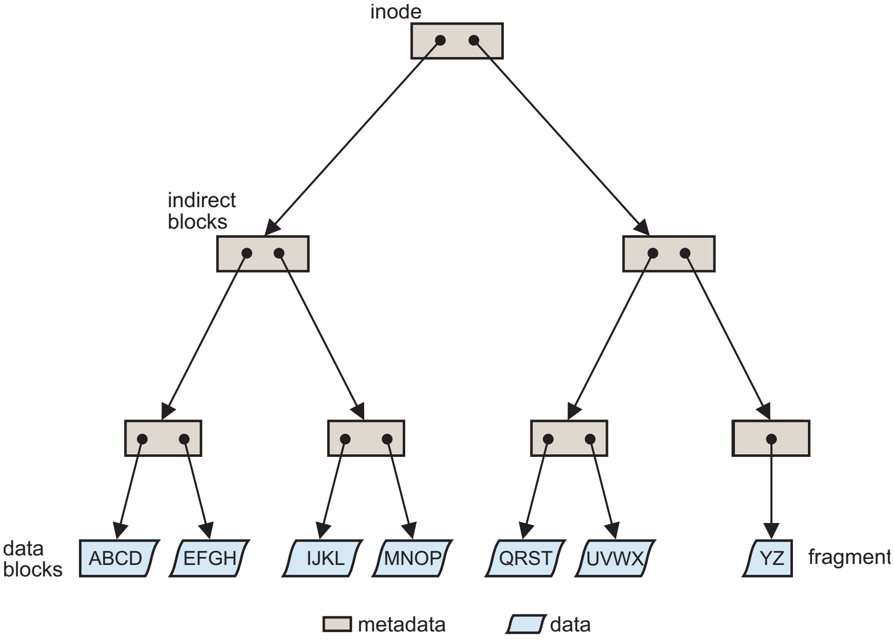


Figure 4. GPFS files have a typical UNIX structure

File system limitations:

1. The maximum number of mounted file systems within a GPFS cluster is 256.
2. The supported file system size depends on the version of GPFS that is installed.
3. The maximum number of files within a file system cannot exceed the architectural limit.

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

GPFS uses the file system descriptor to find all of the disks that make up the file system's stripe group, including their size and order. Once the file system descriptor is processed, it is possible to address any block in the file system. In particular, it is possible to find the first inode, which describes the *inode file*, and a small number of inodes that contain the rest of the file system information. The inode file is a collection of fixed-length records that represent a single file, directory, or link. The unit of locking is the single inode. Specifically, there are fixed inodes within the inode file for the following components:

- Root directory of the file system
- *Block allocation map*, which is a collection of bits that represent the availability of disk space within the disks of the file system. One unit in the allocation map represents a subblock. A subblock is the smallest unit of contiguous disk space that can be allocated to a file. The size of a subblock and the number of subblocks in a block are determined by the block size. The allocation map is broken into regions that reside on disk sector boundaries. The number of regions is set at file system creation time by the parameter that specifies how many nodes will access this file system. The regions are separately locked and as a result, different nodes can be allocating or de-allocating space that is represented by different regions independently and concurrently.
- *Inode allocation map*, which represents the availability of inodes within the inode file. The *Inode allocation map* is located in the *inode allocation file*, and represents all the files, directories, and links that can be created. The **mmchfs** command can be used to change the maximum number of files that can be created in the file system up to the architectural limit.

The data contents of each of these files are taken from the data space on the disks. These files are considered metadata and are allocated only on disks where metadata is allowed.

Quota files

For file systems with quotas enabled, quota files are created at file system creation time.

Note: Starting in GPFS 4.1, quota files no longer exist externally or they are no longer visible from the root directory of the file system; therefore, use **mmbackupconfig** to back up GPFS quota information.

There are three quota files for a file system:

- **user.quota** for users
- **group.quota** for groups
- **fileset.quota** for filesets

For every user who works within the file system, the **user.quota** file contains a record of limits and current usage within the file system for the individual user. If default quota limits for new users of a file system have been established, this file also contains a record for that value.

For every group whose users work within the file system, the **group.quota** file contains a record of common limits and the current usage within the file system of all the users in the group. If default quota limits for new groups of a file system have been established, this file also contains a record for that value.

For every fileset, the **fileset.quota** file contains a record of limits and current usage within the fileset. If default quota limits for filesets have been established, this file also contains a record for that value. The quota limit on blocks and inodes in a fileset are independent of the limits for specific users or groups of users. During allocation, the corresponding the limits for users, groups, and filesets are checked and the lowest threshold is applied.

GPFS recovery logs

GPFS recovery logs are created at file system creation. Additional recovery logs are automatically created as needed. The file system manager assigns a recovery log to each node that accesses the file system.

GPFS maintains the atomicity of the on-disk structures of a file through a combination of rigid sequencing of operations and logging. The data structures maintained are the inode, indirect blocks, the allocation map, and the data blocks. Data blocks are written to disk before any control structure that references the data is written to disk. This ensures that the previous contents of a data block can never be seen in a new file. Allocation blocks, inodes, and indirect blocks are written and logged in such a way that there will never be a pointer to a block marked unallocated that is not recoverable from a log.

Recovery logs are replicated only if default metadata replication is turned on (**-m 2**) or if explicitly enabled for logs (**--log-replicas 2**). You can check to see if log replication is enabled for a file system using the **mmlsfs** command and looking at the value of the **-m** and **--log-replicas** parameters. If both are set, the **--log-replicas** value takes precedence over the **-m** value for log replication.

There are certain failure cases where blocks are marked allocated but not yet assigned to a file, and these can be recovered by running **mmfsck** in online or offline mode. Log recovery is run as part of:

1. The recovery of a node failure affecting the objects that the failed node might have had locked.
2. A **mount** after the file system has been unmounted everywhere.

Note: Space that is not used by metadata, quota files, and recovery logs is used for user data and directories and allocated from the block allocation map as needed.

GPFS and memory

GPFS uses three areas of memory: memory allocated from the kernel heap, memory allocated within the daemon segment, and shared segments accessed from both the daemon and the kernel.

Memory allocated from the kernel heap

GPFS uses kernel memory for control structures such as vnodes and related structures that establish the necessary relationship with the operating system.

Memory allocated within the daemon segment

GPFS uses daemon segment memory for file system manager functions. Because of that, the file system manager node requires more daemon memory since token states for the entire file system are initially stored there. File system manager functions requiring daemon memory include:

- Structures that persist for the execution of a command
- Structures that persist for I/O operations
- States related to other nodes

The file system manager is a token manager, and other nodes may assume token management responsibilities; therefore, any manager node may consume additional memory for token management. For more information, see *Using multiple token servers* in *IBM Spectrum Scale: Administration Guide*.

Shared segments accessed from both the daemon and the kernel

Shared segments consist of both pinned and unpinned memory that is allocated at daemon startup. The initial values are the system defaults. However, you can change these values later using the **mmchconfig** command. See “Cluster configuration file” on page 146.

The pinned memory is called the *pagepool* and is configured by setting the **pagepool** cluster configuration attribute. This pinned area of memory is used for storing file data and for optimizing the performance of various data access patterns. In a non-pinned area of the shared segment, GPFS keeps information about open and recently opened files. This information is held in two forms:

1. A full inode cache

2. A stat cache

Pinned memory

GPFS uses pinned memory (also called page pool memory) for storing file data and metadata in support of I/O operations.

With some access patterns, increasing the amount of page pool memory can increase I/O performance. Increased page pool memory can be useful in the following cases:

- There are frequent writes that can be overlapped with application execution.
- There is frequent reuse of file data that can fit in the page pool.
- The I/O pattern contains various sequential reads large enough that the prefetching of data improves performance.

Pinned memory regions cannot be swapped out to disk, which means that GPFS will always consume at least the value of the **pagepool** attribute in system memory. So consider the memory requirements of GPFS and other applications running on the node when determining a value for the **pagepool** attribute.

Non-pinned memory

There are two levels of cache used to store file metadata.

Inode cache

The inode cache contains copies of inodes for open files and for some recently used files that are no longer open. The **maxFilesToCache** parameter controls the number of inodes cached by GPFS. Every open file on a node consumes a space in the inode cache. Additional space in the inode cache is used to store the inodes for recently used files in case another application needs that data.

The number of open files can exceed the value defined by the **maxFilesToCache** parameter to enable applications to operate. However, when the **maxFilesToCache** number is exceeded, there is no more caching of recently open files, and only open file inode data is kept in the cache.

Stat cache

The stat cache contains enough information to respond to inquiries about the file and open it, but not enough information to read from it or write to it. There is sufficient data from the inode in the stat cache to respond to a **stat()** call (for example, when issuing the **ls -l** command on a UNIX node). A stat cache entry consumes significantly less memory than a full inode. Stat cache entries are kept for the following:

- Recently accessed files
- Directories recently accessed by a number of **stat()** calls

To set the value of the stat cache, use the **maxStatCache** attribute of the **mmchconfig** command.

Notes:

1. GPFS prefetches data for stat cache entries if a pattern of use indicates this will be productive (for example, if a number of **ls -l** commands issued for a large directory).

   On Linux, **maxStatCache** is not used. All node caching is done in the pool controlled by **maxFilesToCache**. Therefore, on Linux, if you are not using local read-only cache (LROC), set **maxStatCache** to 0 so that inodes do not pass through the stat cache to get to the **maxFilesToCache** pool. The passthrough is an extra step in processing an inode. If you are using LROC, the pool used by **maxStatCache** is utilized and it must be tuned. The size of the GPFS shared segment can limit the maximum setting of **maxStatCache**.
2. Each entry in the inode cache and the stat cache requires appropriate tokens:
   a. To ensure the cached information remains correct
   b. For the storage of tokens on the file system manager node

3. Depending on the usage pattern, system performance may degrade when an information update requires revoking a token. This happens when two or more nodes share the same information and the most recent information is moved to a different location. When the current node needs to access the updated information, the token manager must revoke the token from the current node before that node can access the information in the new location.

GPFS and network communication

Within the GPFS cluster, you can specify different networks for GPFS daemon communication and for GPFS command usage.

You make these selections using the **mmaddnode**, **mmchnode**, and **mmcrcluster** commands. In these commands, the node descriptor allows you to specify separate node interfaces for those functions on each node. The correct operation of GPFS is directly dependent upon these selections.

GPFS may not work properly if there is a firewall enabled on the nodes within the cluster. To ensure proper operation, you must either configure the firewall to allow the appropriate ports or disable the firewall. For more information, see *GPFS port usage* in *IBM Spectrum Scale: Administration Guide* and *mmnetverify command* in *IBM Spectrum Scale: Command and Programming Reference*.

GPFS daemon communication

In a cluster environment, the GPFS daemon depends on the correct operation of TCP/IP.

These dependencies exist because:

- The communication path between nodes must be built at the first attempt to communicate.
- Each node in the cluster is required to communicate with the cluster manager and the file system manager during startup and mount processing.
- Once a connection is established, it must remain active until the GPFS daemon is shut down on the nodes.

Note: Establishing other communication paths depends upon application usage among nodes.

The daemon also uses sockets to communicate with other instances of the file system on other nodes. Specifically, the daemon on each node communicates with the file system manager for allocation of logs, allocation segments, and quotas, as well as for various recovery and configuration flows. GPFS requires an active internode communications path between all nodes in a cluster for locking, metadata coordination, administration commands, and other internal functions. The existence of this path is necessary for the correct operation of GPFS. The instance of the GPFS daemon on a node will go down if it senses that this communication is not available to it. If communication is not available to another node, one of the two nodes will exit GPFS.

Using public and private IP addresses for GPFS nodes:

GPFS permits the system administrator to set up a cluster such that both public and private IP addresses are in use. For example, if a cluster has an internal network connecting some of its nodes, it is advantageous to use private IP addresses to communicate on this network, and public IP addresses to communicate to resources outside of this network.

Public IP addresses are those that can be used to access the node from any other location for which a connection exists. Private IP addresses may be used only for communications between nodes directly connected to each other with a communications adapter. Private IP addresses are assigned to each node at hardware setup time, and must be in a specific address range (IP addresses on a 10.0.0.0, 172.16.0.0, or 192.168.0.0 subnet). For more information on private IP addresses, refer to RFC 1597 - Address Allocation for Private Internets(www.ip-doc.com/rfc/rfc1597).

The **subnets** operand on the **mmchconfig** command specifies an ordered list of **subnets** available to GPFS for private TCP/IP communications. Each subnet listed may have a list of cluster names (allowing shell-style wild cards) that specifies other GPFS clusters that have direct access to the same subnet.

When the GPFS daemon starts on a node, it obtains a list of its own IP addresses and associated subnet masks from its local IP configuration. For each IP address, GPFS checks whether that address is on one of the subnets specified on the **subnets** configuration parameter. It records the list of its matching IP addresses and subnet masks, and then listens for connections on any of these addresses. If any IP address for the node (specified when the cluster was created or when the node was added to the cluster), is not specified with the **subnets** configuration parameter, GPFS automatically adds it to the end of the node's IP address list.

Therefore, when you use public IP addresses for a node, there is no need to explicitly list the public IP subnet with the **subnets** configuration parameter. For example, the normal way to configure a system would be to use host names that resolve to the external Ethernet IP address in the **mmcrcluster** command, and then, if the system administrator wants GPFS to use the High Performance Switch within the cluster, add one **subnets** configuration parameter for the HPS subnet. It is acceptable to add two **subnets** configuration parameters, one for the HPS and one for the external Ethernet, making sure that they are in that order. In this case, it does not matter which of each node's two addresses was specified when the cluster was created or when the node was added to the cluster. For example, to add remote access to an existing cluster that was using only switch addresses, it is sufficient to add two **subnets** configuration parameters.

When a node joins a cluster (its own cluster on startup, or another cluster when mounting a file system owned by another cluster), the node sends its list of IP addresses (ordered according to the order of **subnets** configuration parameters) to the cluster manager node, which forwards the list to all other nodes as part of the join protocol. No other additional information needs to be propagated.

When a node attempts to establish a connection to another node, GPFS determines the destination IP address to use according to this procedure:

1. For each of its own IP addresses, it searches the other node's list of IP addresses for an address that is on the same subnet.
   - For normal public IP addresses this is done by comparing IP address values ANDed with the node's subnet mask for its IP address.
   - For private IP addresses GPFS assumes that two IP addresses are on the same subnet only if the two nodes are within the same cluster, or if the other node is in one of the clusters that are explicitly listed in the **subnets** configuration parameter.
2. If the two nodes have more than one IP address pair on a common subnet, GPFS uses the first one found according to the order of **subnets** specified in the initiating node's configuration parameters.
3. If there are no two IP addresses on the same subnet, GPFS uses the last IP address in each node's IP address list. In other words, the last subnet that is specified in the **subnets** configuration parameter is assumed to be on a network that is accessible from the outside.

For more information and an example, see *Using remote access with public and private IP addresses* in *IBM Spectrum Scale: Administration Guide*.

Network communication and GPFS administration commands

Socket communications are used to process GPFS administration commands. Depending on the nature of the command, GPFS may process commands either on the node issuing the command or on the file system manager. The actual command processor merely assembles the input parameters and sends them along to the daemon on the local node using a socket.

Some GPFS commands permit you to specify a separate administrative network name. You make this specification using the **AdminNodeName** field of the node descriptor. For additional information, see *IBM Spectrum Scale: Command and Programming Reference* for descriptions of these commands:

- **mmaddnode**
- **mmchnode**
- **mmcrcluster**

If the command changes the state of a file system or its configuration, the command is processed at the file system manager. The results of the change are sent to all nodes and the status of the command processing is returned to the node, and eventually, to the process issuing the command. For example, a command to add a disk to a file system originates on a user process and:

1. Is sent to the daemon and validated.
2. If acceptable, it is forwarded to the file system manager, which updates the file system descriptors.
3. All nodes that have this file system are notified of the need to refresh their cached copies of the file system descriptor.
4. The return code is forwarded to the originating daemon and then to the originating user process.

Be aware this chain of communication may allow faults related to the processing of a command to occur on nodes other than the node on which the command was issued.

Application and user interaction with GPFS

There are four ways to interact with a GPFS file system.

You can interact with a GPFS file system using:

- The IBM Spectrum Scale management GUI described in "Introduction to IBM Spectrum Scale GUI" on page 94
- Operating system commands, which are run at GPFS daemon initialization time or at file system mount time (see "Operating system commands")
- Operating system calls such as **open()**, from an application requiring access to a file controlled by GPFS (see "Operating system calls" on page 20)
- GPFS commands described in *IBM Spectrum Scale: Command and Programming Reference*
- GPFS programming interfaces described in *IBM Spectrum Scale: Command and Programming Reference*
- IBM Spectrum Scale management API described in "IBM Spectrum Scale management API" on page 97

Operating system commands

Operating system commands operate on GPFS data during the following scenarios.

- The initialization of the GPFS daemon
- The mounting of a file system

Initialization of the GPFS daemon:

GPFS daemon initialization can be done automatically as part of the node startup sequence, or manually using the **mmstartup** command.

The daemon startup process loads the necessary kernel extensions, if they have not been previously loaded by another process. The daemon then waits for the cluster manager to declare that a quorum exists. When quorum is achieved, the cluster manager changes the state of the group from *initializing* to *active*. You can see the transition to active state when the "mmfsd ready" message appears in the GPFS log file (**/var/adm/ras/mmfs.log.latest**) or by running the **mmgetstate** command. When this state changes from *initializing* to *active*, the daemon is ready to accept mount requests.

The mounting of a file system:

GPFS file systems are mounted using the GPFS **mmmount** command.

On AIX or Linux you can also use the operating system's **mount** command. GPFS mount processing builds the structures required to provide a path to the data and is performed on both the node requesting the mount and the file system manager node. If there is no file system manager, a call is made to the cluster manager, which appoints one. The file system manager ensures that the file system is ready to be mounted. The file system manager ensures that each of the following is true:

- There are no conflicting utilities being run by the **mmfsck** command, which checks and repairs a file system.
- There are no conflicting utilities being run by the **mmcheckquota**, which checks file system user, group and fileset quotas.
- All of the disks are available.
- Any necessary file system log processing is completed to ensure that metadata on the file system is consistent.

On the local node, the control structures required for a mounted file system are initialized and the token management function domains are created. In addition, paths to each of the disks that make up the file system are opened. Part of mount processing involves unfencing the disks, which may be necessary if this node had previously failed. This is done automatically without user intervention. If insufficient disks are up, the mount will fail. That is, in a replicated system if two disks are down in different failure groups, the mount will fail. In a non-replicated system, one disk down will cause the mount to fail.

Operating system calls

The most common interface to files residing in a GPFS file system is through normal file system calls to the operating system.

When a file is accessed, the operating system submits the request to the GPFS kernel extension, which attempts to satisfy the application request using data already in memory. If this can be accomplished, control is returned to the application through the operating system interface. If the data is not available in memory, the request is transferred for execution by a daemon thread. The daemon threads wait for work in a system call in the kernel, and are scheduled as necessary. Services available at the daemon level include the acquisition of tokens and disk I/O.

Operating system calls operate on GPFS data during:

- The opening of a file
- The reading of data
- The writing of data

Opening a GPFS file:

The **open** of a file residing in a GPFS file system involves the application making a call to the operating system specifying the name of the file.

Processing of a file **open** involves two stages:

1. Identifying the file specified by the application
2. Building the required data structures based on the inode

The kernel extension code processes the directory search. If the required information is not in memory, the daemon is called to acquire the necessary tokens for the directory or part of the directory needed to resolve the lookup, then reads the directory from disk into memory.

The lookup process occurs one directory at a time in response to calls from the operating system. In the final stage of **open**, the inode for the file is read from disk and connected to the operating system vnode structure. This requires acquiring locks on the inode and a lock that indicates the presence to the metanode. The metanode is discovered or created any time a file is opened.

- If no other node has this file open, this node becomes the metanode.

- If another node has a previous open, then that node is the metanode and this node interfaces directly with the metanode for metadata operations.

If the **open** involves the creation of a new file, the appropriate locks are obtained on the parent directory and the inode allocation file block. The directory entry is created, an inode is selected and initialized and then **open** processing is completed.

Reading file data:

The GPFS **read** function is invoked in response to a **read** system call.

File **read** processing falls into three levels of complexity based on system activity and status:
1. Buffers are available in memory
2. Tokens are available in memory but data must be read
3. Data and tokens must be acquired

At the completion of a **read**, a determination of the need for prefetching is made. GPFS computes a desired read-ahead for each open file based on the performance of the disks, the data access pattern and the rate at which the application is reading data. If additional prefetching is needed, a message is sent to the daemon that processes it asynchronously with the completion of the current **read**.

Buffer and locks available in memory:

The simplest **read** operation occurs when the data is already available in memory, either because it has been pre-fetched or because it has been read recently by another **read** call.

In either case, the buffer is locally locked and the data is copied to the application data area. The lock is released when the copy is complete. Note that no token communication is required because possession of the buffer implies that the node at least has a read token that includes the buffer. After the copying, prefetching is initiated if appropriate.

Tokens available locally but data must be read:

The second, more complex, type of **read** operation is necessary when the data is not in memory.

This occurs under three conditions:
1. The token has been acquired on a previous **read** that found no contention.
2. The buffer has been stolen for other uses.
3. On some random **read** operations.

   In the first of a series of reads, the token is not available locally, but in the second read it might be available.

In such situations, the buffer is not found and must be read from disk. No token activity has occurred because the node has a sufficiently strong token to lock the required region of the file locally. A message is sent to the daemon, which is handled on one of the waiting daemon threads. The daemon allocates a buffer, locks the file range that is required so the token cannot be stolen for the duration of the I/O, and initiates the I/O to the device holding the data. The originating thread waits for this to complete and is posted by the daemon upon completion.

Data and tokens must be acquired:

The third, and most complex **read** operation requires that tokens and data be acquired on the application node.

The kernel code determines that the data is not available locally and sends a message to the daemon waiting after posting the message. The daemon thread determines that it does not have the required tokens to perform the operation. In that case, a token acquire request is sent to the token management server. The requested token specifies a required length of that range of the file, which is needed for this buffer. If the file is being accessed sequentially, a desired range of data, starting at this point of this read and extending to the end of the file, is specified. In the event that no conflicts exist, the desired range is granted, eliminating the need for additional token calls on subsequent reads. After the minimum token needed is acquired, the flow proceeds with the token management process described in "The file system manager" on page 10.

Writing file data:

write processing is initiated by a system call to the operating system, which calls GPFS when the **write** involves data in a GPFS file system.

GPFS moves data from a user buffer into a file system buffer synchronously with the application **write** call, but defers the actual write to disk. This asynchronous I/O technique allows better scheduling of the disk I/O operations and improved performance. The file system buffers come from the memory allocated based on the **pagepool** parameter in the **mmchconfig** command. Increasing the size of the pagepool may allow more writes to be deferred, which can improve performance in certain workloads.

A block of data is scheduled to be written to a disk when:

- The application has specified a synchronous **write**.
- The system needs the memory used to buffer the written data.
- The file token has been revoked.
- The last byte of a block of a file being written sequentially is written.
- A system **sync** command is run.

Until one of these occurs, the data remains in GPFS memory.

write processing falls into three levels of complexity based on system activity and status:

1. Buffer available in memory
2. Tokens available locally but data must be read
3. Data and tokens must be acquired

Metadata changes are flushed under a subset of the same conditions. They can be written either directly, if this node is the metanode, or through the metanode, which merges changes from multiple nodes. This last case occurs most frequently if processes on multiple nodes are creating new data blocks in the same region of the file.

Buffer available in memory:

The simplest path involves a case where a buffer already exists for this block of the file but may not have a strong enough token.

This occurs if a previous **write** call accessed the block and it is still resident in memory. The write token already exists from the prior call. In this case, the data is copied from the application buffer to the GPFS buffer. If this is a sequential **write** and the last byte has been written, an asynchronous message is sent to the daemon to schedule the buffer for writing to disk. This operation occurs on the daemon thread overlapped with the execution of the application.

Token available locally but data must be read:

There are two situations in which the token may exist but the buffer does not.

1. The buffer has been recently stolen to satisfy other needs for buffer space.
2. A previous **write** obtained a desired range token for more than it needed.

In either case, the kernel extension determines that the buffer is not available, suspends the application thread, and sends a message to a daemon service thread requesting the buffer. If the **write** call is for a full file system block, an empty buffer is allocated since the entire block will be replaced. If the **write** call is for less than a full block and the rest of the block exists, the existing version of the block must be read and overlaid. If the **write** call creates a new block in the file, the daemon searches the allocation map for a block that is free and assigns it to the file. With both a buffer assigned and a block on the disk associated with the buffer, the **write** proceeds as it would in "Buffer available in memory" on page 22.

Data and tokens must be acquired:

The third, and most complex path through **write** occurs when neither the buffer nor the token exists at the local node.

Prior to the allocation of a buffer, a token is acquired for the area of the file that is needed. As was true for **read**, if sequential operations are occurring, a token covering a larger range than is needed will be obtained if no conflicts exist. If necessary, the token management function will revoke the needed token from another node holding the token. Having acquired and locked the necessary token, the **write** will continue as in "Token available locally but data must be read."

The stat() system call:

The **stat()** system call returns data on the size and parameters associated with a file. The call is issued by the **ls -l** command and other similar functions.

The data required to satisfy the **stat()** system call is contained in the inode. GPFS processing of the **stat()** system call differs from other file systems in that it supports handling of **stat()** calls on all nodes without funneling the calls through a server.

This requires that GPFS obtain tokens that protect the accuracy of the metadata. In order to maximize parallelism, GPFS locks inodes individually and fetches individual inodes. In cases where a pattern can be detected, such as an attempt to **stat()** all of the files in a larger directory, inodes will be fetched in parallel in anticipation of their use.

Inodes are cached within GPFS in two forms:

1. Full inode
2. Limited stat cache form

The full inode is required to perform data I/O against the file.

The stat cache form is smaller than the full inode, but is sufficient to open the file and satisfy a **stat()** call. It is intended to aid functions such as **ls -l**, **du**, and certain backup programs that scan entire directories looking for modification times and file sizes.

These caches and the requirement for individual tokens on inodes are the reason why a second invocation of directory scanning applications may run faster than the first.

GPFS command processing

GPFS commands fall into two categories: those that are processed locally and those that are processed at the file system manager for the file system involved in the command.

The file system manager is used to process any command that alters the state of the file system. When commands are issued and the file system is not mounted, a file system manager is appointed for the task. The **mmchdisk** command and the **mmfsck** command represent two typical types of commands that are processed at the file system manager.

The mmchdisk command:

The **mmchdisk** command is issued when a failure that caused the unavailability of one or more disks has been corrected. The need for the command can be determined by the output of the **mmlsdisk** command.

mmchdisk performs four major functions:
- It changes the availability of the disk to **recovering**, and to **up** when all processing is complete. All GPFS utilities honor an availability of **down** and do not use the disk. **recovering** means that recovery has not been completed but the user has authorized use of the disk.
- It restores any replicas of data and metadata to their correct value. This involves scanning all metadata in the system and copying the latest to the recovering disk. Note that this involves scanning large amounts of data and potentially rewriting all data on the disk. This can take a long time for a large file system with a great deal of metadata to be scanned.
- It stops or suspends usage of a disk. This merely involves updating a disk state and should run quickly.
- Change disk attributes' metadata.

Subsequent invocations of **mmchdisk** will attempt to restore the replicated data on any disk left in with an availability of **recovering**

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

The mmfsck command:

The **mmfsck** command repairs file system structures.

The **mmfsck** command operates in two modes:
1. online
2. offline

For performance reasons, GPFS logging allows the condition where disk blocks are marked **used** but not actually part of a file after a node failure. The online version of **mmfsck** cleans up that condition. Running **mmfsck -o -n** scans the file system to determine if correction might be useful. The online version of **mmfsck** runs on the file system manager and scans all inodes and indirect blocks looking for disk blocks that are allocated but not used. If authorized to repair the file system, it releases the blocks. If not authorized to repair the file system, it reports the condition to standard output on the invoking node.

The offline version of **mmfsck** is the last line of defense for a file system that cannot be used. It will most often be needed in the case where GPFS recovery logs are not available because of disk media failures. The **mmfsck** command runs on the file system manager and reports status to the invoking node. It is mutually incompatible with any other use of the file system and checks for any running commands or any nodes with the file system mounted. It exits if any are found. It also exits if any disks are **down** and require the use of **mmchdisk** to change them to **up** or **recovering**. The **mmfsck** command performs a full file system scan looking for metadata inconsistencies. This process can be lengthy on large file systems. It seeks permission from the user to repair any problems that are found, which may result in the removal of files or directories that are corrupt. The processing of this command is similar to those for other file systems.

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

NSD disk discovery

When the GPFS daemon starts on a node, it discovers the disks defined as NSDs by reading a disk descriptor that is written on each disk owned by GPFS. This enables the NSDs to be found regardless of the current operating system device name assigned to the disk.

On UNIX, NSD discovery is done by the GPFS shell script **/usr/lpp/mmfs/bin/mmdevdiscover**, which generates a list of available disk devices that appear in the node's local **/dev** file system. To override or enhance NSD discovery, you can create a script and name it **/var/mmfs/etc/nsddevices**. The user-created **nsddevices** script, if it exists, is executed before the default discovery process.

On Windows, NSDs have a GUID Partition Table (GPT) with a single GPFS partition. NSD discovery is done by scanning the system for a list of disks that contain a GPFS partition.

On all platforms, the list of disk devices is then used by the GPFS daemon to determine whether a device interface on the local node maps to an NSD name recorded in the configuration database. The process of mapping a device interface to an NSD involves GPFS opening each device in turn and reading any NSD volume identifier potentially recorded at sector two of the disk.

If GPFS discovers that an NSD volume identifier read from a disk device matches the volume identifier recorded with the NSD name in the GPFS configuration database, I/O for the local node proceeds over the local device interface.

If no device mapping appears on the local node for a particular NSD, I/O proceeds over the IP network to the first NSD server specified in the server list for that NSD. If the first NSD server in the server list is not available, I/O proceeds sequentially through the server list until it finds an available NSD server.

Consult the **/usr/lpp/mmfs/samples/nsddevices.sample** file for configuration guidelines on how to provide additional disk discovery capabilities unique to your configuration.

Failure recovery processing

In general, it is not necessary to understand the internals of GPFS failure recovery processing since it is done automatically. However, some familiarity with the concepts might be useful when failures are observed.

It should be noted that only one state change, such as the loss or initialization of a node, can be processed at a time and subsequent changes are queued. This means that the entire failure processing must complete before the failed node can join the group again. All failures are processed first, which means that GPFS handles all failures prior to completing any recovery.

GPFS recovers from a node failure using join or leave processing messages that are sent explicitly by the cluster manager node. The cluster manager node observes that a node has failed when it no longer receives heartbeat messages from the node. The join or leave processing messages are broadcast to the entire group of nodes running GPFS, and each node updates its current status for the failing or joining node. Failure of the cluster manager node results in a new cluster manager being elected by the cluster. Then the newly elected cluster configuration manager node processes the failure message for the failed cluster manager.

When notified that a node has failed or that the GPFS daemon has failed on a node, GPFS invokes recovery for each of the file systems that were mounted on the failed node. If necessary, new file system managers are selected for any file systems that no longer have one.

The file system manager for each file system ensures the failed node no longer has access to the disks comprising the file system. If the file system manager is newly appointed as a result of this failure, it rebuilds token state by querying the other nodes in the group. After this is complete, the actual recovery of the log of the failed node proceeds. This recovery rebuilds the metadata that was being modified at the time of the failure to a consistent state. In some cases there may be blocks that are allocated that are not part of any file and are effectively lost until **mmfsck** is run, online or offline. After log recovery is complete, the locks held by the failed nodes are released for this file system. When this activity is completed for all file systems, failure processing is done. The last step of this process allows a failed node to rejoin the cluster.

Cluster configuration data files

GPFS commands store configuration and file system information in one or more files collectively known as GPFS cluster configuration data files. These files are not intended to be modified manually.

The GPFS administration commands are designed to keep these files synchronized between each other and with the GPFS system files on each node in the cluster. The GPFS commands constantly update the GPFS cluster configuration data files and any user modification made to this information may be lost without warning. On AIX nodes this includes the GPFS file system stanzas in **/etc/filesystems** and on Linux nodes the lists in **/etc/fstab**.

The GPFS cluster configuration data is stored in the **/var/mmfs/gen/mmsdrfs** file. This file is stored on the nodes designated as the *primary GPFS cluster configuration server* and, if specified, the secondary GPFS cluster configuration server. See “IBM Spectrum Scale cluster configuration information” on page 144. The first record in the **mmsdrfs** file contains a generation number. Whenever a GPFS command causes something to change in the cluster or any of the file systems, this change is reflected in the **mmsdrfs** file and the generation number is increased by one. The latest generation number is always recorded in the **mmsdrfs** file on the primary and secondary GPFS cluster configuration server nodes.

When running GPFS administration commands, it is necessary for the GPFS cluster configuration data to be accessible to the node running the command. Commands that update the **mmsdrfs** file require that both the primary and, if specified, the secondary GPFS cluster configuration server nodes are accessible. If one of the cluster configuration server nodes is inaccessible, you can designate a new primary or secondary cluster configuration servers using the **mmchcluster** command. Similarly, when the GPFS daemon starts up, at least one of the two server nodes must be accessible.

Starting with GPFS 4.1, the master copy of configuration data files may be stored redundantly on all quorum nodes instead of the separately designated primary/backup configuration server nodes. This method of storing configuration data is called the cluster configuration repository (CCR) and is the default for new clusters created on GPFS 4.1 or later. Existing clusters can be converted to the new repository type using the **–ccr-enable** option of the **mmchcluster** command.

Using CCR has the advantage that full read/write access to the configuration data remains available as long as a majority of quorum nodes are accessible. For example, in a cluster with five quorum nodes, commands that update the **mmsdrfs** file will continue to work normally, even if any two of the five quorum nodes have failed. In a two-node cluster with tiebreaker disks, it is still possible to run commands that change the **mmsdrfs** file if one of the two nodes has failed, as long as the surviving node has access to the tiebreaker disks. In general, full configuration command functionality remains available as long as enough nodes and, if specified, tiebreaker disks are accessible for GPFS to reach quorum. The CCR also has the advantage that it allows changing the tiebreaker disk configuration, including switching between node-based quorum and node quorum with tiebreaker, without first shutting down GPFS on all of the nodes.

Based on the information in the GPFS cluster configuration data, the GPFS commands generate and maintain a number of system files on each of the nodes in the GPFS cluster.

Linux /etc/fstab

On Linux nodes, contains lists for all GPFS file systems that exist in the cluster.

AIX /etc/filesystems

On AIX nodes, contains lists for all GPFS file systems that exist in the cluster.

All GPFS nodes

/var/mmfs/gen/mmfsNodeData

Contains GPFS cluster configuration data pertaining to the node.

/var/mmfs/gen/mmsdrfs

Contains a local copy of the **mmsdrfs** file found on the primary and secondary GPFS cluster configuration server nodes.

/var/mmfs/gen/mmfs.cfg

Contains GPFS daemon startup parameters.

GPFS backup data

The GPFS **mmbackup** command creates several files during command execution. Some of the files are temporary and deleted at the end of the backup operation. There are other files that remain in the root directory of the fileset or the file system and they must not be deleted.

The **mmbackup** command creates other files that begin with **.mmbackupShadow.***. These files are associated with the **mmbackup** command and are required for proper backups to be complete, so do not manually delete or change them.

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

Protocols support overview: Integration of protocol access methods with GPFS

IBM Spectrum Scale provides additional protocol access methods. Providing these additional file and object access methods and integrating them with GPFS offers several benefits. It enables users to consolidate various sources of data efficiently in one global namespace. It provides a unified data management solution and enables not only efficient space utilization but also avoids having to make unnecessary data moves just because access methods might be different.

The additional protocol access methods that are integrated with GPFS are file access that use NFS and SMB and object access by using OpenStack Swift. While each of these server functions (NFS, SMB and Object) use open source technologies, this integration adds value by providing scaling and by providing high availability by using the clustering technology in GPFS.

The CES BLOCK service provides iSCSI protocol support. You can export a file resident in a GPFS™ file system as a virtual iSCSI volume to an iSCSI initiator by using the iSCSI protocol.

- The integration of file and object serving with GPFS allows the ability to create NFS exports, SMB shares, and OpenStack Swift containers that have data in GPFS file systems for access by client systems that do not run GPFS.
- Some nodes (at least two recommended) in the GPFS cluster have to be designated as protocol nodes (also called CES nodes) from which (non-GPFS) clients can access data that resides in and managed by GPFS by using the appropriate protocol artifacts (exports, shares, or containers).
- The protocol nodes need to have GPFS server license designations.
- The protocol nodes must be configured with “external” network addresses that are used to access the protocol artifacts from clients. The (external) network addresses are different from the GPFS cluster address that is used to add the protocol nodes to the GPFS cluster.
- The integration that is provided allows the artifacts to be accessed from any of the protocol nodes through the configured network addresses. Further, the integration that is provided allows network addresses that are associated with protocol nodes to fail over to other protocol nodes when a protocol node fails.

All the protocol nodes must be running the supported operating systems, and the protocol nodes must be all Power®, in both big and little endian mode, or all Intel. Although the other nodes in the GPFS cluster could be on other platforms and operating systems.

For information about supported operating systems for protocol nodes and their required minimum kernel levels, see IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

Like GPFS, the protocol serving functionality is also delivered only as software.

- The intent of the functionality is to provide access to data managed by GPFS through more access methods.
- While the protocol function provides several aspects of NAS file serving, the delivery is not a NAS appliance. Therefore, in other words, the GPFSGPFS-style command line interface requiring root access is still available, and itGPFS is not like an appliance from an administrative management perspective.
- Role-based access control of the command line interface is not offered.
- Further, the types of workloads suited for this delivery continue to be workloads that require the scaling/consolidation aspects that are associated with traditional GPFS.

  Note: Some NAS workloads might not be suited for delivery in the current release. For instance, extensive use of snapshots, or support for a large number of SMB users.

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

Along with the protocol-serving function, the delivery includes the installation toolkit as well as some performance monitoring infrastructure.

- The GPFS code, including the server function for the three (NFS, SMB, Object) protocols, along with the installation toolkit and performance monitoring infrastructure, are delivered via a self-extracting archive package just like traditional GPFS.
- The use of the protocol server function requires extra licenses that need to be accepted. A GPFS package without protocols continues to be provided for those users who do not wish to accept these additional license terms.

Note: Even though some of the components that are provided are open source, the specific packages that are provided must be used. If there are existing versions of these open source packages on your system, they must be removed before you install this software.

Several commands are introduced or enhanced to enable the use of the described functions.

- The commands for managing these functions include **spectrumscale**, **mmces**, **mmuserauth**, **mmnfs**, **mmsmb**, **mmobj**, and **mmperfmon**.
- In addition, **mmdumpperfdata** and **mmprotocoltrace** are provided to help with data collection and tracing.
- Existing GPFS commands are expanded with some options for protocols include **mmlscluster**, **mmchnode**, and **mmchconfig**.
- The **gpfs.snap** command is enhanced to include data by gathering about the protocols to help with problem determination.

For information on the use of CES including administering and managing the protocols, see the *Implementing Cluster Export Services* chapter of *IBM Spectrum Scale: Administration Guide*.

IBM Spectrum Scale enables you to build a **data ocean** solution to eliminate silos, improve infrastructure utilization, and automate data migration to the best location or tier of storage anywhere in the world. You can start small with just a few commodity servers by fronting commodity storage devices and then grow to a data lake architecture or even an ocean of data. IBM Spectrum Scale is a proven solution in some of the most demanding environments with massive storage capacity under the single global namespace. Furthermore, your data ocean can store either files or objects and you can run analytics on the data in-place, which means that there is no need to copy the data to run your jobs.

The installation toolkit is provided to help with the installation and configuration of GPFS as well as protocols.

- While it is designed for a user who might not be familiar with GPFS, it can help ease the installation and configuration process of protocols even for experienced GPFS administrators.
- The installation toolkit can help with prechecks to validate environment, distribution of the RPMs from one node to the other nodes, and multiple GPFS administrative steps. **spectrumscale deploy** can be used to configure protocols on an existing GPFS cluster with an existing GPFS file system.

Note: The installation toolkit is only supported on some operating systems. Therefore, all nodes in the cluster must be running on supported operating systems if you want to use the **spectrumscale install** command. All protocol nodes must be on supported operating systems if you want to use the **spectrumscale deploy** command.

In addition to the installation toolkit, IBM Spectrum Scale also includes a performance monitoring toolkit. Sensors to collect performance information are installed on all protocol nodes, and one of these nodes is designated as a collector node. The **mmperfmon query** command can be used to view the performance counters that have been collected.

The **mmhealth** command can be used to monitor the health of the node and the services hosted on that node.

Some protocol use considerations:

- At time of release, several features in GPFS are not explicitly tested with protocol functionality. These include Local Read Only Cache, Multicluster, Encryption, File Placement Optimizer, and Hierarchical Storage Management. This is expected to work with protocols and will be tested with protocols over time. However, if you use one of these features before IBM claims support of it, ensure that it is tested with the expected workloads before you put it into production.
- Use of Clustered NFS (CNFS) is not compatible with use of Clustered Export Service (CES). You must choose one or the other. CNFS (which uses kernel NFS, a different NFS stack than the stack used by the CES infrastructure) continues to be supported; however, if you choose CNFS, you cannot take advantage of the integration of SMB and Object server functionality. If you choose to migrate from CNFS to CES, the CES infrastructure does not support a complete equivalent of CNFS group feature to control failover of IP addresses.
- For information regarding specific limitations about protocols and their integration with GPFS, see the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) and IBM Spectrum Scale in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/ibmspectrumscale_welcome.html).

NFS support overview

The NFS support for IBM Spectrum Scale enables clients to access the GPFS file system by using NFS clients with their inherent NFS semantics.

The following features are provided:

Clustered NFS Support

NFS clients can connect to any of the protocol nodes and get access to the exports defined. A clustered registry makes sure that all nodes see the same configuration data. This means that it does not matter to the client to which of the CES nodes the connections are established. Moreover, the state of opened files is also shared among the CES nodes so that data integrity is maintained. On failures, clients can reconnect to another cluster node as the IP addresses of failing nodes are automatically transferred to another healthy cluster node. The supported protocol levels are NFS version 3 (NFSv3) and NFS version 4 (NFSv4.0).

Export management commands

With the **mmnfs export** command, IBM Spectrum Scale provides a comprehensive entry point to manage all NFS-related export tasks such as creating, changing, and deleting NFS exports. The **mmnfs export** command follows the notions of supported options, that is, a limited set of NFS-related options that have proven useful. For more information, see the **mmnfs** command in the *IBM Spectrum Scale: Command and Programming Reference*.

Export configuration commands

With the **mmnfs config** command, IBM Spectrum Scale provides a tool for administering the global NFS configuration. You can use this command to set and list default settings such as the port number for the NFS service, the default access mode for exported file systems, and the NFS server log level. This command follows the notions of supported options, that is, a limited set of NFS-related options that have proven useful. For more information, see the **mmnfs** command in the *IBM Spectrum Scale: Command and Programming Reference*.

NFS monitoring

The monitoring framework detects issues with the NFS services and triggers failover in case of an unrecoverable error. Moreover, the **mmces** command provides a quick access to current and past system states and these states enable you to diagnose issues with the NFS services on the CES nodes. Issues that are detected and causing failover are, for example, GPFS daemon failures, node failures, or NFS service failures. For more information, see the **mmces** command in the *IBM Spectrum Scale: Command and Programming Reference*.

Integrated install

The integrated installer allows the installation of NFS services, CES framework, and the other protocols (SMB and Object), if desired.

NFS performance metrics

The NFS services provide performance metrics that are collected by the performance monitor framework. The **mmperfmon query** tool provides access to the most important NFS metrics through predefined queries. For more information, see the **mmperfmon** command in the *IBM Spectrum Scale: Command and Programming Reference*.

Cross-protocol integration with SMB

IBM Spectrum Scale enables concurrent access to the same file data by using NFS, SMB, and native POSIX access (limitations apply). The first release does not allow the sharing of data through the traditional file protocols and Object.

Authentication and ID mapping

You can configure NFS services to authenticate against the most popular authentication services such as Microsoft Active Directory and LDAP. Mapping Microsoft security identifiers (SIDs) to the POSIX user and group IDs on the file server can either be done automatically or by using the external ID mapping service like RFC 2307. If none of the offered authentication and mapping schemes match the environmental requirements, the option to establish a user-defined configuration is available. The

mmuserauth service create command can be used to set up all authentication-related settings. For more information, see the **mmuserauth** command in the *IBM Spectrum Scale: Command and Programming Reference*.

SMB support overview

The SMB support for IBM Spectrum Scale 4.1.1 and later allows clients to access the GPFS file system using SMB clients with their 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, that is, for the client it does not matter to which of the CES nodes the connections are established. Moreover, the state of opened files (share modes, open modes, access masks, locks, etc.) is also shared among the CES nodes so that data integrity is maintained. On failures, clients can reconnect to another cluster node as the IP addresses of failing nodes are transferred to another healthy cluster node. The supported protocol levels are SMB2 and the base functionality of SMB3 (dialect negotiation, secure negotiation, encryption of data on the wire).
- Compatibility with SSSD

  Limited SMB protocol compatibility with SSSD. SSSD is the recommended choice by RedHat to connect servers to AD, but to provide full SMB authentication we require winbind to work.
- 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 and administering the global configuration. The **mmsmb** command follows the notions of supported options, that is, a limited set of SMB related options that have proven useful. Moreover, the Microsoft Management Console can be used to administer SMB shares.
- SMB monitoring

  The monitoring framework will detect issues with the SMB services and will trigger failover in case of an unrecoverable error. Moreover, the **mmhealth** and **mmces** command provide a quick access to current and past system states and aid to diagnose issues with the SMB services on the CES nodes. Issues detected and causing failover are, for example, GPFS daemon failures, node failures, or SMB service failures.
- Integrated install

  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. Thus not just the current metrics can be retrieved but also historic data is available (at some lower granularity). The two sets of metrics are firstly global SMB metrics (like number of connects and disconnects) and secondly metrics on for each SMB request (number, time, throughput). The **mmperfmon query** tool provides access to all SMB metrics via predefined queries. Moreover, metrics for the clustered file meta-data base CTDB are collected and exposed via the **mmperfmon query** command.
- Cross-protocol integration with NFS

  IBM Spectrum Scale 4.1.1 and later allows concurrent access to the same file data via SMB, NFS and native POSIX access (limitations apply). IBM Spectrum Scale 4.2 and later releases include unified file and object access using which you can access data using object as well as file interfaces. For more information, see “Unified file and object access overview” on page 33.
- Authentication and ID Mapping

The SMB services can be configured to authenticate against the most popular 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 using the automatic or external ID mapping service like RFC 2307. If none of the offered authentication and mapping schemes matches the environmental requirements the option to establish a user-defined configuration is available. The **mmuserauth service create** command can be used to set up all authentication related settings.

Object storage support overview

IBM Spectrum Scale object storage combines the benefits of IBM Spectrum Scale with the most widely used open source object store, OpenStack Swift.

Data centers are currently struggling to efficiently and cost-effectively store and manage vast amounts of data. The increasing number of application domains, such as analytics, online transaction processing (OLTP), and high-performance computing (HPC), have created silos of storage within data centers. With each new application, a new storage system can be required, forcing system administrators to become experts in numerous storage management tools.

In addition, the set of applications that includes mobile and web-based applications, archiving, backup, and cloud storage has recently created yet another type of storage system for the system administrator to manage: object storage. Objects cannot be updated after they are created (although they can be replaced, versioned, or removed), and in many cases the objects are accessible in an eventually consistent manner. These types of semantics are well suited for images, videos, text documents, virtual machine (VM) images, and other similar files.

IBM Spectrum Scale object storage combines the benefits of IBM Spectrum Scale with the most widely used open source object store today, OpenStack Swift. In IBM Spectrum Scale object storage, data is managed as objects and it can be accessed over the network by using RESTful HTTP-based APIs. This object storage system uses a distributed architecture with no central point of control, providing greater scalability and redundancy. IBM Spectrum Scale object storage organizes data in the following hierarchy:

1. **Account**

   An account is used to group or isolate resources. Each object user is part of an account. Object users are assigned to a project (account) in keystone and can access only the objects that reside within the account. Each user is assigned to a project with a role that defines with the user rights and privileges to perform a specific set of operations on the resources of the account to which it belongs. Users can be part of multiple accounts but it is mandatory that a user must be associated with at least one account. You must create at least one account before adding users. Account contains a list of containers in the object storage. You can also define quota at the account level.

   Note: There is a one-to-one mapping between accounts in the object protocol and the keystone projects.

2. **Container**

   Containers contain objects. Each container maintains a lists of objects and provides a namespace for the objects. You can create any number of containers within an account.

3. **Object**

   Objects store data. You can create, modify, and delete objects. Accounts have containers, and containers store objects. Each object is accessed through a unique URL. The URLs in the API contain the storage account, container name, and object name. You can define quota both at the account and container levels.

IBM Spectrum Scale object storage also offers the following features.

Storage policies for object storage

Storage policies enable segmenting of the object storage within a single cluster for various use cases.

Currently, OpenStack Swift supports storage polices that allow you to define the replication settings and location of objects in a cluster. For more information about storage policies, see OpenStack Documentation for Storage Policies. IBM Spectrum Scale enhances storage policies to add the following functionality for object storage:

- **file-access** (unified file and object access)
- **compression**
- encryption

You can use the **mmobj policy create** command to create a storage policy with the desired functionality from the available options. After a storage policy is created, you can specify that storage policy while creating new containers to associate that storage policy with those containers.

A fileset is associated with the new storage policy. The name of the fileset can be provided optionally as an argument of the **mmobj policy create** command. For information on mapping of storage policy and fileset, see *Mapping of storage policies to filesets* in *IBM Spectrum Scale: Administration Guide*.

For information on creating, listing, and changing storage policies, see *Administering storage policies for object storage* and *mmobj command* in *IBM Spectrum Scale: Administration Guide* and *IBM Spectrum Scale: Command and Programming Reference*.

Unified file and object access overview

Unified file and object access allows use cases where you can access data using object as well as file interfaces.

Some of the key unified file and object access use cases are as follows:

- Object stores are suitable for storing large amounts of data because they are highly scalable and they are an economical storage solution. To analyze large amounts of data, advanced analytics systems are used. However, porting the data from an object store to a distributed file system that the analytics system requires is complex and time intensive. For these scenarios, there is a need to access the object data using file interface so that analytics systems can use it.
- Accessing object using file interfaces and accessing file using object interfaces helps legacy applications designed for file to start integrating into the object world after data migration.
- It allows storage cloud data which is in form of objects to be accessed using files from applications designed to process files.
- It allows files exported using NFS or SMB, or files available on POSIX, to be accessible as objects using http to the end clients. This enables easy availability of file data on mobile devices such as smart phones or tablets which are more suited to REST based interfaces.
- Multi protocol access for file and object in the same namespace allows supporting and hosting **data oceans** of different types with multiple access options.

  For information about **data oceans**, see “Protocols support overview: Integration of protocol access methods with GPFS” on page 27.
- There is a rich set of placement policies for files (using **mmapplypolicy**) available with IBM Spectrum Scale. With unified file and object access, those placement policies can be leveraged for object data.

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

- Ingest data through the object interface and access this data from the file interface
- Ingest data through the file interface and access this data from the object interface
- Ingest and access same data though object and file interfaces concurrently
- Manage authentication and authorization in unified file and object access

For more information, see *Unified file and object access in IBM Spectrum Scale* in *IBM Spectrum Scale: Administration Guide*.

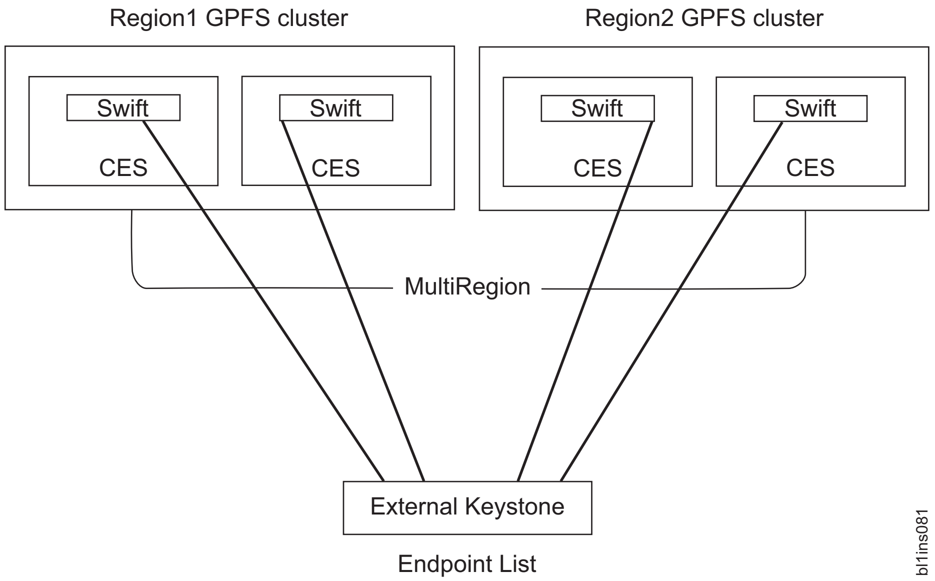
One of the key advantages of unified file and object access is the placement and naming of objects when stored on the file system. For more information, see *File path in unified file and object access* in *IBM Spectrum Scale: Administration Guide*.

Overview of multi-region object deployment

Object multi-region deployment can be used to distribute the object load over several sites to reduce latency for local requests. It can also be used to provide an active-active disaster recovery configuration.

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

With multi-region support in Swift, the replication servers asynchronously copy data between remote clusters to keep them in sync. When the client uploads or downloads objects, the proxy server it connects to initially uses the devices in the same region as the proxy. If the proxy cannot find the data it needs in its own region, it then uses the other regions. In this way, the clients generally get fast access for the data in the close cluster and only be affected by the high-latency penalty when accessing data which has not yet been replicated.



Multi-region object deployment involves creating up to 3 independent CES clusters. Each CES cluster is a single region. The Swift environment and ring files are configured to map each cluster to an associated region index. The configuration is then synced manually between the clusters to enable the active-active replication.

Enabling multi-region support converts the underlying primary Swift account, container, and object rings to include all defined regions. By default, all data is stored in all regions. To store data in just a subset of the regions, storage policies can be used to create object rings which just store the objects in a subset of the regions. For more information on creating custom storage policies, see *mmobj command* in *IBM Spectrum Scale: Command and Programming Reference*.

Only the first cluster can switch to multi-cluster after installation. Subsequent clusters need to be installed as multi-cluster environments due to the need for region numbers and storage policy indices to be globally consistent across clusters.

For information on planning a multi-region object deployment, see “Planning for multi-region object deployment” on page 206.

For information on enabling multi-region object deployment, see “Enabling multi-region object deployment initially” on page 315.

For information on adding a new region to a multi-region object deployment environment, see *Adding a region in a multi-region object deployment* in *IBM Spectrum Scale: Administration Guide*.

S3 API

IBM Spectrum Scale object storage supports the S3 API, in addition to Swift API, for accessing object data.

IBM Spectrum Scale uses Swift3 Middleware for OpenStack Swift, allowing access to IBM Spectrum Scale using the Amazon Simple Storage Service (S3) API. IBM Spectrum Scale for object storage includes S3 API as an optional feature.

S3 API can be either enabled during protocol deployment, initial object configuration, or later on.

- For information on enabling S3 API during protocol deployment using the `-s3` option of the **`spectrumscale config object`** command, see “Deploying protocols” on page 279.
- For information on enabling S3 API during initial object configuration using the `--enable-s3` option of the **`mmobj swift base`** command, see *Configuring and enabling the Object protocol service* in *IBM Spectrum Scale: Administration Guide*.
- For information on enabling S3 API if it is not enabled as part of the object base configuration, see *Changing the object base configuration to enable S3 API* in *IBM Spectrum Scale: Administration Guide*.

Accessing the object storage through swift requests is not affected by enabling the S3 API. When the S3 API is enabled, the object service also recognizes S3 API requests sent to the TCP port used by the object service (8080).

For more information on S3 API, see the S3 API documentation at Amazon S3 REST API Introduction.

For limitations of the S3 API support with IBM Spectrum Scale, see *Managing OpenStack access control lists using S3 API* in *IBM Spectrum Scale: Administration Guide*.

Object capabilities

Object capabilities describe the object protocol features that are configured in the IBM Spectrum Scale cluster.

The following capabilities are supported:

- file-access (Unified file and object access)
- multi-region (Multi-region object deployment)
- S3 (Amazon S3 API)

If unified file and object access has already been configured, changing the file-access capability can be used to enable or disable the related services. Use the command mmobj file-access to change it.

To change multi-region and s3 capabilities, use the **`mmobj multiregion`** and the **`mmobj s3`** commands respectively.

You can create a storage policy with unified file and object access only when file-access (unified file and object access) capability is enabled.

The **ibmobjectizer** and the **openstack-swift-object-sof** services are started only if the **file-access** capability is enabled. Disabling the **file-access** capability stops these services.

For information about enabling, disabling, and listing object capabilities, see *Managing object capabilities* in *IBM Spectrum Scale: Administration Guide*.

Secure communication between the proxy server and other backend servers

Use this feature to establish secure communication between the proxy server and the backend object storage servers.

By default object-server, object-server-sof, container-server, and account-server do not have authentication for the requests that they are serving. Processes, including the proxy-server connecting to these servers over their listening ports, can send requests which can result into updating the database and altering the object data on disk. Additional security between these servers can be enabled. Requesting process signs a request with a secret key kept in swift.conf. This key is verified by the serving object, container, or account server. To enable this feature, set:

```
mmobj config change --ccrfile swift.conf --section node_communication --property secure --value true
```

The signing middleware is added to proxy-server and the validating middleware is added to object-server, object-server-sof, container-server, and account-server. If the secret key is not present in swift.conf, it is randomly chosen and set to key secure_communication_secret under node_communication section. In a multi-region environment, this key must be reset and kept common in all the clusters.

To revert to the original configuration, set:

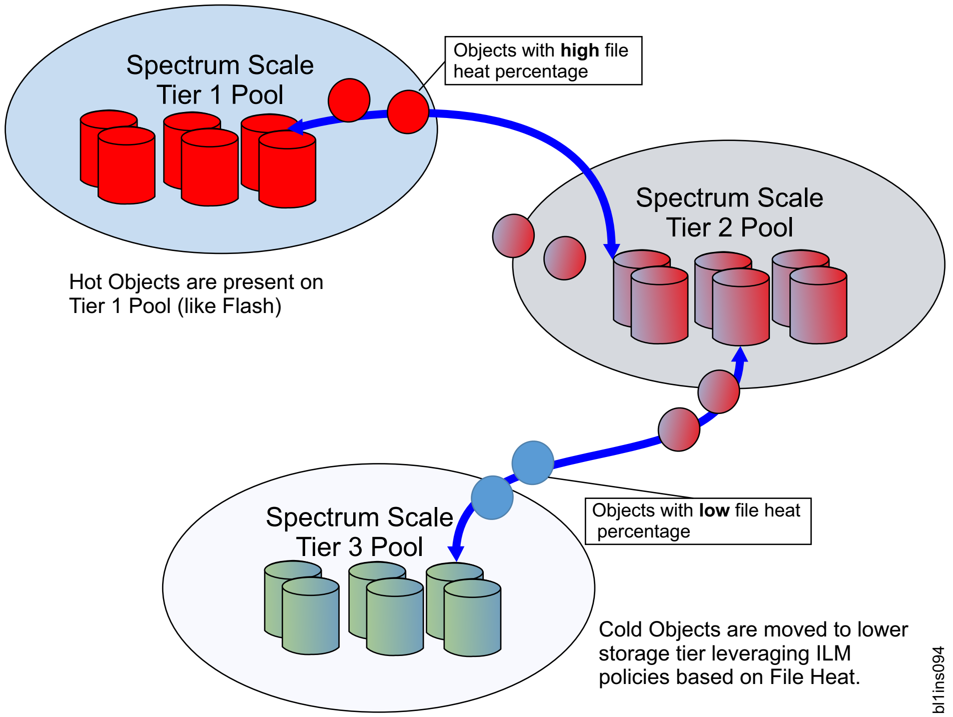
```
mmobj config change --ccrfile swift.conf --section node_communication --property secure --value false
```

Note: Disable SSH access on the protocol nodes on the IBM Spectrum Scale cluster for the users having the same UID and GID as the local swift user.

Object heatmap data tiering

Object heatmap data tiering policies can be applied to data that is frequently accessed.

With the object heatmap data tiering policies, frequently accessed objects are stored in higher-performing storage pools such as the SSD-based pool. The policy also stores objects that are not accessed frequently in slower disk-based storage pools. Data that is not required for regular access anymore is moved from slower disk-based storage pools to tapes.


Spectrum Scale
Tier 1 Pool
Objects with **high** file heat percentage
Hot Objects are present on Tier 1 Pool (like Flash)
Spectrum Scale
Tier 2 Pool
Spectrum Scale
Tier 3 Pool
Objects with **low** file heat percentage
Cold Objects are moved to lower storage tier leveraging ILM policies based on File Heat.
bl1ins094

The object heatmap policy

The object heatmap policy can be applied to object data files in the filesystem.

The policy manages the placement of objects in the gold, silver, and system pools and specifies the maximum threshold so that the capacity in the storage tiers is not over-utilized.

Enabling heatmap tracking

Enable heatmap tracking on the IBM Spectrum Scale file system by running the following command:

```
mmchconfig fileheatperiodminutes=1440,fileheatlosspercent=10
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

Cluster Export Services overview

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. The CES BLOCK service provides iSCSI protocol support.

IBM Spectrum Scale includes Cluster Export Services (CES) infrastructure to support the integration of the NFS, SMB, and object servers.

- The NFS Server supports NFS v3 and the mandatory features in NFS v4.0.
- The SMB server support SMB 2, SMB 2.1, and the mandatory features of SMB 3.0.
- The object server supports the Mitaka release of OpenStack Swift along with Keystone v3.

The CES infrastructure is responsible for:

- Managing the setup for high-availability clustering that is used by the protocols.
- Monitoring the health of these protocols on the protocol nodes and raising events or alerts during failures.

- Managing the addresses that are used for accessing these protocols by including failover and failback of these addresses because of protocol node failures.

High availability

With GPFS, you can configure a subset of nodes in the cluster to provide a highly available solution for exporting GPFS file systems by using the Network File System (NFS), Server Message Block (SMB), Object, and iSCSI protocols. The participating nodes are designated as Cluster Export Services (CES) nodes or protocol nodes. The set of CES nodes is frequently referred to as the *CES cluster*.

A set of IP addresses, the *CES address pool*, is defined and distributed among the CES nodes. As nodes enter and leave the GPFS cluster, the addresses in the pool can be redistributed among the CES nodes to provide high availability. GPFS nodes that are not CES nodes leaving or entering the GPFS cluster do not redistribute the IP addresses. Clients that use these protocols can use these addresses to access the cluster. However, applications using CES IP addresses are not impacted, .

Monitoring

CES monitors the state of the protocol services. Monitoring ensures that the CES addresses are assigned to the appropriate node and that the processes that implement the services in the CES cluster are running correctly. Upon failure detection, CES marks the node as temporarily unable to participate in the CES cluster and reassigns the IP addresses to another node.

Protocol support

CES supports the following export protocols: NFS, SMB, object, and iSCSI (block). Each protocol can be enabled or disabled in the cluster. If a protocol is enabled in the CES cluster, all CES nodes serve that protocol.

The following are examples of enabling and disabling protocol services by using the **mmces** command:

mmces service enable nfs
Enables the NFS protocol in the CES cluster.

mmces service disable obj
Disables the Object protocol in the CES cluster.

Commands

To set or configure CES options, the following commands are used:

mmblock
Manages the BLOCK configuration operations.

mmces
Manages the CES address pool and other CES cluster configuration options.

mmnfs
Manages NFS exports and sets the NFS configuration attributes.

mmobj
Manages the Object configuration operations.

mmsmb
Manages SMB exports and sets the SMB configuration attributes.

mmuserauth
Configures the authentication methods that are used by the protocols.

For more information, see *mmblock command*, *mmces command*, *mmnfs command*, *mmobj command*, *mmsmb command*, and *mmuserauth command* in *IBM Spectrum Scale: Command and Programming Reference*.

Restrictions

For an up-to-date list of supported operating systems, specific distributions, and other dependencies for CES, refer to the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

A CES cluster can contain a maximum of 32 CES nodes and they must all be of the same architecture. If the SMB protocol is enabled, the CES cluster is limited to a total of 16 CES nodes.

Each CES node must have network adapters capable of supporting all IP addresses in the CES address pool. The primary address of these adapters must not be used as a CES address.

CES iSCSI support

The CES BLOCK service provides iSCSI protocol support.

You can export a file resident in a GPFS file system as a virtual iSCSI volume to an iSCSI initiator by using the iSCSI protocol. The iSCSI protocol in IBM Spectrum Scale is referred to as the CES BLOCK service. iSCSI support in IBM Spectrum Scale is intended for remotely booting nodes (non-performance-critical), and is not intended for high-bandwidth block device workloads. Before using the iSCSI protocol, confirm that this matches your use case, or contact scale@us.ibm.com with any question regarding the supported use cases.

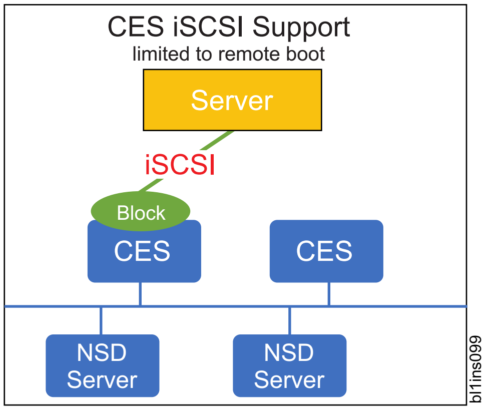


Figure 5. CES BLOCK iSCSI support

Clustered BLOCK service

iSCSI initiators can connect to any protocol node and access the virtual iSCSI volume. A clustered registry ensures that all nodes have the same configuration data. If a CES IP failover is triggered, an iSCSI initiator reconnects transparently to another CES node.

BLOCK service management command

The **mmblock** command provides a comprehensive entry point to manage iSCSI targets and volumes. The block service configuration is controlled by the SCST configuration files. The master version of these files is stored in the CCR, and copies exist in `/etc/scst.conf` on each protocol node. Do not modify the `/etc/scst.conf` file directly because it is overwritten by the information stored in the CCR.

Active File Management

The following topics introduce you to Active File Management (AFM)

Introduction to Active File Management (AFM)

Active File Management (AFM) enables sharing of data across clusters, even if the networks are unreliable or have high latency.

The following figure is a sample of an AFM relationship.

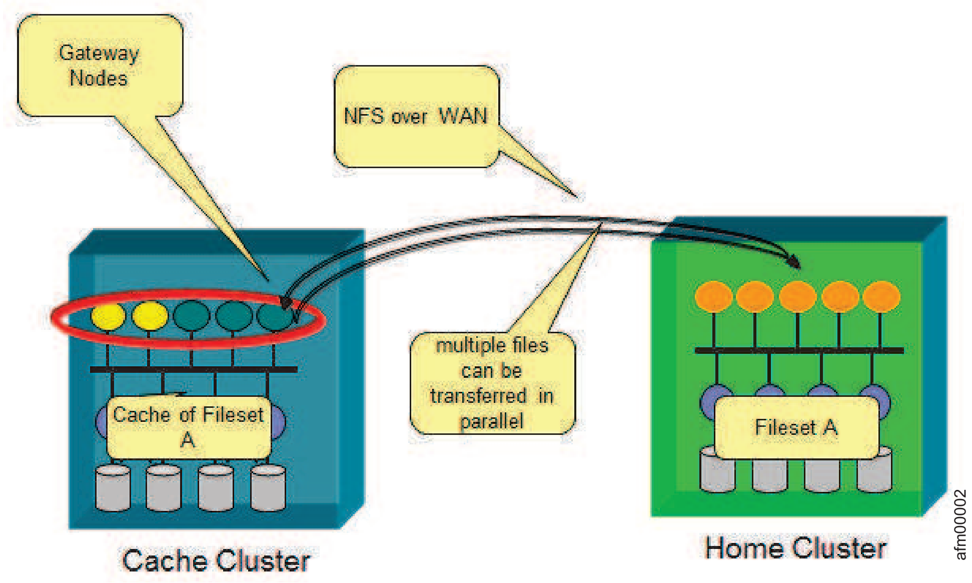


Figure 6. Sample of an AFM relationship

AFM allows you to create associations between IBM Spectrum Scale clusters or between IBM Spectrum Scale clusters and NFS data source. With AFM, you can implement a single name space view across sites around the world making your global name space truly global.

Using AFM, you can build a common name space across locations and automate the flow of file data. You can duplicate data for disaster recovery purposes without suffering from WAN latencies.

Individual files in the AFM filesets can be compressed. Compressing files saves disk space. For more information, see *File compression* in the *IBM Spectrum Scale: Administration Guide*.

Snapshot data migration is also supported. For more information, see *ILM for snapshots* in the *IBM Spectrum Scale: Administration Guide*.

Name space maintenance with AFM occurs asynchronously so that applications can continue operating without being constrained by network bandwidth.

Note: AFM does not offer any feature to check consistency of files across source and destination. However you can use any third-party utility to check consistency, after files are replicated.

Overview and concepts

The following topics provide an overview of AFM and describe the concepts.

Cache and Home

An AFM fileset is an independent fileset. Each fileset has a distinct set of AFM attributes. IBM Spectrum Scale cluster that contains AFM filesets is called a cache cluster. A cache fileset has a relationship with a data source which is called the home.

AFM constantly maintains an active relationship between the cache and the home. Changes are managed per fileset results in modular, scalable architecture capable of supporting billions of files and peta bytes of data. Each AFM-enabled fileset is associated with a single home path.

AFM uses a NFS or NSD protocol to communicate between home and cache. A Home is an NFS v3 export or a remote filesystem mounted from the IBM Spectrum Scale cluster.

Architecturally, AFM works with any NFS home export. However if the home has ACLs, it supports extended attributes and sparse files. You must run **mmafmconfig enable** command on the home cluster to enable this support.

Note: If you do not run **mmafmconfig enable** and configure AFM relationship, ACLs and extended attributes are not supported. Sparse file information is not maintained. The following error message `AFM: AFM is not enabled for fileset cachefs in filesystem gpfs. Some features will not be supported, see documentation for enabling AFM and unsupported features.` appears in `mmfs.log`.

Any cluster can be a home cluster, a cache cluster, or both. In typical setup, a home is in an IBM Spectrum Scale cluster and a cache is defined in another IBM Spectrum Scale cluster. Multiple AFM-enabled filesets can be defined in one cache cluster, each cache having a relationship with targets with a home, or different cluster.

In IW,RO, and LU modes, multiple caches might point to the same home. But in SW mode, only one-to-one relationship between cache and home is supported. AFM can also be configured as a subscription service, where home is the data feed and all caches can subscribe to this data feed.

Within a single cache cluster, application nodes experience POSIX semantics. File locking across cache and home is not supported.

While performing operations on AFM filesets, ensure that the operations are supported on home over the chosen protocol because the operations done from cache are replayed on the remote as normal filesystem operations. While using NSD protocol with UID remapping, operations such as `chown` (change ownership) are not supported.

Communication between cache and home

AFM uses the NFSv3 or NSD protocol for communication between clusters.

Communication for caching between clusters is performed by one or more nodes on the cache cluster designated as gateway nodes by using **mmchnode**. The target path on the home server must be NFS-exported on one or more nodes in the home cluster, or the home filesystem must be mounted on the cache cluster using the NSD protocol. The exported path or the home-mounted path of the home cluster is used in the cache cluster as the target when creating the cache fileset.

AFM maintains a relationship between cache and home by monitoring home availability and reconnects accordingly.

The backend protocol - NFS versus NSD

The NSD protocol is a stateful protocol. The NFSv3 protocol is a stateless protocol which is very resilient to low bandwidth and lossy networks.

The current recommended transport protocol for AFM data transfers is NFS, due to the tolerance of NFS to unstable network connections. It is recommended that you first try using NFS, and shift to the NSD protocol only if NFS does not meet the desired performance requirements even with multiple primary gateways and use of parallel data transfers. The implications of using the NSD protocol between the cache and home cluster are:

1. Network availability fluctuations and instability can affect the NSD protocol connection to the home on the cache cluster primary gateways. This can lead to frequent data access interruptions from the home cluster, and can even cause the connection to the home cluster to stop responding. In these cases, it might be necessary to restart the GPFS daemon on the primary gateway, and possibly even restart the primary gateway server.
2. IBM Spectrum Scale instability issues on the home cluster can affect the cache cluster. The instability issues can also cause the AFM fileset in the cache cluster to stop responding, and might also require a restart of the IBM Spectrum Scale service on both the home and cache clusters.

For more information on setting up primary gateway nodes that communicate with multiple NFS servers at home, see "Parallel data transfers" on page 51.

The following table summarizes the differences between NSD and NFS protocols on various parameters –

Table 4. Comparison between NSD and NFS protocols

	NSD	**NFS**
Ease of use	Customers are familiar with its use in multi-cluster environments. Configuration does not require NFS knowledge or tuning, but requires NSD tuning.	Configuration requires NFS knowledge and performance tuning for both NFS and TCP over WAN.
Performance	By default, uses all primary gateway nodes for parallel data transfers. Large file data transfer performance is better than NFS from a single primary gateway node as it can use the inherent parallelism of striping to multiple NSDs.	Parallel data transfers can be achieved by creating mapping between primary gateway nodes and NFS servers at home. In summary, while both NFS and NSD can do similar forms of parallelism, generally NSD achieves higher performance.
Security	Encryption is built in, which can be turned on optionally.	-
Firewall	Special ports might not be required.	Must be configured for the traffic to pass through.
Stability	Performs well if the network is stable and has low latency.	More resilient to failures within the network such as packet drops which readily happen over WAN and it is more resilient, protecting the cache cluster from being affected by home cluster issues.

Considerations when using the NSD protocol for AFM data transfers

The NSD protocol is more sensitive to packet drops and network latency than NFS. If the network does not respond, or if the packets are dropped, the NSD mount on cache cluster stops responding, causing the cache cluster also to stop responding. More causes of issues when using the NSD protocol are:

1. Deadlock in the home cluster - This might cause the NSD mounts on the cache cluster to stop responding for some time. Due to a non-responsive NSD mount, AFM fileset at cache using these NSD mounts as target might be in the 'unmounted' state. After the home cluster is responsive, the home cluster tries queued operations again.
2. Cluster reconfiguration or higher resource consumption at the home cluster - This might cause a temporary loss of communication between home and cache cluster. If the home cluster does not

respond within AFM wait timeout intervals, AFM filesets at cache using these NSD mounts as target might be in the 'unmounted' state. After the home cluster is responsive, the home cluster tries queued operations again.

3. When a new primary gateway node joins the cluster, the old primary gateway node transfers the fileset to new primary gateway node. If the remote filesystem is not mounted on the new primary gateway node, the fileset remains in an 'unmounted' state. After the remote file system is mounted at gateway node, the fileset automatically moves to Active state.
4. Remote File System cannot be unmounted unless replication is stopped, or primary gateway node is restarted. AFM puts a hold on remote mount, not allowing the file system to be unmounted.
5. Creating an AFM association, using GPFS protocol, to the same local file system is not supported.

Note: If the NSD mount on the gateway node is unresponsive, AFM does not synchronize data with home or secondary. The filesystem might be unmounted at the gateway node. A message `AFM: Remote filesystem` *`remotefs`* `is panicked due to unresponsive messages on fileset <`*`fileset_name`*`>,re-mount the filesystem after it becomes responsive. mmcommon preunmount invoked. File system: fs1 Reason: SGPanic` is written to `mmfs.log`. After the home or secondary is responsive, you must restore the NSD mount on the gateway node.

Primary gateway

Each cache fileset in a cluster is served by one of the nodes designated as a gateway in the cluster. The gateway node that is mapped to a fileset is called the primary gateway of the fileset. The primary gateway acts as the owner of the fileset and communicates with the home cluster.

All other nodes in the cluster, including other gateways, become the application nodes of the fileset. Therefore any node in the cache cluster can function as a gateway node and an application node for different filesets based on configuration of the node.

Application nodes communicate with the primary gateway for a fileset via internal network requests. The gateway function is highly available and can be scaled-out. When a gateway node fails, all cache filesets owned by this gateway node are moved to another gateway node that runs AFM recovery and takes over as the primary gateway of the filesets. When the old primary gateway returns after node failure, the original primary gateway resumes managing its filesets and AFM transfers the queues for all the filesets from the current primary gateway to the old primary gateway.

The primary gateway can be configured to take help from other gateway nodes for parallel data movement of large files to and from home, if the home is an IBM Spectrum Scale cluster for increasing performance during data transfer.

See "Parallel data transfers" on page 51 to setup gateway nodes talking to multiple NFS servers at home.

In a cluster with multiple gateway nodes and AFM cache filesets, AFM uses a hashing algorithm to elect the primary gateway for each of the filesets , **afmHashVersion=2** is set by default. An improved hashing algorithm is set by default on an IBM Spectrum Scale 4.1 cluster. On an upgraded cluster, the old hashing algorithm (**afmHashVersion=2**) is effective by default.

To use the improved algorithm after an upgrade to 5.0, use **mmchconfig afmHashVersion=4** to set **afmHashVersion** value as 4. You need to restart IBM Spectrum Scale for the new algorithm to take effect.

Hashing version '2' is based on static fileset ID mapping, whereas new hashing version '4' is improvement over version '2'. This hashing version is dynamic, ensuring balanced mapping of AFM filesets and gateway nodes.

The gateway node designation is supported only on the Linux operating system. The Windows or AIX nodes cannot be designated as gateway. It is recommended not to assign the role of Gateway and CES/Protocol to the same node.

Recommended configuration of a primary gateway node: Generally, a primary gateway node can have 128 GB of memory and each primary gateway node can have 10 filesets. You can use the **mmchconfig** command to set the parameters as follows - **Pagepool=2G**, **afmHardMemThreshold=40G**, **afmNumFlushThreads=8** , **afmDIO=2**, **maxFilestoCache=10000**. For more information on **pagepool**, see *Parameters for performance tuning and optimization* in *IBM Spectrum Scale: Administration Guide*.

Note: The values stated above are recommendations. For your use case, these values might be different.

Global namespace

You can combine the home and cache entities to create a global namespace.

Any client node can use the same path to connect to the data within any of the IBM Spectrum Scale clusters that are part of the namespace.

In such a global namespace, the following AFM features can improve application performance at a remote site:

1. When a file is being read into a cache, the data can be read after it begins to arrive in the cache.
2. Multiple cache filesets can share a single file system. Data can be transferred between sites in parallel.

AFM also performs on networks that are unreliable, or have high latency. The following example is that of a global namespace, implemented using AFM, with three different sites. An IBM Spectrum Scale client node from any site sees all of the data from all of the sites. Each site has a single file system. Each site is the home for two of the subdirectories and cache filesets pointing to the data originating at the other sites. Every node in all three clusters has direct access to the global namespace.

The following figure illustrates global namespace implemented using AFM.

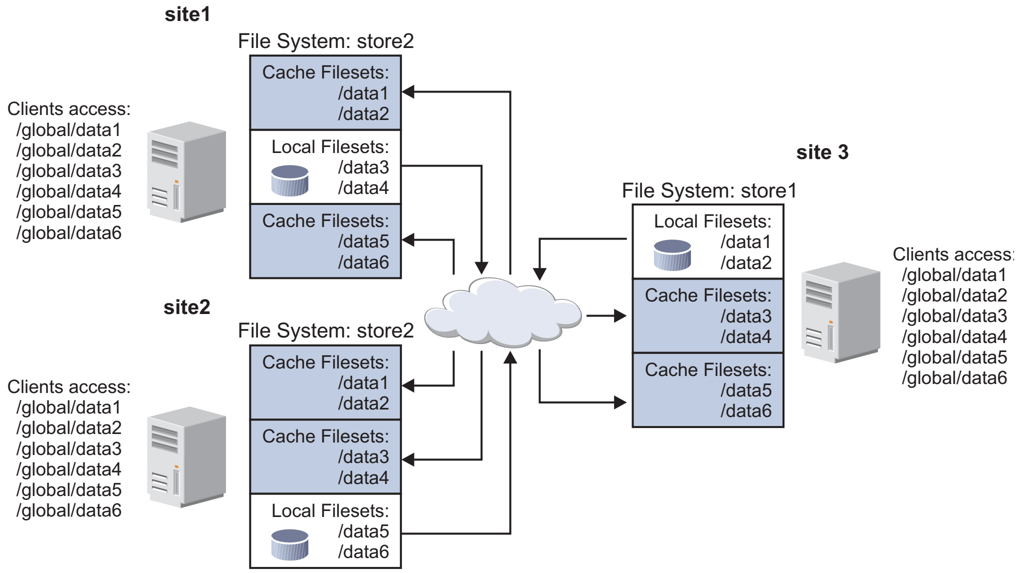


Figure 7. Global namespace implemented using AFM

Caching modes

There are four caching modes in AFM: Read-Only (RO), Single-Writer (SW), Local-Update (LU) and Independent-Writer (IW).

In a Read-Only mode, files in the cache fileset cannot be modified.

In a Single-Writer mode, cache pushes all changes to home but never checks home for any updates.

Independent-Writer mode allows both reads and writes, it pushes changes to home and checks home for file updates.

Local-Update is read only but files can be changed in the cache fileset, though the changes are never pushed to home and any changed file no longer checks home for updated versions.

Revalidation

As users traverse the directory tree of an AFM cache fileset, files and directory metadata information from the home cluster is checked and updated as needed on the cache cluster. This process is called AFM revalidation.

Revalidation performance is dependent upon network latency and bandwidth available between the cache and home. Revalidations are done per node, and not per fileset. If a file or directory is revalidated from one node on the cache cluster, the same fileset goes through another revalidation operation when accessed from another node on the same cache cluster. You can modify the refresh intervals using the **mmchfileset** command as shown:

```
# mmchfileset fs1 sw1 -p afmFileOpenRefreshInterval=160
```

In the above example, the **File Open Refresh Interval** is set to 160 for the filesets in file system *fs1*.

Revalidation intervals can be adjusted to best support the workload and network latency. Setting a parameter using **mmchconfig** command sets the default for all filesets. Parameters set using **mmchfileset** only affect a particular fileset and override the global defaults. You can enable, modify, or disable any of the intervals based on the application needs, though it is recommended to use default values for most cases.

If file or directory refresh intervals are disabled for a fileset, the refresh intervals can be enabled using the **mmchfileset** command. Enabling requires the fileset to be unlinked, and then linked again.

For more information on **mmchfileset**, see the *mmchfilest command* in the *IBM Spectrum Scale: Command and Programming Reference*.

It is recommended not to set the revalidation intervals to 0, as a revalidation request is continuously sent to home, thus resulting in performance degradation. You must set the revalidation interval to as large as possible, depending on how frequently home gets updated, and at what interval the cache needs the updated data.

For more information on revalidation intervals, see the *mmcrfileset command* in the *IBM Spectrum Scale: Command and Programming Reference*.

The revalidation intervals are defined by the following configuration parameters. These are tunable at cluster and fileset level and can be changed using the **mmchconfig** and **mmchfileset** commands respectively:

1. **afmFileLookupRefreshInterval**: The frequency of revalidation triggered by a lookup operation on a file such as ls or stat, from the cache
2. **afmDirLookupRefreshInterval**: The frequency of revalidation triggered by a lookup operation on a directory such as ls or stat, in the cache.
3. **afmFileOpenRefreshInterval**: The frequency of revalidations triggered by read or write operations on a file in the cache. Open requests on that file are served from the cache fileset until the **afmFileOpenRefreshInterval** expires, after which the open requests are sent to home.
4. **afmDirOpenRefreshInterval**: The frequency of revalidations triggered by read or update operations on a directory from the cache. 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 home.

5. **afmRevalOpWaitTimeout**: The time for which AFM waits for completion of revalidation to get response from the home cluster. **afmRevalOpWaitTimeout** can be set only for a cluster and not for an individual fileset.

RO, LU and IW filesets revalidate regularly with home. SW mode populates metadata only once during first access and does not revalidate with home thereafter. To revalidate, AFM sends a message to the home cluster to find out whether the metadata of that file or directory has been modified since the last time it was revalidated. If so, the latest file metadata or data information, depending on the type of request, at home is reflected on the cache.

In some cases one operation such as a lookup can trigger another operation such as readdir asynchronously based on the application. Since such asynchronous operations can get internally triggered by AFM, the behavior of revalidation intervals might show slight variation than the exact set values. For example, a read on a cached file checks with home that the file has not changed if the read request comes after the **afmFileLookupRefreshInterval** has elapsed. So, this read includes a lookup operation for that file.

Cached and uncached files

A readdir operation on a directory populates the metadata of the directory in the cache, but it does not populate contents of each file within the directory. A read operation on file generates a request to home to make contents available in cache. The file contents do not need to be in cache to start reading it.

AFM allows data to be pre-populated before actual read operation using the **mmafmctl prefetch** command. For more information on pre-populating data, see “Prefetch” on page 53.

A file whose contents are completely available in the cache is called a cached file. A file whose contents are not yet present in the cache is called an uncached file. An uncached file cannot be evicted, re-synched with home, or failed over to a new home. See the sections on Asynchronous operations and delay, for AFM eviction and syncing to home.

Files that have all blocks read, or the entire file contents fetched, are marked as cached. AFM does whole-file caching by default. By default, reading more than three blocks of a file drives AFM to cache the full file in the background for performance. Sometimes the whole file might not need to be cached. For example, some applications read only a few bytes of a file to detect the file mime type. In such cases, you can configure partial file read behavior on cache fileset. For more information on partial file caching, see “Partial file caching” on page 53.

Synchronous or asynchronous operations

AFM operations are serviced either synchronously or asynchronously. Reads and revalidations are synchronous operations and update operations from the cache are asynchronous.

Synchronous operations require an application request to be blocked until the operation completes at the home cluster. File data is available while AFM queues are processed asynchronously in the background.

If synchronous operations depend on the results of one or more updates or asynchronous operations, AFM prioritizes the dependent asynchronous operations prior to the execution of the synchronous operations. For example, synchronous operations like file lookup cause dependent asynchronous operations to be flushed, overriding asynchronous delay (**afmAsyncDelay**).

AFM deploys a filtering algorithm for most optimal flushing performance by analyzing the queued requests. For example, if a write and delete are in queue on the same file, the write request is dropped from the queue and the delete is executed at home. Similarly, when **mkdir** and **rmdir** of the same directory name are in queue, both requests are dropped.

AFM sends data from cache to home as root. If home has fileset quotas at home, it should be set such that it can accommodate all writes from cache.

Asynchronous delay

All update operations from the writable cache filesets are on the primary gateway. Queues in the primary gateway are pushed to home asynchronously based on the **afmAsyncDelay** interval.

This interval can be modified using the **mmchfileset** command. You can force a flush of all the pending updates without waiting for Async Delay using the **mmafmctl** command with the `flushPending` option. The `flushPending` option can be issued at the fileset or filesystem level, or with a file that contains the list of files to be flushed.

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

Operations with AFM modes

Following are the operations with AFM modes - Read only (RO), Single writer (SW), Local updates (LU), and Independent writer (IW).

1. Read only (RO)

   The following figure illustrates the Read Only mode.

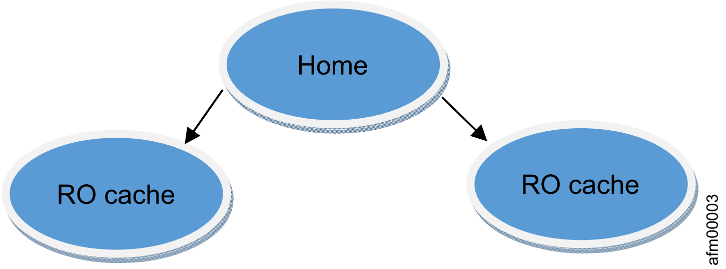


Figure 8. Read Only mode

   In this mode, data in the cache is read-only. Creating or modifying files in the cache fileset is not allowed. If a file is renamed at the home for an RO fileset, the file is recreated in cache and is assigned a new inode number in the cache. If the file is in use by an application while it is recreated (deleted and recreated with the same name) at home, it gets recreated in cache.

2. Single writer (SW)

   The following figure illustrates the Single writer mode.

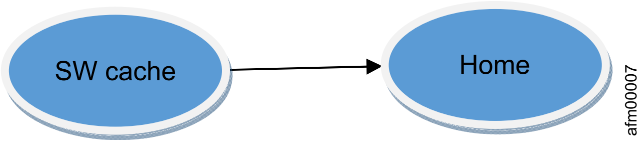


Figure 9. Single writer mode

   In this mode, only the cache fileset does all the writing and the cache does not check home for file or directory updates. The administrator needs to guarantee no writes are done in the home cluster. AFM does not enforce this check.

   A SW home can have some pre-existing data. An SW cache can cache and update this data. Update of an uncached file from SW home caches the file. However, the truncate or append operations on an uncached file does not fetch the contents of the file into the cache, but queues the truncate or append operation to the home.

3. Local updates (LU)

Local update is similar to read-only mode although you can create and modify files in the cache fileset. Updates in the cache are considered local to the cache and get decoupled from the corresponding file at home. Local updates are never pushed back to home. Once a file is modified, it is no longer compared to the version at home during revalidation to verify that it is up to date. Changes of this file on home do not have an impact on the cached copy of the file and vice versa.

Behaviors with local files: In AFM, LU mode files have one of the following states:

- Uncached: Files on the home for which metadata but no data has been copied into the cache are shown in the cache as uncached. At this point, the file is not resident on cache, but only on the home. Changes on the home are reflected in the cache.
- Cached: If an uncached file is read in the cache or pre-fetched, the file's state changes to Cached. In the Cached state, all changes to the file on home are reflected in the cache. The file is resident on the cache.
- Local: File data or metadata modified at cache becomes local to the cache. At this point the cached files relationship to the file in home is broken. Changes on home are not reflected in the cache anymore and file changes are not copied to home.

Operations in the cache that trigger the transitions between these states are shown below:

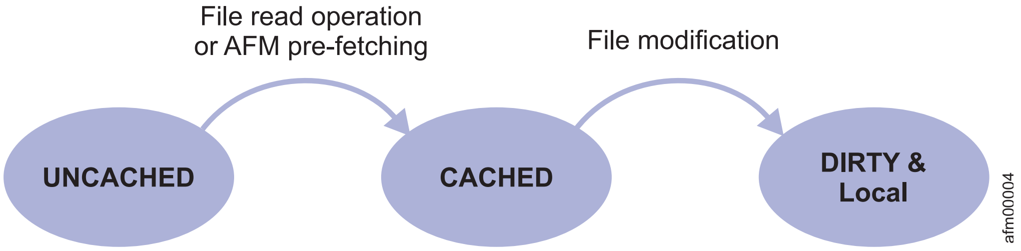


Figure 10. Behaviors with local files

The following tables summarize AFM LU mode behavior with local files:

File operation at home	Uncached file	Cached file	Local file
New Create	Reflects	Reflects	Reflects
Change data or attributes/Rename/Delete	Reflects	Reflects	Does not reflect

File operation at home	Uncached file	Cached file	Local file
File lookup revalidation	Pulls the metadata	Pulls the latest metadata	Does not pull the latest metadata
Read	Pulls the data	Pulls the latest data	Shows data in the local file – does not pull the latest data from the home
Prefetch	Prefetches the data	Prefetches the latest data	Does not prefetch the latest data from home
Change data	Pulls the data into cache and updates the changes	Pulls the latest to cache and updates the changes	Updates the local copy of the file and does not pull the latest data from the home before the update
Change attributes/metadata except delete EA	Updates the file and marks file as local	Updates the file and marks file as local	Updates the file – does not pull the latest metadata from the home before the update

Directories in a local-update cache:

Directories can become local in the LU mode with the following directory operations from the cache:

- Deleting files or sub-directories in the cache directory
- Creating new files or sub-directories in the cache directory
- Renaming files or sub-directories in the cache directory

The following file operations on cache do not cause directories on the cache to become local:

- update the uncached or cached file
- add attributes to the file
- migrate the cached or local files to tape
- recall of the cached or local files from tape

A local directory in a LU cache can contain local, cached and uncached files and directories. A local directory is not revalidated with home. Therefore, operations on the directory at home do not show up on the cache.

The following tables summarize AFM LU mode behavior with local directories:

Operation at home	**Cache behavior - Normal directory**	**Cache behavior - Local directory**
Create a new file or sub-directory	Reflects in cache after next directory lookup	Does not show in cache after next directory lookup. Contents can be read or prefetched if the filename is known in the cache. After read or prefetch, the new file or directory shows up in the cache for future file lookup
Update data or attributes of an existing file or sub-directory	Reflects in cache after next file lookup	Reflects in cache after next file lookup
Rename an existing file or sub-directory	Reflects in cache after next file lookup	Does not show in cache after next directory lookup. On reading the file, contents from renamed file fetched into the cache into the old filename, old filename always prevails in the cache. New filename can arrive at the cache with prefetch, in which case both names will co-exist and point to the same inode at home.
Delete an existing file or sub-directory	Reflects in the cache after next directory lookup	Deleted upon the next file lookup

Directory operation from cache	**Cache behavior - Normal directory**	**Cache behavior - Local directory**
Directory revalidation	Pulls the latest metadata	Does not pull the latest metadata
Read a non-local file in the directory	Pulls the latest data	Pulls the latest data
Prefetch a non-local file	Prefetches the latest data	Prefetches the latest data
Create a new file/directory	Updates and marks the directory local	Updates the local directory
Change data of a non-local file	Pulls the data into the cache, updates the file and marks file as local	Pulls the data into the cache, updates the file and marks the file as local
Change metadata of a non-local file except delete of EA	Updates the file and marks the file as local	Updates the file and marks the file as local

Appending to, truncating, or writing to an uncached file in LU mode fetches the entire file to the cache before making the change locally.

If you expect revalidation, change the LU fileset root directory with caution as this could cause the fileset to be marked as local, and context with the home is lost. For example, running **chmod** or **chown** of the LU fileset root directory causes all subdirectories from fileset root to be out of synchronization with the home.

4. Independent writer (IW)

   The following figure illustrates the Independent writer mode.

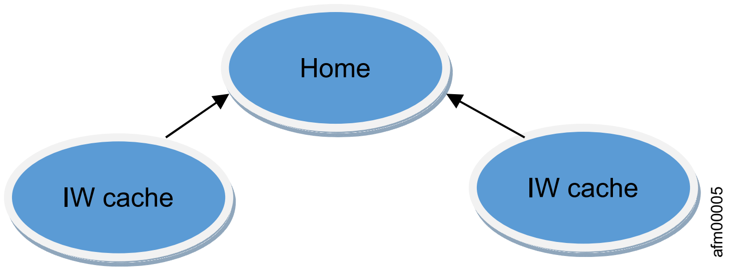


Figure 11. Independent writer mode

This mode allows multiple cache filesets to point to the same home. There is no synchronous locking between clusters while updating home. Each cache reads from home and makes updates to the home independently of each other, based on the revalidation intervals and Async delay.

This mode is used to access different files from each IW cache site, such as, the unique users at each site updating files in their home directory. While this mode allows multiple cache clusters to modify the same set of files, this should be only done by advanced users. This is because there is no locking or ordering between updates. Updates are propagated to the home in an asynchronous manner and can be delayed due to network disconnections. Therefore, conflicting updates from multiple cache sites can cause the data at the home site to be undetermined.

Writing in-place on a pre-existing un-cached file at the home pulls the complete contents of the file into cache. However, truncate and append operations on an uncached file do not fetch the contents of the file into cache, but queues the truncate and append operations to home.

When a large number of IW cache filesets point to the same NFS home, the number of NFS threads at home can be tuned for better efficiency. Increasing the revalidation intervals for the IW cache filesets might reduce the frequency of periodic refreshes from home and improve cache performance.

Note: Do not create hard links at home or in the cache when IW is used, as IW failback does not preserve hard links.

The following file attributes are maintained locally in cache : Control attributes, direct I/O, replication factors, fileset quotas, storage pool flags, special file types such as FIFO, socket, block, character device. Hard links can be created at cache. Hard links at home are displayed as individual files in cache.

Filesets to the same home target

The following are the possibilities of filesets pointing to home as the target:

- One or more RO filesets can point to the same target.
- One or more LU filesets can point to the same target.
- RO and LU filesets can point to the same target.
- SW, RO/LU filesets can point to the same target.
- IW, RO, and LU filesets can point to the same target.
- More than one SW fileset must not point to the same target. More than one SW fileset can technically point to the same target but only can be a writer. The other should behave like a reader fileset.
- One or more IW filesets can point to the same target.
- SW and IW filesets must not point to the same target.

Conversion of mode

An AFM cache can be converted to another mode in some conditions, to meet all types of requirements of data management by unlinking the fileset and using **mmchfileset -p afmmode** command.

Limitations while converting the mode are as follows:

- A SW or IW fileset with pending requests in the queue cannot be converted.
- LU mode filesets cannot be converted to any other mode. Before converting from IW to SW mode, ensure that all the remaining IWs are converted to either RO or LU to avoid the risk of home conflicts caused by writes from other IWs.
- You cannot change the mode or disable AFM when the fileset is linked.

Internal AFM Directories

AFM uses `.afm`, `.ptrash`, and `.pconflicts` as the internal directories.

AFM uses the following internal directories. Do not alter or remove any of these directories.

1. `.afm` directory: Present in both cache and home.

   The **mmafmconfig enable** command creates this directory in the home cluster. The **mmafmconfig disable** command removes the directory from home. The `.afm` directory is created in the cache during fileset create time, and is valid throughout the lifetime of the cache fileset.
2. `.ptrash` directory: Present in the cache.
3. `.pconflicts` directory: Present in the cache.

In an SW or IW mode fileset when there is a delete conflict with a file, the conflict is resolved by removing the file from the cache and moving it to `.ptrash`. A conflict occurs when a file is removed at home but still exists in cache because there are outstanding changes in the cached version. A file with outstanding changes in the cache not yet copied to home is called dirty. Files moved from `.ptrash` or `.pconflicts` (using the **mv** command) are treated as local files to the cache fileset and are not queued to home. However, if they are copied (for example, **cp** command) to the cache, they are queued to home as new files.

Active File Management (AFM) features

The following sections list features of Active File Management (AFM).

Force flushing contents before Async Delay

Requests in the queue of a writable cache (SW/IW) can be flushed to home before they get flushed automatically after Async Delay using the **mmafmctl flushPending** command. Use

```
mmafmctl Device flushPending [-j FilesetName [--list-file ListFile]] [-s LocalWorkDirectory]
```

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

Parallel data transfers

Parallel data transfer improves the AFM data transfer performance.

To help the primary gateway exchange large files with the home cluster, a cache cluster can be configured to leverage all the gateways. When using the NFS for AFM data transfers multiple NFS servers are required at the home cluster. All NFS servers on the home cluster must export the home path using the same parameters.

In a cache cluster, using NFS for AFM data transfer, each gateway node can be mapped to a specific NFS server at home. A map replaces the NFS server name in the **AFMTarget** parameter. Creating an export server map can be used to define more than one NFS server and map those NFS servers to specific AFM gateways. A map can be changed without modifying the **AFMTarget** parameter for a fileset, and needs fileset relink or filesystem remount to change. Use the *mmafmconfig* command in *IBM Spectrum Scale: Command and Programming Reference* to define, display, delete, and update mappings.

To define multiple NFS servers for an **AFMTarget** parameter and use parallel data transfers:
1. Define a mapping.
2. Use the mapping as the **AFMTarget** parameter for one or more filesets.
3. Update parallel read and write thresholds, in chunk size, as required.

The following example shows a mapping for NFS target, assuming four cache gateway nodes hs22n18, hs22n19, hs22n20, and hs22n21, mapped to two home NFS servers js22n01 and js22n02 (192.168.200.11 and 192.168.200.12) and then creating SW filesets using this mapping.

Define the mapping:

```
# mmafmconfig add mapping1 --export-map js22n01/hs22n18,js22n02/hs22n19

mmafmconfig: Command successfully completed
mmafmconfig: Propagating the cluster configuration data to all  affected nodes. This is an asynchronous process.

# mmafmconfig add mapping2 --export-map js22n02/hs22n20,js22n01/hs22n21

mmafmconfig: Command successfully completed
mmafmconfig: Propagating the cluster configuration data to all  affected nodes. This is an asynchronous process.

# mmafmconfig show

Map name:             mapping1
Export server map:    192.168.200.12/hs22n19.gpfs.net,192.168.200.11/hs22n18.gpfs.net

Map name:             mapping2
Export server map:    192.168.200.11/hs22n20.gpfs.net,192.168.200.12/hs22n21.gpfs.net

#Create filesets using these mappings:

mmcrfileset gpfs1 sw1 --inode-space new –p afmmode=sw,afmtarget=mapping1://gpfs/gpfs2/swhome
mmcrfileset gpfs1 ro1 --inode-space new –p afmmode=ro,afmtarget=mapping2://gpfs/gpfs2/swhome
```

All gateway nodes other than the primary gateway that is defined in a mapping are called participating gateway nodes. The primary gateway of a cache fileset communicates with each of the participating gateway nodes, depending on their availability. When parallel data transfer is configured, a single data transfer request is split into multiple chunks. Chunks are sent across to the participating gateway nodes in parallel for transfer to, or from home by using the respective NFS servers. Primary gateway processes the replies from all the participating gateway nodes, handles all data transfer failures, and coordinates activities until all data transfer is completed. If any participating gateway node fails, the primary gateway attempts to retry the failed task on the next available gateway and generates an error message in the IBM Spectrum Scale log.

Parallel reads and writes are effective on files with sizes larger than those specified by the parallel threshold. The threshold is defined by using **afmParallelWriteThreshold** and **afmParallelReadThreshold** parameters, and is true for all types of files except reads on sparse files and files with partial file caching enabled, which is served only by the Primary gateway without splitting.

Use the **afmParallelWriteChunkSize** and **afmParallelReadChunkSize** parameters to configure the size of each chunk.

Some more functions are as under -
1. On an AFM cache fileset that uses the native NSD protocol, all gateway nodes are used for parallel data transfer. A mapping using NSD protocol have nodes from cache cluster to represent the source

and destination servers (the Gateway node and NFS server). In the absence of a mapping definition in such a cache, all gateway nodes are used for parallel data transfer.

2. When using the NFS, if more than one gateway node is mapped to the same NFS server, only one performs a read task. However, a write task is split among all the gateway nodes.
3. One gateway node cannot be mapped to more than one NFS server.
4. Changes in the active mapping take effect after fileset re-link or file system remount.
5. If mapping is not specified or if a matching fails, data cannot be transferred by using parallel data transfer.
6. Gateway designation can be removed from a node only if the node is not participating in an active mapping.
7. If you set new values for **afmParallelReadChunkSize**, **afmParallelReadThreshold**, **afmParallelWriteChunkSize**, and **afmParallelWriteThreshold**; you need not relink filesets for the new values to take effect.

Note: If an AFM home is a mix of architectures (x86 and ppc), parallel data transfer works only for the set of nodes that belong to any one architecture, depending on which architecture serves the data transfer first.

Partial file caching

With partial file caching, the cache can fetch only the blocks that are read and not the entire file, thereby utilizing network and local disk space more efficiently. This is useful when an application does not need to read the whole file. Partial file caching is enabled on an IBM Spectrum Scale block boundary.

Partial file caching is controlled by the **afmPrefetchThreshold** parameter which can be updated using the **mmchfileset** command. The default value of this parameter is 0, which means complete file caching and all blocks of a file are fetched after any three blocks have been read by the cache and the file is marked as cached. This is useful for sequentially accessed files that are read in their entirety, such as image files, home directories, and development environments.

The valid **afmPrefetchThreshold** values are between 1 and 100. This specifies the file size percentage that must be cached before the rest of the data blocks are automatically fetched into the cache. A large value is suitable for a partially-accessed file.

An **afmPrefetchThreshold** value of 100 disables full file prefetching. This value caches only the data blocks that are read by the application. This is useful for large random-access files, that are either too big to fit in the cache or are never expected to be read in their entirety. When all data blocks are available in the cache, the file is marked as cached.

For sparse files, the percentage for prefetching is calculated as the ratio of the size of data blocks allocated in the cache and the total size of data blocks at home. Holes in the home file are not considered in the calculation.

Writes on partially cached files

If a write is queued on a file that is partially cached, then the file is completely cached first. Only then the write is queued on the file. Appending to a partially cached file does not cache the whole file. In the LU mode alone, the write inset or append on a partially-cached file caches the whole file even if the prefetch threshold is set on the fileset.

Note: As Partial file caching is not backward compatible, all nodes must be on GPFS 3.5.0.11 or later.

Prefetch

Prefetch fetches the file metadata (inode information) and data from home before an application requests the contents.

Prefetch is a feature that allows fetching the contents of a file into the cache before actual reads.

Prefetching files before an application starts can reduce the network delay when an application requests a file. Prefetch can be used to pro-actively manage WAN traffic patterns by moving files over the WAN during a period of low WAN usage.

Prefetch can be used in the following ways:

- Populate data
- Populate metadata
- View prefetch statistics

Use the following command to perform these activities.

```
mmafmctl Device prefetch -j FilesetName [--metadata-only]
                    [{--list-file ListFile} |
                    {--home-list-file HomeListFile} |
                    {--home-inode-file PolicyListFile}]
                    [--home-fs-path  HomeFileSystemPath]
               [-s LocalWorkDirectory]
```

For more information on the command, see *mmafmctl* command in *IBM Spectrum Scale: Command and Programming Reference*. If no options are given for prefetch, the statistics of the last prefetch command run on the fileset are displayed.

`--metadata-only` - Prefetches only the metadata and not the actual data. This is useful in migration scenarios. This option requires the list of files whose metadata is to be populated. It has to be combined with a list file option.

`--list-file` **`ListFile`** - The specified file is a file containing a list of files that need to be pre-populated, one file per line. All files must have fully qualified path names. If the list of files to be prefetched have filenames with special characters then a policy should be used to generate the listfile. This should be hand-edited to remove all other entries except the filenames. The list of files can be:

1. Files with fully qualified names from cache.
2. Files with fully qualified names from home
3. Fist of files from home generated using policy. The file must not be edited.

`--home-list-file` **`HomeListFile`** - The specified file is a file containing a list of files from home that need to be pre-populated, one file per line. All files must have fully qualified path names. If the list of files to be prefetched have filenames with special characters then a policy should be used to generate the listfile. A policy generated file should be hand-edited to remove all other entries except the filenames. As of version 4.2.1, this option is deprecated. The **`–list-file`** option can handle this.

`--home-inode-file` **`PolicyListFile`** - The specified file is a file containing the list of files from home that need to be pre-populated in the cache and this file is generated using policy. This should not be hand-edited. This option is deprecated. The –list-file option can handle this.

`--home-fs-path HomeFileSystemPath` - Specifies the full path to the fileset at the home cluster and can be used in conjunction with –list-file. You must use this option, when in the NSD protocol the mount point on the gateway nodes of the afmTarget filesets does not match the mount point on the Home cluster. For example, the home filesystem is mounted on the home cluster at `/gpfs/homefs1`. The home filesystem is mounted on the cache using NSD protocol at `/gpfs/remotefs1`.

For example, **`mmafmctl gpfs1 prefetch -j cache1 –list-file /tmp/list.allfiles --home-fs-path /gpfs/remotefs1`**.

Prefetch is an asynchronous process and the fileset can be used while prefetch is in progress. Prefetch completion can be monitored by using the **afmPrepopEnd** callback event or looking at **mmafmctl Device prefetch** command with no options.

Prefetch pulls the complete file contents from home (unless the **—metadata-only** flag is used), so the file is designated as cached when it is completely prefetched. Prefetch of partially cached files caches the complete file.

Prefetch can be run in parallel on multiple filesets, although only one prefetch job can run on a fileset.

If a file is in the process of getting prefetched, it is not evicted.

If parallel data transfer is configured, all gateways participate in the prefetch process.

If the filesystem unmounts during prefetch on the gateway, prefetch needs to be issued again.

Prefetch can be triggered on inactive filesets.

Directories are also prefetched to the cache if specified in the prefetch file. If you specify a directory in the prefetch file and if that directory is empty, the empty directory is prefetched to cache. If the directory contains files or sub-directories, you must specify the names of the files or sub-directories which you want to prefetch. If you do not specify names of individual files or sub-directories inside a directory, that directory is prefetched without its contents.

Prefetch Recovery: If the primary gateway of a cache is changed while prefetch is running, prefetch is stopped. The next access to the fileset automatically re-triggers the interrupted prefetch on the new primary gateway. The list file used when prefetch was initiated must exist in a path that is accessible to all gateway nodes. Prefetch recovery on a single-writer fileset is triggered by a read on some file in the fileset. Prefetch recovery on a read-only, independent-writer and local-update fileset is triggered by a lookup or readdir on the fileset. Prefetch recovery occurs on the new primary gateway and continues where it left off. It looks at which files did not complete prefetch and it rebuilds the prefetch queue. Examples of messages in the `mmfs.log` are as below:

```
Wed Oct  1 13:59:22.780 2014: [I] AFM: Prefetch recovery started for the file system gpfs1 fileset iw1.
mmafmctl: Performing prefetching of fileset: iw1
Wed Oct  1 13:59:23 EDT 2014: mmafmctl: [I] Performing prefetching of fileset: iw1
Wed Oct  1 14:00:59.986 2014: [I] AFM: Starting 'queue' operation for fileset 'iw1' in filesystem '/dev/gpfs1'.
Wed Oct  1 14:00:59.987 2014: [I] Command: tspcache /dev/gpfs1 1 iw1 0 257 42949 67295 0 0 1393371
Wed Oct  1 14:01:17.912 2014: [I] Command: successful tspcache /dev/gpfs1 1 iw1 0 257 4294967295 0 0 1393371
Wed Oct  1 14:01:17.946 2014: [I] AFM: Prefetch recovery completed for the filesystem gpfs1 fileset iw1. error 0
```

1. Metadata population using prefetch:

   ```
   # mmafmctl fs1 getstate -j ro
   Fileset Name Fileset Target           Cache State Gateway Node Queue Length Queue  numExec
   ------------ --------------                    ----------- ------------ ------------    -------------
   ro     nfs://c26c3apv1/gpfs/homefs1/dir3 Active  c26c2apv2    0           7
   List Policy:
   RULE EXTERNAL LIST 'List' RULE 'List' LIST 'List' WHERE PATH_NAME LIKE'%'
   Run the policy at home:mmapplypolicy /gpfs/homefs1/dir3 -P px -f px.res -L 1 -N mount -I defer
   Policy creates a file which should be manually edited to retain only the file names.  Thereafter
   this file is used at the cache to populate metadata.

   # mmafmctl fs1 prefetch -j ro --metadata-only --list-file=px.res.list.List
   mmafmctl: Performing prefetching of fileset: ro

   Prefetch end can be monitored by using this event:
   Thu May 21 06:49:34.748 2015: [I] Calling User Exit Script prepop: event afmPrepopEnd,
   Async command prepop.sh.

   The statistics of the last prefetch command can be viewed by running the following command:
   ```

```
mmafmctl fs1 prefetch -j ro
Fileset Name Async Read (Pending) Async Read (Failed) Async Read (Already Cached)
Async Read (Total) Async Read (Data in Bytes)
------------ -------------------- ------------------  --------------------------  ---------------------
       ro          0                    1                  0                        7                0
```

2. Prefetch of data by giving list of files from home: # **cat /listfile1**

```
/gpfs/homefs1/dir3/file1
/gpfs/homefs1/dir3/dir1/file1
```

# **mmafmctl fs1 prefetch -j ro --list-file=/listfile1**

```
mmafmctl: Performing prefetching of fileset: ro
```

# **mmafmctl fs1 prefetch -j ro**

```
Fileset Name Async Read (Pending) Async Read (Failed) Async Read (Already Cached) Async
Read (Total) Async Read (Data in Bytes)
------------ -------------------- ------------------   -------------------------- ------------------

ro           0                    0                   0                            2           122880
```

3. Prefetch of data using list file that is generated using policy at home:

Inode file is created using the above policy at home, and must be used as such without hand-editing.

```
List Policy:
RULE EXTERNAL LIST 'List' RULE 'List' LIST 'List' WHERE PATH_NAME LIKE '%'
```

For files with special characters, path names must be encoded with ESCAPE %.

```
RULE EXTERNAL LIST 'List' ESCAPE '%' RULE 'List' LIST 'List' WHERE PATH_NAME LIKE '%'
```

Run the policy at home:# **mmapplypolicy /gpfs/homefs1/dir3 -P px -f px.res -L 1 -N mount -I defer**

# **cat /lfile2**

```
113289 65538 0 -- /gpfs/homefs1/dir3/file2
113292 65538 0 -- /gpfs/homefs1/dir3/dir1/file2
```

# **mmafmctl fs1 prefetch -j ro –list-file=/lfile2**

```
mmafmctl: Performing prefetching of fileset: ro
mmafmctl fs1 prefetch -j ro
Fileset Name Async Read (Pending) Async Read (Failed) Async Read (Already Cached) Async
Read (Total) Async Read (Data in Bytes)
-------------------- ------------------    ------------------     --------------------------
 ro          0                    0                   2                        2                  0
```

4. Prefetch using --home-fs-path option for a target with NSD protocol:

# **mmafmctl fs1 getstate -j ro2**

```
Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
------------ --------------     ----------- ------------ ------------ -------------
ro2          gpfs:///gpfs/remotefs1/dir3       Active      c26c4apv1    0           7
```

# **cat /lfile2**

```
113289 65538 0 -- /gpfs/homefs1/dir3/file2
113292 65538 0 -- /gpfs/homefs1/dir3/dir1/file2
```

# **mmafmctl fs1 prefetch -j ro2 –list-file=/lfile2 --home-fs-path=/gpfs/homefs1/dir3**

```
mmafmctl: Performing prefetching of fileset: ro2
```

# **mmafmctl fs1 prefetch -j ro2**

```
 Fileset Name Async Read (Pending) Async Read (Failed) Async Read (Already Cached) Async
 Read (Total) Async Read (Data in Bytes)
 ------------ -------------------- ------------------    ------------------ ----------------
 ro2         0                    0                   0                2              122880
```

Peer snapshot -psnap

The peer snapshot function provides a snapshot at home and cache separately, ensuring application consistency on both sides (cache and home).

When you take a peer snapshot, it creates a snapshot in the cache fileset then queues a snapshot to be executed at the home. When all of the queued requests outstanding at the time the snap was performed in cache have been flushed to the home fileset so that the home data is consistent with cache, a snapshot of the corresponding home fileset is performed. The result is a pair of peer snapshots, one at the cache and one at the home. Both refer to the same copy.

Peer snapshots are created using the **mmpsnap** command on the cache. To create a peer snapshot, the cache fileset must be in the active state. The last successful snapshot is saved and can be viewed by running the **mmlssnapshot** command at home or cache. Multiple outstanding peer snapshots can be queued on the gateway. Use the **mmpsnap** command to ensure that both the cache and home snapshots are removed. The **mmpsnap** command works only if the home cluster has run the **mmafmconfig enable ExportPath** command. The **mmpsnap** command can be used only in an SW cache. It cannot be used for RO, IW, or LU caches.

Note: Do not use the **mmdelsnapshot** command to delete peer snapshots.

If the cache fileset is disconnected from the home fileset when the cache snapshot is created, the cache records that the peer snapshot on the home fileset has not been created. When connection is restored, it attempts to create the snapshot.

Peer snapshots are not allowed on a SW cache that uses the NSD protocol for communicating with home.

1. To create a fileset level snapshot in cache of a single-writer fileset called sw in file system fs1, run the following command:

   ```
   mmpsnap fs1 create -j sw
   ```

   The system displays the following output:

   ```
   Writing dirty data to disk.
   Quiescing all file system operations.
   Writing dirty data to disk again.
   Snapshot psnap-13882361812785649740-C0A80E85:4F44B305-59-12-03-01-02-27-28 created with id 8.
   Snapshot psnap-13882361812785649740-C0A80E85:4F44B305-59-12-03-01-02-27-28 created at the satellite.
   Core snapshot has been queued.
   ```

2. To view the snapshot, run the following command:

   ```
   mmlssnapshot fs1 -j sw
   ```

   The system displays the following output:

   ```
   Snapshots in file system fs1:
   Directory SnapId Status Created Fileset
   psnap-13882361812785649740-C0A80E85:4F44B305-59-12-03-01-02-27-28 8 Valid Thu Oct 27 02:27:29 2016 sw
   ```

3. To view the snapshot at home, run the following command at home.:

   ```
   mmlssnapshot fs1
   ```

   The system displays the following output:

   ```
   Snapshots in file system fs1:
   Directory SnapId Status Created Fileset
   psnap-13882361812785649740-C0A80E85:4F44B305-59-12-03-01-02-27-28 8 Valid Thu Oct 27 02:23:16 2016
   ```

Gateway node failure and recovery

When the Primary gateway of a fileset fails, another gateway node takes over the ownership of the fileset.

Gateway node failures are not catastrophic and do not result in the loss of data or the loss of the ability of AFM to communicate with the home cluster with updates and revalidations.

AFM internally stores all the information necessary to replay the updates made in the cache at the home cluster. When a gateway node fails the in-memory queue is lost. The queue is rebuilt in memory by the node taking over for the failed gateway. The process of rebuilding the queue is called Recovery. As an administrator, ensure that you create sufficient disk space in `/var/mmfs/afm` for a smooth recovery. Approximately, one million files require 250 MB disk space in `/var/mmfs/afm`.

During recovery, outstanding cache updates are placed on the in-memory queue and the gateway starts processing the queue. AFM collects the pending operations by running a policy scan in the fileset. AFM uses the policy infrastructure in IBM Spectrum Scale to engage all the nodes mounting the file system. Pending requests are queued in a special queue called the priority queue which is different from the normal queue where normal requests get queued. After the priority queue is flushed to home, the cache and home become synchronized and recovery is said to be completed and the cache returns to an Active state. In some cases, the files or directories deleted at cache might not be deleted at home. Hence, these undeleted files or directories remain at home. However, the recovery operations remain unaffected.

The beginning and end of the AFM recovery process can be monitored by using the **afmRecoveryStart** and **afmRecoveryEnd** callback events.

Recovery is only used for single-writer and the independent-writer mode filesets. It is triggered when the cache fileset attempts to move to Active state, for example when the fileset is accessed for the first time after the failure.

Recovery can run in parallel on multiple filesets, although only one instance can run on a fileset at a time. The time taken to synchronize the contents of cache with home after the recovery of a gateway node depends on the number of files in the fileset and the number of outstanding changes since the last failure.

In multiple AFM cache environment, recovery is triggered after a failure. You can configure **afmMaxParallelRecoveries** to specify the number of filesets in the cluster on all filesystems. The recovery process is run on the number of filesets you specify for **afmMaxParallelRecoveries**. The number of filesets specified in **afmMaxParallelRecoveries** are accessed for recovery. After recoveries are complete on these filesets, the next set of filesets are accessed for recovery. By default, **afmMaxParallelRecoveries** is set to 0, and the recovery process is run on all filesets. Specifying **afmMaxParallelRecoveries** restricts the number of recoveries, thus conserving hardware resources. For more information, see *Configuration parameters for AFM* in *IBM Spectrum Scale: Administration Guide*.

Peer snapshots created in cache and queued to home might get lost due to gateway node failure. These peer snapshots cannot be recovered through AFM recovery. For details, see "Peer snapshot -psnap" on page 57.

If there are no updates to any AFM-enabled fileset, the failure of a gateway node is harmless and application nodes do not experience delays. During recovery, application requests to all AFM filesets are momentarily blocked.

The following example indicates changes in the AFM fileset state during recovery:

```
node2:/gpfs/cache/fileset_SW # mmafmctl fs1 getstate
         Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
         ------------ -------------- ------------- ------------ ------------ -------------
         fileset_SW nfs://node4/gpfs/fshome/fset001 FlushOnly node2 0 0

         node2:/gpfs/cache/fileset_SW # mmafmctl fs1 getstate
         Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
         ------------ -------------- ------------- ------------ ------------ -------------
         fileset_SW nfs://node4/gpfs/fshome/fset001 Recovery node2 0 0

         node2:/gpfs/cache/fileset_SW # mmafmctl fs1 getstate
```

```
        Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
        ------------ -------------- ------------- ------------ ------------ -------------
        fileset_SW nfs://node4/gpfs/fshome/fset001 Active node2 0 3
```

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

An example of the messages in mmfs.log is as follows:

```
Thu Oct 27 15:28:15 CEST 2016: [N] AFM: Starting recovery for fileset 'fileset_SW' in fs 'fs1'
Thu Oct 27 15:28:15 CEST 2016: mmcommon afmrecovery invoked: device=fs1 filesetId=1....
Thu Oct 27 15:28:16 CEST 2016: [N] AFM: mmafmlocal /usr/lpp/mmfs/bin/mmapplypolicy....
Thu Oct 27 15:28:31.325 2016: [I] AFM: Detecting operations to be recovered...
Thu Oct 27 15:28:31.328 2016: [I] AFM: Found 2 update operations...
Thu Oct 27 15:28:31.331 2016: [I] AFM: Starting 'queue' operation for fileset 'fileset_SW'in filesystem'fs1'.
Thu Oct 27 15:28:31.332 2016: [I] Command: tspcache fs1 1 fileset_SW 0 3 1346604503 38 0 43
Thu Oct 27 15:28:31.375 2016: [I] Command: successful tspcache fs1 1 fileset_SW 0 3 1346604503 38 0 43
Thu Oct 27 15:28:31 CEST 2016: [I] AFM: Finished queuing recovery operations for /gpfs/cache/fileset_SW
```

Failures during recovery

Filesets can be in recovery state and may not complete recovery due to some conditions. The fileset might go to **Dropped** or **NeedsResync** state, implying that recovery has failed.

The mmfs.log might contain the following lines: `AFM: File system fs1 fileset adrSanity-160216-120202-KNFS-TC11-DRP encountered an error synchronizing with the remote cluster. Cannot synchronize with the remote cluster until AFM recovery is executed. remote error 28.`

After the recovery fails, the next recovery is triggered after 120 seconds or when some operation is performed on the SW or IW fileset. After a successful recovery, modifications at the cache are synchronized with the home and the fileset state is 'Active'.

The following checks are needed from the administrator to ensure that the next recovery is successful:
1. Check the inode or block quota on cache and at home.
2. Ensure home is accessible. Remount home filesystem and restart NFS at home.
3. Ensure memory is not reached. If memory is reached, increase **afmHardMemThreshold**.
4. Check network connectivity with home.
5. If recovery keeps failing as eviction is triggered due to exceeding block quotas, increase block quotas or disable eviction to get recovery working.

Cache eviction

Cache eviction is a feature where file data blocks in the cache are released when fileset usage exceeds the fileset soft quota, and space is created for new files.

The process of releasing blocks is called eviction. However, file data is not evicted if the file data is dirty. A file whose outstanding changes are not flushed to home is a dirty file.

You can use automatic cache eviction or define your own policy to decide which file data is evicted. To automatically enable cache eviction, define a fileset soft quota on the cache fileset. Eviction starts when fileset usage reaches the soft quota limit. A time lag might result when an eviction is triggered and the data is evicted. Tuning the soft and hard quota limits minimizes application delays that are caused by the data being cached at a faster rate than it is being evicted.

Cache eviction based on inode quotas is not supported. Cached data from partially fetched files are not evicted from cache. When files are evicted from cache, all data blocks of those files are cleared. The files are then uncached. When a read operation is performed on the files the next time, the files fetch data from home.

Cache eviction is enabled by default on all AFM nodes and is controlled by the **afmEnableAutoEviction** parameter, and fileset block quota. Cache eviction can also be manually triggered by using the **mmafmctl evict** command. When a file is evicted, file data is removed from the cache, but the inode stays in the cache. Using eviction, you can build environments, where all objects from home are available but running in limited amount of space.

For example, a cache could be created in flash storage. File eviction opens a powerful method of providing small but high speed and low latency cache filesets to clusters.

Manual eviction can be done by using the **mmafmctl evict** command as under:

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

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

This option is applicable for RO/SW/IW/LU filesets. This command can be run manually or run in a script with a custom policy to implement a custom eviction policy. Options can be combined.

--safe-limit **SafeLimit** - This is a mandatory parameter for the manual evict option, for order and filter attributes. It specifies target quota limit which is used as the low water mark for eviction in bytes – the value must be less than the soft limit. This parameter can be used alone or can be combined with one of the following parameters (order or filter attributes). Specify the parameter in bytes.

--order **LRU | SIZE** - The order in which files are to be chosen for eviction: LRU - Least recently used files are to be evicted first. SIZE - Larger-sized files are to be evicted first.

--log-file **Log File**- The file where the eviction log is to be stored. By default no logs are generated.

--filter Attribute=**Value** - The attributes that you can use to control the way data is evicted from the cache. Valid attributes are: FILENAME=File Name - The name of a file to be evicted from the cache. This option uses an SQL-type search query. If the same file name exists in more than one directory, the cache will evict all the files with that name. The complete path to the file must not be given here. MINFILESIZE=Size - The minimum size of a file to evict from the cache. This value is compared to the number of blocks allocated to a file (KB_ALLOCATED), which might differ slightly from the file size. MAXFILESIZE=Size - The maximum size of a file to evict from the cache. This value is compared to the number of blocks allocated to a file (KB_ALLOCATED), which might differ slightly from the file size.

Possible combinations of options could be:

- Only Safe limit
- Safe limit + LRU
- Safe limit + SIZE
- Safe limit + FILENAME
- Safe limit + MINFILESIZE
- Safe limit + MAXFILESIZE
- Safe limit + LRU + FILENAME
- Safe limit + LRU + MINFILESIZE
- Safe limit + LRU + MAXFILESIZE
- Safe limit + SIZE + FILENAME
- Safe limit + SIZE + MINFILESIZE
- Safe limit + SIZE + MAXFILESIZE

`--list-file` **`ListFile`** - The specified file is a file containing a list of files to be evicted, one file per line. All files must have fully qualified path names. Filesystem quotas need not be specified. If the list of files to be evicted have filenames with special characters then a policy should be used to generate the listfile. This policy output should be hand-edited to remove all other entries except the filenames and can be passed to the evict command.

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

`enforceFilesetQuotaOnRoot` - Enable to evict files created by root. By default, files created by root are not evicted.

1. Evicting using filter option -

```
node1:/gpfs/cache/fileset_IW # mmlsfs fs1 -Q
          flag value description
          ------------------- -------------------------------------------------------------
          Q user;group;fileset Quotas accounting enabled
          none Quotas enforced
          none Default quotas enabled
```

```
node1:/gpfs/cache/fileset_IW # mmquotaon -v fs1
          mmquotaon on fs1
```

```
node1:/gpfs/cache/fileset_IW # mmlsfs fs1 -Q
          flag value description
          ------------------- -------------------------------------------------------------
          Q user;group;fileset Quotas accounting enabled
          user;group;fileset Quotas enforced
          none Default quotas enabled
```

```
node1:/gpfs/cache/fileset_IW # mmlsquota -j fileset_IW fs1
          Block Limits | File Limits
          Filesystem type KB quota limit in_doubt grace | files quota limit in_doubt grace Remarks
          fs1 FILESET 96 102400 2097152 0 none | 20 0 0 0 none
```

```
node1:/gpfs/cache/fileset_IW # mmafmctl fs1 evict -j fileset_IW --filter FILENAME='%work%'
          mmafmctl: Run mmcheckquota command before running mmafmctl with evict option
          mmafmctl: Command failed. Examine previous error messages to determine cause.
```

```
node1:/gpfs/cache/fileset_IW # mmcheckquota fs1

          fs1: Start quota check2 % complete on Thu Oct 27 09:19:41 2016
          [...]
          95 % complete on Thu Oct 27 09:20:03 2016
          100 % complete on Thu Oct 27 09:20:03 2016
          Finished scanning the inodes for fs1.
          Merging results from scan.
```

```
node1:/gpfs/cache/fileset_IW # mmafmlocal ls workfile1
          -rw-r--r-- 1 root root 104857600 Oct 27 08:30 workfile1
```

```
node1:/gpfs/cache/fileset_IW # mmafmctl fs1 evict -j fileset_IW --filter FILENAME='%work%'
          node1:/gpfs/cache/fileset_IW # mmafmlocal ls workfile1
          mmafmlocal: Command failed. Examine previous error messages to determine cause.
```

2. Manual evict using the **`--list-file`** option -

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

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

[root@c21f2n08 ~]# mmapplypolicy /gpfs/fs1/evictCache -I defer -P /tmp/evictionPolicy.pol -f
/tmp/evictionList
           #Edited list of files to be evicted

[root@c21f2n08 ~]# cat /tmp/evictionList.list.HomePREPDAEMON
           27858306 605742886 0   --/gpfs/fs1/evictCache/file3M

[root@c21f2n08 ~]# mmafmctl fs1 evict –j evictCache --list-file /tmp/
evictionList.list.HomePREPDAEMON
[root@c21f2n08 ~]# ls -lshi /gpfs/fs1/evictCachetotal 3.0M
            27858308 1.0M -rw-r--r--. 1 root root 1.0M Feb  5  02:07 file1M
            27858307 2.0M -rw-r--r--. 1 root root 2.0M Feb  5  02:07 file2M
            27858306    0 -rw-r--r--. 1 root root 3.0M Feb  5 02:07 file3M
```

3. Manual evict using the **--file** option -

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

[root@c21f2n08 ~]# mmafmctl fs1 evict –j evictCache --file /gpfs/fs1/evictCache/file1M
[root@c21f2n08 ~]# ls -lshi /gpfs/fs1/evictCache/file1M
         total 0
         27858308 0 -rw-r--r--. 1 root root 1.0M Feb  5 02:07 file1M
```

Operation with disconnected home

With a cache and home cluster separated by a WAN (wide area network), it might result in intermittent outages and possibly long-term disruptions.

If the primary gateway determines that the home cannot be accessed, the primary gateway waits until the interval time specified in the **afmDisconnectTimeout** parameter passes, and then changes the cache state to disconnected. This feature is available on all AFM filesets.

In a disconnected state, cached files are served to applications from the cache. Application requests for uncached data return an I/O error. All update operations from the cache complete, and return successfully to the application. These requests remain queued at the gateway until they can be flushed to home.

In a disconnected state, the home cannot be accessed for revalidation. Therefore, the latest updates from home are not available in the cache. Writes, which require revalidation with home might appear temporarily stuck if it is done after the Home fails, and before the cache moves to disconnected state. With revalidation stuck waiting on the unavailable home, the request times out as per the value set in the **afmDisconnectTimeout** parameter.

In the case of cache filesets based on the NSD protocol, when the home file system is not mounted on the gateway, the cache cluster puts the cache filesets into unmounted state. These cache filesets never enter the disconnected state.

If the remote cluster is unresponsive at the home cluster due to a deadlock, operations that require remote mount access, such as revalidation or reading uncached contents, stop responding until the remote mount becomes available again. This is true for AFM filesets that use the NSD protocol to connect to the home cluster. You can continue accessing cached contents without disruption by temporarily disabling all of the revalidation intervals until the remote mount is accessible again.

If a cache fileset is disconnected for an extended period, the number of file system updates might exceed the buffering capacity of the gateway nodes. In this situation, operations continue in the cache. When the connection to home is restored, AFM runs recovery and synchronizes its local updates to the home cluster.

AFM automatically detects when home is available and moves the cache into the Active state. The callback events **afmHomeDisconnected** and **afmHomeConnected** can be used to monitor when a cache changes state.

Filesets using a mapping target go into disconnected state if the NFS server of the Primary Gateway is unreachable, even when the NFS servers of all participating gateways are reachable.

Prefetch tasks that fail due to home disconnection, continue when home is available again.

The following example shows the number of read/write operations that were executed while the home was in the disconnected mode.

```
node1:/gpfs/cache/fileset_IW # mmafmctl fs1 getstate
             Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
             ------------ -------------- ------------- ------------ ------------------------
             fileset_IW nfs://node4/gpfs/fshome/fset002new Disconnected node2 0 102
```

```
node1:/gpfs/cache/fileset_IW # ls -la
             total 128
             drwx------ 65535 root root 32768 Oct 13 16:50 .afm
             -rw-r--r-- 1 root root 8 Oct 22 17:02 newfile2
              drwx------ 65535 root root 32768 Oct 22 05:23 .pconflicts
              drwx------ 65535 root root 32768 Oct 22 17:02 .ptrash
              dr-xr-xr-x 2 root root 32768 Oct 13 16:20 .snapshots
```

```
node1:/gpfs/cache/fileset_IW # for i in 1 2 3 4 5 ; do date > file$i ; done
```

```
node1:/gpfs/cache/fileset_IW # mmafmctl fs1 getstate
             Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
             ------------ -------------- ------------- ------------ ------------------------
             fileset_IW nfs://node4/gpfs/fshome/fset002new Disconnected node2 27 102
```

# turn on NFS at home

```
node1:/gpfs/cache/fileset_IW # mmafmctl fs1 getstate
             Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
             ------------ -------------- ------------- ------------ ------------------------
            fileset_IW nfs://node4/gpfs/fshome/fset002new Dirty node2 27 102
```

```
node1:/gpfs/cache/fileset_IW # mmafmctl fs1 getstate
            Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
            ------------ -------------- ------------- ------------ ------------------------
           fileset_IW nfs://node4/gpfs/fshome/fset002new Dirty node2 27 102
```

```
node1:/gpfs/cache/fileset_IW # date > file10
```

```
node1:/gpfs/cache/fileset_IW # mmafmctl fs1 getstate
```

```
Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
------------ -------------- ------------- ------------ -------------------------
fileset_IW nfs://node4/gpfs/fshome/fset002new Active node2 0 134
```

Expiring a disconnected RO cache

Read-only filesets can be configured to expire cached data after the gateway nodes are in a disconnected state for a specified amount of time.

This feature provides control over how long a network outage between the cache and home can be tolerated before the data in the cache is considered stale. This prevents access to old data, where old is defined by the amount of time that the WAN cache is out of synchronization with the data at the home site. Cached data is only available until the time period set in the **afmExpirationTimeout** parameter expires, at which point the cached data is considered 'expired' and cannot be read until a reconnect occurs.

This feature is not available in other AFM modes.

RO cache filesets created using the NSD protocol do not automatically go to the expired state, even if the expiration time-out is set for them. They continue to remain in the unmounted state for as long as the home file system is unavailable. The administrator can manually expire the data in an RO cache before the expiration timeout is reached, using the **mmafmctl expire** command only if the expiration time-out is configured for the fileset. An RO cache created using NSD protocol can also be manually expired. Use the following command - **mmafmctl Device {resync | expire | unexpire} -j FilesetName**. For more information, see *mmafmctl* command in *IBM Spectrum Scale: Command and Programming Reference*.

1. Expiring a RO fileset -

   ```
   #  mmafmctl fs1 expire -j ro1

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

2. Unexpiring a RO fileset -

   ```
   # mmafmctl fs1 unexpire -j ro1

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

The **afmFilesetExpired** callback event is triggered when a fileset moves into expired state. The **afmFilesetUnexpired** callback event is triggered when a fileset moves out of the expired state.

Viewing snapshots at home

All snapshots at home for RO and LU filesets can be viewed in the cache by setting the **afmShowHomeSnapshot** parameter to yes. This variable is not applicable for SW and IW filesets.

The home and cache can have different snapshot directories for clarity. If the **afmShowHomeSnapshot** parameter is changed after fileset creation, the change is reflected in the cache only after IBM Spectrum Scale is restarted, or the file system is remounted. However, if the value of **afmShowHomeSnapshot** parameter is changed from yes to no during the lifetime of a fileset, it continues to show the home snapshots even after a restart, or file system remount.

For RO and LU filesets using NFS, when a lookup is performed from the parent directory of .snapshot directory, the changes at home are not reflected in the cache, as NFS cannot detect the changes in

.snapshot. The **mtime/ctime** parameter of this directory is not modified at home despite snapshots creates or deletes at home. This is not true for RO and LU filesets using GPFS backend as the latest snapshot directory gets reflected in cache.

Changing home of AFM cache

AFM filesets continue to function in the event of home failures.

AFM filesets serve the cache applications with cached data. If the home is permanently lost, you can create a new home. An existing SW/IW AFM cache can make the following changes to the home. This is not supported on RO/LU filesets.

1. Replace the home with a completely new empty home where the new target is created using NFS/NSD mapping:

   The administrator must run the **mmafmctl failover** command without **–target-only** option to point to the new home. The new home is expected to be empty. To ensure that extended attributes are synchronized, run the **mmafmconfig** command on the new home before running the failover command. If the new target is a mapping, failover does not split data transfers and queues them as normal requests without parallel data transfer.
2. Replace any of the following on an existing home:
   a. Replace the communication protocol (NSD, NFS): The administrator must run the **mmafmctl failover** command without the **–target-only** option, using the new target protocol.

      Note: Only the protocol changes. The home path does not change.
   b. Enable or disable parallel data transfer by shifting between NFS and a mapping: The administrator must run the **mmafmctl failover** command without the **–target-only** option, using the new target protocol or mapping.

      Note: Only the protocol changes. The home path does not change.
   c. Replace either the IP address or the NFS server using the same communication protocol and home path: The administrator must run the **mmafmctl failover** command with the **–target-only** option. The IP/NFS server must be on the same home cluster and must be of the same architecture as the old NFS server.

      Note: Only the IP or NFS server changes.

During failover ensure that the cache filesystem is mounted on all gateway nodes, and the new home filesystem is mounted on all the nodes in the home cluster.

Note: Failover does not use parallel data transfers.

When creating a new home, all cached data and metadata available in the cache are queued in the priority queue to the new home during the failover process. The failover process is not synchronous and completes in the background. The **afmManualResyncComplete** callback event is triggered when failover is complete. Resync does not split data transfers even if parallel data transfer is configured, and the target is a mapping.

If the failover is interrupted due to a gateway node failure or quorum loss, failover is restarted automatically when the cache fileset attempts to go to Active state.

When there are multiple IW caches, the administrator must choose a primary IW cache and fail this over to a new empty home. All of the other IW cache filesets to the old home must be deleted and recreated. If another IW cache is failed over to the same home after failing over another IW cache, all the data in that cache overwrites existing objects at home. For this reason, failover to a non-empty home is discouraged.

When the failover function is likely to be used due to the failure of the old home, the admin must be cautious and disable automatic eviction. This is to ensure that eviction does not free the cached data on the cache. The evicted data cannot be recovered if the old home is lost.

Each AFM fileset is independently managed and has a one-to-one relationship with a target, thus allowing different protocol backends to coexist on separate filesets in the same file system. However, AFM does not validate the target for correctness when a fileset is created. The user must specify a valid target. Do not use a target that belongs to the same file system as the AFM fileset. For more information, see **mmafmctl** command in *IBM Spectrum Scale: Command and Programming Reference*.

The following example shows changing the target for an SW fileset.

```
node2:/gpfs/cache/fileset_SW # mmlsfileset fs1 fileset_SW --afm
             Filesets in file system 'fs1':
             Name Status Path afmTarget
             fileset_SW Linked /gpfs/cache/fileset_SW nfs://node4/gpfs/fshome/fset001

             Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
             ------------ -------------- ------------- ------------ -------------------------
             fileset_SW nfs://node4/gpfs/fshome/fset001 Active node2 0 6

            node2:/gpfs/cache/fileset_SW # mmafmctl fs1 failover -j fileset_SW --new-target
            nfs://node4/gpfs/fshome/fset002new
            mmafmctl:Performing failover to nfs://node4/gpfs/fshome/fset002new
            Fileset fileset_SW changed.
            mmafmctl: Failover in progress. This may take while...
            Check fileset state or register for callback to know the completion status.

            node2:/gpfs/cache/fileset_SW # mmafmctl fs1 getstate
            Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
            ------------ -------------- ------------- ------------ -------------------------
            fileset_SW nfs://node4/gpfs/fshome/fset002new Inactive

            node2:/gpfs/cache/fileset_SW # mmafmctl fs1 getstate
            Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
            ------------ -------------- ------------- ------------ -------------------------
            fileset_SW nfs://node4/gpfs/fshome/fset002new NeedsResync node2 6 0

            node2:/gpfs/cache/fileset_SW # mmafmctl fs1 getstate
            Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
            ------------ -------------- ------------- ------------ -------------------------
            fileset_SW nfs://node4/gpfs/fshome/fset002new Recovery node2 6 0

           node2:/gpfs/cache/fileset_SW # mmafmctl fs1 getstate
           Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
           ------------ -------------- ------------- ------------ -------------------------
           fileset_SW nfs://node4/gpfs/fshome/fset002new Active node2 0 6

          node2:/gpfs/cache/fileset_SW # mmlsfileset fs1 --afm
          Filesets in file system 'fs1':
          Name Status Path afmTarget
          root Linked /gpfs/cache --
          fileset_SW Linked /gpfs/cache/fileset_SW nfs://node4/gpfs/fshome/fset002new
```

Out-band Failover - You can choose to copy all cached data offline from the AFM cache to the new home with any tool that preserves modification time (**mtime**) with nanoseconds granularity. An example of such a tool is - rsync version 3.1.0 or later with protocol version 31. After the data is copied, you can run **mmafmctl failover** to compare **mtime** and **filesize** at home, and avoid queuing unnecessary data to home.

Resync on SW filesets

A conflict situation might arise due to an inadvertent write on a SW home. AFM runs a resynchronization automatically.

Writing data on a SW home is not allowed under normal circumstances. However, AFM does not enforce this condition. If there is an inadvertent write or corruption at a SW home due to some error situation, it results in a conflict scenario for the single-writer fileset.

Note: Resync does not use parallel data transfers. Even if parallel data transfer is configured and the target is a mapping, the data transfers are not split.

If AFM detects inconsistencies during flushing, the fileset goes into `NeedsResync` state. When this occurs AFM runs a resynchronization automatically during the next request to the primary gateway, and the inconsistencies are resolved.

If resync is not triggered automatically, the inconsistencies must be manually resolved by running the resync command . When running a resync the fileset must be in the Active or the `NeedsResync` state. Resync is not allowed on IW filesets as multiple cache filesets can update the same set of data. Use the following command - **`mmafmctl Device {resync | expire | unexpire} -j FilesetName`**. For more information, see *mmafmctl* command in *IBM Spectrum Scale: Command and Programming Reference*.

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

         # mmafmctl fs1 getstate -j sw1
         Fileset Name Fileset Target     Cache State Gateway Node Queue Length   Queue numExec
         ------------ --------------      ----------- ------------ ------------ -------------
         sw1    nfs://c26c3apv2/gpfs/homefs1/newdir1 Dirty      c26c2apv1    4067         10844
```

During a resync all cached data and metadata are queued in the priority queue. The resync process is asynchronous and completes in the background. The callback event `afmManualResyncComplete` is triggered when the resync completes. Once the priority queue is completely flushed, the fileset state changes to Active.

Evicted files are not synchronized to home.

A resync of partially cached files pushes only the cached data blocks to the new home. Uncached blocks are filled with null bytes.

Important: Do not use on a SW fileset, except when it is required to correct the home under such conflict scenarios.

Out-band Resync - You can choose to copy all cached data offline from the AFM cache to the new home with any tool that preserves modification time (**`mtime`**) with nanoseconds granularity. An example of such a tool is - rsync version 3.1.0 or later with protocol version 31. After the data is copied, you can run **`mmafmctl failover`** to compare **`mtime`** and **`filesize`** at home, and avoid queuing unnecessary data to home.

Resync does not delete new files or directories that are created at home. Similarly, rename of files/directories at home are not deleted by resync

Using IBM Spectrum Protect for Space Management (HSM)

IBM Spectrum Protect for Space Management (HSM) can be used on the AFM filesets or on the home.

The following figure illustrates HSM connected to home.

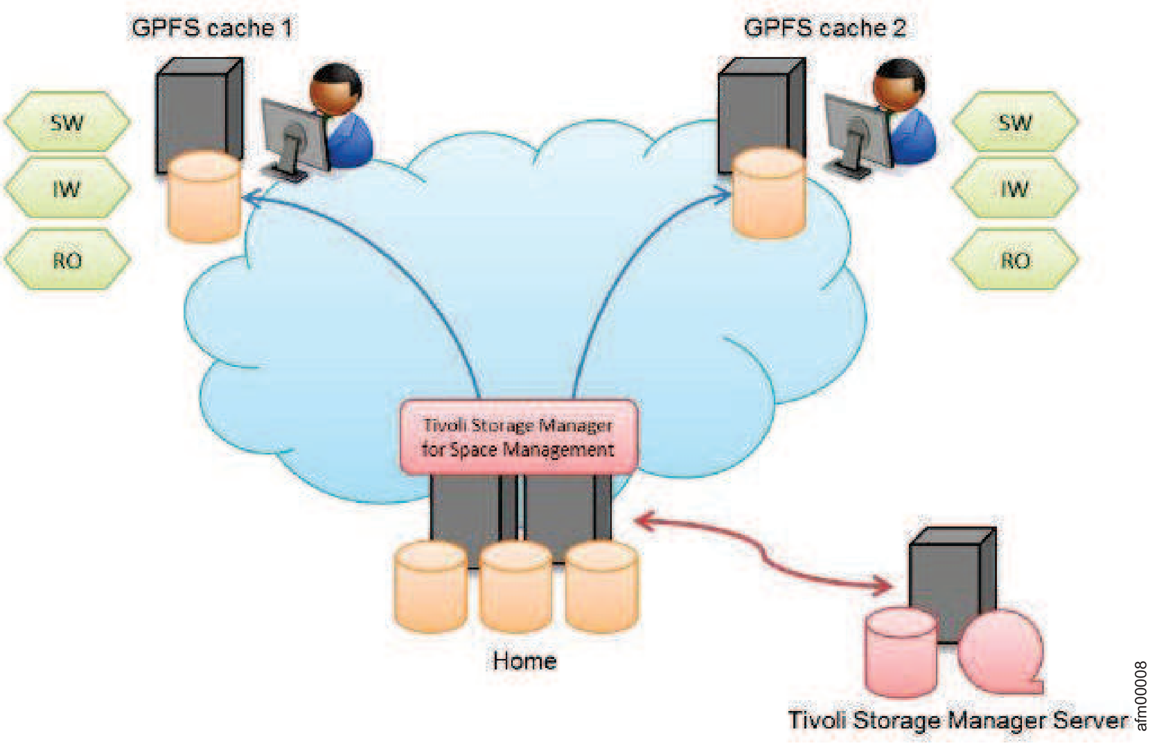

GPFS cache 1
GPFS cache 2
SW
IW
RO
SW
IW
RO
Tivoli Storage Manager for Space Management
Home
Tivoli Storage Manager Server
afm00008

Figure 12. Sample setup of HSM connected to home

A new file created at home becomes candidate for migration to the IBM Spectrum Protect server. When a migrated file at home is read from cache, the file is recalled at home and served to the cache. If the cache does a write on a migrated file at home, the file is recalled at home and written to when the AFM queue is flushed. If multiple migrated files are written at the same time, the home issues recalls for all files at the same time. It is recommended that you exclude frequently changing files from HSM migration process to avoid recalls.

The following figure illustrates HSM connected to both home and cache.

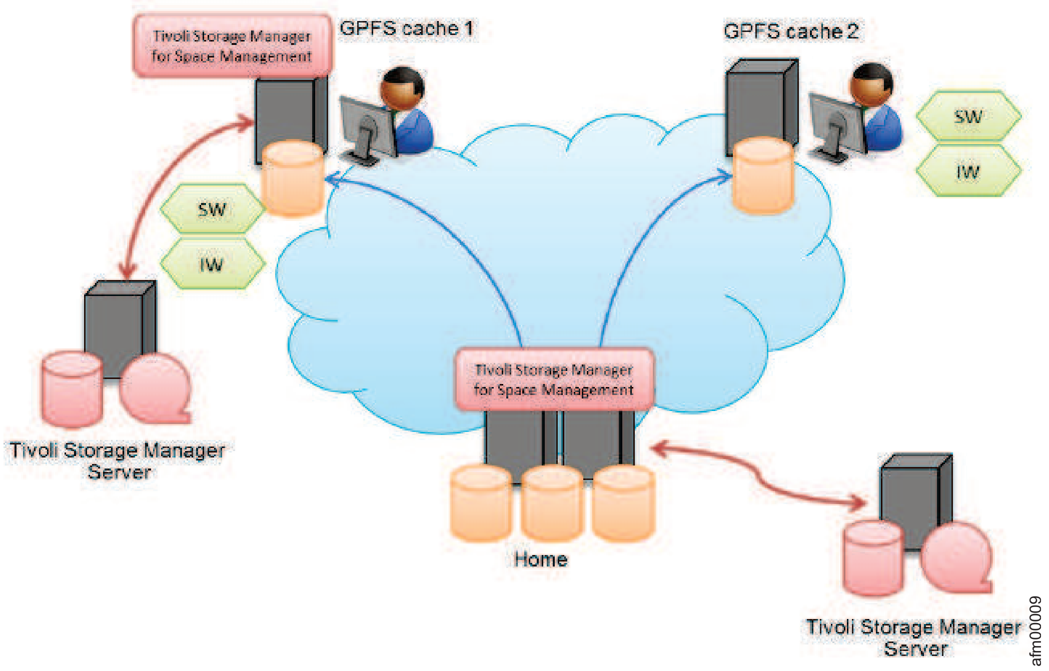

Tivoli Storage Manager for Space Management
GPFS cache 1
GPFS cache 2
SW
IW
SW
IW
Tivoli Storage Manager Server
Tivoli Storage Manager for Space Management
Home
Tivoli Storage Manager Server
afm00009

Figure 13. HSM connected to both home and cache

When using HSM on an AFM fileset, the flag **AFMSKIPUNCACHEDFILES** must be set in the dsm.sys configuration file (IBM Spectrum Protect-related) to yes. For example - **AFMSKIPUNCACHEDFILES** yes. This parameter should be used for read-write cache filesets. It prevents the migration of dirty and uncached files. If this flag is not set, it might result in long waiters or unexpected cache states and unexpected errors in terms of the HSM migration processing. In the LU mode when this parameter is set and a file is made local because of updating data, migration of the file to tape might be prevented. For migration in the LU mode, do not unset this parameter. Migrated files cannot be evicted. File operations such as read, write, truncate, append or rename on migrated files recalls the files into the cache. When multiple migrated files are modified at the same time all recalls are submitted at the same time, IBM recommends that you exclude frequently changing files or frequently read files from HSM migration process on both cache and home to avoid recalls.

It is recommended the following guidelines while using IBM Spectrum Scale AFM and IBM Spectrum Protect:

- Prevent cache eviction in combination with IBM Spectrum Protect on the same fileset. Both techniques have the same goal to reduce the space required in the fileset. The combination of both techniques unnecessarily increases the complexity of the environment.
- IBM Spectrum Scale snapshots and IBM Spectrum Protect have a limited compatibility. The deletion of a stub file that is reflected in a snapshot (the snapshot was generated before or after the file was migrated) causes the recall of the file data. The file data is stored in the snapshot so that it can be accessed later. Therefore, do not use snapshots for an AFM fileset (in home or cache) and in the file system hosting the AFM fileset, if you are using HSM.
- When using IBM Spectrum Protect on home or cache be aware that access (read or write) to multiple migrated files at the same time causes bulk recalls. Access to multiple files can be caused by users such as when they copy an entire directory or by AFM when changed files are replicated to home where the previous versions are migrated. You can avoid these issues by using the optimized tape recall process, which requires a list of files to be recalled before processing.

When running IBM Spectrum Scale AFM and IBM Spectrum Protect backup operations, prevent cache eviction in combination with IBM Spectrum Protect backup on the same fileset, if possible. Evicted (uncached) files will be skipped from backup processing. This might lead to errors in terms of the versioning of files on the IBM Spectrum Protect server.

For detailed description about the setup and configuration of HSM for AFM, see Configuring IBM Spectrum Scale Active File Management.

Performing a planned maintenance by using the IW cache

You can perform a planned maintenance by using IW cache.

Let us assume that we have IW cache on Side A that is hosting applications. IW cache points to a home in Side B. Complete the following steps for a planned maintenance of Side A:

1. Stop all applications in IW cache (Side A).
2. Flush the pending queue on IW cache (Side A) to home (Side B) by using **`mmafmctl Device flushPending -j`** *FileSetName*.
3. Unlink IW cache (Side A).
4. Start applications at home (Side B).
5. Relink the cache after maintenance window.
6. Stop the applications at home (Side B).
7. Start the applications at cache (Side A).
8. Schedule the background prefetch if required (Side A).

   The following figure illustrates IW cache (Side A) to home (Side B).

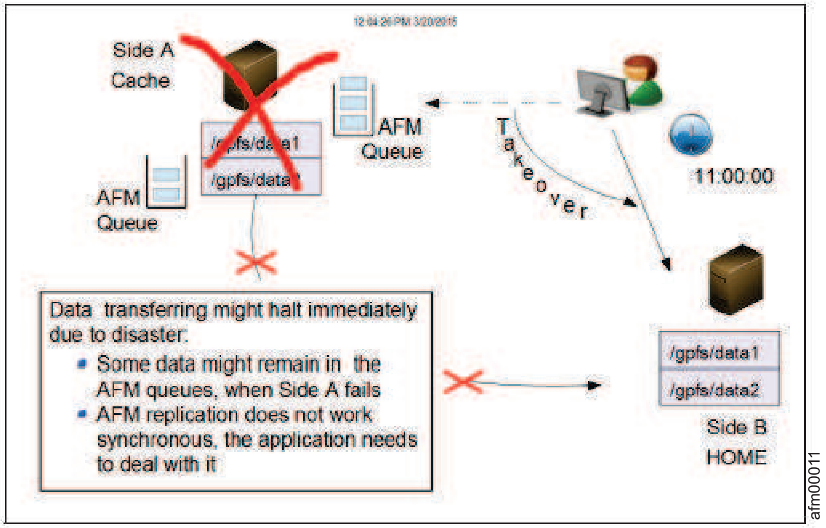

12:04:26 PM 3/20/2015
Side A
Cache
/gpfs/data1
/gpfs/data2
AFM
Queue
AFM
Queue
Takeover
11:00:00
Data transferring might halt immediately due to disaster:
• Some data might remain in the AFM queues, when Side A fails
• AFM replication does not work synchronous, the application needs to deal with it
/gpfs/data1
/gpfs/data2
Side B
HOME
afm00011

Figure 14. IW cache (Side A) to home (Side B)

Handling IW cache disaster

You can perform a planned or unplanned maintenance by using IW cache

Performing an unplanned maintenance using IW cache:

You can perform an unplanned maintenance by using IW cache.

For an unplanned outage, some changes might not be synchronized with the home. IW can handle requests lost from the cache due to a disaster, while allowing application updates at home to the same data.

The cache became unavailable because of the disaster at the IW cache. Therefore IW cache is down with pending updates (Side A). Steps to follow in an unplanned outage are:

1. Start applications at home (Side B). Record the time.
2. When the IW cache site (Side A) is ready to take over again, stop applications at the home site (Side B).

   The following figure illustrates IW cache site (Side A) to home site (Side B).

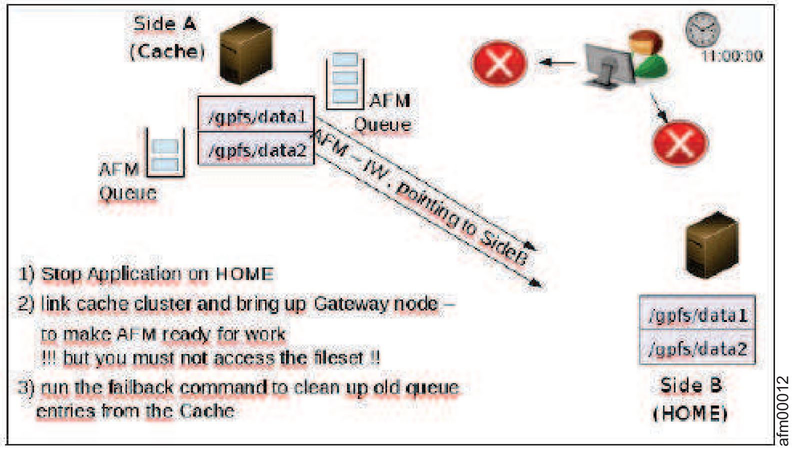

Side A
(Cache)
AFM
Queue
/gpfs/data1
/gpfs/data2
AFM
Queue
AFM – IW , pointing to SideB
11:00:00
1) Stop Application on HOME
2) link cache cluster and bring up Gateway node –
to make AFM ready for work
!!! but you must not access the fileset !!
3) run the failback command to clean up old queue
entries from the Cache
/gpfs/data1
/gpfs/data2
Side B
(HOME)
afm00012

Figure 15. IW cache site (Side A) and home site (Side B)

3. Link the cache site back (Side A) and ensure that the gateways are up. Do not access the fileset directories.

   If the cache directories are accessed, recovery is triggered. Old content from cache which were pending get synchronized with the home at the end of recovery, over-writing latest application changes at home. Perform the following steps so that the latest data is preserved and old or stale data is discarded. The data staleness is determined based on file mtime and failover time. The failover time is the time when applications were moved to the home, given as input to **failback** command.
4. Run the following command from the cache site to synchronize the latest data, specifying the time when the home site became functional and including the complete time zone, in case home and cache are in different time zones:

   ```
   mmafmctl FileSystem failback -j Fileset --start --failover-time 'TimeIncludingTimezone'
   ```

   The failback command resolves conflicts between pending updates in cache at the time of failover with the data changes at home. When the synchronization is in progress, the fileset state is `FailbackInProgress`. The fileset is read-only when the failback is in progress.

   After failback completes, the fileset state is `FailbackCompleted`. The failback process resolves conflicts in the following way:

   a. Dirty data that was not synchronized is identified.
   b. If the changed files or directories were not modified at the home when the applications were connected to the home, the cache pushes the change to the home to avoid any data loss.
   c. If the files or directories are recently modified at the home, the cache discards the earlier updates from the cache. The next lookup on the cache brings the latest metadata from the home.

   If the conflicted dirty data is a directory, it is moved to `.ptrash`. If the conflicted dirty data is a file, it is moved to `.pconflict`. The administrator must clean `.pconflict` directory regularly. If IW is converted to the other modes, `.ptrash` and `.pconflict` directories remain.
5. After achieving the `FailbackCompleted` state, run the following command to move the fileset from the Read-Only to the Active, where it is ready for use. If the command is not run successfully, run the command again.

   ```
   mmafmctl FileSystem failback -j Fileset --stop
   ```

Note:

If failback is not complete or if the fileset moves to `NeedFailback`, run the failback command again.

The cache site is ready for use. All applications can start functioning from the cache site. New files created at home are reflected in the cache on the next access, based on the revalidation interval. Failback does not pull in data of uncached files from home, which needs to be done explicitly by the administrator using **mmafmctl prefetch**. If failback might be used, the `ctime` of RENAME operations in the home file system must be updated. This is enabled using **setCtimeOnFileRename** at home:

```
mmchconfig setCtimeOnFileRename=yes –i
```

Note: Failback does not work if revalidation is disabled in the IW cache site.

Disabling AFM

An AFM fileset can be disabled and converted to a GPFS-independent fileset. However, GPFS-independent fileset cannot be converted to an AFM fileset.

To dissociate home and cache relationship of an AFM fileset, you can convert the AFM fileset to an independent fileset using **mmchfileset -p afmTarget=disable**. Thereafter, the fileset is a GPFS or IBM Spectrum Scale-independent fileset and requests are not queued to home. An example is the case of NFS migration, where AFM fileset must be disabled after migration.

You must disable AFM filesets while the filesets are linked. In some cases, disabling of AFM fileset might fail with an error message.

Complete the following steps to disable an AFM fileset -

1. If the fileset is unlinked, run **mmlinkfileset gpfs0** *FilesetName* **-J gpfs0/***FilesetName* to link the fileset.
2. Run **mmchfileset gpfs0** *FilesetName* **-p afmTarget=disable**.

   The following warning message displays - `Warning! Once disabled, AFM cannot be re-enabled on this fileset. Do you wish to continue? (yes/no)`

   Enter `yes`.

   The following warning message displays - `Warning! Fileset should be verified for uncached files and orphans. If already verified, then skip this step. Do you wish to verify same? (yes/no)`

   Enter `yes`.

   - If error conditions are not found, the following message displays

     `File system gpfs0 is mounted on` *n* `nodes or fileset` *FilesetName* `is not unlinked.`

     Unlink the fileset and run **mmchfileset gpfs0** *FilesetName* **-p afmTarget=disable** again. Enter `no` to the warning message about verification of the fileset. The AFM fileset is disabled.
   - In case of error conditions, disabling AFM fileset fails. See the following table for the error conditions with the corresponding messages.

Table 5. Conditions in which disabling AFM fileset fails, with corresponding messages

Condition	**Error message**
Incomplete directories present	`mmchfileset: 6027-2307 [E] Uncached files present, run prefetch first.`
Uncached files present	`Uncached files present, run prefetch first using policy output:` */var/mmfs/tmp/cmdTmpDir.mmchfileset.18149/list-file.mmchfileset.18149*

Table 5. Conditions in which disabling AFM fileset fails, with corresponding messages (continued)

Condition	Error message
Orphans present	`Orphans are present, run prefetch first using policy output:` */var/mmfs/tmp/cmdTmpDir.mmchfileset.18149/list-file.mmchfileset.18149*

You must run prefetch as suggested in the message to fix the error condition.

3. If uncached files or orphans are present, run prefetch using the following command - **mmafmctl gpfs0 prefetch -j** *FilesetName* **--list-file /***var/mmfs/tmp/cmdTmpDir.mmchfileset.18149/list-file.mmchfileset.18149*
4. If incomplete directories are present, generate a list file to prefetch the uncached files first. Use this generated list file containing uncached file entries to run the command - **mmafmctl gpfs0 prefetch -j** *FilesetName* **--list-file** *list file path*. Here, *list file path* is the generated list file.
5. Ensure that all data in the AFM fileset is prefetched.
6. Run step 2 on page 72 again and complete disabling AFM fileset..

Using AFM with encryption

AFM supports file encryption. Encryption can be applied to AFM-managed filesets.

AFM home sites and cache sites can be enabled with encryption, independent of each other. The data is encrypted while at rest (on disk) and is decrypted on the way to the reader/application hosted on home and caches; however, AFM communication between home and cache is not encrypted.

With the data that is flowing between home and cache filesets not being encrypted by the adoption of file encryption, communication between the clusters needs to be encrypted explicitly (if the privacy of the data over the network is a concern), by ensuring that a cipher list is configured. To ensure that the data is transmitted in the encrypted form, a cipher other than AUTHONLY must be adopted. AES128-GCM-SHA256 is one of the recommended ciphers. Run the **mmauth show** command to view the cipher lists used to communicate within the local and with the remote clusters. To ensure that all file content on disk and on the network is encrypted, configure file encryption at home and on the caches. Also configure a cipher list on all the clusters, ensuring that ciphers are configured within and across clusters. Even though file encryption results in the data being transmitted in the encrypted form between NSD clients and servers (both directions), neither file metadata nor RPC headers are encrypted. Only the use of encrypted communications (cipher list) ensures that the entire message content gets encrypted.

Using mmbackup

When **mmbackup** is run on a file system that has AFM filesets, only the cached data from the filesets is backed up.

The **mmbackup** does not back up the uncached data from the AFM filesets.

In a file system that is HSM managed, a backup operation in either the cache or home fileset will skip over migrated offline files that were not yet backed up. The backup of a file that is evicted from a cache causes the file to be read again from home to cache. To avoid reading the file again when eviction on the cache site is enabled, ensure that the backup occurs before the eviction.

AFM Limitations

AFM limitations include:

- If you change an AFM SW/IW home export from Kernel NFS or CNFS to the new IBM Spectrum Scale CES in 4.1.1 or later, AFM will not recognize the home. The best way to manage this situation is to run the **mmafmctl failover** command. See the IBM Spectrum Scale documentation for more details.
- AFM RO/LU home export cannot be changed from Kernel NFS or CNFS to new Spectrum Scale CES in 4.1.1. or later.

- Customers who have enabled DMAPI/HSM at the AFM home cluster should be running at RHEL 6.3, or later, or SLES 11 SP2, or later, on the cache cluster. There has been a bug identified in earlier levels of RHEL which requires that the exported path at home be excluded from DMAPI/HSM. See http://www-01.ibm.com/support/docview.wss?uid=ssg1S1004310 .
- Code upgrade limitations:
  - If there are AFM filesets using the Kernel NFS backend to communicate with a GPFS home that is running V4.1 or earlier, upgrading home to IBM Spectrum Scale V4.1.1 or later will cause the AFM filesets to disable synchronization to home with a message as follows:

    ```
    GPFS: 6027-3218 Change in home export detected. Synchronization with home
    is suspended until the problem is resolved
    ```

    Note: This only affects AFM filesets that were originally created with a GPFS home running V4.1 or earlier. This does not affect AFM filesets that were created with a home running IBM Spectrum Scale V4.1.1 or later. This issue is fixed in 4.1.1.4 and 4.2.0.1 available at Fix Central (IBM Fix central).
- AFM filesets do not support the gpfs API - **gpfs_prealloc**.

AFM and AFM DR Limitations

The common limitations for both AFM and AFM DR functions include:

- The OS/architecture support matrix is given below:

Table 6. Availability and OS/architecture supported

Arch/OS	Fileset	Supported	Not Supported
Linux for Power Systems™ and x86	AFM SW/IW/RO/LU AFM primary/secondary	Gateway Application Filesystem Manager	
Linux on Z	AFM SW/IW/RO/LU AFM primary/secondary	Gateway Application Filesystem Manager	
AIX	AFM SW/IW/RO/LU AFM primary/secondary	Application Filesystem Manager	AIX node as a gateway for AFM/AFM DR fileset and NFS server at home or secondary.

- 3rd party NFS servers at home/secondary:
  - Unsupported for AFM DR secondary.
  - Untested for all AFM modes. The IBM Spectrum Scale support team will help customers who are using other file systems at home to solve problems directly related to GPFS or IBM Spectrum Scale, but will not be responsible for solving problems deemed to be issues with the other file systems.
- AIX NFS servers at home/secondary:
  - Unsupported for AFM DR secondary.
  - Tested and supported for all AFM modes, but running **mmafmconfig** on the home cluster is not supported, which means, EA's, ACLs, sparseness of files, and psnap will not be supported.
- The **mmclone** command is not supported on AFM cache and AFM DR primary filesets. Clones created at home for AFM filesets are treated as separate files in the cache.
- Quality of Service is not tested on AFM and AFM Async DR filesets.
- AFM and AFM DR is not tested with IPv6 on gateway nodes.
- AFM and AFM DR filesets on Linux on Z is not supported with HSM.
- AFM and AFM DR filesets on Linux on Z is not supported with the NSD protocol for communication.
- Connecting AFM cache or home or AFM primary or secondary to IBM Spectrum Archive™ is not supported.

- Linking of a dependent or an independent fileset is not allowed inside the inode space of any AFM or AFM DR fileset (in other words, linking them under the junction path within the fileset is not allowed).
- AFM and Async DR is not supported on clusters that have Windows nodes.
- Cascading relationships with AFM caches and AFM primary filesets are not tested.
- Fileset snapshot restore is not supported for AFM and AFM DR filesets.
- AFM and AFM DR is not supported when SELinux is enabled.
- Files in an AFM or AFM DR fileset which are shared by using SMB, an error message `E_ACCESS` might appear if SMB holds locks on the files. When the locks are released by SMB, AFM replicates again and processes the files from the AFM or AFM DR fileset.

AFM-based Asynchronous Disaster Recovery (AFM DR)

The following topics introduce you to AFM-based Asynchronous Disaster Recovery (AFM DR).

The following figure illustrates the AFM disaster recovery process.

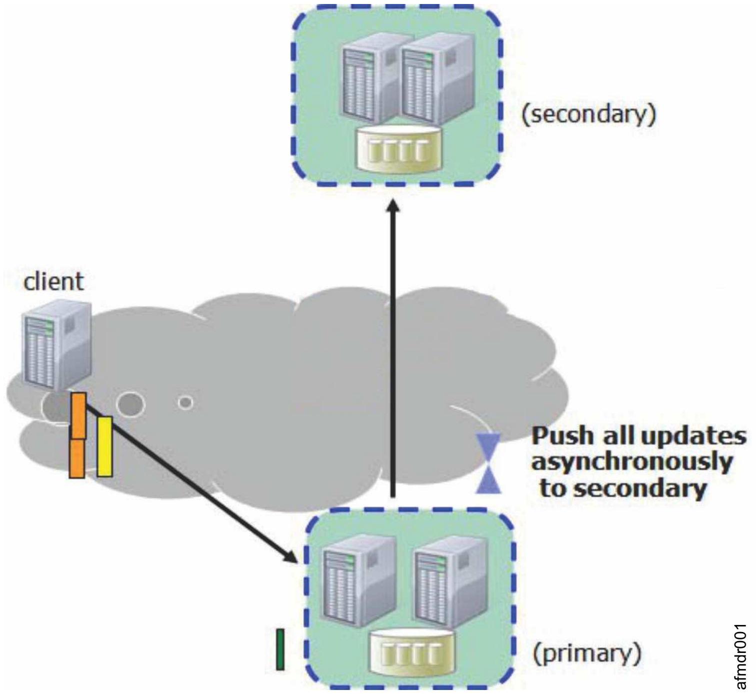


Figure 16. Asynchronous disaster recovery

Introduction

AFM-based asynchronous disaster recovery (AFM DR) is a fileset-level replication disaster-recovery capability.

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 helps to optimize your tuning parameters and enable the feature. Please include this message while contacting IBM Support.

For more information, see Flash (Alert): IBM Spectrum Scale (GPFS) V4.2 and V4.1.1 AFM Async DR requirement for planning.

The Disaster recovery solution includes:

- providing business stability during a disaster
- restoring business stability after the disaster has been repaired
- enduring multiple disasters
- minimizing data loss in the event of a disaster

AFM-based asynchronous disaster recovery is an AFM-based fileset-level replication disaster-recovery capability that augments the overall business recovery solution. This capability is a one-to-one active-passive model and is represented by two sites: primary and secondary.

The primary site is a read-write fileset where the applications are currently running and have read-write access to the data. The secondary site is read-only. All the data from the primary site is asynchronously synchronized with the secondary site. The primary and secondary sites can be independently created in storage and network configuration. After the sites are created, you can establish a relationship between the two filesets. The primary site is available for the applications even in the event of communication failures or secondary failures. When the connection with the secondary site is restored, the primary site detects the restored connection and asynchronously updates the secondary site.

The following data is replicated from the primary site to the secondary site:

- File-user data
- Metadata including the user-extended attributes except the inode number and atime
- Hard links
- Renames

The following file system and fileset-related attributes from the primary site are not replicated to the secondary:

- User, group, and fileset quotas
- Replication factors
- Dependent filesets

A consistent view of the data in the primary fileset can be propagated to the secondary fileset by using fileset-based snapshots (psnaps). Recovery Point Objective (RPO) defines the frequency of these snapshots and can send alerts through events when it is unable to achieve the set RPO. RPO is disabled by default. The minimum time you can set as RPO is 720 minutes. AFM-based Asynchronous DR can reconfigure the old primary site or establish a new primary site and synchronize it with the current primary site.

Individual files in the AFM DR filesets can be compressed. Compressing files saves disk space. For more information, see *File compression* in the *IBM Spectrum Scale: Administration Guide*.

Snapshot data migration is also supported. For more information, see *ILM for snapshots* in the *IBM Spectrum Scale: Administration Guide*.

In the event of a disaster at the primary site, the secondary site can be failed over to become the primary site. When required, the filesets of the secondary site can be restored to the state of the last consistent RPO snapshot. Applications can be moved or failed over to the acting primary site. This helps to ensure stability with minimal downtime and minimal data loss. This makes it possible for applications to eventually be failed back to the primary site as soon as the (new) primary is on the same level as the acting primary.

Note: AFM DR does not offer any feature to check consistency of files across primary and secondary. However, you can use any third-party utility to check consistency after files are replicated.

Recovery time objective (RTO)

Recovery time objective (RTO) is the time taken to convert secondary fileset into active primary fileset and move applications to that site, when a disaster occurs at the primary site.

Whenever a disaster occurs at the primary site, the secondary site becomes the acting primary by running the **`mmafmctl failoverToSecondary`** command. `--norestore` is set by default. The secondary fileset is converted to acting primary but avoids restoration of data from the last RPO snapshot and maintains the existing data. As an administrator, you can run the **`mmafmctl failoverToSecondary`** command with `--restore` option to clean the existing data at the secondary fileset and restore all data from the last RPO snapshot.

RTO can be estimated if the administrator knows the volume of data generated in an RPO interval. The volume of data that must be restored from the last snapshot depends on the number of files that have changed in an RPO interval as a snapshot restore requires a complete scan of the snapshot to determine which files have changed. Fail over time is one of the components that an administrator should use to calculate the RTO for the application.

Modes and concepts

AFM DR uses the same underlying infrastructure as AFM. AFM DR is characterized by two modes; the fileset in the primary cluster uses the primary mode, and the fileset in the secondary cluster uses the secondary mode.

AFM DR is supported over both NFS v3 and GPFS protocol. The primary fileset is owned by the MDS, which communicates with the NFS server on the secondary side. The primary-secondary relationship is strictly one-to-one.

AFM revalidation does not apply to primary filesets. All files are always cached, because primary is the only writer and secondary is in the read-only mode. After the applications failover to secondary and are failed back to primary, all updated directories and files with contents on the secondary site are pulled to the primary site by the failback process. The asynchronous delay is applicable to primary filesets and can be configured.

You can convert the SW/IW relationship to a DR relationship. However, you cannot convert a DR relationship to a SW/IW relation.

AFM-based Asynchronous Disaster Recovery features

The following sections describe the AFM-DR features.

The following AFM features are offered on AFM DR filesets:

- Force flushing contents before Async Delay
- Parallel data transfers
- Peer snapshot - psnap
- Gateway node failure and recovery
- Operation with disconnected secondary
- Using IBM Spectrum Protect for Space Management (HSM)
- Disabling AFM DR
- Using AFM DR with encryption

You can use **`mmbackup`** command to back up all files from primary, as all files are in a cached state on the primary fileset. Similar to AFM filesets, IBM Spectrum Protect (HSM) can be connected to primary or secondary, or both sides. When HSM is connected to the primary side, set `AFMSKIPUNCACHEDFILES yes` in dsm.sys file. AFM features such as revalidation, eviction, prefetch, partial file caching, expiration, resynchronization, failover, and showing home snapshots are not offered on AFM DR filesets.

RPO snapshots

Recovery point objective (RPO) snapshots are peer snapshots that are taken at the same time on the primary and the secondary sides. RPO is disabled by default, and the minimum value you can set is 720 minutes and increment in multiples of 5 minutes . You can update **afmRPO** parameter while creating the primary to change the interval, or after creating the fileset by using the **mmchfileset -p** command with the **afmRPO** parameter.

```
mmchfileset -p
```

The appropriate RPO intervals for each application setup are determined by the following factors:

- The rate of data change
- The fileset parameters such as **afmAsyncDelay** and **afmNumFlushThreads**
- The network bandwidth between the two sites

Each RPO interval triggers a snapshot at fileset level on both the primary and the secondary sides, that results in the file system being quiesced. Quiescing the file system is a data-transfer intensive action. Therefore, as a performance optimization, if multiple caches have similar RPOs, their snapshots are batched together so that the file system is quiesced only once. As further optimization, RPO snapshots are only taken for primary filesets whose data or metadata is modified since the last RPO period.

afmAsyncDelay specifies the minimum time that the primary site must wait before flushing the pending data that needs to be transferred to the secondary site. An RPO request in queue flushes the queue before creating a snapshot on the secondary site. The general guidelines for **afmAsyncDelay** is to set the asynchronous delay less than the RPO interval.

afmRPOMiss event occurs when RPO is missed because of a network delay or failure to create the RPO snapshot on the secondary site. If RPOs are missed, AFM waits for the expiration of the next RPO and during next RPO, a message is logged in mmfs.log and a new RPO snapshot is created. Failed RPOs are queued on the gateway again, and are run again at the secondary site until they succeed. At any point in time, two RPO snapshots that are not **psnap0** are retained on both sides. If RPOs are not played at the secondary due to some reason and primary does not get acknowledgment for the create from the secondary, the RPO is also deleted from primary. To improve the performance of more than one fileset taking RPOs at the same time, the RPO snapshots are batched together. In this process, the RPOs of few filesets might get slightly adjusted to make use of the batching facility.

Normal RPO deletions from the primary site are performed and queued when the new RPO snapshots are received. While there is every attempt to ensure that the delete operations are successful on the secondary site, there can be some extra snapshots as some delete operations might not be successful. In such cases, you must manually delete the extra snapshots on the secondary site by using **mmdelsnapshot -p**. Apart from automatic RPO snapshots, you can create user snapshots by running **mmpsnap create** with - rpo. These are intermediate snapshots between scheduled RPOs and are not deleted during the RPO snapshot delete process.

When the primary site fails, you can roll back the last consistent snapshot present on the secondary site while converting it to the current primary site. However, it is recommended to convert the secondary as-is into the acting primary, because in most cases the data in the secondary is always greater than or equal to the last RPO snapshot. When applications are moved to the current primary fileset after the failover operation, RPO snapshots are not taken because no secondary is associated with the current primary. Therefore, RPO is not applicable to the current primary site. The gateway node mapped to serve a fileset is called the primary gateway of the fileset. The primary gateway acts as the owner of the fileset. If the primary gateway fails, another gateway node takes over the fileset as primary gateway. The recovery is triggered on the new primary gateway while taking over the fileset. If the next RPO snapshot is due at the time when recovery gets triggered, a new RPO snapshot is taken and used for recovery. Whenever RPO snapshots are used for recovery, the normal cleanup process of the older RPO snapshots does not take place. Therefore, some extra RPO snapshots are temporarily displayed on the primary and secondary sites and are cleaned up on subsequent RPO intervals. No regular RPO snapshots are taken on a fileset that is running recovery until the recovery process is complete.

All user-created and system-created RPO snapshots except the **psnap0** belonging to an active primary-secondary must be deleted before the primary-secondary is deleted. You can change the RPO interval by running **mmchfileset**. The time for the next RPO snapshot (**Tnext**) is calculated by adding the RPO interval that you set (**Irpo**) to the time at which the previous snapshot occurred (**Tprev**): **Tnext** = **Tprev** + **Irpo**. If no file in the RPO snapshot changes during the interval, AFM does not take the snapshot, and the time for the next RPO snapshot is calculated as **Tnext** = **Tnext** + **Irpo**.

Note: The RPO interval is not added to the time at which you set the RPO interval, but to the time of the last RPO snapshot. For example, the previous snapshot was at 9:00 AM and the RPO interval is 720 minutes. If you change the RPO interval to 780 minutes during the 12 hours from 9:00 AM, the next RPO snapshot occurs after 780 minutes from 9:00 AM.

The RPO management deletes the oldest snapshot when it creates a new one but it never deletes the **psnap0**. The **mmpsnap** command can be run to delete **psnap0**. By deleting **psnap0**, storage utilization improves because data blocks used by **psnap0** are not held down. These data blocks can be significant over a period of time. Also, deletion of **psnap0** can improve the performance of subsequent creation and deletion of RPO snapshots. However, you must delete **psnap0** only when other RPO snapshots are present in the primary and secondary filesets to handle disaster recovery. If **psnap0** is deleted without any other RPO snapshots, data is lost. Also, **psnap0** can be deleted from secondary in cases like failing over to the secondary, and a new primary being set up.

How to decide if you need RPO:
Failover

RPO should be enabled if you need to go back in time after a failover by restoring the data on the secondary from the last RPO snapshot. But this is NOT recommended, as the data in the secondary is always greater than or equal to last RPO snapshot. Hence in most cases, RPO is not needed for failover.

Failback

If the primary comes back after a temporary failure, you would only need to copy the data that has changed in the secondary to the primary. For this, RPO function needs to be enabled. However, the RPO can be a large value, such as 24 hours. This value will dictate how much data is copied back to primary from the secondary during a failback. The higher the value, the larger the amount of data copied.

Factors for deciding the RPO interval: RPO interval dictates how often snapshots are created on the primary and secondary. The entire filesystem must be quiesced before taking a snapshot, even if the snapshot is for a single fileset in a massive filesystem.

CAUTION:
Setting RPO intervals on primary fileset has significant implications on the performance of the file system.

- Classify your filesets based on how critical the data is - the more critical the data the lower the async delay.
- To calculate this approximately, review the frequency/pattern of data generation at the primary and considering the number of GW nodes, network latency and other parameters, predict the time taken for this data to get synchronized with the secondary. The RPO must be set to this time.
- Do not set many filesets to the same RPO interval.

Failover to the secondary site

When the primary site stops functioning, all applications must be moved to the secondary site to ensure stability. Run **mmafmctlDevice failoverToSecondary -j FilesetName [--norestore |--restore]** to convert the secondary site to the active primary site.

You can restore the latest snapshot data on the secondary site during the failover process, or keep the existing data. The `--norestore` option is the default value, which keeps the existing data. After failover, the secondary site becomes acting primary and is ready to host applications.

The customer must ensure that the NFS export used for the secondary site is not removed so that after failback to the primary, the original primary site can communicate with the secondary site, which is the current primary . Also, the RPO snapshots existing on the current primary must not be deleted because the RPO snapshots are used during the failback process when the original primary site starts functioning again.

RPO snapshots are temporarily disabled on the active primary site. The new data is not replicated to any other site until a new primary is set up, or the original primary is reinstalled, and the current primary is converted back to the secondary site.

Failing back to the old primary site

This topic lists the steps to bring back the old primary site.

Complete the following steps to bring back the old primary site when it is repaired and it is back online after the disaster:

1. Run the following command to configure the old primary while the applications continue to run on the current primary site: **mmafmctl Device failbackToPrimary -j FilesetName { --start | --stop }[--force]** The `--start` option restores the primary to the contents from the last RPO on the primary before the disaster. With the `--start` option , the primary is in the read-only mode. This mode avoids accidental corruption until the failback process is completed. If the old primary site starts functioning again, all RPOs before the disaster are present and accessible. If a common RPO snapshot `psnap0` is not present, the old primary site can be converted to a normal GPFS fileset, and the steps described in failing back to new primary section must be followed to set up the primary site.

   If `--start` run on the fileset is unsuccessful, next `--start failbackToPrimary` might not be allowed. You can use the `--force` to start failback again.

   While the primary is coming back, as an administrator ensure that I/O does not occur on the AFM DR before the start of the failback process.
2. Run the following command to apply differences created by applications on the old primary site as the current primary site has taken over the applications: **mmafmctl Device { applyUpdates | getPrimaryId } -j FilesetName**. All the differences can be brought over in a single or multiple iterations. For minimizing the application downtime, this command can be run repeatedly to synchronize the contents of the original primary site with the current primary site. When the contents on both the sites are as close as possible or have minimal differences, applications must take a downtime and this command must be run one last time. `applyUpdates` might fail with an error during instances when the acting primary is overloaded. In such cases the command needs to be run again.
3. Complete the failback process by running **mmafmctl** with **failbackToPrimary --stop** option. With this command, the fileset is in the read-write mode. The primary site is ready for starting the applications. If `--stop` option of the failback does not complete due to errors and it does not allow for failback to be stopped, it can be forced to stop with the `--force` option.
4. Convert the current primary site back to secondary, and set the primary ID. Run the following command - **mmchfileset device fileset -p afmMode=secondary -p afmprimaryid=primaryid** Unlink the acting primary site, change it to secondary, and link the secondary site. NFS can be restarted on the secondary site, to ensure that the secondary export is accessible to the primary site. The primary and secondary sites are connected back as before the primary disaster and all data from the primary is played on the secondary site. Regular RPO also resumes on the primary site.

Failing back to the new primary site

This topic lists the steps to configure a new primary.

You can configure a new primary if the old primary site has stopped functioning. Complete the following steps:

1. Create a new GPFS fileset on the new primary site. This fileset is configured as the new primary. Create the latest snapshot from the acting primary by running **mmafmctl** command with replacePrimary option. A new psnap0 with all the latest contents is created on the current primary server. This snapshot is used in the subsequent steps in this task.
2. Copy the contents from the snapshot to the new primary site by using scp or other means outside of AFM.
3. The snapshot created in the above step, is an argument to **--secondary-snapname** option. Run the following command to convert the fileset on the primary site:

   ```
   mmafmctl Device convertToPrimary -j FilesetName
   [--afmtarget Target { --inband | --secondary-snapname SnapshotName}]
   [ --check-metadata | --nocheck-metadata ][--rpo RPO] [-s LocalWorkDirectory]
   ```

   The primary id is generated and a psnap0 with the same name is created on the primary site. After this, run all the steps in failing back to old primary.

Changing the secondary site

This topic lists the steps to follow when the secondary site stops functioning.

Complete the following steps if the secondary site stops functioning:

1. Create a new GPFS independent fileset on the new secondary site.
2. Copy the data on the primary site to the new GPFS fileset by using **ftp** or **scp**. This is known as outband trucking.
3. Run the following command to establish a new secondary site for the primary site:

   ```
   mmafmctlDevice changeSecondary-j FilesetName
   --new-target NewAfmTarget [ --target-only |--inband ]
   [-s LocalWorkDirectory]
   ```

   The **--inband** option is used depending on the method that is being used to truck data. **--target-only** can be used if you want to change the mount path or the IP address in the target path. The new NFS server must be in the same home cluster and must be of the same architecture as the existing NFS server in the target path. **--target-only** must not be used if the home is completely changed, or if you want to change the protocol while using the same home cluster. While using the same secondary path, if you want to change only the target protocol from GPFS to NFS and vice-versa, you can use **mmafmctl changeSecondary** command.
4. Convert the GPFS fileset on the new secondary site to a secondary and set the primary id by running **mmchfileset** or **mmafmctl** with **convertToSecondary** option.

   Ensure that the NFS export on the secondary site is accessible on all the gateway nodes on the primary cluster if NFS is used for defining AFM target on the primary site. If the GPFS protocol is used for the target, the secondary file system must be mounted on all gateway nodes on the primary site.

   After the primary and secondary sites are linked, the psnap0 queued from the primary fileset is played on the secondary fileset. The new secondary site is now linked to this primary site. Ensure that the primary filesystem is mounted on all gateway nodes and new secondary filesystem is mounted on all the nodes in the secondary cluster.

Using the primary fileset to make changes to the secondary fileset:

1. Replace the secondary with a completely new empty secondary where the new target is using NFS/NSD/mapping. The administrator must run the **mmafmctl changeSecondary** command without

`--target-only` option to point to the new secondary. The new secondary is expected to be empty. If the new target is a mapping, `changeSecondary` does not split I/Os and queues them as normal requests without parallel data transfer.

2. Replace any of the following on an existing secondary:
   - Replace the communication protocol (NSD/NFS):

     The administrator must run the `mmafmctl changeSecondary` command without `--target-only` option, using the new target protocol.

     Note: Only the protocol changes and not the secondary path.
   - Enable or disable parallel data transfer PIO by shifting between NFS/NSD and a mapping:

     The administrator must run the `mmafmctl changeSecondary` command without `--target-only` option, using the new target protocol or mapping.

     Note: Only the protocol changes and not the secondary path.
   - Replace only the IP address or NFS server using the same communication protocol and secondary path:

     The administrator must run the `mmafmctl changeSecondary` command with the `–target-only` option.

     Note: Only the IP or NFS server changes. The IP/NFS server must be on the same secondary cluster and must be of the same architecture as the old NFS server.

AFM DR Limitations

AFM DR Limitations include:

- 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 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.

  Note: This limitation does not apply to base AFM support. This applies only to Async DR available with the IBM Spectrum Scale Advanced Edition V4.1.1 and later.

  Have this information available when requesting a development led feasibility assessment for Async DR related proposals.
  - Detailed use case description including any application description for applications running in DR filesets.
  - Network latency and bandwidth between sites.
  - Scale of the solution including the number of filesets, number of files, type of files (small or large), any special file usage (clones or encryption).
  - RPO time.
  - Rate of change of data that needs to be transferred between two RPOs from Async DR primary site to secondary.
  - Number of Async DR filesets.
  - Any plans to use snapshots in Async DR filesets.
- If you change an AFM DR Secondary export from Kernel NFS or CNFS to the new IBM Spectrum Scale CES in 4.1.1 or later, AFM will not recognize the secondary. The best way to manage this situation is to run the **mmafmctl changeSecondary** command. See the IBM Spectrum Scale documentation for more details.
- DR administration using the **mmafmctl** command is not supported when run from the AIX OS.
- AFM ADR (primary/secondary filesets) is not supported on an FPO enabled system.

- AFM features such as revalidation, eviction, prefetch, show home snapshots, partial file caching, expire and unexpire, mmafmctl resync and failover, and IW failback are not applicable for AFM DR filesets.

AFM and AFM DR Limitations

The common limitations for both AFM and AFM DR functions include:

- The OS/architecture support matrix is given below:

Table 7. Availability and OS/architecture supported

Arch/OS	Fileset	Supported	Not Supported
Linux for Power Systems and x86	AFM SW/IW/RO/LU AFM primary/secondary	Gateway Application Filesystem Manager	
Linux on Z	AFM SW/IW/RO/LU AFM primary/secondary	Gateway Application Filesystem Manager	
AIX	AFM SW/IW/RO/LU AFM primary/secondary	Application Filesystem Manager	AIX node as a gateway for AFM/AFM DR fileset and NFS server at home or secondary.

- 3rd party NFS servers at home/secondary:
  - Unsupported for AFM DR secondary.
  - Untested for all AFM modes. The IBM Spectrum Scale support team will help customers who are using other file systems at home to solve problems directly related to GPFS or IBM Spectrum Scale, but will not be responsible for solving problems deemed to be issues with the other file systems.
- AIX NFS servers at home/secondary:
  - Unsupported for AFM DR secondary.
  - Tested and supported for all AFM modes, but running **mmafmconfig** on the home cluster is not supported, which means, EA's, ACLs, sparseness of files, and psnap will not be supported.
- The **mmclone** command is not supported on AFM cache and AFM DR primary filesets. Clones created at home for AFM filesets are treated as separate files in the cache.
- Quality of Service is not tested on AFM and AFM Async DR filesets.
- AFM and AFM DR is not tested with IPv6 on gateway nodes.
- AFM and AFM DR filesets on Linux on Z is not supported with HSM.
- AFM and AFM DR filesets on Linux on Z is not supported with the NSD protocol for communication.
- Connecting AFM cache or home or AFM primary or secondary to IBM Spectrum Archive is not supported.
- Linking of a dependent or an independent fileset is not allowed inside the inode space of any AFM or AFM DR fileset (in other words, linking them under the junction path within the fileset is not allowed).
- AFM and Async DR is not supported on clusters that have Windows nodes.
- Cascading relationships with AFM caches and AFM primary filesets are not tested.
- Fileset snapshot restore is not supported for AFM and AFM DR filesets.
- AFM and AFM DR is not supported when SELinux is enabled.
- Files in an AFM or AFM DR fileset which are shared by using SMB, an error message `E_ACCESS` might appear if SMB holds locks on the files. When the locks are released by SMB, AFM replicates again and processes the files from the AFM or AFM DR fileset.

AFM DR deployment considerations and best practices

This document describes the characteristics of AFM DR including advantages and limitations, its deployment method, and best practices.

AFM DR is a file-level asynchronous disaster recovery solution. Replication of data to a secondary cluster is asynchronous, which means that the data created on the primary cluster is copied to the secondary cluster with delay. The amount of delay is determined by the network bandwidth between the sites and some IBM Spectrum Scale tuning parameters.

| AFM DR copies only the data blocks that have changed on the primary cluster to the secondary cluster, which makes it a very efficient replication mechanism.

| Since data replication is asynchronous using AFM DR to recover data after a primary cluster failure might cause some data loss. The amount of data lost depends on the amount of data that is in the primary queue when a failure occurs. Therefore, AFM DR is not the right disaster recovery solution if your application cannot tolerate possible data loss. If this loss is not acceptable,customers should implement a synchronous disaster recovery mechanism.

Characteristics of AFM DR

The following sections give a brief description of the various characteristics of AFM DR.

Independent filesets in AFM DR:

The primary and secondary fileset clusters are independent filesets.

Features

- The disaster protection of data happens at an independent fileset level.
- A single file system can contain multiple AFM DR relationships, one per fileset.

Advantages

- Fileset-level replication allows a high level of granularity and flexibility to protect data.
- An RPO can be specified for each fileset, which allows you to set it based on the criticality of the data.
- AsyncDelay can be set on a fileset level allowing you to control rate of replication.

Limitations

- Can only be defined on an independent fileset.
- A single AFM DR relationship cannot be created at the file system level.

One-on-one relationships in AFM DR:

AFM DR creates a one-to-one relationship between the primary and the secondary filesets.

Advantages

- RPO intervals can be defined per AFM DR relationship.
- AsyncDelay can be specified per AFM DR relationship.
- Quotas can be specified for the primary and secondary filesets.

Limitations

You can only create two copies of the data.

Active-passive relationships in AFM DR:

AFM DR creates an active-passive relationship between the primary and the secondary clusters.

Features

- Only the primary can write to the secondary fileset. The secondary is read-only for all other users.

- You can take a snapshot of the secondary fileset and use the secondary as a source to create backups or additional copies.

Advantages

The secondary fileset is protected, so no other program or application other than the primary can write to the secondary.

Limitations

- All operations are allowed on the primary fileset, but only snapshot operations are allowed on the secondary fileset.
- The secondary can be converted into the primary even when the primary is writing to the secondary. There is no protection against an administrator doing a failover operation on the secondary, even if the DR relationship is healthy.
- The administrator must ensure that the primary cannot accidentally write to secondary after a failover.

NFSv3 versus NSD exports in AFM DR:

The primary accesses the secondary fileset using either NFSv3 or NSD protocol.

Features of NFSv3

NFSv3 is a stateless protocol, which is resilient on low-bandwidth and high-latency networks.

Advantages of NFSv3

It is recommended to use NFSv3 because NFSv3 is more tolerant when deployed on an unstable or high-latency network.

Note: It is recommended to use the NSD protocol only if the NFS protocol does not meet the expected performance requirements.

Limitations of NFSv3

For parallel I/O to work using NFSv3, create a map between the gateway nodes in the primary to the NFS servers in the secondary. For more information on parallel I/O configuration, see *Parallel I/O configuration parameters for AFM-based DR* in the *IBM Spectrum Scale: Administration Guide*.

Features of NDS

The NSD protocol is sensitive to packet drops and network latency.

Advantages of NSD

NSD provides high performance to LAN-based installations.

Limitations of NSD

If the secondary cluster is inaccessible, the NSD mount hangs, which in turn might cause the application to hang on the primary cluster.

Trucking features in AFM DR:

AFM DR trucking feature can be used to synchronize data between the primary and secondary filesets when the DR relationship is established for the first time.

Features

- AFM DR requires that the data between the primary and secondary be the same before establishing a DR relationship using the peer-to-peer snapshot (PSNAP) mechanism. Once the data on the primary and secondary are the same, the peer-to-peer snapshot mechanism creates a special snapshot called PSNAP0. The PSNAP0 is created on both the primary and the secondary fileset.
- Two trucking methods are supported by AFM DR:
  - Inband trucking: AFM DR copies the data from the primary to the secondary using the DR transport protocol (NFSv3 or NSD). After the copy is complete, the primary and secondary have the exact same data and PSNAP0 is created.
  - Outband trucking: It is the user's responsibility to copy all the data from the primary and secondary using any mechanism they choose. - for example, rsync with **mtime** preserved. After all the data is copied, the user can use the AFM DR commands to establish the DR relationship that creates the PSNAP0.

Note: Both methods use the **--inband** option of **mmafmctl** command while setting up relationship between the primary and secondary sites.

For more information on the two trucking methods, see "Converting GPFS filesets to AFM DR" on page 354. See the following use cases for recommendation on when to use inband versus outband trucking.

AFM DR trucking use cases:
Create the primary and secondary filesets from scratch

The primary and secondary clusters are empty and AFM DR establishes the relationship by creating PSNAP0 that is empty.

Trucking method recommendation: Use the inband trucking method for this scenario as there is no data to copy and create PSNAP0.

Create the primary and secondary fileset from scratch

In this case, either the inband or the outband mechanism can be used to copy the data to the secondary cluster.

- Inband:

  Using the inband method the existing data in the primary is copied by AFM DR to the secondary. To copy this data a work queue is created on the gateway that contains all of the files that need to be copied and file system operations that need to be performed on the secondary. Then the gateway processes this queue as quickly as possible copying data to the secondary. When all of the work outstanding when the command was executed is flushed to the secondary, a PSNAP0 is created. This marks the end of the trucking process.

  It is recommended to start applications on the primary after the trucking process is complete. However, if applications are started while the trucking process is in progress, the gateway node queues the data generated by the application and the requests from the trucking process at the same time. If this results in too many requests, it might fill up the gateway node queue buffer, resulting in queue memory overflow. In such a case the queue gets dropped and the gateway node goes into recovery mode. This can result in a cycle where the primary cluster will always lag behind the secondary cluster. The potential solutions are:

  - Add bandwidth to handle the initial synchronization workload. This may involve tuning - for example, increasing the number of flush threads to increase the gateway throughput, additional hardware or network bandwidth.
  - Keep the application from writing to the primary until the trucking process is complete and PSNAP0 is created.
  - Increase the gateway node memory allocation to prevent memory overflows.

- Outband:

  In this case a complete copy of the data exists before the primary-secondary relationship is created. The data can be copied to the secondary using any method including a restore from a backup or an rsync process. Once an identical copy of the data exists at the secondary AFM DR can establish the DR relationship. The primary and secondary need to be identical when the relationship is established, so during this time the data must not change in the primary. After the PSNAP0 is created, applications can start using the AFM DR filesets.

Note: Both methods use the `--inband` option of `mmafmctl` command while setting up relationship between the primary and secondary sites. For data that is already copied to the secondary site, `mtime` of files at the primary and secondary site is checked. If `mtime` values of both files match, data is not copied again and existing data on the secondary site is used. If `mtime` values of both files do not match, existing data on the secondary site is discarded and data from the primary site is written to the secondary site.

The primary fileset is empty and the secondary fileset has data

Trucking method recommendation: : In such a situation, the recommended mechanism is outband trucking where the administrator copies the data from the secondary to the primary and then uses the AFM DR conversion mechanism to establish the DR relationship.

Data exists on the 'to-be' primary and the secondary filesets

Ensure that the data is identical on the primary and secondary before the AFM DR conversion is performed.

Trucking method recommendation: This can be done only using outband trucking. AFM DR checks if the data is the same when the relationship is established.

Failover and Failback:
Failover

After a failure of the primary, applications can fail over to the secondary fileset. Performing a failover operation on the secondary fileset converts it to the primary and makes it writable. AFM DR failover has two modes: “As is” and “Revert to the latest RPO snapshot on the secondary”:

- As is :

  The “as is” data corresponds to the last byte of data that was copied to the secondary cluster before the primary cluster failed. This is always the latest data available on the secondary cluster. It is the recommended option and also the default behavior.
- Revert to the latest RPO snapshot on the secondary:

  Restoring data from the last RPO snapshot results in data being restored from the past. As a result, this process is not recommended in AFM DR as it might lead to data loss. For example, if the RPO interval was set to 24 hours and if the failover occurred at the 23rd hour, then restoring from the last snapshot (taken 24 hours ago) can result all the data created in the last 23 hours being lost. This failover option can take extra time as the data from the snapshot must be restored completely before the application can start using the fileset.

Failback

There are two options in AFM DR to failback to the original primary:

- Failback Option 1: New primary

  Create the primary from scratch because all the data in the primary was lost. Creating a primary from scratch requires that all the data be copied to the original primary from the active primary using whichever mechanism the administrator chooses

- Fail back Option 2: Temporary loss of primary

  A temporary failure of the primary that results in a failover of the application to the secondary. To re-establish the original primary:

  AFM DR uses the last RPO snapshot to calculate exactly what data changed on the secondary since the failover occurred. So for this to work there needs to be a common snapshot between the primary and secondary. The following steps give a simplified illustration of such a scenario:

  1. A PSNAP was created between the primary and secondary at 8 A.M. At this time the snapshot in the primary and secondary have the exact samedata.
  2. At9 A.M, the primary fails.
  3. At 9 A.M the applications failover to secondary.
  4. Primary is back up at 10 A.M., that is 2 hours later.
  5. Restore the T0 snapshot on the primary.
  6. Restore the 8 A.M snapshot on the primary.
  7. Copy the changed data calculated in step 6 from the secondary to primary. Now the primary is in sync with secondary.

Note: For more information on failback, see *Failback procedures* in the *IBM Spectrum Scale: Administration Guide*.

For a temporary primary failure, the PSNAP (peer-to-peer snapshot) mechanism can be used to periodically take synchronized snapshot between the primary and the secondary. The snapshots can be scheduled at fairly large intervals maybe once or twice a day. Only the data created in the secondary since the last PSNAP needs to be copied back to the primary during a failback.

Note:

The amount of data copied back to the primary depends on the following:

- How long the primary is unavailable.
- The amount of data created since the primary failure.
- How long the failure was from the last snapshot. For example, if the RPO snapshot is captured every 24 hours and the primary fails at the 23rd hour, then changes for the last 23 hours need to be recovered.

Deployment considerations for AFM DR

See the following considerations to understand if AFM DR fulfills your production requirements for disaster recovery:

Amount of data created per hour or per day

With data replication the amount of data created in a day determines the network bandwidth and the gateway node design, and dictates whether the replication rates can be supported by AFM DR. You can determine the requirements by looking at:

- Network bandwidth:

  The network must have the bandwidth to accommodate a transfer rate equivalent to the rate data is generated. You should consider how you want to handle high traffic events like fail-back where I/O requirements can be greatly increased for a short period of time.
- Gateway nodes(s)

  A gateway tracks all the changes and replicated the data from the primary to the secondary. As the number of filesets increase you can increase the number of gateway nodes in the primary cluster to increase throughput. The number of gateway nodes needed depends on:

  - The number of primary filesets.
  - The rate of data changes generated by each fileset.

- – The bandwidth of the gateway hardware and the network connections between clusters.
- – The method in which the filesets are distributed across gateway nodes. For example, it is possible that all the heavily loaded filesets are allocated to the same gateway node.

  Note: There is no manual way of controlling the fileset allocation to a gateway node. Currently, the method in which AFM DR allocates fileset to the gateway node is ad hoc and creates another challenge for the user.

The number of filesets on the primary cluster

The number of filesets impact the following factors:

- You might need to increase the number of gateway nodes for even distribution of the workload.
- The AFM DR RPO mechanism creates and deletes a snapshot after every RPO interval. As the number of filesets increase, creating the RPO snapshots simultaneously for all the filesets might cause a significant load on the system. The RPO can be disabled if there is no need to failback after a temporary primary failure. Alternatively, setting a large RPO interval can relieve some of this pressure however, once the RPO is enabled then all of the above points need to be considered carefully.

Note: A gateway node failure and recovery causes redistribution of the fileset workload, which in turn might cause changes in the performance of the replication characteristics.

Best Practices for AFM DR

Minimizing data loss during primary failure

AFM DR is asynchronous, so there is always a chance data can be lost in the event of a failure. The amount of data lost is affected by:

- The network bandwidth between the primary and secondary.
- The performance of the gateway which depends on Spectrum Scale tuning, the amount of memory and CPU available to gateway node and the number of filesets allocated to each gateway node. If the gateway node is overloaded it can result in replication rate reduction and reduced network bandwidth utilization.
- The ability of the primary and secondary to read and write the data to disk.

The AFM DR replication should be tuned to minimize the data loss in case of primary failure by keeping-up with the data creation rate. Some workloads occur in bursts, in this case you can design for average data transfer rates though keep in mind the time to sync is much greater right after a burst of changes.

Note: The AFM DR replication rate is independent of the RPO interval. The RPO interval does not affect the data loss sustained during primary failure.

Generating notification for failed replication

There is no automated notification mechanism in AFM DR to monitor the replication rates that are falling behind as long as they are within the RPO. However, a script can be written to periodically test the gateway node to see how fast the message queue is being processed. This provides an estimate of the replication rate sustained by the gateway node.

To monitor the RPO you can use the AFMRPOMISS callback event. This event is triggered if the RPO snapshot is not taken at the set interval. The event indicates something is wrong within the system, and can be used as a trigger to start an analysis of what needs to be rectified within the system to bring it back to optimal performance.

Using tuning parameters to improve performance

There are several AFM DR tuning parameters that can be used to tune performance. For more information on tuning parameters, see *Configuration parameters for AFM-based DR* in the *IBM Spectrum Scale: Administration Guide*.

AFM DR use case

This use case describes the AFM-based Asynchronous Disaster Recovery (AFM DR) process when NFS and SMB clients are mounted from the primary IBM Spectrum Scale cluster.

Overview of AFM DR when NFS and SMB clients are mounted from the primary cluster

During scheduled maintenance on the primary cluster or a failure at the primary site cluster, those NFS, and SMB clients that mount exports from the primary cluster experience a brief and temporary outage, when the **mmafmctl failover commands** are executed. After the **mmafmctl failover** commands complete, the NFS and SMB clients can mount from the new acting primary cluster (formerly the secondary cluster).

Note: To ensure a successful failover, system administers at each site need to create identical export definitions on the secondary cluster. It is important to keep the definitions up to date when changes occur on the primary cluster.
After you create initial AFM DR relationships, you can create NFS and SMB exports so that applications can access the IBM Spectrum Scale data via protocols.

Use case 1: The path to the data is identical on both the primary and secondary clusters

1. On both the primary and the secondary clusters, enter the same path and client (-c) definitions as shown in the following example:
```
mmnfs export add /ibm/gpfs0/export1 -c
9.11.102.0/24(Access_Type=RO,Squash=root_squash;
9.11.136.0/23(Access_Type=RO,Squash=root_squash;
10.18.40.0/22(Access_Type=RW,Squash=root_squash;
10.18 48.0/22(Access_Type=RW,Squash=root_squash;
```
2. On the NFS clients,
   - Issue the **showmount -e** command and the IP address of the primary cluster as shown in the following example:
```
showmount -e 10.18.96.100
Export list for 10.18.96.100:
/ibm/gpfs0/export1
9.11.102.0,911.136.0,10.1840.0,10.18.48.0
```
   - Issue the **showmount -e** command and the IP address of the secondary cluster as shown in the following example:
```
showmount -e 10.18.50.30
Export list for 10.18.50.30:
/ibm/gpfs0/export1
9.11.102.0,911.136.0,10.18.40.0,10.18.48.0
```
3. On the NFS clients, issue the following command to mount from the primary CES IP addresses as shown in the following example:
```
mount -o hard, sync
10/18.96.100:/ibm/gpfs0/export0 /mnt_pnt
```
4. If a failure occurs on the primary cluster, initiate failover. For details, see *Performing failover for protocols cluster when primary cluster fails* in *IBM Spectrum Scale: Administration Guide.*.
5. On the NFS clients, issue the following commands to unmount from the primary CES IP addresses and then mount from the secondary CES IP addresses as shown in the following example:
```
umount -f /mnt_pnt
mount -o hard, sync
10.18.50.30:/ibm/gpfs0/export0 /mnt_pnt
```

Use case 2: The path to the data is different on the primary and secondary cluster

1. Create the export with the same client (-c) definitions.
   - Issue the following command and the IP address of the primary cluster:

     ```
     mmnfs export add /ibm/gpfs0/export1 -c
     9.11.102.0/24(Access_Type=RO,Squash=root_squash;
     9.11.136.0/23(Access_Type=RO,Squash=root_squash;
     10.18.40.0/22(Access_Type=RW,Squash=root_squash;
     10.18 48.0/22(Access_Type=RW,Squash=root_squash;
     ```

   - Issue the following command and the IP address of the secondary cluster:

     ```
     mmnfs export add /ibm/gpfs3/export9 -c
     9.11.102.0/24(Access_Type=RO,Squash=root_squash;
     9.11.136.0/23(Access_Type=RO,Squash=root_squash;
     10.18.40.0/22(Access_Type=RW,Squash=root_squash;
     10.18.48.0/22(Access Type=RW,_quash=root_squash;
     ```

2. On the NFS clients,
   - Issue the **showmount -e** command and the IP address of the primary cluster:

   ```
   showmount -e 10.18.96.100
   Export list for 10.18.96.100:
   /ibm/gpfs0/export1
   9.11.102.0,911.136.0,10.18.40.0,10.18.48.0
   ```

   - Issue the **showmount -e** command and the IP address of the secondary cluster:

   ```
   showmount -e 10.18.50.30
   Export list for 10.18.50.30:
   /ibm/gpfs3/export9
   9.11.102.0,911.136.0,10.18.40.0,10.18.48.0
   ```

3. On the NFS clients, issue the following command to mount from the primary CES IP addresses:

   ```
   mount -o hard, sync
   10/18.96.100:/ibm/gpfs0/export0 /mnt_pnt
   ```

4. If a failure occurs on the primary cluster, initiate failover. For more information, see *Performing failover for protocols cluster when primary cluster fails* in the *IBM Spectrum Scale: Administration Guide.*.
5. On the NFS clients, issue the following commands to unmount from the primary CES IP addresses and then mount from the secondary CES IP addresses:

   ```
   umount -f /mnt_pnt
   mount -o hard, sync
   10.18.50.30:/ibm/gpfs3/export9 /mnt_pnt
   ```

Data protection and disaster recovery in IBM Spectrum Scale

The IBM Spectrum Scale installation should be protected against data loss to ensure continuity of operation after a malfunction.

Data loss can be prevented by protecting the three types of key data:
- Cluster configuration data
- File system configuration data
- File system contents (user data, metadata, configuration)

Cluster configuration data is the administrative data which associates the nodes, addresses, networks, and software installation on each node. The system administrators should save the output of the **mmlscluster** command to ensure reconstruction of this data is possible if needed.

File system configuration data consists of a wide variety of information involving all the file systems in the cluster. To protect this data, it is essential to use the **mmbackupconfig** command for each file system and save the output files for future use. This configuration data details which disks are associated as NSD components of which file systems, how much storage is in use, the filesets defined, the quotas defined, and other valuable configuration data which describes the file system structure. These do not include the file data in the user files. User file data is the lowest level of information and most frequently changing contents that need protection.

IBM Spectrum Scale has built in data protection processes that allows the users not only to backup and restore valuable data, but also recover data that could be potentially lost or corrupted by their own actions. For example, unintentional deletion or overwriting of a user's file.

Data backup options in IBM Spectrum Scale

This section describes the various options available in IBM Spectrum Scale to back up data.

Backing up the data

IBM Spectrum Scale creates a second copy of the data within the system as backup. These secondary copies are usually point-in-time copy of the original data, and can be created on a daily, monthly, or weekly basis. In a situation where the actual data is corrupted, the system uses the backed-up copy to restore data to a previous point in time.

You can use the **mmbackup** command to back up the user data from a GPFS file system or independent fileset to a TSM server or servers. The **mmbackup** command can be used only to back up file systems owned by the local cluster. For more information on how to back up your data, see *Backup your data* in the *IBM Spectrum Scale: Problem Determination Guide* and *mmbackup command* in the *IBM Spectrum Scale: Command and Programming Reference*.

Fileset and backup

Starting from version 4.1.1, IBM Spectrum Scale supports backup from independent filesets in addition to backup of the whole file system. For more information on fileset backup, see *Filesets and backup* in the *IBM Spectrum Scale: Administration Guide*. If you are planning to use IBM Spectrum Protect to back up IBM Spectrum Scale file systems, see “Backup considerations for using IBM Spectrum Protect” on page 169

Data restore options in IBM Spectrum Scale

This section describes the various options available in IBM Spectrum Scale to restore data.

Restoring data

The system can restore the data if it becomes corrupted due to user errors, hardware failure, and loss or even if the data gets corrupted due to bugs in other application software. For more information on how to restore system data, see *Restoring data and system configuration* in the *IBM Spectrum Scale: Problem Determination Guide*.

Data mirroring in IBM Spectrum Scale

In IBM Spectrum Scale you can copy data from one location to a secondary storage location, thus creating a mirror image of the original. This replication is called data mirroring.

In data mirroring the data is copied in real time, so the information stored in the secondary copy is always an exact replica of the data in the primary copy. Data mirroring is useful in the speedy recovery of critical data after a disaster. Data mirroring can be implemented locally or offsite at a different location. . For more information on data mirroring and replication, see *Data mirroring and replication* in the *IBM Spectrum Scale: Administration Guide*.

Protecting file data using snapshots

A snapshot of an entire file system or of an independent fileset can be created to preserve the contents of the file system or the independent fileset at a single point in time.

A fileset snapshot is a snapshot of the entire inode space. Any snapshot of an independent fileset also includes any dependent filesets contained within that independent fileset. You cannot create a snapshot of a dependent fileset. For more information on creating and maintaining snapshots, see *Creating and maintaining snapshots of file systems* in the *IBM Spectrum Scale: Administration Guide*.

Snapshots can be used in environments where multiple recovery points are necessary. Ensure that the file system is mounted before restoring the snapshot. For information about mounting a file system, see *Mounting a file system* in *IBM Spectrum Scale: Administration Guide*. For more information on restoring data using snapshots, see *Restoring a file system from a snapshot* in the *IBM Spectrum Scale: Administration Guide*.

Note:

For object protocol, the best method to ensure consistency in the event that data has to be restored from a snapshot, is to configure the `cesSharedRoot` directory to be in the same file system as the object filesets.

Introduction to Scale Out Backup and Restore (SOBAR)

Scale Out Backup and Restore (SOBAR) is a data protection mechanism used specifically for disaster recovery situations.

SOBAR is used to back up and restore GPFS files that are being managed by IBM Spectrum Protect for Space Management (HSM). For more information on SOBAR, see *Scale Out Backup and Restore (SOBAR)* in the *IBM Spectrum Scale: Administration Guide*.

Introduction to protocols cluster disaster recovery (DR)

Protocols cluster disaster recovery (DR) allows an IBM Spectrum Scale cluster to fail over to another cluster and fail back once the primary cluster is back up. It also backs up and restores the protocol configuration information using the **mmcesdr** command in cases where a secondary cluster is not available.

During failover, the files can be re-created or restored in the new cluster. However, the default is set to re-create, and the restore option can be selected if complete configuration information needs to be restored (such as due to corruption). Once the failover is complete, and the original cluster is active once again, the secondary cluster can be failed back to the old primary, or a new primary. The data can be re-created or restored during the failback as well. Failback is used when there is a new primary cluster, while restore is used when no existing configurations exist. For more information on disaster recovery and how it can be set up in IBM Spectrum Scale, see *Protocols cluster disaster recovery* in the *IBM Spectrum Scale: Administration Guide*.

Note: The re-create option for failover keeps existing SMB and NFS exports and customer data within their corresponding filesets. The restore option in failback deletes any SMB and NFS exports (but not data within their corresponding filesets) that were on the secondary system before failover occurred. However, neither re-create nor restore affects object configuration or object data.

Commands for data protection and recovery in IBM Spectrum Scale

The following section lists the commands used for data protection and recovery in IBM Spectrum Scale.

mmbackup

Used to back up the user data from a GPFS file system or independent fileset to a IBM Spectrum Protect server or servers. For more information on **mmbackup**, see *mmbackup command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmbackupconfig
Used to collect basic file system configuration information. The configuration information backed up by this command includes block size, replication factors, storage pool layout, filesets and junction points, quota information, and a number of other file system attributes. This command forms a part of the SOBAR utilities along with the **mmrestoreconfig**, **mmimgbackup** and **mmimgrestore** commands. For more information on **mmbackupconfig**, see *mmbackupconfig command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmcesdr
Used to perform initial configuration for protocols disaster recovery on the primary cluster and to generate a configuration file that is used on the secondary cluster. For more information on **mmcesdr**, see *mmcesdr command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmfsck
Used to find and repairs conditions that can cause problems in your file system. The **mmfsck** command operates in two modes: online and offline. The online mode only checks and recovers unallocated blocks on a mounted file system. In general, it is unnecessary to run **mmfsck** in the offline mode, unless so specified by the IBM Support Center. For more information on **mmfsck**, see *mmfsck command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmimgbackup
Used to perform a backup of a single GPFS file system metadata image. This command forms a part of the SOBAR utilities along with the **mmbackupconfig**, **mmrestoreconfig** and **mmimgrestore** commands. Make sure that you run the **mmbackupconfig** command before you run the **mmimgbackup** command. For more information on **mmimgbackup**, see *mmimgbackup command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmimgrestore
Used to restore a single GPFS file system metadata image. This command forms a part of the SOBAR utilities along with the **mmbackupconfig**, **mmrestoreconfig** and **mmimgbackup** commands. The **mmrestoreconfig** command must be run before running the **mmimgrestore** command. For more information on **mmimgrestore**, see *mmimgrestore command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmrestoreconfig
Used to restore file system configuration information. The **mmrestoreconfig** command allows you to either query or restore, or both query and restore the output file generated by the **mmbackupconfig** command. This command forms a part of the SOBAR utilities along with the **mmbackupconfig**, **mmimgbackup** and **mmimgrestore** commands. For more information on **mmrestoreconfig**, see *mmrestoreconfig command* in the *IBM Spectrum Scale: Command and Programming Reference*.

mmrestorefs
Used to restore user data and attribute files to a file system or an independent fileset by using those of the specified snapshot. For more information on **mmrestorefs**, see *mmrestorefs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

Introduction to IBM Spectrum Scale GUI

The IBM Spectrum Scale management GUI provides an easy way to configure and manage various features that are available with the IBM Spectrum Scale system.

You can perform the following important tasks through the IBM Spectrum Scale management GUI:

- Monitoring the performance of the system based on various aspects
- Monitoring system health
- Creating and managing file systems
- Creating filesets and snapshots
- Managing Objects and NFS and SMB data exports

- Creating administrative users and defining roles for the users
- Creating object users and defining roles for them
- Defining default, user, group, and fileset quotas
- Monitoring the capacity details at various levels such as file system, pools, filesets, users, and user groups
- Configuring call home
- Configuring and monitoring thresholds

The following table provides an overview of the features associated with each GUI page.

Table 8. Features associated with IBM Spectrum Scale GUI pages

GUI Page	Function
Home	Provides an overall summary of the IBM Spectrum Scale system configuration, status of its components, and services that are hosted on it. This page is displayed in the GUI only if the minimum release level of IBM Spectrum Scale is 4.2.2 or later.
Monitoring > **Dashboards**	Provides a dashboard to display the predefined performance charts and the customized charts that are marked as favorite charts in the Performance page.
Monitoring > **Events**	Monitor and troubleshoot the issues that are reported in the system.
Monitoring > **Tips**	Monitor events of type “tips”. The tip events give recommendations to the user to avoid certain issues that might occur in the future.
Monitoring > **Statistics**	Provides a list of pre-defined performance charts to monitor the performance of the system based on various aspects.
Monitoring > **Thresholds**	Configure and various thresholds based on the performance monitoring metrics. You can also monitor the threshold rules and the events that are associated with each rule.
Nodes	Monitor the performance and health status of the nodes that are configured in the cluster.
Network	Monitor the performance and health status of various types of networks and network adapters.
Files > **File Systems**	View and manage file systems. File system creation through GUI is not supported in 4.2.x release.
Files > **Filesets**	Create, view, and manage filesets.
Files > **Snapshots**	Create snapshots or snapshot rules to automate snapshot creation and retention.
Files > **User Capacity**	Monitor the capacity details of users and user groups.
Files > **Quotas**	Create and manage default, user, group, and fileset quotas at the file system level.
Files > **Information Lifecycle**	Create, manage, and delete policies that manage automated tiered data storage.
Files > **Transparent Cloud Tiering**	Provides both a summarized and attribute-wise details of the Transparent Cloud Tiering service, which is integrated with the IBM Spectrum Scale system.

Table 8. Features associated with IBM Spectrum Scale GUI pages (continued)

GUI Page	Function
Files > **Active File Management**	Helps to monitor the performance, health status, and configuration aspects of AFM, AFM DR, and gateway nodes. You can see this page in the GUI that is part of the AFM cache cluster with at least one AFM-enabled fileset.
Storage > **Pools**	Helps to monitor the performance, health status, and configuration aspects of all available pools in the IBM Spectrum Scale cluster.
Storage > **NSDs**	Provides an easy way to monitor the performance, health status, and configuration aspects of the all network shared disks (NSD) that are available in the IBM Spectrum Scale cluster.
Protocols > **NFS Exports**	Create and manage NFS exports. Protocols pages are displayed in the GUI only when the protocol feature is enabled on the cluster.
Protocols > **SMB Shares**	Create and manage SMB shares. Protocols pages are displayed in the GUI only when the protocol feature is enabled on the cluster.
Object > **Accounts**	Create and manage accounts and containers in the object storage. Object pages are displayed in the GUI only when the object feature is enabled on the cluster.
Object > **Users**	Create object users.
Object > **Roles**	Define roles for the object users.
Access > **GUI Users**	Create users and groups. This page is visible only if internal user repository is used for GUI user authentication.
Access > **GUI Access**	Create users and groups. The GUI Access page is available only if external authentication server is enabled to manage the GUI user authentication.
Access > **File System ACL**	Define ACL templates and apply ACLs on files and directories.
Access > **Remote Connections**	Request access to the GUI node of the remote cluster to establish communication between the local and remote clusters. You can also use this page to grant access to the remote clusters when an access request is received.
Settings > **Event Notifications**	Configure event notifications to notify administrators when events occur in the system.
Settings > **Diagnostic Data**	Download diagnostic data to troubleshoot the issues that are reported in the system.
Settings > **NFS Service**	Specify NFS server settings and start or stop NFS services.
Settings > **SMB Service**	Specify SMB server settings and start or stop SMB services.
Settings > **Object Administrator**	Define object administrator who can manage accounts in the object storage.
Settings > **Object Service**	View and change the Object service status.
Settings > **GUI Preferences**	Specify a message that can be displayed in the login page.

Table 8. Features associated with IBM Spectrum Scale GUI pages (continued)

GUI Page	Function
Settings > **Call Home**	Configure call home feature that automatically notifies the IBM Support personnel about the issues reported in the system. You can also manually upload diagnostic data files and associate them with a PMR through the GUI.

Note: The default user name and password to access the IBM Spectrum Scale management GUI are `admin` and `admin001` respectively.

Assistance for understanding the features associated with a GUI page

The following three levels of assistance are available for the GUI users:

1. **Hover help:** When you hover the mouse over the tiny question mark that is placed next to the field label, the system displays a brief description of the feature that is associated with that field. Hover help is only available for the important and complex fields.
2. **Context-sensitive help:** Provides a detailed explanation of the features that are associated with the page. The context-sensitive help files are available in the help menu, which is placed in the upper right corner of the GUI page.
3. **Knowledge Center:** This is the third level of information where the users can find entire details of the product. Link to the IBM Spectrum Scale Knowledge Center is also available in the help menu, which is placed in the upper right corner of the GUI page.

Related concepts:

"Manually installing IBM Spectrum Scale management GUI" on page 252
The management GUI provides an easy way for the users to configure, manage, and monitor the IBM Spectrum Scale system.

"Manually upgrading the IBM Spectrum Scale management GUI" on page 397
You can upgrade the IBM Spectrum Scale management GUI to the latest version to get the latest features. You can upgrade one GUI node at a time without shutting down IBM Spectrum Scale on other nodes to ensure high availability.

IBM Spectrum Scale management API

With the IBM Spectrum Scale management API, you can develop scripts to automate labor-intensive cluster management tasks. These APIs provide a useful way to integrate and use the IBM Spectrum Scale system.

The IBM Spectrum Scale management API is a REST-style interface for managing IBM Spectrum Scale cluster resources. It runs on HTTPS and uses JSON syntax to frame data inside HTTP requests and responses. For more information, see "Functional overview" on page 99.

Based on the architecture and syntax, the following two different versions are available for IBM Spectrum Scale management API:

IBM Spectrum Scale management API Version 1
Introduced in the IBM Spectrum Scale 4.2.2 release. Implementation was based on Python and the deployment limited only to the manager nodes that runs on RHEL7.

IBM Spectrum Scale management API Version 2
Introduced in the 4.2.3 release. The implementation is based on GUI stack. GUI server is managing and processing the API requests and commands. The following figure illustrates the architecture of API Version 2:

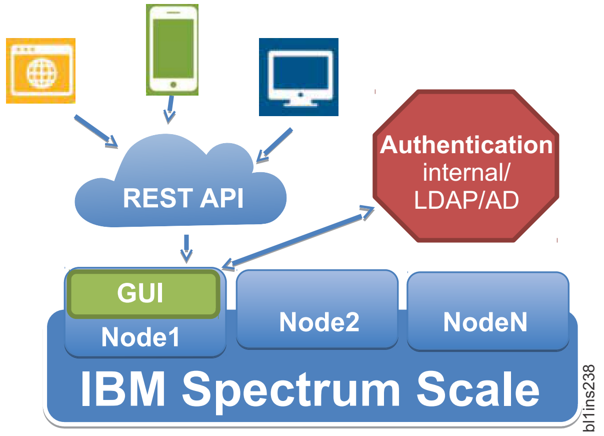


Figure 17. IBM Spectrum Scale management API Version 2 architecture

The Version 2 has the following features:

- Reuses the GUI deployment's backend infrastructure, which makes introduction of new API commands easier.
- Uses the same role-based access feature that is available to authenticate and authorize the GUI users. No additional configuration required for the API users.
- Makes deployment easier as the GUI installation takes care of the basic deployment.
- Fixes scalability issues and introduces new features such as filter and field parameters and it supports paging.
- Highly scalable and can support large clusters with thousands of nodes.
- The APIs are driven by the same WebSphere® server and object cache that is used by the IBM Spectrum Scale GUI.

Important: Both version 1 and version 2 are supported in IBM Spectrum Scale 5.0.0 release. However, new API commands and features are added only in the version 2 architecture. Those who are using the APIs in the version 1 architecture are recommended to switch to version 2 to avail the advanced features and flexibility.

Installing and configuring IBM Spectrum Scale management API

For information about how to configure the IBM Spectrum Scale system to start using the APIs, see Chapter 9, "Installing and configuring IBM Spectrum Scale management API," on page 343.

IBM Spectrum Scale management API command reference

For information about the API commands, see the topic *IBM Spectrum Scale management API commands* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

You can also access the documentation corresponding to each API command from the GUI itself. The API documentation is available in the GUI at: *https://<IP address or host name of API server>:<port>/ibm/api/explorer/*. For example: *https://scalegui.ibm.com:443/ibm/api/explorer*.

Using the IBM Spectrum Scale management API

The following topics describe the API features in detail.

Functional overview

The IBM Spectrum Scale management API is a REST-style API that provides interoperability between a client and server over a network. These APIs allow authenticated users to perform day-to-day storage management tasks.

The following list provides the salient features of the REST-style APIs:

- Resource-based
- Stateless
- Client-server
- Cacheable
- Layered system

REST is a resource-based service system in which requests are made to the resource's Universal resources identifier (URI). This invokes a response from the resource in the JSON format. An IBM Spectrum Scale management API *resource* is a collection of all the objects of the same type in a cluster, such as all the filesets. A *resource element* is an instance of a resource, such as a particular fileset named fileset1. In a REST API request, a resource or a resource followed by a resource element appears as the last term in the request, except for any trailing HTTPS parameters. The following example request ends with a resource, `filesets`, and therefore it indicates this as an operation that affects all the filesets that are part of a file system named *gpfs0*:

```
https://198.51.100.1:443/scalemgmt/v2/filesystems/gpfs0/filesets
```

Corresponding URL in the IBM Spectrum Scale management API version 1 is given in the following example:

```
https://198.51.100.1:8191/scalemgmt/v1/filesets?filesystemName=fs1
```

In contrast, the following example ends with a resource followed by a resource element `filesets/fileset1`, and therefore indicates an operation that affects this particular fileset, such as creating or deleting the fileset:

```
https://198.51.100.1:443/scalemgmt/v2/filesystems/gpfs0/filesets/fileset1
```

Corresponding URL in the IBM Spectrum Scale management API version 1 is given in the following example:

```
https://198.51.100.1:8191/scalemgmt/v1/filesets/fileset1?filesystemName=fs1
```

The kind of operations that can be performed on the resources or a resource element are directed by the HTTP methods such as GET, POST, PUT, DELETE, and in some cases by parameters in the HTTPS request. The following list provides the meanings of the basic HTTP methods used in the requests:

- GET: Reads a specific resource or a collection of resources and provides the details as the response.
- PUT: Updates a specific resource or a collection of resources.
- DELETE: Removes or deletes a specific resource.
- POST: Creates a new resource.

API requests

API requests include a URL and in some cases HTTP parameters or JSON data.

URL for an API request

The URL for an API request has the following format:

```
https://domain_name:port/scalemgmt/v2/resource_type/[resource_element]
```

where

domain_name:*port*

Specifies the domain name or address and the port of the IBM Spectrum Scale management API server, which is IBM Spectrum Scale GUI server.

/scalemgmt/v2

Specifies the path of the API server. This path is not configurable; it must be `/scalemgmt/v2`.

resource_type

Specifies a resource type. The supported resource types are as follows:

ces/addresses

ces/services

cluster

config

filesets

filesystems

info

jobs

nfs/export

nodes

nodes/health

nodeclasses

quotas

smb/shares

snapshots

perfmon

/filesystems/{filesystemName}/filesets/{filesetName}/psnaps

thresholds

resource_element

Specifies the name of a particular resource, such as the name of a file system, a fileset, a node, or a snapshot. Optional.

As described earlier, an API request that ends with a resource type indicates a request that affects all the instances of that resource in the cluster. For example, the following HTTP request, sent with a GET method, returns information about all the nodes in the cluster:

In contrast, an API request that ends with a resource type followed by a resource element indicates a request that affects only the specified resource element. For example, the following HTTP request, sent with a GET method, returns information about the specified node, `node1`:

Data in API requests

Some types of API requests require one or more input parameters that follow the resource or resource element. For example, sent with a GET method, the following request returns information about all the filesets in the specified file system. :

Table 9. Getting information about filesets

Request method	GET
URL	`https://198.51.100.1:443/scalemgmt/v2/filesystems/gpfs0/filesets/fs1`
JSON data	None.

Also, some types of the API requests require input parameters to be included in the body of the HTTP request in JSON format. For example, the following request creates a new snapshot in a file system. The JSON data specifies the name of the file system and the name of the new fileset:

Table 10. Creating a snapshot

Request method	**POST**
URL	`https://198.51.100.1:443/scalemgmt/v2/filesystems/gpfs0/snapshots`

JSON data	`{`
	`"snapshotName":"snap1"`
	`}`

Request headers

The following headers are required for requests:

```
'content-type: application/json' 'accept: application/json'
```

Fields parameter

The field parameter can be used to specify the fields of the returned object by adding the query parameter "?fields" to a GET URL. For example, `?fields=field1,field2,parentField1.field1`.

You can use certain keywords to get the desired results in the response. The following list provides the basic keywords that can be used with the fields parameter:

- '**:none:**' Returns only the fields that are required to uniquely identify an object. For example, filesystemName and filesetName for a fileset.

  Use this keyword for queries that return more than one object. This means that only the keys that are needed to retrieve the full object are returned.
- '**:all:**' Returns all fields. Use this in queries to a single object.

Apart from with these keywords, you can also use the following options to customize the response by using the fields parameter:

- Use the field name "." to query fields that are wrapped in a parent object.

  For example, use `?fields=config.iamMode` to query for "iamMode" in the following object:

  ```
  "config" : {
  "iamMode" : "off"
  }
  ```
- Use parent name to query for all fields of a parent. For example, `?fields=config,afm` includes all fields of the parent objects "config" and "afm"
- Multiple fields can be separated using ",". For example, `?fields=config.iamMode,status.id`

Examples

This section lists out the examples of using various keywords in the fields parameter.

1. `curl -k -u admin:admin001 -X GET --header 'accept:application/json' 'https://198.51.100.1:443/scalemgmt/v2/filesystems/gpfs0/filesets?fields=:all:'` . This GET call returns the following response:

   ```
   {
     "status": {
       "code": "200",
       "message": "..."
     },
     "paging": {
       "next": "https://localhost:443/scalemgmt/v2/filesystems/gpfs0/filesets?lastId=1001"
     },
     "filesets": [
       {
   ```

```
      "config": {
        "filesetName": "myFset1",
        "filesystemName": "gpfs0",
        "path": "/mnt/gpfs0/myFset1",
        "inodeSpace": "0",
        "maxNumInodes": "4096",
        "permissionChangeMode": "chmodAndSetAcl",
        "comment": "Comment1",
        "iamMode": "off",
        "oid": "158",
        "id": "5",
        "status": "Linked",
        "parentId": "1",
        "created": "2016-12-13 13.59.15",
        "isInodeSpaceOwner": "1",
        "inodeSpaceMask": "0",
        "snapId": "0",
        "rootInode": "131075"
      },
      "afm": {
        "afmTarget": "nfs://10.0.100.11/gpfs/afmHomeFs/afmHomeFileset2",
        "afmAsyncDelay": "0",
        "afmDirLookupRefreshInterval": "60",
        "afmDirOpenRefreshInterval": "60",
        "afmEnableAutoEviction": "false",
        "afmExpirationTimeout": "100",
        "afmFileLookupRefreshInterval": "30",
        "afmMode": "read-only",
        "afmNumFlushThreads": "4",
        "afmParallelReadChunkSize": "134217728",
        "afmParallelReadThreshold": "1024",
        "afmParallelWriteChunkSize": "0",
        "afmParallelWriteThreshold": "0",
        "afmPrefetchThreshold": "0",
        "afmPrimaryID": "string",
        "afmRPO": "0",
        "afmShowHomeSnapshots": "false"
      }
    }
  ]
}
```

2. **GET /filesystems/gpfs0/filesets** : The default keyword `:none:` is used in the GET request and this call returns only the key fields as shown in the following example:

```
{
  "filesets": [
    {
     "config": {
        "filesetName": "(fs1)",
        "filesystemName": "gpfs0"
      },
      "links": {
        "self": "https://198.51.100.1:8191/scalemgmt/v2/filesystems/gpfs0/filesets/fs1"
      }
    },
    {
     "config": {
        "filesetName": "afmFset1",
        "filesystemName": "gpfs0"
      },
      "links": {
        "self": "https://198.51.100.1:8191/scalemgmt/v2/filesystems/gpfs0/filesets/afmFset1"
      }
    }
    ]
  "status": {
    "code": 0,
    "message": ""
  }
}
```

3. **GET /filesystems/gpfs0/filesets?fields=state.rootInode,state.id**: Uses the field name "." to query fields that are wrapped in a parent object. This GET call return the following response:

```
{
  "filesets": [
    {
      "config": {
        "filesetName": "(fs1)",
        "filesystemName": "gpfs0"
      },
      "links": {
        "self": "https://198.51.100.1:8191/scalemgmt/v2/filesystems/gpfs0/filesets/fs1"
      },
      "state": {
        "id": 1,
        "rootInode": "latest",
      }
    },
    {
       "config": {
        "filesetName": "afmFset1",
        "filesystemName": "gpfs0"
      },
      "links": {
        "self": "https://198.51.100.1:8191/scalemgmt/v2/filesystems/gpfs0/filesets/afmFset1"
      },
      "state": {
        "id": 2,
        "rootInode": 50692
      }
    }
    ]
  "status": {
    "code": 0,
    "message": ""
  }
}
```

Filter parameter

Use the filter parameter to filter the retrieved objects. For example, all filesets with ID < 2 can be retrieved by using `?filter=status.id<2`. The fields used in the filter parameter are always added to the result set and therefore they do not have to be specified using the field parameter. Filters can be concatenated using",".

The following list provides the supported operators for different field types:

- **String**: "=","!=" using a regular expression as the value is supported. Example: `?filter=config.filesetName=' ' 'fs.*' ' '`
- **Numeric**: "=", "!=", "<", ">" Example: `?filter=status.id<2`
- **Boolean**: "=","!=" using "true" or "false" as the value. Example: `?filter=config.isIndependent=false`

The following examples shows how to use filter parameter in a request:

```
curl -k -u admin:admin001 -X GET -H content-type:application/json
'https://198.51.100.1:443/scalemgmt/v2/filesystems/gpfs0/filesets?filter=status.rootInode>3'
{
  "filesets" : [ {
    "config" : {
      "filesetName" : "fs1",
      "filesystemName" : "gpfs0"
    },
    "status" : {
      "rootInode" : 10752
    }
  } ],
  "status" : {
    "code" : 0,
    "message" : "List of filesets for filesystem"
  }
}
```

Retrieve all file systems with encryption="yes":

```
curl -H content-type:application/json -K -u admin:admin001 -X GET
'https://198.51.100.1:443/scalemgmt/v2/filesystems?filter=encryption=yes'
(
"filesystems" : [ (
"encryption: : "yes"
"filesystemName" : "objfs"
} ],
"status" : {
"code" : 200,
"message" : "List of all file systems"
}
}
```

API responses

API responses provide a response code from the IBM Spectrum Scale management API server and a return code and optional return message from IBM Spectrum Scale.

Return codes and messages

A return code and an optional return message from IBM Spectrum Scale are stored in JSON format in a `status` structure at the end of any returned data. A return code of 0 indicates that IBM Spectrum Scale processed the request successfully:

```
  "status": {
    "message": "",
    "code": 200
  },
```

A response code from the IBM Spectrum Scale management API server describes the result of the API request. The following table lists the response codes:

Table 11. Response codes

Description	Code	Client retry	Return code
Success	20x	None	200: Success (GET or PUT) 201: New resource created (POST) 202: Request accepted (long running command) 204: No content (DELETE)
Client error	4xx	No, the command will continue to fail.	400: Invalid request (format error in request data) 401: Unauthorized request (wrong credentials) 403: Resource not accessible 404: Resource not found (wrong URL) 405: Method not supported for this resource
Server error	5xx	Yes, in most cases	500: Internal server error, retry (system problem) 503: Service not available (the server is busy with other requests or is down)

Data in REST API responses

IBM Spectrum Scale management API responses always include some data in JSON format. The `status` structure with a return code and an optional return message is always present. See the `status` structure at the end of the following JSON example. Also, GET methods return the requested information in JSON format. The following example is from a response to a request to GET information about CES addresses. In JSON, square brackets [ ] indicate an array. The following example contains an array that is named `cesaddresses` that contains two elements, each of which describes one CES address:

```
{
  "cesaddresses": [
     {
      "cesNode": 1,
      "attributes": "",
      "cesAddress": "198.51.100.10",
      "cesGroup": "",
     }
    },
    {
      "cesNode": 2,
      "attributes": "",
      "cesAddress": "198.51.100.14",
      "cesGroup": "",
      }
    }
  ]
  "status": {
    "message": "",
    "code": 200
  }
}
```

Response headers

The same header types appear in all responses from the REST API server. The content type must be `application/json`. The following set of response headers is typical:

```
{
 HTTP/1.1 200 OK
 X-Frame-Options: SAMEORIGIN
 Content-Type: application/json
 Content-Language: en-US
 Content-Length: 210
 Set-Cookie:
      LtpaToken2=yyYwMkAHceggOe74hZhsqf5iCU9r2nl9QBGO9nH7uPm3Mt/vpQfVEHuhwRIWKfq1fi1t8EVn6sZJx7
      +6EpVvUqxqs9PdmIXX28DzU/wwQFxGIMA5a2AuNAFYmZ7lFWqYEEhWq5tx0oQMBHQOL7AbkyR6TUq+F1wzvuZnTe1
      cwu0AYmwZr6WrWdLDj8ZMJ22k95s2PmLbmNMsuSEeSbUFmc1nZdYueieRBgL7QgokS9Ol4lX2gN8YSwWxxljfzCwsed
      MsdYvhawLhYJNA3IkOFnFVJpeopLP4EQtcSdMPXzpX+AQHn/0XQdd6iaWfHppt;
      Path=/; HttpOnly
 Set-Cookie: JSESSIONID=0000SRpFCu8e03i3WXhDyJS2qnn:8bf7532d-fb68-4d4b-90fc-4aa4e3deefd5;
      Path=/; Secure; HttpOnly
 Date: Thu, 16 Feb 2017 15:14:52 GMT
 Expires: Thu, 01 Dec 1994 16:00:00 GMT
 Cache-Control: no-cache="set-cookie, set-cookie2"
}
```

Paging

Paging happens on the response data if more than 1000 objects are returned by the query to avoid server overload. Every response object contains an object called "paging", which contains a link which can then be used to retrieve the next set of objects, if available.

The URL to retrieve the next page can then be found under “paging”→ “next” as shown in the following example:

```
curl -k -u admin:admin001 -X GET -H content-type:application/json
"https://198.51.100.1:443/scalemgmt/v2/filesystems/gpfs0/filesets?filter=filesetName='''myFset1.*'''"
{
  "filesets" : [ {
    "config" : {
      "filesetName" : "myFset134",
      "filesystemName" : "gpfs0"
    },
"status" : {}
}, {
"config" : {
      "filesetName" : "myFset133",
      "filesystemName" : "gpfs0"
"status" : {}
} ],
"paging": {
"next" : "https://198.51.100.1:443/scalemgmt/v2/filesystems/gpfs0/filesets?filter=filesetName='''myFset1.*'''"
    },
  "status" : {
    "code" :200,
    "message" : "List of filesets for filesystem"
  }
}
```

List of IBM Spectrum Scale management API commands

The following table lists the currently supported operations for each type of resource and resource element. The first and second columns list the resource type and the operation. The third column lists the resource and resource element terms that appear in the HTTPS request. The braces {} are a notation for indicating a resource element in text and do not appear in the actual HTTPS request. The fourth column lists the HTTP method that is called to do the operation. A footnote indicates a request that requires HTTP parameters.

Table 12. Operations supported for resources and resource elements in API version 2

Resource type	Operation	Resource/{element}	Method
CES addresses	Get information about CES addresses.	/ces/addresses	GET
	Get information about a CES address.	/ces/addresses/{cesAddress}	GET
CES services	Get information about CES services.	ces/services	GET
	Get information about a CES service.	/ces/services/{service}	GET
Cluster	Get information about the current cluster.	/cluster	GET
Config	Get the current configuration attributes.	/config	GET
File systems	Get information about file systems in the cluster.	/filesystems	GET
	Get information about a file system.	/filesystems/{filesystemName}	GET
ACLs	Get access control list of a file or directory in a file system.	/filesystems/{filesystemName}/acl/{path}	GET
	Get access control list for a file or directory in a file system.	/filesystems/{filesystemName}/acl/{path}	PUT
Disks	Get details about disks that are part of a file system.	/filesystems/{filesystemName}/disks	GET
	Get details about a disk that is part of a file system.	/filesystems/{filesystemName}/disks/(diskName}	GET

Table 12. Operations supported for resources and resource elements in API version 2 (continued)

Resource type	Operation	Resource/{element}	Method
Filesets	Get information about filesets in a file system.	/filesystems/{filesystemName}/filesets	GET
	Get information about a fileset.	/filesystems/{filesystemName}/filesets/{filesetName}	GET
	Create a fileset.	/filesystems/{filesystemName}/filesets	POST
	Modify a fileset.	/filesystems/{filesystemName}/filesets/{filesetName}	PUT
	Delete a fileset.	/filesystems/{filesystemName}/filesets/{filesetName}	DELETE
	Link an existing fileset.	/filesystems/{filesystemName}/filesets/{filesetName}/link	POST
	Unlink a fileset	/filesystems/{filesystemName}/filesets/{filesetName}/link	DELETE
Quotas	Get information about quota defined at the fileset level.	/filesystems/{filesystemName}/filesets/{filesetName}/quotas	GET
	Set quota limits at the fileset level.	/filesystems/{filesystemName}/filesets/{filesetName}/quotas	POST
	Get information about quota defined at the file system level.	/filesystems/{filesystemName}/quotas	GET
	Set quota limits at the fileset level.	/filesystems/{filesystemName}/quotas	POST
Owner	Get details of the owner of the file or directory in a file system.	/filesystems/{filesystemName}/owner/{path}	GET
	Set owner of the file or directory in a file system.	/filesystems/{filesystemName}/owner/{path}	PUT
Information	Get information about the IBM Spectrum Scale management API.	/info	GET
Jobs	Get details about asynchronous jobs.	/jobs	GET
	Get details about an asynchronous job.	/jobs/{jobID}	GET
	Cancel a specific asynchronous job.	/jobs/{jobId}	DELETE
NFS exports	Get information about NFS exports configured in the system.	/nfs/exports	GET
	Get information about an NFS export configured in the system.	/nfs/exports/{exportPath}	GET
	Create an NFS export.	/nfs/exports	POST
	Modify an existing NFS export.	/nfs/exports/{exportPath}	PUT
	Delete an NFS export.	/nfs/exports/{exportPath}	DELETE
Nodes	Get information about nodes in the cluster.	/nodes	GET
	Get information about a node.	/nodes/{nodeName}	GET
	Add one or more nodes in the cluster.	/nodes	POST
	Delete one or more nodes from the cluster.	/nodes/{name}	DELETE

Table 12. Operations supported for resources and resource elements in API version 2 (continued)

Resource type	Operation	Resource/{element}	Method
Node health	Get details about system health events that are reported on a node.	/nodes/{name}/heath/events	GET
	Get details about system health states of the events that are reported in a node.	/nodes/{name}/heath/states	GET
Node class	Get details of the node classes.	/nodeclasses	GET
	Create a node class.	/nodeclasses	POST
	Delete a specific node class.	/nodeclasses/{nodeclassName}	DELETE
	Get details of a specific node class.	/nodeclasses/{nodeclassName}	GET
	Make changed to a specific node class.	/nodeclasses/{nodeclassName}	PUT
NSDs	Gets information about NSDs that are part of the system.	/nsds	GET
	Gets information about an NSD.	/nsds/{nsdName}	GET
Snapshots	Get information about fileset snapshots.	/filesystems/{filesystemName}/ filesets/{filesetName}/snapshots	GET
	Get information about a fileset snapshot.	/filesystems/{filesystemName}/ filesets/{filesetName}/snapshots/ {snapshotName}	GET
	Get information about file system snapshots.	/filesystems/{filesystemName}/ snapshots	GET
	Get information about a file system snapshot.	/filesystems/{filesystemName}/ snapshots/{snapshotName}	GET
	Create a fileset snapshot.	/filesystems/{filesystemName}/ filesets/{filesetName}/snapshots	POST
	Create a file system snapshot.	/filesystems/{filesystemName}/ snapshots	POST
	Delete a fileset snapshot.	/filesystems/{filesystemName}/ filesets/{filesetName}/snapshots	DELETE
	Delete a file system snapshot.	/filesystems/{filesystemName}/ snapshots	DELETE
AFM peer snapshots	Create a peer snapshot	/filesystems/{filesystemName}/ filesets/{filesetName}/psnaps	POST
	Delete a peer snapshot	/filesystems/{filesystemName}/ filesets/{filesetName}/psnaps/ {snapshotName}	DELETE
SMB shares	Get information about SMB shares.	smb/shares	GET
	Get information about an SMB share.	/smb/shares/{shareName}	GET
	Create an SMB share.	/smb/shares/{shareName}	POST
	Modify an existing SMB share.	/smb/shares/{shareName}	PUT
	Delete an SMB share.	/smb/shares/{shareName}	DELETE
Performance monitoring	Get performance data.	/perfmon/data	GET
Thresholds	Get list of threshold rules.	/thresholds	GET
	Create a threshold rule.	/thresholds	POST
	Delete an existing threshold rule.	/thresholds/{name}	DELETE
	Get details of a threshold rule	thresholds/{name}	GET

Table 12. Operations supported for resources and resource elements in API version 2 (continued)

Resource type	Operation	Resource/{element}	Method
For more information about the commands, see *REST API commands* in the *IBM Spectrum Scale: Command and Programming Reference* guide.			

Table 13. Operations supported for resources and resource elements in API version 1

Resource type	Operation	Resource/{element}	Method
CES addresses	Get information about CES addresses.	/cesaddresses	GET
	Get information about a CES address.	/cesaddresses/{cesAddress}	GET
CES services	Get information about CES services.	/cesservices	GET
	Get information about a CES service.	/cesservices/{service}	GET
Cluster	Get information about the current cluster.	/cluster	GET
Config	Get the current configuration attributes.	/config	GET
Filesets	Get information about filesets in the cluster.	/filesets[1]	GET
	Get information about a fileset.	/filesets/{filesetName}[1]	GET
	Create a fileset.	/filesets	POST
	Modify a fileset.	/filesets/{filesetName}	PUT
	Delete a fileset.	/filesets/{filesetName}[1]	DELETE
File systems	Get information about file systems in the cluster.	/filesystems	GET
	Get information about a file system.	/filesystems/{filesystemName}	GET
REST API	Get information about the REST API server.	/info	GET
Nodes	Get information about nodes in the cluster.	/nodes	GET
	Get information about a node.	/nodes/{nodeName}	GET
Quotas	Get information about quotas.	/quotas[1]	GET
	Create or set a quota.	/quotas	POST
Snapshots	Get information about snapshots.	/snapshots[1]	GET
	Get information about a snapshot.	/snapshots/{snapshotName}[1]	GET
	Create a snapshot.	/snapshots	POST
	Delete a snapshot.	/snapshots/{snapshotName}[1]	DELETE
[1] This operation requires parameters in the HTTP request. For more information, see the description of the command under the topic *REST API commands* in the *IBM Spectrum Scale: Command and Programming Reference* guide.			

The following table lists the HTTP methods that are currently supported for each resource type. A check mark ✓ indicates that a method is supported:

Table 14. HTTP methods supported for each type of resource in API version 2

Object	DELETE	GET	POST	PUT
ces/addresses		✓		
ces/services		✓		
cluster		✓		
config		✓		
filesets	✓	✓	✓	✓
filesystems		✓		

Table 14. HTTP methods supported for each type of resource in API version 2 (continued)

Object	DELETE	GET	POST	PUT
info		✓		
jobs	✓	✓		
nfs/exports	✓	✓	✓	✓
nodes		✓		
nodes/health		✓		
node class	✓	✓	✓	✓
nsds		✓		
quotas		✓	✓	
snapshots	✓	✓	✓	
peer snapshots	✓		✓	
smb/shares	✓	✓	✓	✓
performance monitoring		✓		
thresholds	✓	✓	✓	

Table 15. HTTP methods supported for each type of resource in API version 1

Object	DELETE	GET	POST	PUT
cesaddresses		✓		
cesservices		✓		
cluster		✓		
config		✓		
filesets	✓	✓	✓	✓
filesystems		✓		
info		✓		
nodes		✓		
quotas		✓	✓	
snapshots	✓	✓	✓	

Introduction to Cloud services

This topic provides a brief introduction to the Cloud services available in IBM Spectrum Scale.

Cloud services have the following two components:

- Transparent cloud tiering
- Cloud data sharing

Transparent cloud tiering is a feature of IBM Spectrum Scale that provides a native cloud storage tier. It allows data center administrators to free up IBM Spectrum Scale storage capacity, by moving out cooler data to the cloud storage, reducing capital and operational expenditures. Tiering can also be used to archive an extra copy of your data using pre-migration, a function that copies data rather than moving it. The Transparent cloud tiering feature leverages the existing ILM policy query language semantics available in IBM Spectrum Scale, and administrators can define policies to tier data to cloud storage. On an IBM Spectrum Scale cluster with multiple storage tiers configured, this external cloud storage can be used as the cooler storage tier to store infrequently accessed data from a cool storage pool. For performance reasons, it is recommended not to move any active or hot data to this external storage pool, as it drives excessive data traffic on the Transparent cloud tiering which in turn can cause delays, leading to problems like application timeouts. Copying hot data to the cloud by using the pre-migration function is acceptable as long as the file is not likely to be updated or deleted soon.

Cloud data sharing is an IBM Spectrum Scale Cloud services that allows a way to set up sharing between IBM Spectrum Scale and various types of object storage, including IBM Softlayer and IBM Cloud Object Storage. Furthermore, for export of data to object storage the service can be invoked from an ILM policy to allow for periodic sharing of data based on when the policy is run. For import, you can specify a list of files that you can use to move object storage (by using your IBM Cloud data sharing service) into the IBM Spectrum Scale cluster.

This is useful when data needs to be distributed across multiple domains or when large amounts of data needs to be transferred.

You can associate your Cloud services file systems and file sets with a Cloud services node group. Up to four node groups are supported in a cluster to allow for scaling (of the solution). For each file system or file set, you can associate up to two cloud storage tiers. For example, you should define an IBM Cloud Object Storage service locally to act as your cool tier. You should use IBM's Public Cloud Object Storage (on IBM Cloud) for your cold tier. The diagram below is an example of a Cloud services configuration that is on a cluster.

Figure 18 illustrates these features.

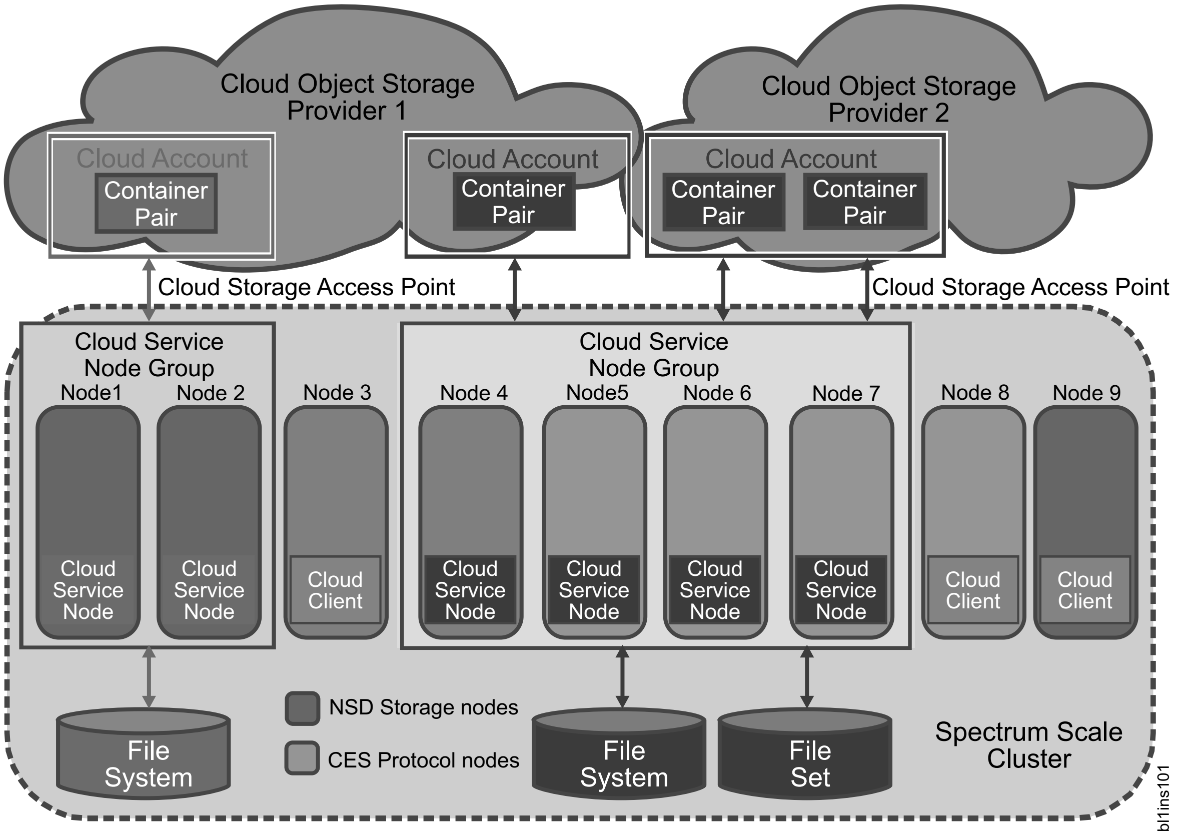


Figure 18. Transparent cloud tiering and Cloud data sharing features

Note:

- For performance reasons, the median file size to be migrated to the cloud tier must be greater than 1 MB. Migration is supported for file size less than one MB, but performance is slower because of the overhead that is associated with small files.
- For Transparent cloud tiering, data on the cloud object storage is opaque and cannot be accessed directly by applications. All I/O operations must happen through your IBM Spectrum Scale system.
- Transparent cloud tiering works with IBM Spectrum Scale on multi-site stretch clusters to allow continued access to cloud storage data even if failure of an entire site.
- For applications that require tiering of data that scales beyond 100 million files per file system, you can create extra containers on cloud storage to hold your files. For applications that require more I/O bandwidth than a single maximum node group (4 nodes) is capable of, you can create extra node groups.

  Note: You must create a new container when the existing container has approximately 100,000,000 files.

Unsupported use cases

Transparent cloud tiering does not support the following use cases:

- Using Transparent cloud tiering to migrate/recall hot (active) data
- Using Transparent cloud tiering as a backup mechanism
- Using Transparent cloud tiering to restore data in disaster recovery scenarios.

Note: IBM Spectrum Protect cloud-container storage pools can be used to back up data to cloud storage providers.

Note: To enable Cloud services 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. 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.

How Transparent cloud tiering works

This topic describes how you can use the Transparent cloud tiering (which tiers files to object storage) feature works in the IBM Spectrum Scale cluster.

Transparent cloud tiering is a IBM Spectrum Scale feature that allows the IBM Spectrum Scale cluster to tier files to object storage. The Transparent cloud tiering service stores a file as two objects out in the cloud in two separate containers. These containers are also called vaults or buckets depending on which cloud storage vendor you use. One object holds the file's data and the other object holds the file's metadata. The metadata contains items such as ACLs, extended attributes, and other information that allows Transparent cloud tiering to do a full restore of the file from the cloud tier.

When a file is migrated to the cloud, the metadata of the file is retained in the IBM Spectrum Scale file system. The only reason for copying the metadata to the cloud is to manage scenarios where a file is destroyed on the cluster for some reason (such as accidental deletion) and the file needs to be restored from the cloud.

For the usual case, only the file's data is recalled from the cloud. You can tier the files by migrating the data to the cloud. This means that the data is no longer on the IBM Spectrum Scale cluster and is stored only on the cloud. Such files are considered to be in the non-resident state. Alternatively, you can pre-migrate the data to the object storage so that the data is retained in the IBM Spectrum Scale cluster, but is also copied out to object storage so that the data is on both tiers. Such files are considered to be in the co-resident state.

Transparent cloud tiering stores information about files that are migrated to the cloud tier in a database called *cloud directory*. A separate database or cloud directory is kept for every Transparent cloud tiering container pair (the pair of containers that hold the file data and metadata). This database contains a list and versions of all files that are migrated to the cloud. The metadata contains items such as ACLs, extended attributes, and other information that allows Transparent cloud tiering to do a full restore of the file from the cloud tier in the future.

You can use a policy to configure Transparent cloud tiering to automatically remove files that are deleted in the cluster from the cloud tier. There is a background task that periodically runs and looks for files to clean up. The removal is not immediate, but you can use the policy to set the number of days to retain the file in the cloud tier (prior to removal).

Note: Do not rely on this background process to strictly comply with any deletion deadlines that are contractually guaranteed.
In such cases, you must run the **reconcile** command to make sure that the files that are deleted are guaranteed to be deleted on the next deletion policy run after the reconcile.

Note: Because of high overhead, it is recommended that you not run the **reconcile** command to manage deletion unless it is required for compliance reasons.

Additionally, if a file is migrated and only the metadata is changed later, a subsequent migration copies only the metadata to the cloud. This metadata references the original data object.

Note: Transparent cloud tiering data is migrated to the cloud in a way that is not designed for direct use in the cloud. If data must be consumed in the cloud, you must consider the Cloud data sharing service.

How Cloud data sharing works

This topic describes how the Cloud data sharing feature works in the IBM Spectrum Scale cluster.

Cloud data sharing allows you to import data from object storage into your IBM Spectrum Scale file system using the **import** command. You can also export data to the cloud for use in the cloud by using the **export** command. Optionally, there is a manifest that can be built that allows those sharing to very quickly ascertain what files are being shared.

Cloud data sharing is different from Transparent cloud tiering because it is meant as a means of sharing and distributing data, whereas with Transparent cloud tiering the data that is migrated to object storage is concealed so it can be consumed only by recalling the data back into the IBM Spectrum Scale system.

Note: There is no implicit consistency link that is established by the export or import of a file. The import or export is just a transfer of a file. Once the transfer is complete, if the original file is changed at one place it has no bearing on its copies. And if a copy of a file is changed or deleted it has no bearing on the original.

How exporting file system data to cloud storage works

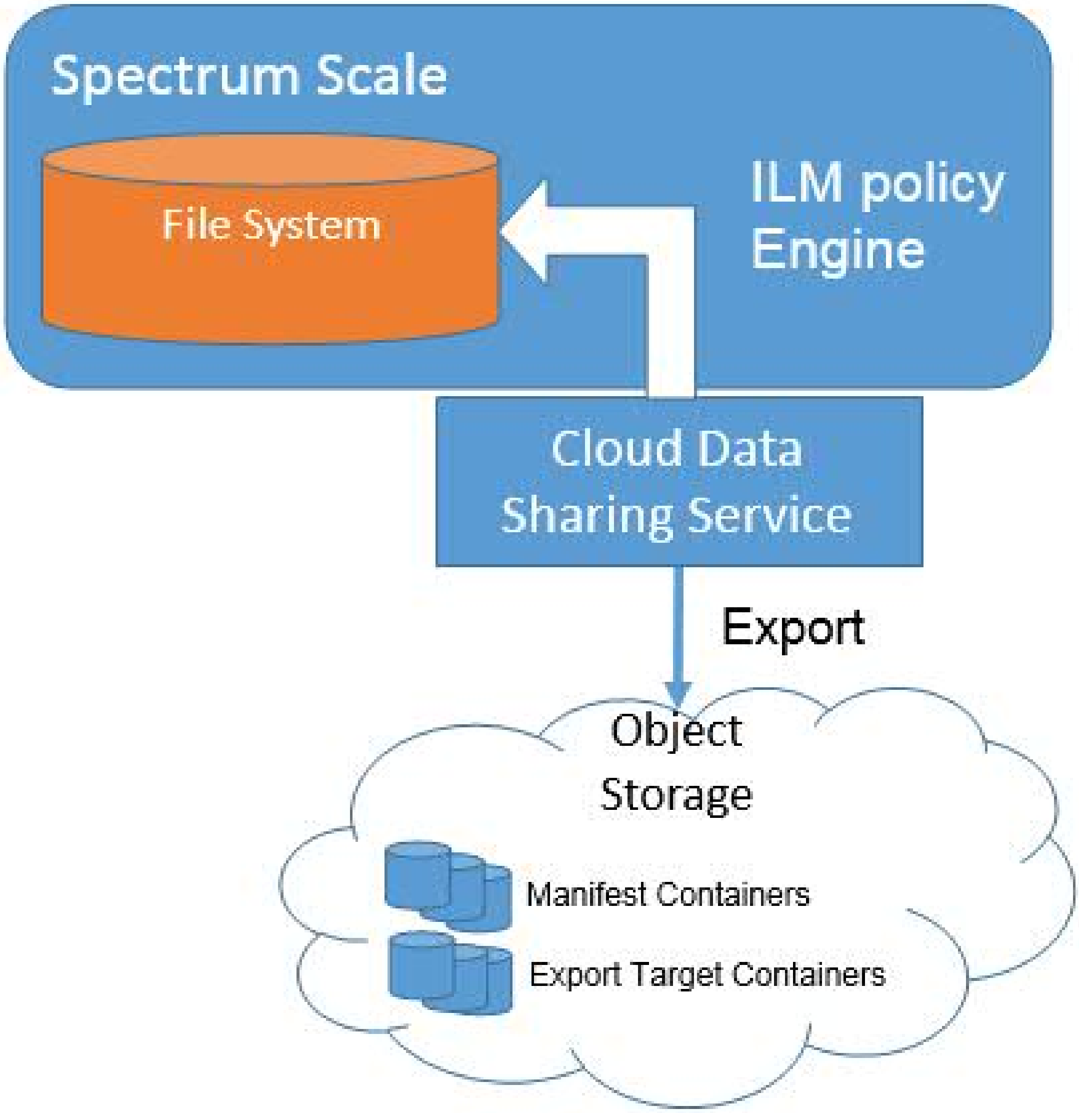


Figure 19. Exporting file system data to a cloud storage tier

You can use the Cloud data sharing service to export data to the cloud. Data can be exported manually, but typically it is done leveraging the ILM Policy Manager that comes with IBM Spectrum Scale. When data transfer requests are generated, they are distributed evenly between all nodes in the Cloud services node group providing the Cloud data sharing service.

Also, optionally as a part of the export request, a list of what was sent is also put in a manifest file that can be used to track what was exported. Export is very flexible because for each command the target container can be specified. Any container accessible by the given cloud account can be used. For each **export** command, you can also specify the target manifest file. The manifest file is not automatically stored in the cloud storage but rather remains stored within the Cloud data sharing service. The manifest is exported on demand to the desired target location.

How importing object storage data into the Spectrum Scale file system works

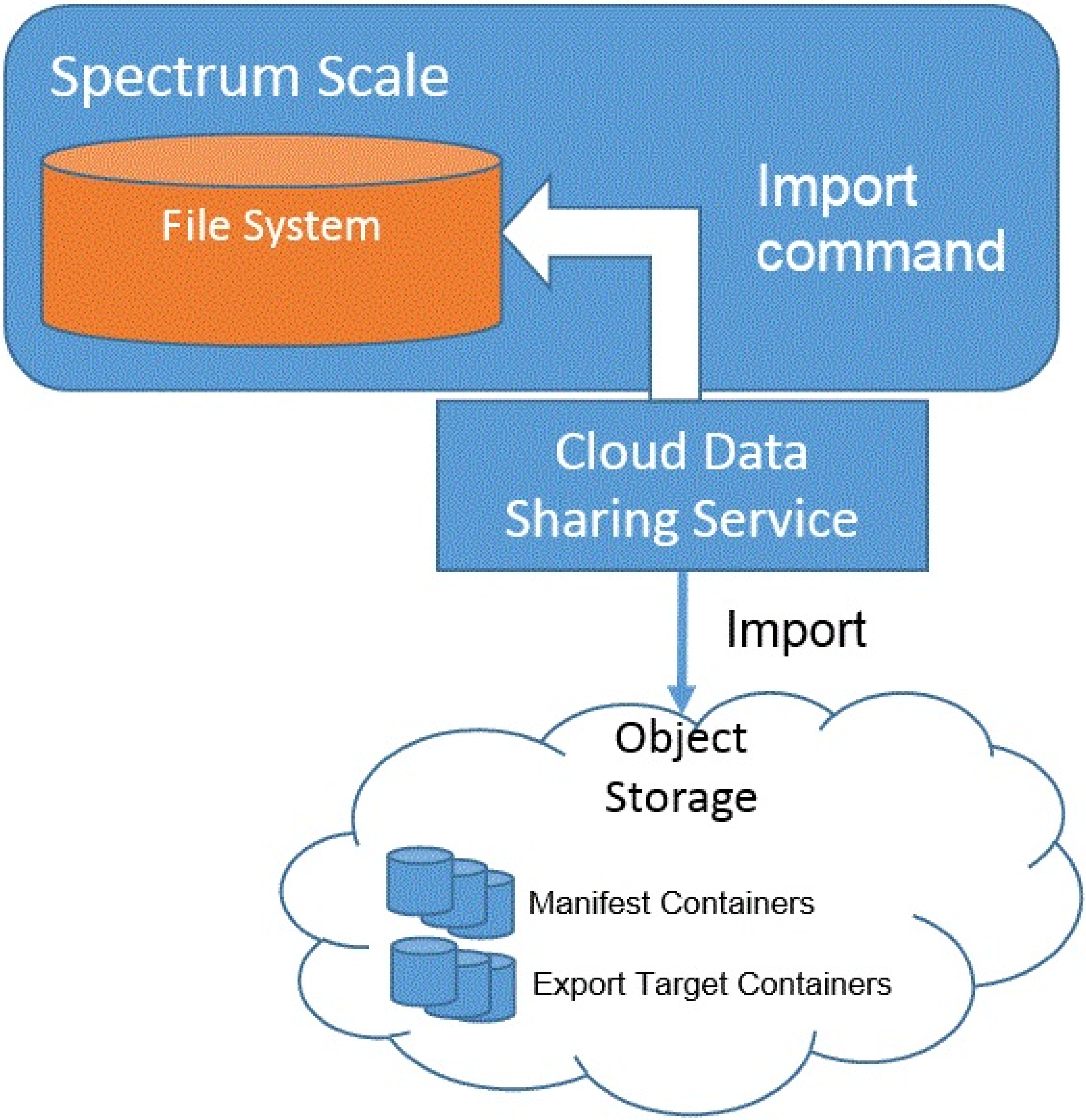


Figure 20. Importing object storage data into the file system

You can import data directly by providing a list of objects to be imported in the command itself or by providing a pointer to a manifest that contains the list of objects to be imported. The files are imported and placed in a target directory in the file system. Since the files are not yet present in the file system, there is no way to use the policy engine to initially import the files. You can also import the stub of the file and then use the policy engine to look at the file stub information that you use to import the desired data subset. You can delete the stubs that you are not interested in.

Note: The import of the stub only works if the data on the object storage that is being imported was originally created using the IBM Spectrum Scale export service so that the stub has the appropriate format. (Native object storage stubs are not mapped or normalized as of yet).

How to be aware of and exploit what data is shared

A manifest might be kept which lists information on objects that are in cloud storage available for sharing. When the Cloud data sharing service exports files to the object storage, it can be directed to add entries on what it exported to a manifest. Other applications that generate cloud data can also generate a manifest. The location of where the manifest is stored and when it is exported to be looked at is the decision of the application – it is not automatic. A typical time to export the manifest to object storage is in the same cron job as the policy invocation immediately following the policy execution completion so that it represents all the files that are exported by that policy run.

The `mmcloudmanifest` manifest utility program is available that can read a manifest and provide a comma-separated value (CSV) output stream of all the files in the manifest, or a subset of those files as provided by some simple filtering. This manifest utility is written in Python and can run separately from the cloud data sharing service most anywhere python can run. The utility is also built into cloud data sharing and can be used to get input for import operations.

Currently, there is no built-in way to do asynchronous notification or to have a database that tracks all updates at a scale larger than is reasonable with manifest files, but it is possible to add notification in the script where the export policy is invoked. Also, manifests can be readily consumed by a database (for example the Elastic Search, Logstash, Kibana ELK Stack) since the data structure is very simple. On larger deployments, such an approach with asynchronous notification that is built into the scripts (and uses a database) is worth consideration.

How a native cloud storage application can prepare a set of objects for sharing

There is a simple way for native cloud storage applications or other generators of cloud object data to generate a manifest. The manifest utility can build the manifest from a list of objects that it is passed.

How Write Once Read Many (WORM) storage works

This topic describes how you can use Write Once Read Many (WORM) solutions to take advantage of functions that are offered by IBM Spectrum Scale, Transparent cloud tiering, and IBM Cloud Object Storage.

WORM (Write Once-Read Many) storage solutions leverage IBM Spectrum Scale immutability, Transparent cloud tiering, and IBM Cloud Object Storage locked vaults. Essentially, you can create the immutable file sets on the IBM Spectrum Scale file system, and you can set files as immutable either through IBM Spectrum Scale commands or through POSIX interface. You can also set a retention period on the immutable files.

A locked vault on the IBM Cloud Object Storage cannot be deleted by the IBM Cloud Object Storage administrator, and its Access Control Lists (ACLs) cannot be changed. Additionally, you cannot rename it or enable the proxy settings. RSÁ private key and private certificate are used to create and access the locked vaults. Transparent cloud tiering is configured with the RSA private key and private certificate to create locked vaults. Once configured, you can use configured private keys and certificates to use REST APIs against the Accesser® nodes.

You can perform migrate and recall operation on immutable files. Since the immutable files cannot be deleted, the data on the IBM Cloud Object Storage locked vaults is only deleted according to locked vault deletion policy, which provides a WORM solution for compliance archival.

The following diagram provides an overview of the feature:

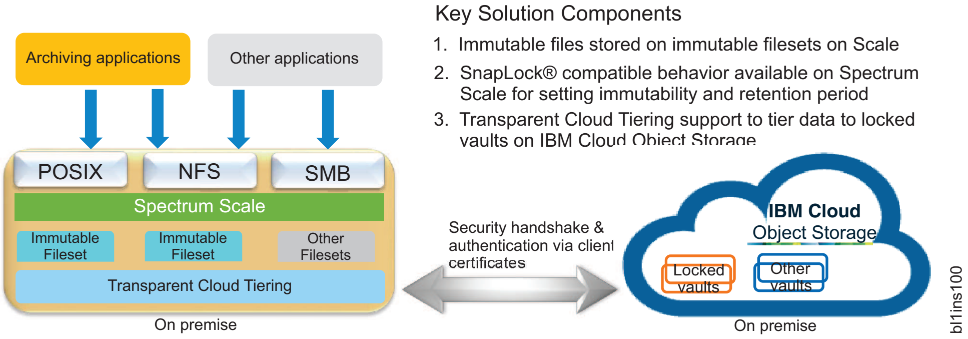

Key Solution Components
1. Immutable files stored on immutable filesets on Scale
2. SnapLock® compatible behavior available on Spectrum Scale for setting immutability and retention period
3. Transparent Cloud Tiering support to tier data to locked vaults on IBM Cloud Object Storage
Archiving applications
Other applications
POSIX
NFS
SMB
Spectrum Scale
Immutable Fileset
Immutable Fileset
Other Filesets
Transparent Cloud Tiering
On premise
Security handshake & authentication via client certificates
IBM Cloud Object Storage
Locked vaults
Other vaults
On premise
bl1ins100

Figure 21. WORM storage overview

Supported cloud providers

This topic describes the cloud providers that the IBM Spectrum Scale supports.

The Cloud services use the following cloud providers:

- IBM Cloud Object Storage (version 3.7.3.2 and above)
- IBM Cloud Object Storage on IBM SoftLayer® and IBM Bluemix®.
- OpenStack Swift (Swift version 2.3 and above and Keystone version 2 only)
- Amazon Web Services S3
- Swift3 on OpenStack Swift version 1.1 and above

Note: OpenStack Swift is supported only with Keystone authentication.

For the delete functionality to work in Swift, verify that the Bulk middleware to be in pipeline, as follows:

```
vim /etc/swift/proxy-server.conf
[pipeline:main] pipeline = healthcheck cache bulk authtoken keystone proxy-server
```

Note: For information on Swift DLO configuration, see the *Known limitations of Cloud services* topic in the *IBM Spectrum Scale: Administration Guide.*

Interoperability of Transparent cloud tiering with other IBM Spectrum Scale features

This topic describes the interoperability of Transparent cloud tiering with other IBM Spectrum Scale features.

IBM Spectrum Archive (LTFS) and HSM

Running IBM Spectrum Archive and Transparent cloud tiering on the same file system is not supported.

However, both HSM and Transparent cloud tiering can co-exist on the same systems if they are configured with different file systems.

It is advised not to enable the Data Management API (DMAPI) on the file system that is managed by the Transparent cloud tiering.

AFM

Running Transparent cloud tiering service on AFM Gateway nodes is not supported.

Data from AFM or AFM DR filesets cannot be accessed by Transparent cloud tiering.

Using Transparent cloud tiering on AFM home is not supported.

Multi-clusters

Multi-cluster support is limited to remotely mounted clients.

File Placement Optimizer (FPO)

Transparent cloud tiering is supported on FPO cluster. Recalled files might not have optimal placement.

IBM Spectrum Scale Object

Transparent cloud tiering can be configured on Object file sets.

Snapshots

Transparent cloud tiering should not be used to migrate/recall snapshots. Space that is contained in snapshots is not freed if a file is migrated to cloud object storage.

Sparse files

Transparent cloud tiering can be used to migrate/recall sparse files. However, sparseness is not maintained and on recall, full blocks are allocated.

Encryption

Transparent cloud tiering can be used with Scale file system level encryption feature (available in the Advanced and Data Management Editions). However, all the data that is migrated to the cloud object storage is migrated with the key that is configured for Transparent cloud tiering. Essentially, when data is read from file system, the files get decrypted, and encrypted again at user space by Transparent cloud tiering and pushed into the cloud storage.

Compression

Transparent cloud tiering can be used along with Scale file system level compression capability. Essentially, when data is read from the file system, the files are uncompressed, Transparent cloud tiering push uncompressed, but by default encrypted, files onto the cloud storage.

CES (protocol services)

Transparent cloud tiering can co-exist with active or inactive NFS, SMB, or Object services on the CES nodes.

Spectrum Protect (TSM)

Beginning with IBM Spectrum Scale V4.2, for the file systems that are managed by an HSM system, ensure that hot data is backed up to TSM by using the **mmbackup** command, and as the data gets cooler, migrate them to the cloud storage tier. This ensures that the **mmbackup** command has already backed up the cooler files that are migrated to the cloud.

Elastic Storage Server (ESS)

Transparent cloud tiering cannot be deployed directly on ESS nodes. However, it can be deployed on X86 protocol nodes that can be attached to ESS. On a Power Linux cluster, a few X86 protocol nodes can be added and Transparent cloud tiering can be configured on those nodes.

Mixed-node cluster configuration

Transparent cloud tiering service runs on x86 and Power LE Linux nodes. Transparent cloud tiering does not run on Windows, IBM Z, or Power BE Linux node. No mixed-node cluster support with Windows or IBM Z. Both x86 Linux and Power Linux nodes can initiate migrations/recalls of data, and these nodes can initiate a transparent recall on file access.

Transparent cloud tiering is supported for use only in IBM Spectrum Scale clusters and on any associated remotely mounted clients, with Intel x86 Linux and Power LE Linux nodes. Use of Windows, PowerPC® Big Endian, Linux on Z or AIX nodes that access file systems, where Transparent cloud tiering is used (either within the cluster or via a remotely mounted client), is untested and unsupported. Inclusion of such nodes is allowed only if there is no possibility that they will ever access Transparent cloud tiering; for example, such nodes are tested and supported for PowerPC Big Endian nodes in Elastic Storage Server (ESS) BE & LE servers, where no user applications are supported on Elastic Storage Server (ESS).

SELinux

Supported

SOBAR
Running SOBAR and Transparent cloud tiering on the same file system is not supported.

IPV6 Support
Not supported

Linux GNOME and Windows Explorer
Supported. GNOME/Windows Explorer opens all files and reads the first few Kilobytes to obtain data for displaying the icon view of the file. There is an option available to cache the start of every file to prevent a recall storm in this case.

IBM Spectrum Scale Stretch Clusters

This service can be used in conjunction with stretch clusters. Cloud services node classes can be set up all on one site or, most usually, can be split across sites to allow for continued access to cloud data even through failure of an entire site.

installation toolkit
Not supported.

Linux on Power8 Little Endian
Supported.

Note: For information on known limitations, see the *Known Limitations for Transparent cloud tiering* topic in the *IBM Spectrum Scale: Administration Guide*.

Interoperability of Cloud data sharing with other IBM Spectrum Scale features

This topic describes the interoperability of Cloud data sharing with other IBM Spectrum Scale features.

IBM Spectrum Archive (LTFS) and HSM

- Running IBM Spectrum Archive and Cloud data sharing on the same file system is not supported.
- However, both HSM and Cloud data sharing can co-exist on the same systems if they are configured with different file systems.
- It is advised not to enable the Data Management API (DMAPI) on the file system that is managed by the Cloud data sharing.

AFM

- Data from AFM or AFM DR filesets might be accessed by Cloud data sharing.
- Using Cloud data sharing on AFM home is supported.

Multi-clusters
Running Cloud data sharing on multiple clusters to the same or different cloud accounts is supported.

File Placement Optimizer (FPO)

Cloud data sharing is supported on FPO cluster. Transferred files might not have an optimal placement.

IBM Spectrum Scale Object

- Cloud data sharing works only with Unified File Object based IBM Spectrum Scale object storage at this time.
- Running Cloud data sharing in the same file system with another data set is supported.

Snapshots
Cloud data sharing can be used with the snapshot function to export snapshot files to the cloud since it is simply doing the file replication.

Sparse files

Cloud data sharing can be used to import or export sparse files. However, sparseness is not maintained and on re-import, full blocks are allocated.

Encryption

Cloud data sharing can be used with the IBM Spectrum Scale file system level encryption feature (available in the Advanced Edition). However, all the data that is exported to the cloud object storage is unencrypted or if cloud data sharing encryption is enabled it is exported with the encryption key configured for Cloud services. Essentially, when data is read from file system, the files get decrypted, and may be encrypted again at user space by Cloud data sharing and pushed into the cloud storage.

Compression

Cloud data sharing can be used along with IBM Spectrum Scale file system level compression capability. Essentially, when data is read from the file system, the files will get uncompressed, Cloud data sharing will push uncompressed file onto the cloud storage.

CES (protocol services)

Cloud data sharing can co-exist with active or inactive NFS, SMB, or Object services on the CES nodes.

Spectrum Scale stretched cluster

Spectrum Scale stretched cluster and Cloud data sharing on the same file system is supported.

Spectrum Protect (TSM)

IBM Spectrum Scale file systems backed up using Spectrum Protect may leverage Cloud data sharing.

Elastic Storage Server (ESS)

Cloud data sharing cannot be deployed directly on ESS nodes. However, it can be deployed on X86 Protocol (CES) nodes that can be attached to ESS that effectively allows sharing of ESS data.

Mixed-node cluster configuration

- Cloud data sharing service runs only on x86 and Power Linux nodes. Cloud data sharing is not supported to be deployed on Power Linux,Windows, or IBM Z. However, it can co-exist in a Linux cluster that contains both x86 and Power nodes, as long as the Power nodes are not running the Cloud services.
- No mixed-node cluster support with Windows or IBM Z. Only x86 Linux nodes can initiate migrations/recalls of data and only x86 Linux nodes can initiate a transparent recall on file access.

SELinux

Supported

SOBAR

Running SOBAR and Cloud data sharing on the same file system is supported.

IPV6 Support

Not supported

Linux on Power8 Little Endian

Supported

installation toolkit

Not supported.

Note: For information on known limitations, see the *Known Limitations for Cloud services* topic in the *IBM Spectrum Scale: Administration Guide*.

Introduction to file audit logging

File audit logging captures file operations on a file system and logs them to a retention enabled fileset.

Each file operation is generated as a local event on the node that serves the file operation. These events are published to a distributed multinode message queue from which they are consumed to be written into the fileset. These events are called lightweight events. Lightweight events occur at the file system level and are protocol agnostic, which means that they capture all access to a monitored file system from protocol exports to even root access that occurs directly on nodes. For more information, see "Producers in file audit logging." The most common file operations such as open, close, destroy (delete), rename, unlink, create, remove directory, extended attribute change, ACL change, and GPFS attribute change are the events that are captured. Events are created in a highly parsable, JSON formatted string as they are written to the designated fileset. For each file system enabled for file audit logging, a fileset is designated where the audit logs will go. This fileset keeps the logs currently being written to in append only mode and, as it rotates to a new log file, compresses the old log file and makes it immutable for the retention period. Configurable options for file audit logging filesets include the device where they are mounted, their name, and the retention period in days. A separate file audit logging fileset can store the audit log for each file system or they can be shared among multiple file systems that are being audited. File audit logging is integrated into the system health infrastructure, so alerts are generated for elements of the message queue and the processes that consume the events and create the audit logs.

For more information about file audit logging, see *Monitoring file audit logging* in *IBM Spectrum Scale: Administration Guide*.

Producers in file audit logging

Producers create lightweight events and publish them to a list of message queue servers/brokers.

Lightweight events are generated when file operations are served locally on a node in the data path. These lightweight events are generated as node local events that are then published in the multinode message queue. Producers publish file access events to the message queue. They reside in the GPFS daemon and are activated and configured by IBM Spectrum Scale policy code. Once the events are generated, they are published to a specific topic in the message queue that has a 1:1 correspondence to the file system for which the event is generated.

The message queue in file audit logging

The message queue collects and stores events that are published by the producers running on nodes in an IBM Spectrum Scale cluster.

Messages are added to the queue as part of registered topics. In the case of file audit logging, a unique topic is created for each file system that is being audited. The topic provides a way to track messages that are being added by producers for a specific file system and later read via a consumer group. A file audit logging topic is composed of four items:

```
<Audit_Device_Minor_Number>_<Cluster_ID>_<Gen_Number>_audit
```

- **Audit_Device_Minor_Number**: A minor number represents the file system device. This number is used because it will not change even if the device name of the file system changes.
- **Cluster_ID**: The cluster ID represents the cluster where file audit logging is configured and enabled on one or more file system devices.
- **Gen_Number**: A generation number allows file audit logging to be disabled and later re-enabled on the same file system even if the message queue server containing one of the replicas of the data associated with the topic is unavailable.
- **audit**: Audit notifies the user that the topic is being used for file audit logging.

Other message queue components consist of the message queue servers, Zookeepers, and the actual message queue data.

A message queue server provides an interface for producers to write messages to the queue and for consumers to read messages from the queue. When enabling file audit logging for a file system device, the default is to set the list of message queue servers to be the protocol nodes in the IBM Spectrum Scale cluster. If protocol nodes do not exist or if the user wishes to specify their own list of nodes that will comprise the message queue servers, the **-N** option of the **mmmsgqueue** command can be used. This option is not available when utilizing the Install Toolkit.

Zookeeper nodes are paired with Linux quorum nodes in the IBM Spectrum Scale cluster. Zookeepers provide configuration and consensus services for the message queue. To allow the message queue system to be enabled (and therefore to enable file audit logging), there must be a minimum of three Linux quorum nodes within the IBM Spectrum Scale cluster.

Data in the message queue is retained on local storage on each of the message queue server nodes. The replication factor is three by default, which means that there are three copies of each message. The messages in the queue are retained in log files stored on disk. When enabling file audit logging for the file system device, the local space available on all of the message queue server nodes is queried to ensure that at least the minimum required amount of free space is available. The critical limit is 5 GB per file system that is being audited and a minimum threshold of 10 GB (a suggested amount of free space). File audit logging cannot be enabled if the minimum space requirements are not met. A warning will be given if there is less than 10 GB of free space for message queue server node.

Consumers in file audit logging

Consumers subscribe to a given topic on the message queue, read messages, and write them to a designated fileset within a specified file system.

A new node class is created for consumers (similar to the new node classes for the message queue servers/brokers and Zookeepers). The consumer node class consists of the same nodes that are used by the message queue servers/brokers. For every file system that is being audited, a separate consumer runs on each node within the consumer node class. All consumers associated with the same file system cluster together to form a consumer group. If one fails, then the remaining consumers can perform the work of the failed consumer until it is available again.

For more information, see *Stopping consumers in file audit logging*, *Starting consumers in file audit logging*, and *Monitoring the consumer status* in the *IBM Spectrum Scale: Administration Guide*.

The file audit logging fileset

The audited events are logged in an independent file audit logging fileset.

The user can also supply a previously created fileset as long as it is already in IAM mode non-compliant. This is a requirement so that audit logs are not altered accidentally. By having the file audit logging fileset be IAM mode non-compliant, a retention period can be set on all of the files within the fileset. However, unlike in a compliant IAM mode fileset, the root user can change the expiration date of the individual files if space needs to be cleaned up in the fileset before the retention period expires.

File audit logging records for a given file system are arranged in the following manner:

```
<FS_Mount_Point>/<Fileset_Name>/Topic/Year/Month/Day
```

- **FS_Mount_Point**: The default mount point of the device is designated to hold the file audit logging destination fileset. By default, this is the same as the file system that is being audited. The fileset can reside on the same file system that is being audited or a different one. The user can make that choice using the **mmaudit** command.
- **Fileset_Name**: The fileset name indicates the file audit logging destination fileset. If no other fileset name is given, then the default is `.audit_log`.
- **Topic**: The topic that is associated with the file system that is being audited. It consists of the following structure:

```
<Device_Minor Number>_<Cluster_ID>_<Generation_Number>_audit
```

For more information about this structure, see "The message queue in file audit logging" on page 121.

- **Year/Month/Day**: Details when the individual file that is being written was first created.

Note: The **mmbackupconfig** command does not save the file audit logging enabled state of a file system. However, the IAM noncompliant mode of the file audit logging fileset is saved. Therefore, when restoring configuration information using the **mmrestoreconfig** command, file audit logging will have to be re-enabled on the file system using the **mmaudit** command. You will need to specify **--log-fileset** and possibly **--log-fileset-device**.

File audit logging records

Files that hold the JSON formatted event records.

Files in the file audit logging fileset are named in the following format:

```
auditLogFile_<Consumer_Node_Name >_YYYY-MM-DD_HH:MM:SS
```

When a file audit logging file is created, it is put in append only mode and an expiration date is set for it. The expiration date corresponds to the number of days after the current date given in the retention option of the **mmaudit** command. The default retention period is 365 days, which means that a given file audit logging file is set to expire one year from when it is created.

When auditing events are read by the consumers in the message queue, they are appended to the files in the file audit logging fileset. After approximately 500,000 entries have been made in a file, it is made immutable, GPFS native compression is enabled on it, and a new file with the current date and time is created for the next batch of audit logs. Therefore, new files are created at a rate that corresponds to the file system activity of the device that is being audited.

Important: Since files in the file audit logging fileset are created in the directory based on their creation date, there might be audit records for several days within the same file if there is not much file system activity.
Furthermore, since the events are written to these files in batches, the file audit logging event information might be placed in a previous day's or next day's directory.

For more information, see "JSON attributes in file audit logging."

JSON attributes in file audit logging

Use this information to learn more about the JSON attributes that are associated with the ten events in file audit logging.

LWE_JSON
The version of the record.

path The path name of the file that is involved in the event.

oldPath
The previous path name of the file during the **RENAME** event. For all other events, it is NULL.

clusterName
The name of the cluster where the event took place.

nodeName
The name of the node where the event took place.

nfsClientIp
The IP address of the remote client that is involved in the event.

fsName
The name of the file system that is involved in the event.

event

This is one of the following events: **OPEN**, **CREATE**, **CLOSE**, **RENAME**, **XATTRCHANGE**, **ACLCHANGE**, **UNLINK**, **DESTROY**, or **RMDIR**.

inode The inode number of the file that is involved in the event.

openFlags
The open flags that are specified during the event, as defined in

```
fcntl.h ( O_RDONLY,O_WRONLY,O_RDWR, O_CREAT, ...)
```

For example:

```
"openFlags": "32962" = 0x80C2 = o100302 translates to ( O_RDWR | O_CREAT | O_EXCL | O_LARGEFILE)
```

poolName
The pool name where the file resides.

fileSize
The current size of the file in bytes.

ownerUserId
The owner id of the file that is involved in the event.

ownerGroupId
The group id of the file that is involved in the event.

atime The time in UTC format of the last access of the file that is involved in the event.

ctime The time in UTC format of the last status change of the file that is involved in the event.

eventTime
The time in UTC format of the event.

clientUserId
The user id of the process that is involved in the event.

clientGroupId
The group id of the process that is involved in the event.

processId
The process id that is involved in the event.

permissions
The permissions on the file that is involved in the event.

acls The access control lists that are involved in the event.

xattrs The extended attributes that are involved in the event.

subEvent
The type of IBM Spectrum Scale attribute change. Only applies to the immutability and appendOnly flags.

The following table describes the JSON attributes that are provided for the ten events in file audit logging.

Table 16. JSON attributes in file audit logging

Attribute	OPEN	CREATE	CLOSE	RENAME	XATTR CHANGE	ACL CHANGE	UNLINK	DESTROY	RMDIR	GPFS ATTRCHANGE
LWE_JSON	X	X	X	X	X	X	X	X	X	X
path	X	X	X	X	X	X	X	X[1]	X	X
oldPath	Null	Null	Null	X	Null	Null	Null	Null	Null	Null

Table 16. JSON attributes in file audit logging (continued)

Attribute	OPEN	CREATE	CLOSE	RENAME	XATTR CHANGE	ACL CHANGE	UNLINK	DESTROY	RMDIR	GPFS ATTRCHANGE
clusterName	X	X	X	X	X	X	X	X	X	X
nodeName	X	X	X	X	X	X	X	X	X	X
nfsClientIp	X[2]									
fsName	X	X	X	X	X	X	X	X	X	X
event	X	X	X	X	X	X	X	X	X	X
inode	X	X	X	X	X	X	X	X	X	X
openFlags	X	0	X	0	0	0	0	0	0	0
poolName	X	X	X	X	X	X	X	X	X	X
fileSize	0	0	X	X	X	X	X	X	X	X
ownerUserId	X	X	X	X	X	X	X	X	X	0
ownerGroupId	X	X	X	X	X	X	X	X	X	0
atime	X	X	X	X	X	X	X	X	X	X
ctime	X	X	X	X	X	X	X	X	X	X
eventTime	X	X	X	X	X	X	X	X	X	X
clientUserId	X	X	X	X	X	X	X		X	0
clientGroupId	X	X	X	X	X	X	X		X	0
processId	X	X	X	X	X	X	X	NA	X	NA
permissions	X	X	X	X	X	X	X	X	X	X
acls	Null	Null	Null	Null	Null	X	Null	Null	Null	Null
xattrs	Null	Null	Null	Null	X[3]	Null	Null	Null	Null	Null
subEvent	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE	APPENDONLY IMMUTABILITY

Note:

1. The **path** attribute is sometimes available for the **DESTROY** event, but it is not guaranteed.
2. The **nfsClientIp** attribute is provided for NFS clients using Ganesha. The value is NULL for kernel NFS versions and SMB.
3. The **xattrs** attribute only shows the **xattr** that was changed.

IBM Spectrum Scale in an OpenStack cloud deployment

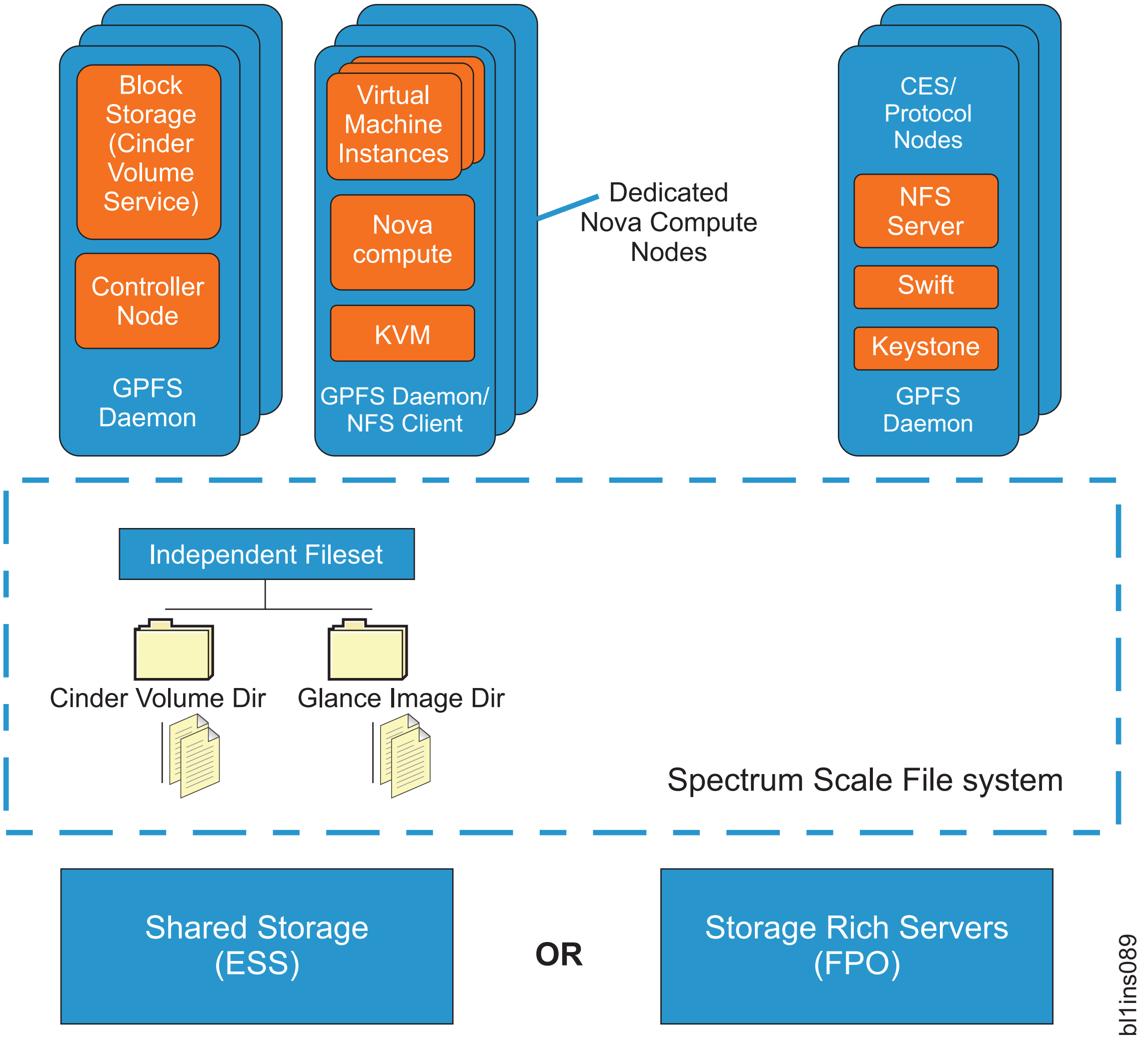
OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter. Using IBM Spectrum Scale in an OpenStack cloud environment offers many benefits, including the enterprise features of IBM Spectrum Scale and consolidated storage options.

An introduction to OpenStack

OpenStack is an open source software platform that is widely used as the base to build cloud infrastructure as a service solutions. OpenStack is typically deployed on commodity hardware and is used to virtualize various parts of the infrastructure (compute, storage, and network) to ease the sharing of the infrastructure across applications, use cases and workloads.

Using IBM Spectrum Scale in an OpenStack cloud deployment

Deploying OpenStack over IBM Spectrum Scale offers benefits provided by the many enterprise features in IBM Spectrum Scale as well as the ability to consolidate storage for various OpenStack components and applications running on top of the OpenStack infrastructure under a single storage management plane. One key benefit of IBM Spectrum Scale is that it provides uniform access to data under a single namespace with integrated analytics.



OpenStack components

- Cinder: Provides virtualized block storage for virtual machines. The IBM Spectrum Scale Cinder driver, also known as the GPFS driver, is written to take full advantage of the IBM Spectrum Scale enterprise features.
- Glance: Provides the capability to manage virtual machine images. When Glance is configured to use the same IBM Spectrum Scale fileset that stores Cinder volumes, bootable images can be created almost instantly by using the copy-on-write file clone capability.
- Swift: Provides object storage to any user or application that requires access to data through a RESTful API. The Swift object storage configuration has been optimized for the IBM Spectrum Scale environment, providing high availability and simplified management. Swift object storage also supports native the Swift APIs and Amazon S3 APIs for accessing data. Finally, the Swift object storage also supports access to the same data through either object interface or file interface (POSIX, NFS, SMB) without creating a copy.
- Manila: Provides a shared file system access to client, virtual, and physical systems. The IBM Spectrum Scale share driver (GPFS driver) is written to take full advantage of the IBM Spectrum Scale enterprise features.
- Keystone: Although not a storage component, internal keystone with in-built HA is provided by IBM Spectrum Scale as part of the Object protocol. In deployments that already have keystone support, the Object protocol can be configured to use the external keystone server rather than the internal one.

The following table lists the available features that IBM Spectrum Scale supports for deploying the OpenStack cloud storage:

Table 17. Features that IBM Spectrum Scale supports for deploying the OpenStack cloud storage

Feature	Support	Feature	Support
Volume and Image Management (Cinder and Glance)			
Volume Creation and Deletion	Yes	Volume Snapshot management	Yes
Volume Creation from Snapshot	Yes	Extend volumes	Yes
Copy Image to Volume and Copy Volume to Image	Yes	Instantaneous Boot Volume Create From Glance Repo	Yes
Live migration of instances	Yes	Backup of volumes	Yes
Data sharing between instances (with file system support)	Yes	Quality of service using multi-tier storage (with Flash support)	Yes
Encryption of volumes	Yes	Tiering of volumes	Yes
Compression of volumes	Yes	Volume Attach and Detach to a VM instance	Yes
Identity Management (Keystone)			
Integrated High Availability across the IBM Spectrum Scale protocol nodes	Yes	Easy configuration, management, and monitoring	Yes
AD and LDAP Support	Yes	External keystone support	Yes
Object store features (Swift)			
Unified file and object support	Yes	In-place analytics with Hadoop compatibility	Yes
High performance and High Availability	Yes	Object compression	Yes
Object encryption	Yes	Multi-region support	Yes
WAN caching with Active File Management (AFM)	Yes	Policy based information life cycle management	Yes
Easy install, configuration, and management	Yes	Integrated monitoring	Yes
Swift and S3 API Support	Yes	Large object support (5 TB)	Yes
Support for OpenStack Swift object store features	Yes		
Manila			
Support for NFS	Yes	Support for SMB (CIFS)	No

For more information on OpenStack, see the OpenStack Redpaper™.

IBM Spectrum Scale product editions

IBM Spectrum Scale has two different editions based on functional levels. Each edition can be licensed by IBM Spectrum Scale Client license, IBM Spectrum Scale Server license, and IBM Spectrum Scale FPO license.

For more information, see "IBM Spectrum Scale license designation" on page 129.

IBM Spectrum Scale Standard Edition

Available on AIX, Linux, and Windows. This edition provides extended features in addition to the base GPFS functions. On AIX and Linux, the extended features include Information Lifecycle Management (ILM), Active File Management (AFM), and Clustered NFS (CNFS). CNFS is not available on Linux on Z. On Windows, the extended features include limited Information Lifecycle Management (ILM).

On Red Hat Enterprise Linux 7.x, SLES 12, and Ubuntu 16.04 LTS, the extended features include the ability to enable and use the additional protocol functionality integration (NFS, SMB, and Object). On SLES 12, Object functionality is not supported in this release. Two packages are provided for this edition: one that includes protocols and needs additional (opensource/GPL) license acceptance, and another that does not include protocols and requires the traditional GPFS license acceptance.

IBM Spectrum Scale Advanced Edition

Available on AIX and Linux. This edition provides all the features of the Standard Edition and also provides AFM-based Asynchronous Disaster Recovery and high-level data protection using the GPFS cryptographic subsystem. For additional information, see the *Encryption* topic in *IBM Spectrum Scale: Administration Guide*.

IBM Spectrum Scale Data Management Edition

This edition provides identical functionality as IBM Spectrum Scale Advanced Edition under capacity-based licensing. For more information, see “Capacity-based licensing” on page 132.

Note: All nodes in a cluster must have the same edition installed.

The only exception is when a remote cluster without encryption needs access to a home cluster with encryption. For more information, see *Encryption with Multicluster* in “IBM Spectrum Scale license designation” on page 129.

Attention: Starting with release 4.2.3, IBM Spectrum Scale Express Edition is no longer available. For information on upgrading from IBM Spectrum Scale Express Edition 4.2.2.x or earlier to IBM Spectrum Scale Standard Edition 4.2.3.x or later, see “Migrating from Express Edition to Standard Edition” on page 417.

Table 18 lists the availability of key features in the IBM Spectrum Scale editions.

Table 18. Features in IBM Spectrum Scale editions

Feature	Standard Edition	Advanced or Data Management Edition
Support for maximum file system size, number of files, file systems and number of nodes	✓	✓
Node roles: Collector node, Admin node	No charge	No charge
Node roles: Cluster Config Server, Manager, quorum, tie-breaker, NSD server	✓	✓
Multi-cluster	✓	✓
Quotas (user and group only)	✓	✓
Snapshots	✓ (fileset)	✓ (fileset)
Management GUI	✓	✓
Compression (2:1 to 5:1 ratio)	✓	✓
Quality of Service	✓	✓
Filesets	✓	✓

Table 18. Features in IBM Spectrum Scale editions (continued)

Feature	Standard Edition	Advanced or Data Management Edition
Multi-protocol access	✓	✓
Storage pools	✓	✓
ILM placement and management policies	✓	✓
HSM with IBM Spectrum Protect or IBM Spectrum Archive	✓	✓
AFM caching	✓	✓
Transparent cloud tiering		✓
AFM-based Asynchronous Disaster Recovery (gateway nodes)		✓
Encryption of data at rest and secure erase		✓

Consult the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) for the latest features included in each edition.

IBM Spectrum Scale license designation

According to the IBM Spectrum Scale Licensing Agreement, each server in the cluster must be designated as possessing an IBM Spectrum Scale Client license, an IBM Spectrum Scale File Placement Optimizer (FPO) license, or an IBM Spectrum Scale Server license. Starting with release 4.2.2, you can also use capacity-based licensing which is available only with IBM Spectrum Scale Data Management Edition.

There are three basic storage cluster models: storage area network (SAN), network shared disk (NSD), and shared nothing. IBM Spectrum Scale supports all three of them. For SAN and NSD clusters, a number of Server licenses are required in addition to Client licenses if the customer is planning to connect Client nodes to the IBM Spectrum Scale server using the IBM Spectrum Scale high-speed native driver (GPFS daemon). For shared nothing clusters of storage rich servers, FPO licenses are required on each of those servers.

The type of license that is associated with any one server depends on the functional roles that the server has been designated to perform.

IBM Spectrum Scale Client license

The IBM Spectrum Scale Client license permits exchange of data between servers that locally mount the same IBM Spectrum Scale file system. No other export of the data is permitted.

IBM Spectrum Scale Server license

The IBM Spectrum Scale Server license permits the licensed server to perform management functions such as cluster configuration manager, quorum node, manager node, and Network Shared Disk (NSD) server. In addition, the IBM Spectrum Scale Server license permits the licensed server to share IBM Spectrum Scale data directly through any application, service protocol or method such as Network File System (NFS), Server Message Block (SMB), File Transfer Protocol (FTP), Hypertext Transfer Protocol (HTTP), Object Protocol (OpenStack Swift, Amazon S3 API). Therefore, protocol nodes also require an IBM Spectrum Scale Server license.

IBM Spectrum Scale FPO license

The IBM Spectrum Scale FPO license permits the licensed server to perform NSD server functions for sharing IBM Spectrum Scale data with other server that have an IBM Spectrum Scale FPO or an IBM Spectrum Scale Server license.

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

Use the guidance in Figure 22 to decide which IBM Spectrum Scale license to buy for your requirements.

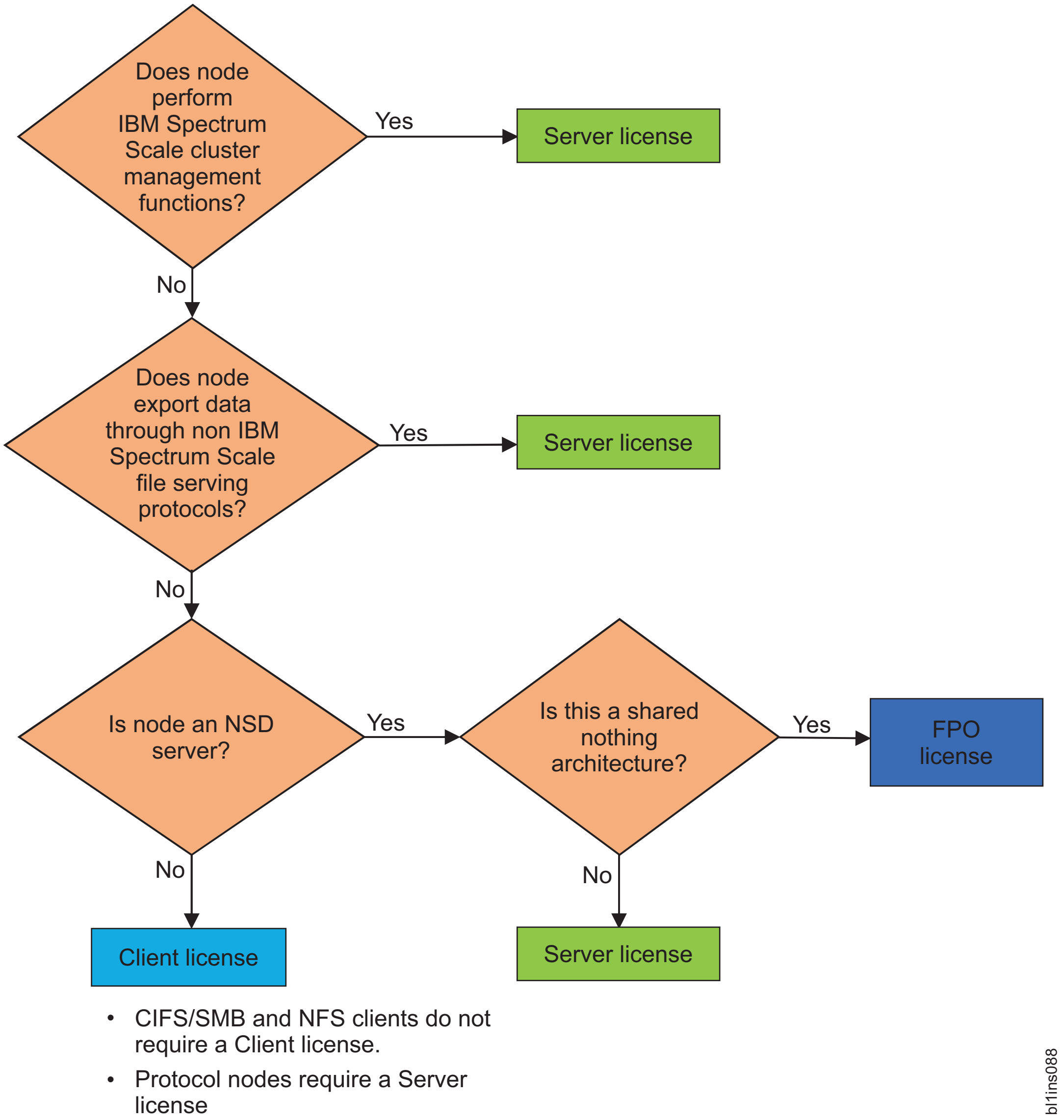

Does node perform IBM Spectrum Scale cluster management functions?
Yes
Server license
No
Does node export data through non IBM Spectrum Scale file serving protocols?
Yes
Server license
No
Is node an NSD server?
Yes
Is this a shared nothing architecture?
Yes
FPO license
No
Client license
No
Server license
• CIFS/SMB and NFS clients do not require a Client license.
• Protocol nodes require a Server license
bl1ins088

Figure 22. Guidance on which license to buy

These licenses are all valid for use in IBM Spectrum Scale Standard Edition and IBM Spectrum Scale Advanced Edition. For more information, see "IBM Spectrum Scale product editions" on page 127. With IBM Spectrum Scale Data Management Edition, you must use capacity-based licensing. For more information, see "Capacity-based licensing" on page 132.

- You can view the number and type of licenses currently in effect for the cluster using the **mmlslicense** command.
- If needed, you can change the type of license assigned to a node using the **mmchlicense** command.

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

The following are IBM Spectrum Scale licensing considerations including considerations for using IBM Spectrum Scale with other offerings.

AFM-based Async Disaster Recovery (AFM DR) with multicluster

When using AFM-based Async Disaster Recovery (AFM DR) in a multicluster environment, both the home and the cache cluster require the IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition.

Encryption

The encryption function available in IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition requires a separate IBM Security Key Lifecycle Manager (ISKLM) license.

Encryption with multicluster

When using the IBM Spectrum Scale encryption function in a multicluster environment, each server in an IBM Spectrum Scale cluster requiring access to another cluster's encrypted file system must be licensed with IBM Spectrum Scale Advanced Edition (Client, Server, or FPO as appropriate) or IBM Spectrum Scale Data Management Edition. IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition is not required on all nodes of the remote cluster.

Hadoop access

The IBM Spectrum Scale Hadoop connector can be used with all license types (Client, Server, FPO) and all Editions (Standard, Advanced, and Data Management). There is no additional license requirement for Hadoop access. A Hadoop node using the connector needs no IBM Spectrum Scale license. The Hadoop node can connect to a node in the IBM Spectrum Scale cluster, which is licensed as Server because it is exporting data, and access the file system directly using the Hadoop connector.

IBM Spectrum Scale with IBM Spectrum Protect

When using IBM Spectrum Scale with IBM Spectrum Protect for backup and restore, each IBM Spectrum Scale node performing data movement requires an IBM Spectrum Protect license.

IBM Spectrum Scale with IBM Spectrum Protect for Space Management

When using IBM Spectrum Scale with IBM Spectrum Protect for Space Management for migration and recall, each IBM Spectrum Scale node performing data movement requires an IBM Spectrum Protect for Space Management license.

IBM Spectrum Scale with IBM Spectrum Archive Enterprise Edition (EE)

In addition to an IBM Spectrum Archive Enterprise Edition (EE) license, a server in the IBM Spectrum Scale cluster which is being used for IBM Spectrum Archive Enterprise Edition (EE) requires IBM Spectrum Scale to be installed with the correct license (ordered separately). The IBM Spectrum Archive Enterprise Edition (EE) server requires an IBM Spectrum Scale Server license, if the IBM Spectrum Archive Enterprise Edition (EE) server is also being used for any of the functions requiring an IBM Spectrum Scale Server license such as NSD server, quorum, management, etc. Otherwise, the IBM Spectrum Archive Enterprise Edition (EE) server can be licensed with an IBM Spectrum Scale Client license. The IBM Spectrum Archive Enterprise Edition (EE) server's function of moving data from the file system to a tape drive is not considered as a type of data export that requires an IBM Spectrum Scale Server license.

Virtualization licensing

In an IBM Spectrum Scale environment containing VMs or LPARs, licenses are required for the sockets available to the IBM Spectrum Scale VMs or LPARs on a physical server. The number of licenses required is the number of cores available to IBM Spectrum Scale. For more information, see *Virtualization questions* in IBM Spectrum Scale FAQ on IBM Knowledge Center.

Capacity-based licensing

You can use capacity-based licensing to license IBM Spectrum Scale based on the storage capacity being managed in an IBM Spectrum Scale cluster.

The storage capacity to be licensed is the capacity in terabytes (TiB) from all NSDs in the IBM Spectrum Scale cluster. The capacity to be licensed is not affected by using functions such as replication or compression or by doing tasks such as creating or deleting files, file systems, or snapshots.

For example, if you have 100 TiB of capacity from all NSDs in the IBM Spectrum Scale cluster and if you have set the replication factor to 2, for 50 TiB of application data, 100 TiB of disk capacity is consumed. In this case, the capacity to be licensed is 100 TiB, not 50 TiB.

The capacity to be licensed changes only when an NSD is added or deleted from the IBM Spectrum Scale cluster.

You can view the cluster capacity for licensing by using one of the following methods:

- Select the **About** option from the menu that is available in the upper right corner of the IBM Spectrum Scale management GUI.
- Issue the **`mmlslicense --capacity`** command.

Capacity-based licensing can be used only with IBM Spectrum Scale Data Management Edition. The Data Management Edition provides identical functionality as the Advanced Edition.

Capacity-based licensing can also be used to license an Elastic Storage Server (ESS) system. For ESS, the capacity to be licensed is the capacity after applying IBM Spectrum Scale RAID. The exact capacity to be licensed depends on the RAID code selected, 8+2p or 8+3p.

IBM Spectrum Storage Suite

IBM Spectrum Scale is a part of IBM Spectrum Storage Suite.

IBM Spectrum Storage Suite gives you unlimited access to the IBM Software Defined Storage product portfolio with licensing on a flat, cost-per-TB basis to make pricing easy to understand and predictable as capacity grows.

Licenses are calculated on the amount of storage capacity you are managing, not the number of software products you are using.

For more information, see IBM Spectrum Storage Suite website.

Chapter 2. Planning for IBM Spectrum Scale

Planning for GPFS

Although you can modify your GPFS configuration after it has been set, a little consideration before installation and initial setup will reward you with a more efficient and immediately useful file system.

During configuration, GPFS requires you to specify several operational parameters that reflect your hardware resources and operating environment. During file system creation, you can specify parameters that are based on the expected size of the files or you can let the default values take effect.

The installation toolkit is also available to assist with GPFS installation on Linux nodes. For more information, see "Overview of the installation toolkit" on page 258

Planning for GPFS includes:
- "Hardware requirements"
- "Software requirements" on page 134
- "IBM Spectrum Scale product editions" on page 127
- "Recoverability considerations" on page 135
- "GPFS cluster creation considerations" on page 141
- "IBM Spectrum Scale license designation" on page 129
- "Disk considerations" on page 146
- "File system creation considerations" on page 151

Hardware requirements

You can validate that your hardware meets GPFS requirements by taking the steps outlined in this topic.

1. Consult the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) for the latest list of:
   - Supported server hardware
   - Tested disk configurations
   - Maximum cluster size
   - Additional hardware requirements for protocols
2. Provide enough disks to contain the file system. Disks can be:
   - SAN-attached to each node in the cluster
   - Attached to one or more NSD servers
   - A mixture of directly-attached disks and disks that are attached to NSD servers

   For additional information, see "Network Shared Disk (NSD) creation considerations" on page 147.
3. When doing network-based NSD I/O, GPFS passes a large amount of data between its daemons. For NSD server-to-client traffic, it is suggested that you configure a dedicated high-speed network solely for GPFS communications when the following are true:
   - There are NSD disks configured with servers providing remote disk capability.
   - Multiple GPFS clusters are sharing data using NSD network I/O.

   For additional information, see "Disk considerations" on page 146.

GPFS communications require static IP addresses for each GPFS node. IP address takeover operations that transfer the address to another computer are not allowed for the GPFS network. Other IP addresses within the same computer that are not used by GPFS can participate in IP takeover. To provide availability or additional performance, GPFS can use virtual IP addresses created by aggregating several network adapters using techniques such as EtherChannel or channel bonding.

Cluster Export Services (CES) have dedicated network addresses to support SMB, NFS, Object, and failover or failback operations. File and Object clients use the public IPs to access data on GPFS™ file systems.

For additional information, see *CES network configuration* in *IBM Spectrum Scale: Administration Guide*.

Software requirements

GPFS planning includes understanding the latest software requirements.

Note: This topic lists the additional software requirements to use specific functions after successful installation. It does not list the dependencies for installing IBM Spectrum Scale packages.

- GPFS is supported on AIX, Linux, and Windows.
- For existing GPFS 3.5 clusters, OpenSSL is required for remote cluster access.
- Kernel development files and compiler utilities are required to build the GPFS portability layer on Linux nodes. The required RPMs or packages for each supported Linux distribution are:
  - **SLES Linux RPMs**
    - `kernel-default-devel`
    - `cpp`
    - `gcc`
    - `gcc-c++`
    - `binutils`
  - **Red Hat Linux RPMs**
    - `kernel-devel`
    - `cpp`
    - `gcc`
    - `gcc-c++`
    - `binutils`
  - **Ubuntu Linux packages**
    - `linux-headers`
    - `cpp`
    - `gcc`
    - `g++`
    - `binutils`
- To use active file management (AFM), the following is required:
  `nfs-utils`
- To use CNFS, the following are required:
  `ethtool`
  `nfs-utils` (on Ubuntu: `nfs-common`)
  `rpcbind`
  `psmisc`
  `iputils` (on Ubuntu: `iputils-arping`)
- To use the `mmchconfig` `numaMemoryInterleave` parameter, the following is required:
  `numactl`
- To use the **`mmhealth`** command on Ubuntu Linux, the following is required:
  `sqlite3`
- For the IBM Spectrum Scale monitoring service on AIX and Linux, the following is required:

`python` 2.7 or later

Note: `python` 2.7.5 or later is recommended on AIX. This package is a part of AIX Toolbox for Linux Applications. However, `python` 3.0 is not supported for Linux or AIX.

For a list of installation prerequisites including those specific to protocols, see "Installation prerequisites" on page 224.

Consult the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) for the latest list of:

- AIX environments
- Linux distributions
- Linux kernel versions
- Windows environments

Recoverability considerations

Sound file system planning requires several decisions about recoverability. After you make these decisions, GPFS parameters enable you to create a highly-available file system with rapid recovery from failures.

- At the disk level, consider preparing disks for use with your file system by specifying failure groups that are associated with each disk. With this configuration, information is not vulnerable to a single point of failure. See "Network Shared Disk (NSD) creation considerations" on page 147.
- At the file system level, consider replication through the metadata and data replication parameters. See "File system replication parameters" on page 160.

Additionally, GPFS provides several layers of protection against failures of various types:

1. "Node failure"
2. "Network Shared Disk server and disk failure" on page 139
3. "Reduced recovery time by using Persistent Reserve" on page 141

Node failure

If a node fails, GPFS prevents the continuation of I/O from the failing node and replays the file system metadata log for the failed node.

GPFS prevents the continuation of I/O from a failing node through a GPFS-specific fencing mechanism called *disk leasing*. When a node has access to file systems, it obtains disk leases that allow it to submit I/O. However, when a node fails, that node cannot obtain or renew a disk lease. When GPFS selects another node to perform recovery for the failing node, it first waits until the disk lease for the failing node expires. This allows for the completion of previously submitted I/O and provides for a consistent file system metadata log. Waiting for the disk lease to expire also avoids data corruption in the subsequent recovery step.

To reduce the amount of time it takes for disk leases to expire, you can use Persistent Reserve (SCSI-3 protocol). If Persistent Reserve (configuration parameter: **usePersistentReserve**) is enabled, GPFS prevents the continuation of I/O from a failing node by fencing the failed node that uses a feature of the disk subsystem called Persistent Reserve. Persistent Reserve allows the failing node to recover faster because GPFS does not need to wait for the disk lease on the failing node to expire. For additional information, refer to "Reduced recovery time by using Persistent Reserve" on page 141. For further information about recovery from node failure, see *Installation and configuration issues* in *IBM Spectrum Scale: Problem Determination Guide*.

There is a temporary impact to I/O during file system recovery from node failure. Recovery involves rebuilding metadata structures that might be under modification at the time of the failure. If the failing node is acting as the file system manager when it fails, the delay is longer and proportional to the level of activity on the file system at the time of failure. In this case, the failover file system management task happens automatically to a surviving node.

Managing node failures also involves sizing the solution adequately so that remaining nodes in the cluster can support a node down situation such as a planned system maintenance or an unplanned node failure in terms of bandwidth and throughput. For protocols, this includes supporting SMB, NFS, or Object connections that have to fail over to another CES node in the cluster if a node fails.

Quorum:

GPFS uses a cluster mechanism called quorum to maintain data consistency in the event of a node failure.

Quorum operates on the principle of majority rule. This means that a majority of the nodes in the cluster must be successfully communicating before any node can mount and access a file system. This keeps any nodes that are cut off from the cluster (for example, by a network failure) from writing data to the file system.

During node failure situations, quorum needs to be maintained in order for the cluster to remain online. If quorum is not maintained due to node failure, GPFS unmounts local file systems on the remaining nodes and attempts to reestablish quorum, at which point file system recovery occurs. For this reason it is important that the set of quorum nodes be carefully considered (refer to “Selecting quorum nodes” on page 139 for additional information).

GPFS quorum must be maintained within the cluster for GPFS to remain active. If the quorum semantics are broken, GPFS performs recovery in an attempt to achieve quorum again. GPFS can use one of two methods for determining quorum:
- Node quorum
- Node quorum with tiebreaker disks

Node quorum:

Node quorum is the default quorum algorithm for GPFS.

With node quorum:
- Quorum is defined as one plus half of the *explicitly defined* quorum nodes in the GPFS cluster.
- There are no default quorum nodes; you must specify which nodes have this role.

For example, in Figure 23 on page 137, there are three quorum nodes. In this configuration, GPFS remains active as long as there are two quorum nodes available.

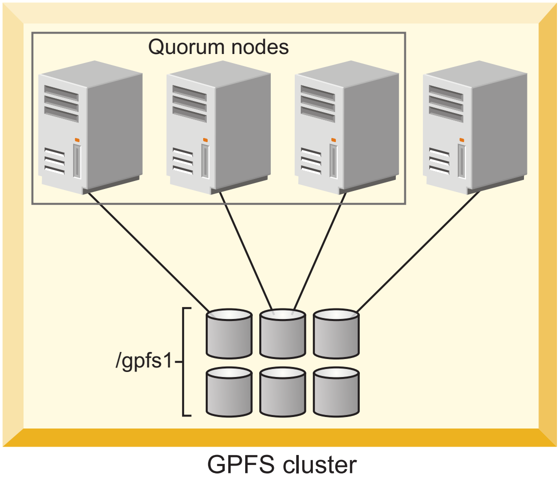


Figure 23. GPFS configuration using node quorum

Node quorum with tiebreaker disks:

When running on small GPFS clusters, you might want to have the cluster remain online with only one surviving node.

To achieve this, you need to add a tiebreaker disk to the quorum configuration. 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 (refer to Figure 24 on page 138). 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.

Cluster node rules:

1. There is a maximum of eight quorum nodes.
2. All quorum nodes need to have access to all of the tiebreaker disks.
3. When using the traditional server-based (non-CCR) configuration repository, you should include the primary and secondary cluster configuration servers as quorum nodes.
4. You may have an unlimited number of non-quorum nodes.
5. If a network connection fails, which causes the loss of quorum, and quorum is maintained by tiebreaker disks, the following rationale is used to re-establish quorum. If a group has the cluster manager, it is the “survivor”. The cluster manager can give up its role if it communicates with fewer than the minimum number of quorum nodes as defined by the **minQuorumNodes** configuration parameter. In this case, other groups with the minimum number of quorum nodes (if they exist) can choose a new cluster manager.

Changing quorum semantics:

When using the cluster configuration repository (CCR) to store configuration files, the total number of quorum nodes is limited to eight, regardless of quorum semantics, but the use of tiebreaker disks can be enabled or disabled at any time by issuing an **mmchconfig tiebreakerDisks** command. The change will take effect immediately, and it is not necessary to shut down GPFS when making this change.

When using the traditional server-based (non-CCR) configuration repository, it is possible to define more than eight quorum nodes, but only when no tiebreaker disks are defined:

1. To configure more than eight quorum nodes under the server-based (non-CCR) configuration repository, you must disable node quorum with tiebreaker disks and restart the GPFS daemon. To disable node quorum with tiebreaker disks:
   a. Issue the **mmshutdown -a** command to shut down GPFS on all nodes.
   b. Change quorum semantics by issuing **mmchconfig tiebreakerdisks=no**.
   c. Add additional quorum nodes.
   d. Issue the **mmstartup -a** command to restart GPFS on all nodes.
2. If you remove quorum nodes and the new configuration has less than eight quorum nodes, you can change the configuration to node quorum with tiebreaker disks. To enable quorum with tiebreaker disks:
   a. Issue the **mmshutdown -a** command to shut down GPFS on all nodes.
   b. Delete the appropriate quorum nodes or run **mmchnode --nonquorum** to drop them to a client.
   c. Change quorum semantics by issuing the **mmchconfig tiebreakerdisks="***diskList***"** command.
      - The *diskList* contains the names of the tiebreaker disks.
      - The list contains the NSD names of the disks, preferably one or three disks, separated by a semicolon (;) and enclosed by quotes.
   d. Issue the **mmstartup -a** command to restart GPFS on all nodes.

Tiebreaker disk rules:

- You can have one, two, or three tiebreaker disks. However, you should use an odd number of tiebreaker disks.
- Among the quorum node groups that appear after an interconnect failure, only those having access to a majority of tiebreaker disks can be candidates to be the survivor group.
- Tiebreaker disks must be connected to all quorum nodes.
- In a CCR-based cluster, when adding tiebreaker disks:
  – GPFS should be up and running, if tiebreaker disks are part of the file system.
  – GPFS can be either running or shut down, if tiebreaker disks are not a part of the file system.

In Figure 24 GPFS remains active with the minimum of a single available quorum node and two available tiebreaker disks.

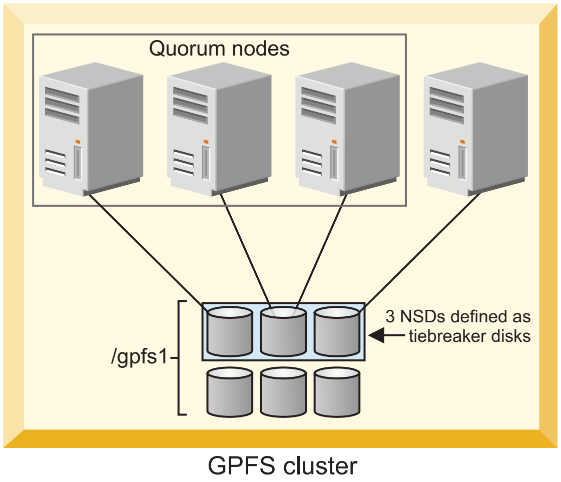


Figure 24. GPFS configuration using node quorum with tiebreaker disks

When a quorum node detects loss of network connectivity, but before GPFS runs the algorithm that decides if the node will remain in the cluster, the **tiebreakerCheck** event is triggered. This event is generated only in configurations that use quorum nodes with tiebreaker disks. It is also triggered on the cluster manager periodically by a challenge-response thread to verify that the node can still continue as cluster manager.

Selecting quorum nodes:

To configure a system with efficient quorum nodes, follow these rules.

- Select nodes that are likely to remain active
  - If a node is likely to be rebooted or require maintenance, do not select that node as a quorum node.
- Select nodes that have different failure points such as:
  - Nodes that are located in different racks
  - Nodes that are connected to different power panels
- You should select nodes that GPFS administrative and serving functions rely on such as:
  - Primary configuration servers
  - Secondary configuration servers
  - Network Shared Disk (NSD) servers
- Select an odd number of nodes as quorum nodes
  - The suggested maximum is eight quorum nodes.
- Having a large number of quorum nodes might increase the time required for startup and failure recovery.
  - Having more than seven quorum nodes does not guarantee higher availability.
  - All quorum nodes must have access to all of the tiebreaker disks.

Network Shared Disk server and disk failure

The three most common reasons why data becomes unavailable are disk failure, disk server failure with no redundancy, and failure of a path to the disk.

In the event of a disk failure in which GPFS can no longer read or write to the disk, GPFS will discontinue use of the disk until it returns to an available state. You can guard against loss of data availability from disk failure by:

- Utilizing hardware data protection as provided by a Redundant Array of Independent Disks (RAID) device (see Figure 25)

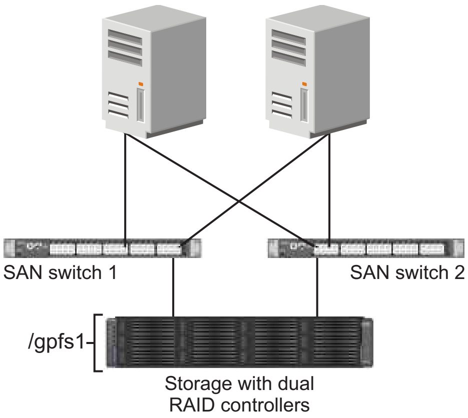


Figure 25. An example of a highly available SAN configuration for a GPFS file system

- Utilizing the GPFS data and metadata replication features (see "Increased data availability" on page 2) along with the designation of failure groups (see "Network Shared Disk (NSD) creation considerations" on page 147 and Figure 26)

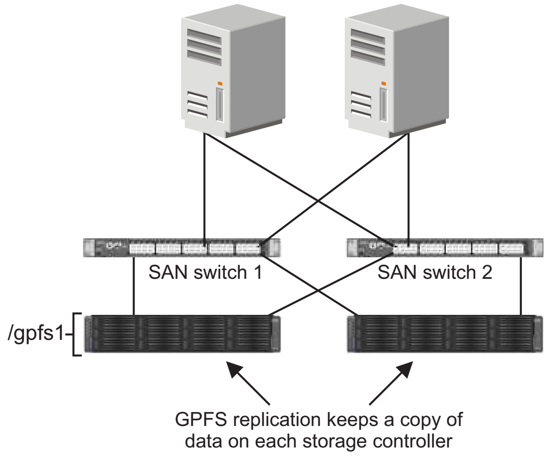

SAN switch 1
SAN switch 2
/gpfs1
GPFS replication keeps a copy of data on each storage controller

Figure 26. Configuration using GPFS replication for improved availability

It is suggested that you consider RAID as the first level of redundancy for your data and add GPFS replication if you desire additional protection.

In the event of an NSD server failure in which a GPFS client can no longer contact the node that provides remote access to a disk, GPFS discontinues the use of the disk. You can guard against loss of an NSD server availability by using common disk connectivity on multiple NSD server nodes and specifying multiple Network Shared Disk servers for each common disk.

Note: In the event that a path to a disk fails, GPFS reports a disk failure and marks the disk **down**. To bring the disk back online, first follow the directions supplied by your storage vendor to determine and repair the failure.

Guarding against loss of data availability due to path failure

You can guard against loss of data availability from failure of a path to a disk by doing the following:

- Creating multiple NSD servers for each disk

  As GPFS determines the available connections to disks in the file system, it is recommended that you always define more than one NSD server for each disk. GPFS allows you to define up to eight NSD servers for each NSD. In a SAN configuration where NSD servers have also been defined, if the physical connection is broken, GPFS dynamically switches to the next available NSD server (as defined on the server list) and continues to provide data. When GPFS discovers that the path has been repaired, it moves back to local disk access. This is the default behavior, which can be changed by designating file system mount options. For example, if you never want a node to use the NSD server path to a disk, even if the local path fails, you can set the **-o useNSDserver** mount option to **never**. You can set the mount option using the **mmchfs**, **mmmount**, **mmremotefs**, and **mount** commands.

  Important: In Linux on Z, it is mandatory to have multiple paths to one SCSI disk (LUN) to avoid single path of failure. The coalescing of the paths to one disk is done by the kernel (via the device-mapper component). As soon as the paths are coalesced, a new logical, multipathed device is created, which is used for any further (administration) task. (The single paths can no longer be used.)

  The multipath device interface name depends on the distribution and is configurable:

  SUSE **/dev/mapper/***Unique_WW_Identifier*

For example: /dev/mapper/36005076303ffc56200000000000010cc

Red Hat

/dev/mapper/mpath*

To obtain information about a multipathed device, use the multipath tool as shown in the following example:

```
# multipath -ll
```

The system displays output similar to this:

```
36005076303ffc56200000000000010cc dm-0 IBM,2107900
[size=5.0G][features=1 queue_if_no_path][hwhandler=0]
\_ round-robin 0 [prio=2][active]
 \_ 1:0:0:0 sdb   8:16  [active][ready]
 \_ 0:0:0:0 sda   8:0   [active][ready]
```

See the question, "What considerations are there when setting up DM-MP multipath service" in the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

- Using an I/O driver that provides multiple paths to the disks for failover purposes

  Failover is a path-management algorithm that improves the reliability and availability of a device because the system automatically detects when one I/O path fails and reroutes I/O through an alternate path.

Reduced recovery time by using Persistent Reserve

Persistent Reserve (PR) provides a mechanism for reducing recovery times from node failures.

To enable PR and to obtain recovery performance improvements, your cluster requires a specific environment:

- *All* disks must be PR-capable.
- On AIX, all disks must be hdisks. Starting with 3.5.0.16, it is also possible to use a logical volume as a **descOnly** disk without disabling the use of Persistent Reserve. See the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

  On Linux, typically all disks must be generic (/dev/sd*) or DM-MP (/dev/dm-*) disks.

  However, for Linux on Z, multipath device names are required for SCSI disks, and the names depend on the distribution and are configurable. For more information, see "Guarding against loss of data availability due to path failure" on page 140.
- If the disks have NSD servers that are defined, all NSD server nodes must be running the same operating system (AIX or Linux).
- If the disks are SAN-attached to all nodes, all nodes in the cluster must be running the same operating system (AIX or Linux).

For quicker recovery times when you use PR, set the **failureDetectionTime** configuration parameter on the **mmchconfig** command. For example, for quick recovery a recommended value would be 10:

```
mmchconfig failureDetectionTime=10
```

You must explicitly enable PR by specifying the **usePersistentReserve** parameter on the **mmchconfig** command. If you set usePersistentReserve=yes, GPFS attempts to set up PR on all of the PR capable disks. All subsequent NSDs are created with PR enabled if they are PR capable. However, PR is only supported in the home cluster. Therefore, access to PR-enabled disks from another cluster must be through an NSD server that is in the home cluster and not directly to the disk (for example, through a SAN).

GPFS cluster creation considerations

A GPFS cluster is created using the **mmcrcluster** command.

Table 19 details the cluster creation options, how to change the options, and the default values for each option.

Table 19. GPFS cluster creation options

Options	Command to change the option	Default value
"Creating an IBM Spectrum Scale cluster" on page 143	**mmaddnode** **mmdelnode**	None
Node designation: **manager** or **client**. See "Creating an IBM Spectrum Scale cluster" on page 143.	**mmchnode**	**client**
Node designation: **quorum** or **nonquorum**. See "Creating an IBM Spectrum Scale cluster" on page 143.	**mmchnode**	**nonquorum**
"Remote shell command" on page 144	**mmchcluster**	**/usr/bin/ssh**
"Remote file copy command" on page 145	**mmchcluster**	**/usr/bin/scp**
"Cluster name" on page 145	**mmchcluster**	The node name of the primary GPFS cluster configuration server.
Cluster configuration information. See "IBM Spectrum Scale cluster configuration information" on page 144.	**mmchcluster**	The Cluster Configuration Repository (CCR) is enabled.
GPFS administration adapter port name. See "GPFS node adapter interface names"	**mmchnode**	Same as the GPFS communications adapter port name.
GPFS communications adapter port name. See "GPFS node adapter interface names."	**mmchnode**	None
"User ID domain for the cluster" on page 145.	**mmchconfig**	The name of the GPFS cluster.
"Starting GPFS automatically" on page 146	**mmchconfig**	**no**
"Cluster configuration file" on page 146	Not applicable	None

GPFS node adapter interface names

An adapter interface name refers to the hostname or IP address that GPFS uses to communicate with a node. Specifically, the hostname or IP address identifies the communications adapter over which the GPFS daemons or GPFS administration commands communicate.

The administrator can specify two node adapter interface names for each node in the cluster:

GPFS node name
Specifies the name of the node adapter interface to be used by the GPFS daemons for internode communication.

GPFS admin node name
Specifies the name of the node adapter interface to be used by GPFS administration commands when communicating between nodes. If not specified, the GPFS administration commands use the same node adapter interface used by the GPFS daemons.

These names can be specified by means of the node descriptors passed to the **mmaddnode** or **mmcrcluster** command and can later be changed with the **mmchnode** command.

If multiple adapters are available on a node, this information can be communicated to GPFS by means of the **subnets** parameter on the **mmchconfig** command.

Creating an IBM Spectrum Scale cluster

When you create an IBM Spectrum Scale cluster, you can either create a small cluster of one or more nodes and later add nodes to it or you can create a cluster with all its nodes in one step.

Two methods are available for creating an IBM Spectrum Scale cluster:

- Run the **mmcrcluster** command to create a cluster that contains one or more nodes, and later run the **mmaddnode** command as needed to add nodes to the cluster. This method is more flexible and is appropriate when you want to build a cluster step by step.
- Run the **mmcrcluster** command to create a cluster and at the same time to add a set of nodes to the cluster. This method is quicker when you already know which nodes you want to add to the cluster.

Whichever method you choose, you can later add nodes to the cluster with the **mmaddnode** command or delete nodes with the **mmdelnode** command. For more information, see the topics *mmcrcluster command* and *mmaddnode command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

With both commands, you specify node descriptors to identify the nodes that you are adding to the cluster. You can list node descriptors either on the command line or in a separate node descriptor file. A node descriptor has the following format:

```
NodeName:NodeDesignations:AdminNodeName
```

NodeName is a required parameter. *NodeDesignations* and *AdminNodeName* are optional parameters.

NodeName

The host name or IP address of the node for GPFS daemon-to-daemon communication.

The host name or IP address that is used for a node must refer to the communication adapter over which the GPFS daemons communicate. Alias names are not allowed. You can specify an IP address at NSD creation, but it will be converted to a host name that must match the GPFS node name. You can specify a node using any of these forms:

- Short hostname (for example, h135n01)
- Long hostname (for example, h135n01.frf.ibm.com)
- IP address (for example, 7.111.12.102)

Note: Host names should always include at least one alpha character, and they should not start with a hyphen (-).

Whichever form you specify, the other two forms must be defined correctly in DNS or the hosts file.

NodeDesignations

An optional, "-" separated list of node roles.

- **manager** | **client** – Indicates whether a node is part of the node pool from which file system managers and token managers can be selected. The default is **client**, which means do not include the node in the pool of manager nodes. For detailed information on the manager node functions, see "The file system manager" on page 10.

  In general, it is a good idea to define more than one node as a manager node. How many nodes you designate as manager depends on the workload and the number of GPFS server licenses you have. If you are running large parallel jobs, you may need more manager nodes than in a four-node cluster supporting a Web application. As a guide, in a large system there should be a different file system manager node for each GPFS file system.
- **quorum** | **nonquorum** – This designation specifies whether or not the node should be included in the pool of nodes from which quorum is derived. The default is **nonquorum**. You need to designate at least one node as a quorum node. It is recommended that you designate at least the primary and secondary cluster configuration servers and NSD servers as quorum nodes.

  How many quorum nodes you designate depends upon whether you use node quorum or node quorum with tiebreaker disks. See "Quorum" on page 136.

AdminNodeName

Specifies an optional field that consists of a node name to be used by the administration commands to communicate between nodes.

If *AdminNodeName* is not specified, the *NodeName* value is used.

Follow these rules when you add nodes to a IBM Spectrum Scale cluster:

- While a node may mount file systems from multiple clusters, the node itself may only reside in a single cluster. Nodes are added to a cluster using the **mmcrcluster** or **mmaddnode** command.
- The nodes must be available when they are added to a cluster. 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 then issue the **mmaddnode** command to add those nodes.
- Designate at least one but not more than seven nodes as quorum nodes. When not using tiebreaker disks, you can designate more quorum nodes, but it is recommended to use fewer than eight if possible. When using server-based configuration repository, it is recommended that you designate the cluster configuration servers as quorum nodes. How many quorum nodes altogether you will have depends on whether you intend to use the node quorum with tiebreaker algorithm or the regular node based quorum algorithm. For more details, see “Quorum” on page 136.

IBM Spectrum Scale cluster configuration information

You can use the Cluster Configuration Repository (CCR) to maintain cluster configuration information.

It is a good practice to use the Cluster Configuration Repository (CCR) to maintain the cluster configuration information. By default the **mmcrcluster** command enables the CCR. For more information, see the topics *mmcrcluster command* and *mmchcluster command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

If you do not use the Cluster Configuration Repository (CCR), you must designate one of the nodes in your IBM Spectrum Scale cluster as the primary cluster configuration server, to maintain configuration information. It is a good practice to also specify a secondary cluster configuration server. If your primary server fails and you do not have a secondary server, the cluster configuration data files are inaccessible, which results in the failure of any IBM Spectrum Scale administration commands that you enter. Note the following information:

- When the GPFS daemon starts up, at least one of the two GPFS cluster configuration servers must be accessible. See “Cluster configuration data files” on page 26.
- If the primary server fails, you can use the **mmchcluster** command with the **-p** option to designate another node as the primary cluster configuration server. Similarly, you can use the **mmchcluster** command with the **-s** option to define a secondary cluster configuration server.

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

Remote shell command

GPFS commands need to be able to communicate across all nodes in the cluster. To achieve this, the GPFS commands use the remote shell command that you specify on the **mmcrcluster** command or the **mmchcluster** command.

The default remote shell command is **ssh**. You can designate the use of a different remote shell command by specifying its fully-qualified path name on the **mmcrcluster** command or the **mmchcluster** command. The remote shell command must adhere to the same syntax as **ssh**, but it can implement an alternate authentication mechanism.

Clusters that include both UNIX and Windows nodes must use **ssh** for the remote shell command. For more information, see “Installing and configuring OpenSSH on Windows nodes” on page 334.

Clusters that only include Windows nodes may use the **mmwinrsh** utility that comes with GPFS. The fully-qualified path name is **/usr/lpp/mmfs/bin/mmwinrsh**. For more information about configuring Windows GPFS clusters, see **mmwinservctl command** in *IBM Spectrum Scale: Command and Programming Reference*.

By default, you can issue GPFS administration commands from any node in the cluster. Optionally, you can choose a subset of the nodes that are capable of running administrative commands. In either case, the nodes that you plan to use for administering GPFS must be able to run remote shell commands on any other node in the cluster as user **root** without the use of a password and without producing any extraneous messages. You can also use sudo wrappers for this purpose.

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

Remote file copy command

The GPFS commands must maintain a number of configuration files across all nodes in the cluster. To achieve this, the GPFS commands use the remote file copy command that you specify on the **mmcrcluster** command or the **mmchcluster** command.

The default remote file copy program is **scp**. You can designate the use of a different remote file copy command by specifying its fully-qualified path name on the **mmcrcluster** command or the **mmchcluster** command. The remote file copy command must adhere to the same syntax as **scp**, but it can implement an alternate authentication mechanism.

Clusters that include both UNIX and Windows nodes must use **scp** for the remote copy command. For more information, see “Installing and configuring OpenSSH on Windows nodes” on page 334.

The file copy command **rcp** cannot be used in a cluster that contains Windows Server nodes. However, clusters that include only Windows nodes may use the **mmwinrcp** utility that comes with GPFS. The fully-qualified path name is /usr/lpp/mmfs/bin/mmwinrcp. For more information about configuring Windows GPFS clusters, see *mmwinservctl command* in *IBM Spectrum Scale: Command and Programming Reference*.

The nodes that you plan to use for administering GPFS must be able to copy files using the remote file copy command to and from any other node in the cluster without the use of a password and without producing any extraneous messages.

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

Cluster name

Provide a name for the cluster by issuing the **-C** option on the **mmcrcluster** command.

If the user-provided name contains dots, it is assumed to be a fully qualified domain name. Otherwise, to make the cluster name unique in a multiple cluster environment, GPFS appends the domain name of the primary cluster configuration server. If the **-C** option is not specified, the cluster name defaults to the hostname of the primary cluster configuration server. The name of the cluster may be changed at a later time by issuing the **-C** option on the **mmchcluster** command.

The cluster name is applicable when GPFS file systems are mounted by nodes belonging to other GPFS clusters. See the **mmauth** and the **mmremotecluster** commands.

User ID domain for the cluster

This option is the user ID domain for a cluster when accessing a file system remotely.

This option is further explained in *IBM Spectrum Scale: Administration Guide* and 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).

Starting GPFS automatically

You can specify whether to start the GPFS daemon automatically on a node when it is started.

Whether or not GPFS automatically starts is determined using the **autoload** parameter of the **mmchconfig** command. The default is *not* to automatically start GPFS on all nodes. You may change this by specifying **autoload=yes** using the **mmchconfig** command. This eliminates the need to start GPFS by issuing the **mmstartup** command when a node is booted.

The autoload parameter can be set the same or differently for each node in the cluster. For example, it may be useful to set **autoload=no** on a node that is undergoing maintenance since operating system upgrades and other software can often require multiple reboots to be completed.

Cluster configuration file

GPFS provides default configuration values, so a cluster configuration file is not required to create a cluster.

You can provide an optional cluster configuration file at the time of cluster creation. This optional file can be useful if you already know the correct parameter values based on previous testing or if you are restoring a cluster and have a backup copy of configuration values that apply to most systems. Typically, however, this option is not used at cluster creation time, and configuration parameters are modified after the cluster is created (using the **mmchconfig** command).

For more information about the cluster configuration file, see the description of the **`mmcrcluster -c`** *`ConfigFile`* option in *IBM Spectrum Scale: Command and Programming Reference*.

Disk considerations

Designing a proper storage infrastructure for your IBM Spectrum Scale file systems is key to achieving performance and reliability goals. When deciding what disk configuration to use, consider three key areas: infrastructure, performance, and disk access method.

Infrastructure

- Ensure that you have sufficient disks to meet the expected I/O load. In IBM Spectrum Scale terminology, a disk may be a physical disk or a RAID device.
- Ensure that you have sufficient connectivity (adapters and buses) between disks and network shared disk servers.
- Determine whether you are within IBM Spectrum Scale limits. Starting with GPFS 3.1, the structural limit on the maximum number of disks in a file system increased from 2048 to 4096; however, IBM Spectrum Scale still enforces the original limit of 2048. Should your environment require support for more than 2048 disks, contact the IBM Support Center to discuss increasing the enforced limit. (However, the number of disks in your system is often constrained by products other than IBM Spectrum Scale.)
- For a list of storage devices tested with IBM Spectrum Scale, see the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).
- For Linux on Z, see the "Storage" topics "DASD device driver" and "SCSI-over-Fibre Channel device driver" in Device Drivers, Features, and Commands(www.ibm.com/support/knowledgecenter/api/content/linuxonibm/liaaf/lnz_r_dd.html) in the Linux on Z library overview.

Disk access method

- Decide how your disks will be connected. Supported types of disk connectivity include the following configurations:
  1. All disks SAN-attached to all nodes in all clusters that access the file system

     In this configuration, every node sees the same disk simultaneously and has a corresponding disk device entry.
  2. Each disk connected to multiple NSD server nodes (up to eight servers), as specified on the server list

     In this configuration, a single node with connectivity to a disk performs data shipping to all other nodes. This node is the first NSD server specified on the NSD server list. You can define additional NSD servers on the server list. Having multiple NSD servers guards against the loss of a single NSD server. When using multiple NSD servers, all NSD servers must have connectivity to the same disks. In this configuration, all nodes that are not NSD servers will receive their data over the local area network from the first NSD server on the server list. If the first NSD server fails, the next available NSD server on the list will control data distribution.
  3. A combination of SAN-attached and an NSD server configuration.

     Configuration consideration:
     - If the node has a physical attachment to the disk and that connection fails, the node switches to using a specified NSD server to perform I/O. For this reason, it is recommended that you define NSDs with multiple servers, even if all nodes have physical attachments to the disk.
     - Configuring IBM Spectrum Scale disks without an NSD server stops the serving of data when the direct path to the disk is lost. This may be a preferable option for nodes requiring a higher speed data connection provided through a SAN as opposed to a lower speed network NSD server connection. Parallel jobs using MPI often have this characteristic.
     - The **-o useNSDserver** file system mount option on the **mmmount**, **mount**, **mmchfs**, and **mmremotefs** commands can be used to specify the disk discovery, and limit or eliminate switching from local access to NSD server access, or the other way around.
- Decide whether you will use storage pools to manage your disks.

  Storage pools allow you to manage your file system's storage in groups. You can partition your storage based on such factors as performance, locality, and reliability. Files are assigned to a storage pool based on defined policies.

  Policies provide for the following:
  - Placing files in a specific storage pool when the files are created
  - Migrating files from one storage pool to another
  - File deletion based on file characteristics

  For more information, see *Storage pools* in *IBM Spectrum Scale: Administration Guide*.

Disk considerations include:
1. "Network Shared Disk (NSD) creation considerations"
2. "NSD server considerations" on page 148
3. "File system descriptor quorum" on page 149
4. "Preparing direct access storage devices (DASD) for NSDs" on page 150

Network Shared Disk (NSD) creation considerations

You must prepare each physical disk you intend to use with GPFS by first defining it as a Network Shared Disk (NSD) using the **mmcrnsd** command.

On Windows, GPFS will only create NSDs from empty disk drives. **mmcrnsd** accepts Windows *Basic* disk or *Unknown/Not Initialized* disks. It always re-initializes these disks so that they become *Basic GPT Disks* with a single *GPFS partition*. NSD data is stored in GPFS partitions. This allows other operating system components to recognize that the disks are used. **mmdelnsd** deletes the partition tables created by **mmcrnsd**.

A new NSD format was introduced with GPFS 4.1. The new format is referred to as NSD v2, and the old format is referred to as NSD v1. The NSD v1 format is compatible with GPFS releases prior to 4.1. The latest GPFS release recognizes both NSD v1 and NSD v2 formatted disks.

The NSD v2 format provides the following benefits:

- On Linux, includes a partition table so that the disk is easily recognized as a GPFS device
- Adjusts data alignment to support disks with a 4 KB physical block size
- Adds backup copies of some key GPFS data structures
- Expands some reserved areas to allow for future growth

Administrators do not need to select one format or the other when managing NSDs. GPFS will always create and use the correct format based on the **minReleaseLevel** for the cluster and the file system version. When **minReleaseLevel** (as reported by **mmlsconfig**) is less than 4.1.0.0, **mmcrnsd** will only create NSD v1 formatted disks. When **minReleaseLevel** is at least 4.1.0.0, **mmcrnsd** will only create NSD v2 formatted disks. In this second case, however, the NSD format may change dynamically when the NSD is added to a file system so that the NSD is compatible with the file system version.

On Linux, NSD v2 formatted disks include a GUID Partition Table (GPT) with a single partition. The GPT allows other operating system utilities to recognize when a disk is owned by GPFS, which helps prevent inadvertent data corruption. After running **mmcrnsd**, Linux utilities like **parted** can show the partition table. When an NSD v2 formatted disk is added to a 3.5 or older file system, its format is changed to NSD v1 and the partition table is converted to an MBR (MS-DOS compatible) type.

Note: Leftover persistent reserve (PR) keys can cause problems such as reservation conflicts in multipath, which can in turn cause I/O failure. In such cases, it is necessary to clean up leftover PR keys on a fresh install. For a detailed procedure, see *Clearing a leftover Persistent Reserve reservation* in *IBM Spectrum Scale: Problem Determination Guide*.

The **mmcrnsd** command expects a stanza file as input. For details, see the following topics in *IBM Spectrum Scale: Command and Programming Reference* and *IBM Spectrum Scale: Administration Guide*:

- *Stanza files*
- **mmchdisk** *command*
- **mmchnsd** *command*
- **mmcrfs** *command*
- **mmcrnsd** *command*

NSD server considerations

If you plan to use NSD servers to remotely serve disk data to other nodes, as opposed to having disks SAN-attached to all nodes, you should consider the total computing and I/O load on these nodes.

- Will your Network Shared Disk servers be dedicated servers or will you also be using them to run applications? If you will have non-dedicated servers, consider running less time-critical applications on these nodes. If you run time-critical applications on a Network Shared Disk server, servicing disk requests from other nodes might conflict with the demands of these applications.
- The special functions of the file system manager consume extra processing time. If possible, avoid using a Network Shared Disk server as the file system manager. The Network Shared Disk server consumes both memory and processor cycles that could impact the operation of the file system manager. See “The file system manager” on page 10.

- The actual processing capability required for Network Shared Disk service is a function of the application I/O access patterns, the type of node, the type of disk, and the disk connection. You can later run **iostat** on the server to determine how much of a load your access pattern will place on a Network Shared Disk server.
- Providing sufficient disks and adapters on the system to yield the required I/O bandwidth. Dedicated Network Shared Disk servers should have sufficient disks and adapters to drive the I/O load you expect them to handle.
- Knowing approximately how much storage capacity you will need for your data.

You should consider what you want as the default behavior for switching between local access and NSD server access in the event of a failure. To set this configuration, use the **-o useNSDserver** file system mount option of the **mmmount**, **mount**, **mmchfs**, and **mmremotefs** commands to:

- Specify the disk discovery behavior
- Limit or eliminate switching from either:
  - Local access to NSD server access
  - NSD server access to local access

You should consider specifying how long to wait for an NSD server to come online before allowing a file system mount to fail because the server is not available. The **mmchconfig** command has these options:

nsdServerWaitTimeForMount

When a node is trying to mount a file system whose disks depend on NSD servers, this option specifies the number of seconds to wait for those servers to come up. If a server recovery is taking place, the wait time you are specifying with this option starts after recovery completes.

Note: The decision to wait for servers is controlled by the **nsdServerWaitTimeWindowOnMount** option.

nsdServerWaitTimeWindowOnMount

Specifies a window of time (in seconds) during which a mount can wait for NSD servers as described for the **nsdServerWaitTimeForMount** option. The window begins when quorum is established (at cluster startup or subsequently), or at the last known failure times of the NSD servers required to perform the mount.

Notes:

1. When a node rejoins a cluster, it resets all the failure times it knew about within that cluster.
2. Because a node that rejoins a cluster resets its failure times within that cluster, the NSD server failure times are also reset.
3. When a node attempts to mount a file system, GPFS checks the cluster formation criteria first. If that check falls outside the window, it will then check for NSD server fail times being in the window.

File system descriptor quorum

A GPFS structure called the *file system descriptor* is initially written to every disk in the file system and is replicated on a subset of the disks as changes to the file system occur, such as the adding or deleting of disks.

Based on the number of failure groups and disks, GPFS creates one to five replicas of the descriptor:

- If there are at least five different failure groups, five replicas are created.
- If there are at least three different disks, three replicas are created.
- If there are only one or two disks, a replica is created on each disk.

Once it decides how many replicas to create, GPFS picks disks to hold the replicas, so that all replicas are in different failure groups, if possible, to reduce the risk of multiple failures. In picking replica locations, the current state of the disks is taken into account. Stopped or suspended disks are avoided. Similarly, when a failed disk is brought back online, GPFS might rebalance the file system descriptors in order to assure reliability across the failure groups. The disks used to hold the file system descriptor replicas can be seen by running the **mmlsdisk** *fsname* **-L** command and looking for the string **desc** in the **remarks** column.

GPFS requires that a majority of the replicas on the subset of disks remain available to sustain file system operations:

- If there are at least five different replicas, GPFS can tolerate a loss of two of the five replicas.
- If there are at least three replicas, GPFS can tolerate a loss of one of the three replicas.
- If there are fewer than three replicas, a loss of one replica might make the descriptor inaccessible.

The loss of all disks in a disk failure group might cause a majority of file systems descriptors to become unavailable and inhibit further file system operations. For example, if your file system is backed up by three or more disks that are assigned to two separate disk failure groups, one of the failure groups will be assigned two of the file system descriptor replicas, while the other failure group will be assigned only one replica. If all of the disks in the disk failure group that contains the two replicas were to become unavailable, the file system would also become unavailable. To avoid this particular scenario, you might want to introduce a third disk failure group consisting of a single disk that is designated as a **descOnly** disk. This disk would exist solely to contain a replica of the file system descriptor (that is, it would not contain any file system metadata or data). This disk should be at least 128MB in size.

For more information, see “Network Shared Disk (NSD) creation considerations” on page 147 and *Establishing disaster recovery for your GPFS cluster* in *IBM Spectrum Scale: Administration Guide*.

Preparing direct access storage devices (DASD) for NSDs

When preparing direct access storage devices (DASD) for NSDs, see the table “Disk hardware tested with GPFS for Linux on Z” in the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

Preparing your environment for use of extended count key data (ECKD) devices:

If your GPFS cluster includes Linux on Z instances, do not use virtual reserve/release.

Instead, follow the process described in Sharing DASD without Using Virtual Reserve/Release (www.ibm.com/support/knowledgecenter/SSB27U_6.3.0/com.ibm.zvm.v630.hcpa5/hcpa5259.htm). Data integrity is handled by GPFS itself.

Preparing an ECKD™ device for GPFS

To prepare an ECKD device for GPFS, complete these steps on a single node:

1. Ensure that the ECKD device is online. To set it online, issue the following command:

   ```
   chccwdev -e device_bus_id
   ```

   where *device_bus_id* identifies the device to be configured. *device_bus_id* is a device number with a leading **0.***n*, where *n* is the subchannel set ID. For example:

   ```
   chccwdev -e 0.0.3352
   ```

2. Low-level format the ECKD using one of the following commands.

   Note: GPFS supports ECKD disks in either compatible disk layout (CDL) format or Linux disk layout (LDL) format. The DASD must be formatted with a block size of 4096.

   - To specify CDL format, issue the following command:

     ```
     dasdfmt -d cdl device
     ```

     There is no need to specify a block size value, as the default value is 4096.

- To specify LDL format, issue the following command:

  ```
  dasdfmt -d ldl device
  ```

  There is no need to specify a block size value, as the default value is 4096.

  In both of these commands, *device* is the node of the device. For example:

  ```
  dasdfmt -d cdl /dev/dasda
  ```

3. This step is for *CDL disks only*. It is an *optional* step because partitioning is optional for CDL disks.

   If you wish to partition the ECKD and create a single partition that spans the entire device, use the following command:

   ```
   fdasd -a device
   ```

   Notes:

   - This step is not required for LDL disks because the **dasdfmt -d ldl** command issued in the previous step automatically creates a single Linux partition on the disk.
   - If a CDL disk is partitioned, the partition name should be specified in the stanza input file for **mmcrnsd**. If a CDL disk is not partitioned, the disk name should be specified in the stanza input file.

For more information about all of these commands, see the following:

- "Commands for Linux on Z" topic in Device Drivers, Features, and Commands(www.ibm.com/support/knowledgecenter/api/content/linuxonibm/liaaf/lnz_r_dd.html) in the Linux on Z library overview.
- "Getting started with Elastic Storage for Linux on Z based on GPFS technology" white paper, available on the Welcome Page for IBM Spectrum Scale in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/ibmspectrumscale_welcome.html).

Repeat these steps for each ECKD to be used with GPFS.

After preparing the environment, set the ECKD devices online on the other nodes.

Note: After partitioning and formatting an ECKD device, the partition information is not distributed automatically to other cluster nodes. If the ECKD device is shared with different cluster nodes when they are online. It is recommended to set the ECKD device offline and then back online. This procedure will update the partition information for the specific ECKD device on the cluster node.

Always ensure that the ECKD devices are online before starting GPFS. To automatically set ECKD devices online at system start, see the documentation for your Linux distribution.

File system creation considerations

File system creation involves anticipating usage within the file system and considering your hardware configurations. Before creating a file system, consider how much data will be stored and how great the demand for the files in the system will be.

Each of these factors can help you to determine how much disk resource to devote to the file system, which block size to choose, where to store data and metadata, and how many replicas to maintain. For the latest supported file system size, see the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

Your GPFS file system is created by issuing the **mmcrfs** command. Table 20 on page 152 details the file system creation options specified on the **mmcrfs** command, which options can be changed later with the **mmchfs** command, and what the default values are.

To move an existing file system into a new GPFS cluster, see *Exporting file system definitions between clusters* in *IBM Spectrum Scale: Administration Guide*.

Table 20. File system creation options

Options	**mmcrfs**	**mmchfs**	**Default value**
Device name of the file system. See "Device name of the file system" on page 155.	X	X	none
DiskDesc for each disk in your file system. **Note:** The use of disk descriptors is discouraged. See "Disks for your file system" on page 155.	X	Issue the **mmadddisk** or **mmdeldisk** command to add or delete disks from the file system.	none
-F *StanzaFile* specifies a file that contains a list of NSD stanzas. For more information, see *Stanza files* in *IBM Spectrum Scale: Administration Guide*.	X	Issue the **mmadddisk** or **mmdeldisk** command to add or delete disks as indicated in the stanza file.	none
-A {**yes** | **no** | **automount**} to determine when to mount the file system. See "Deciding how the file system is mounted" on page 155.	X	X	**yes**
-B *BlockSize* to set the data block size: **64K**, **128K**, **256K**, **512K**, **1M**, **2M**, **4M**, **8M** or **16M**. See "Block size" on page 155.	X	This value cannot be changed without re-creating the file system.	**4 MiB**
-D {**posix** | **nfs4**} semantics for a deny-write open lock See "NFS V4 deny-write open lock" on page 155.	X	X	**nfs4**
-E {**yes** | **no**} to report exact mtime values. See "mtime values" on page 158.	X	X	**yes**
-i *InodeSize* to set the size of inodes: **512**, **1024**, or **4096** bytes.	X	This value cannot be changed.	**4096**
-j {**cluster** | **scatter**} to determine the block allocation map type. See "Block allocation map" on page 159.	X	NA	See "Block allocation map" on page 159.
-k {**posix** | **nfs4** | **all**} to determine the authorization types supported by the file system. See "File system authorization" on page 159.	X	X	**all**
-K {**no** | **whenpossible** | **always**} to enforce strict replication. See "Type of replication" on page 159.	X	X	**whenpossible**

Table 20. File system creation options (continued)

Options	mmcrfs	mmchfs	Default value
-L *LogFileSize* to specify the size of the internal log files. See "GPFS recovery logs" on page 15.	X	X	The default is **4 MiB** or the metadata block size, whichever is larger.
-m *DefaultMetadataReplicas* See "File system replication parameters" on page 160.	X	X	**1**
-M *MaxMetadataReplicas* See "File system replication parameters" on page 160.	X	This value cannot be changed.	**2**
-n *NumNodes* that will mount the file system. See "Number of nodes mounting the file system" on page 161.	X	X	**32**
-o *MountOptions* to be passed to the **mount** command. See "Assign mount command options" on page 161.	NA	X	none
-Q {**yes** | **no**} to activate quota. See "Enabling quotas" on page 161.	X	X	**no**
-r *DefaultDataReplicas* See "File system replication parameters" on page 160.	X	X	**1**
-R *MaxDataReplicas* See "File system replication parameters" on page 160.	X	This value cannot be changed.	**2**
-S {**yes** | **no** | **relatime**} to control how the **atime** value is updated. See "atime values" on page 157.	X	X	The default value is **relatime** if **minReleaseLevel** is 5.0.0 or greater when the file system is created. Otherwise the default value is **no**.
-t *DriveLetter* See "Windows drive letter" on page 161.	X	X	none
-T *Mountpoint* See "Mountpoint directory" on page 161.	X	X	**/gpfs/***DeviceName*
-V {**full** | **compat**} to change the file system format to the latest level. See "Changing the file system format to the latest level" on page 163	NA	X	none

Table 20. File system creation options (continued)

Options	mmcrfs	mmchfs	Default value
-v {**yes** | **no**} to verify disk usage. See "Verifying disk usage" on page 163.	X	NA	**yes**
-W *NewDeviceName* to assign a new device name to the file system.	NA	X	none
-z {**yes** | **no**} to enable DMAPI See "Enabling DMAPI" on page 163.	X	X	**no**
--filesetdf to specify (when quotas are enforced for a fileset) whether the **df** command will report numbers based on the quotas for the fileset and not for the total file system.	X	X	**--nofilesetdf**
--inode-limit *MaxNumInodes* [:*NumInodesToPreallocate*] to determine the maximum number of files in the file system. See "Specifying the maximum number of files that can be created" on page 164.	X	X	file system size/**1** MiB
--log-replicas *LogReplicas* to specify the number of recovery log replicas.	X	X	none
--metadata-block-size *MetadataBlockSize* to specify the block size for the system storage pool. See "Block size" on page 155.	X	NA	The default is the same as the value set for **-B** *BlockSize*.
--mount-priority *Priority* to control the order in which the individual file systems are mounted at daemon startup or when one of the **all** keywords is specified on the **mmmount** command.	X	X	**0**
--perfileset-quota to set the scope of user and group quota limit checks to the individual fileset level.	X	X	**--noperfileset-quota**
--rapid-repair to keep track of incomplete replication on an individual file block basis (as opposed to the entire file).	NA	X	none
--version *VersionString* to enable only the file system features that are compatible with the specified release. See "Enabling file system features" on page 164.	X	NA	The default value is defined by the current committed function level for the cluster **minReleaseLevel**, for example 4.2.2.0 or other values, depending on the cluster.

Notes:

1. X – indicates that the option is available on the command.
2. NA (not applicable) – indicates that the option is not available on the command.

Device name of the file system

File system names must be unique within a GPFS cluster. However, two different clusters can have two distinct file systems with the same name.

The device name of the file system does not need to be fully qualified. **fs0** is as acceptable as **/dev/fs0**. The name cannot be the same as an existing entry in **/dev**.

Note: If your cluster includes Windows nodes, the file system name should be no longer than 31 characters.

NFS V4 deny-write open lock

You can specify whether a deny-write open lock blocks writes, which is expected and required by NFS V4, Samba, and Windows.

For more information, see *Managing GPFS access control lists* in *IBM Spectrum Scale: Administration Guide*.

nfs4 Must be specified for file systems supporting NFS V4 and file systems mounted on Windows. This is the default.

posix Specified for file systems supporting NFS V3 or ones which are not NFS exported.

posix allows NFS writes even in the presence of a deny-write open lock.

Disks for your file system

Disks must be defined as network shared disks (NSDs) before they can be added to a GPFS file system.

NSDs are created using the **mmcrnsd** command. Use the **mmlsnsd -F** command to display a list of available NSDs. Use the **mmadddisk** to add the NSD after you create it.

See "Disk considerations" on page 146.

Deciding how the file system is mounted

Specify when the file system is to be mounted:

yes When the GPFS daemon starts. This is the default.

no Manual mount.

automount
When the file system is first accessed.

This can be changed at a later time by using the **-A** option on the **mmchfs** command.

Considerations:

1. GPFS mount traffic may be lessened by using the automount feature to mount the file system when it is first accessed instead of at GPFS startup. Automatic mounts only produce additional control traffic at the point that the file system is first used by an application or user. Mounts at GPFS startup on the other hand produce additional control traffic at every GPFS startup. Thus startup of hundreds of nodes at once may be better served by using automatic mounts.
2. Automatic mounts will fail if the node does not have the operating system's automount support enabled for the file system.
3. When exporting file systems for NFS mounts, it may be useful to mount the file system when GPFS starts.

Block size

Choose the file system block size based on the projected workload of the file system and the type of storage that it uses.

A block is the largest contiguous amount of disk space that can be allocated to a file and is therefore the largest amount of data that can be accessed in a single I/O operation. A subblock is the smallest unit of contiguous disk space that can be allocated. The size of a subblock and the number of subblocks in a block are determined by the block size. IBM Spectrum Scale supports the following file system block sizes: 64 KiB, 128 KiB, 256 KiB, 512 KiB, 1 MiB, 2 MiB, 4 MiB, 8 MiB and 16 MiB (for IBM Spectrum Scale RAID only). You can choose the file system block size based on the workload and the type of storage. For more information, see the topic *mmcrfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

Note: A *fragment* is a contiguous group of one or more subblocks. Files smaller than one block size are stored in fragments. Large files are stored in a number of full blocks plus zero or more subblocks to hold the data at the end of the file.

See the following tables for the suggested file system block sizes.

Table 21. Suggested file system block size per I/O type with examples of applications

I/O Type	**Examples of applications**	**Block size**
Large sequential I/O	Scientific computing, Digital media, File-based analytics	1 -16 MiB
Relational database	DB2®, Oracle, Small files on ESS	512 KiB
Small sequential I/O	General File Service, Email, Web applications	256 KiB

Table 22. Suggested file system block size per workload and configuration type.

Workload	**Configuration type**	**Block size**
SAP HANA	ESS GL	1 MiB for metadata 16 MiB for data
SAP HANA	FPO	1 MiB for single pool 256 KiB for metadata only 2 MiB for data only
Hadoop	ESS GL	1 MiB for metadata only 8 MiB for data only
Hadoop	FPO	256 KiB for metadata only 2 MiB for data only
Spark	FPO	256 KiB for metadata only 2 MiB for data only
SAP Sybase IQ	ESS GL	256 KiB - 1 MiB for metadata only 16 MiB for data only
Healthcare (Medical Imaging)	ESS GL	256 KiB for metadata only 1 MiB for data only
Healthcare (Medical Imaging)	Other Storage - IBM Spectrum Scale deployed with Block Storage at the backend. See the tip below.	256 KiB for metadata and data
Archive	Other Storage - IBM Spectrum Scale deployed with Block Storage at the backend. See the tip below.	Depends on the filesize.

Table 22. Suggested file system block size per workload and configuration type. (continued)

Workload	**Configuration type**	**Block size**
ECM	Other Storage - IBM Spectrum Scale deployed with Block Storage at the backend. See the tip below.	256 KiB Unless the content is very large files (For example - Videos).
Oracle	Other Storage - IBM Spectrum Scale deployed with Block Storage at the backend. See the tip below.	256 KiB for metadata and data
Technical Computing	ESS GL	1 MiB metadata 4 - 16 MiB data depending on importance of peak sequential performance.
SAS	ESS GL	1 MiB metadata 8 - 16 MiB depending on the SASBUF size (128 KiB or 256 KiB)
Enterprise File (Misc Projects, data sharing)	Other Storage - IBM Spectrum Scale deployed with Block Storage at the backend. See the tip below.	256 KiB metadata and data

Note: To determine the ideal file system block size for your workload, it is best to run performance tests on IBM Spectrum Scale using your workload, or a representative file system benchmark with different file system block sizes

For more information about IBM Spectrum Scale RAID, see *IBM Spectrum Scale RAID: Administration* in the Elastic Storage Server documentation on IBM Knowledge Center.

The **--metadata-block-size** option on the **mmcrfs** command allows a different block size to be specified for the system storage pool, provided its usage is set to **metadataOnly**. This can be beneficial if the file system block size is larger than 1 MB. Valid values are the same as those listed for the **-B** option.

Tip: If you plan to use RAID 5 or 6 devices in your file system, a larger block size might be more effective by reducing the penalties involved in small block-write operations. For example, in a RAID configuration using 4-8 data disks and 2 parity disk (a 84+2P - RAID 6 configuration), which uses a 64 KB stripe size, the optimal file system block size would be an integral multiple of 512 KB (8 data disks × 64 KB stripe size = 512 KB). A block size of an integral multiple of 256 KB results in a single data **write** that encompasses the 8 data disks and a parity-write to the parity disk. If a block size smaller than 512 KB, such as 256 KB, is used with the same RAID configuration, **write** performance might be degraded by the read-modify-write behavior depending on the Storage hardware.

The choice of block size affects the performance of certain metadata operations, in particular, block allocation performance. The IBM Spectrum Scale block allocation map is stored in blocks, similar to regular files. When the block size is small:

- More blocks are required to store a given amount of data resulting in additional work to allocate those blocks
- One block of allocation map data contains less information

Note: The choice of block size is particularly important for large file systems. For file systems larger than 100 TB, you should use a block size of at least 256 KB.

atime values

atime is a standard file attribute that represents the time when the file was last accessed.

The file attribute **atime** is updated locally in memory, but the value is not visible to other nodes until after the file is closed. To get an accurate value of **atime**, an application must call subroutine **gpfs_stat** or **gpfs_fstat**.

You can control how **atime** is updated with the **-S** option of the **mmcrfs** and **mmchfs** commands:

-S no
The **atime** attribute is updated whenever the file is read. This setting is the default if the minimum release level (**minReleaseLevel**) of the cluster is less than 5.0.0 when the file system is created.

-S yes
The **atime** attribute is not updated. The subroutines **gpfs_stat** and **gpfs_fstat** return the time that the file system was last mounted with **-S no**.

Note: This setting can cause a problem if you have a file management policy that depends on the **ACCESS_TIME** attribute. For more information, see the topic *Exceptions to Open Group technical standards* in the *IBM Spectrum Scale: Administration Guide*.

-S relatime
The **atime** attribute is updated whenever the file is read, but only if one of the following conditions is true:

- The current file access time (**atime**) is earlier than the file modification time (**mtime**).
- The current file access time (**atime**) is greater than the **atimeDeferredSeconds** attribute. For more information, see the topic *mmchconfig command* in the *IBM Spectrum Scale: Command and Programming Reference*.

This setting is the default if the minimum release level (**minReleaseLevel**) of the cluster is 5.0.0 or greater when the file system is created.

You can temporarily override the value of the **-S** setting by specifying a mount option specific to IBM Spectrum Scale when you mount the file system. The mount option persists until the file system is unmounted. The following table shows how the mount options correspond to the **-S** settings:

Table 23. Correspondence between **mount** *options and the* **-S** *option in* **mmcrfs** *and* **mmchfs**

-S option of mmcrfs and mmchfs commands.	Equivalent mount option; persists until file system is unmounted.	Effect – see topic text for details.
no	**norelatime**	Allow **atime** to be updated.
yes	**noatime**	Do not allow **atime** to be updated.
relatime	**relatime**	Allow **atime** to be updated if a condition is met.

For more information, see the topic *Mount options specific to IBM Spectrum Scale* in the *IBM Spectrum Scale: Administration Guide*.

mtime values

mtime is a standard file attribute that represents the time when the file was last modified.

The **-E** parameter controls when the **mtime** is updated. The default is **-E yes**, which results in standard interfaces including the **stat()** and **fstat()** calls reporting exact **mtime** values. Specifying **-E no** results in the **stat()** and **fstat()** calls reporting the **mtime** value available at the completion of the last sync period. This may result in the calls not always reporting the exact **mtime**. Setting **-E no** can affect backup operations, AFM, and AFM DR functions that rely on the last modified time or the operation of policies using the **MODIFICATION_TIME** file attribute.

For more information, see *Exceptions to the Open Group technical standards* in *IBM Spectrum Scale: Administration Guide*.

Block allocation map

GPFS has two different methods of allocating space in a file system. The **-j** parameter specifies the block allocation map type to use when creating a file system. The block allocation map type cannot be changed once the file system is created.

When allocating blocks for a given file, GPFS first uses a round-robin algorithm to spread the data across all of the disks in the file system. After a disk is selected, the location of the data block on the disk is determined by the block allocation map type.

The two types of allocation methods are **cluster** and **scatter**:

cluster
GPFS attempts to allocate blocks in clusters. Blocks that belong to a given file are kept next to each other within each cluster.

This allocation method provides better disk performance for some disk subsystems in relatively small installations. The benefits of clustered block allocation diminish when the number of nodes in the cluster or the number of disks in a file system increases, or when the file system free space becomes fragmented. The **cluster** allocation method is the default for GPFS clusters with eight or fewer nodes and for file systems with eight or fewer disks.

scatter GPFS chooses the location of the blocks randomly.

This allocation method provides more consistent file system performance by averaging out performance variations due to block location (for many disk subsystems, the location of the data relative to the disk edge has a substantial effect on performance). This allocation method is appropriate in most cases and is the default for GPFS clusters with more than eight nodes or file systems with more than eight disks.

This parameter for a given file system is specified at file system creation by using the **-j** option on the **mmcrfs** command, or allowing it to default. This value *cannot* be changed after the file system has been created.

File system authorization

The type of authorization for the file system is specified on the **-k** option on the **mmcrfs** command or changed at a later time by using the **-k** option on the **mmchfs** command.

posix Traditional GPFS access control lists (ACLs) only (NFS V4 and Windows ACLs are not allowed).

nfs4 Support for NFS V4 and Windows ACLs only. Users are not allowed to assign traditional ACLs to any file system objects.

all Allows for the coexistence of POSIX, NFS V4, and Windows ACLs within a file system. This is the default.

If the file system will be used for NFS and Swift protocol exports, **nfs4** is recommended.

If the file system will be used for SMB protocol exports, **nfs4** is required.

Type of replication

You can control the type of replication that IBM Spectrum Scale uses.

Run the `mmchfs` command with the `-K` option set to the preferred type of replication. The `-K` option can have the following values:

always
Indicates that strict replication is enforced.

whenpossible

Strict replication is enforced if the disk configuration allows it. If the number of failure groups is insufficient, strict replication is not enforced. This is the default value.

no Strict replication is not enforced. GPFS tries to create the needed number of replicas, but returns an **errno** of EOK if it can allocate at least one replica.

For more information on changing the attributes for replication, see the `mmchfs` command.

Internal log file

You can specify the internal log file size. Refer to “GPFS recovery logs” on page 15 for additional information.

File system replication parameters

The metadata (inodes, directories, and indirect blocks) and data replication parameters are set at the file system level and apply to all files. They are initially set for the file system when issuing the **mmcrfs** command.

They can be changed for an existing file system using the **mmchfs** command. When the replication parameters are changed, files created after the change are affected. To apply the new replication values to existing files in a file system, issue the **mmrestripefs** command.

Metadata and data replication are specified independently. Each has a default replication factor of 1 (no replication) and a maximum replication factor with a default of 2. Although replication of metadata is less costly in terms of disk space than replication of file data, excessive replication of metadata also affects GPFS efficiency because all metadata replicas must be written. In general, more replication uses more space.

Default metadata replicas:

The default number of copies of metadata for all files in the file system may be specified at file system creation by using the **-m** option on the **mmcrfs** command or changed at a later time by using the **-m** option on the **mmchfs** command.

This value must be equal to or less than *MaxMetadataReplicas*, and cannot exceed the number of failure groups with disks that can store metadata. The allowable values are 1, 2, or 3, with a default of 1 (no replication).

Maximum metadata replicas:

The maximum number of copies of metadata for all files in the file system can be specified at file system creation by using the **-M** option on the **mmcrfs** command.

The default is 2. The allowable values are 1, 2, or 3, but it cannot be less than the value of *DefaultMetadataReplicas*. This value cannot be changed after the file system is created.

Default data replicas:

The default replication factor for data blocks may be specified at file system creation by using the **-r** option on the **mmcrfs** command or changed at a later time by using the **-r** option on the **mmchfs** command.

This value must be equal to or less than *MaxDataReplicas*, and the value cannot exceed the number of failure groups with disks that can store data. The allowable values are 1, 2, and 3, with a default of 1 (no replication).

If you want to change the data replication factor for the entire file system, the data disk in each storage pool must have a number of failure groups equal to or greater than the replication factor. For example, you will get a failure with error messages if you try to change the replication factor for a file system to 2 but the storage pool has only one failure group.

Maximum data replicas:

The maximum number of copies of data blocks for a file can be specified at file system creation by using the **-R** option on the **mmcrfs** command. The default is 2.

The allowable values are 1, 2, and 3, but cannot be less than the value of *DefaultDataReplicas*. This value cannot be changed after the file system is created.

Number of nodes mounting the file system

The estimated number of nodes that will mount the file system may be specified at file system creation by using the **-n** option on the **mmcrfs** command or allowed to default to 32.

When creating a GPFS file system, over-estimate the number of nodes that will mount the file system. This input is used in the creation of GPFS data structures that are essential for achieving the maximum degree of parallelism in file system operations (see "GPFS architecture" on page 9). Although a larger estimate consumes a bit more memory, insufficient allocation of these data structures can limit the ability to process certain parallel requests efficiently, such as the allotment of disk space to a file. If you cannot predict the number of nodes, allow the default value to be applied. Specify a larger number if you expect to add nodes, but avoid wildly overestimating as this can affect buffer operations.

You can change the number of nodes using the **-n** option on the **mmchfs** command. Changing this value affects storage pools created after the value was set; so, for example, if you need to increase this value on a storage pool, you could change the value, create a new storage pool, and migrate the data from one pool to the other.

Windows drive letter

In a Windows environment, you must associate a drive letter with a file system before it can be mounted. The drive letter can be specified and changed with the **-t** option of the **mmcrfs** and **mmchfs** commands. GPFS does not assign a default drive letter when one is not specified.

The number of available drive letters restricts the number of file systems that can be mounted on Windows.

Note: Certain applications give special meaning to drive letters **A:**, **B:**, and **C:**, which could cause problems if they are assigned to a GPFS file system.

Mountpoint directory

Every GPFS file system has a default mount point associated with it. This mount point can be specified and changed with the **-T** option of the **mmcrfs** and **mmchfs** commands.

If you do not specify a mount point when you create the file system, GPFS will set the default mount point to **/gpfs/***DeviceName*.

Assign mount command options

Options may be passed to the file system **mount** command using the **-o** option on the **mmchfs** command.

In particular, you can choose the option to perform quota activation automatically when a file system is mounted.

Enabling quotas

The GPFS quota system can help you control file system usage.

Quotas can be defined for individual users, groups of users, or filesets. Quotas can be set on the total number of files and the total amount of data space consumed. When setting quota limits for a file system, the system administrator should consider the replication factors of the file system. Quota management takes replication into account when reporting on and determining if quota limits have been exceeded for both block and file usage. In a file system that has either data replication or metadata replication set to a value of two, the values reported on by both the **mmlsquota** and **mmrepquota** commands are double the value reported by the **ls** command.

Whether or not to enable quotas when a file system is mounted may be specified at file system creation by using the **-Q** option on the **mmcrfs** command or changed at a later time by using the **-Q** option on the **mmchfs** command. After the file system has been mounted, quota values are established by issuing the **mmedquota** command and activated by issuing the **mmquotaon** command. The default is to *not* have quotas activated.

GPFS levels are defined at three limits that you can explicitly set using the **mmedquota** and **mmdefedquota** commands:

Soft limit

Defines levels of disk space and files below which the user, group of users, or fileset can safely operate.

Specified in units of kilobytes (k or K), megabytes (m or M), or gigabytes (g or G). If no suffix is provided, the number is assumed to be in bytes.

Hard limit

Defines the maximum amount of disk space and number of files the user, group of users, or fileset can accumulate.

Specified in units of kilobytes (k or K), megabytes (m or M), or gigabytes (g or G). If no suffix is provided, the number is assumed to be in bytes.

Grace period

Allows the user, group of users, or fileset to exceed the soft limit for a specified period of time. The default period is one week. If usage is not reduced to a level below the soft limit during that time, the quota system interprets the soft limit as the hard limit and no further allocation is allowed. The user, group of users, or fileset can reset this condition by reducing usage enough to fall below the soft limit; or the administrator can increase the quota levels using the **mmedquota** or **mmdefedquota**.

For implications of quotas for different protocols, see *Implications of quotas for different protocols* in *IBM Spectrum Scale: Administration Guide*.

Default quotas:

Applying default quotas provides all new users, groups of users, or filesets with established minimum quota limits. If default quota values are not enabled, new users, new groups, or new filesets have a quota value of zero, which establishes no limit to the amount of space that can be used.

Default quota limits can be set or changed only if the **-Q yes** option is in effect for the file system. To set default quotas at the fileset level, the **--perfileset-quota** option must also be in effect. The **-Q yes** and **--perfileset-quota** options are specified when creating a file system with the **mmcrfs** command or changing file system attributes with the **mmchfs** command. Use the **mmlsfs** command to display the current settings of these quota options. Default quotas may then be enabled by issuing the **mmdefquotaon** command. Default values are established by issuing the **mmdefedquota** command.

Quota system files:

The GPFS quota system maintains three separate files that contain data about usage and limits.

These files reside in the root directory of the GPFS file systems when quotas are enabled:
- **user.quota**
- **group.quota**
- **fileset.quota**

All three **.quota** files are:
- Built with the information provided in the **mmedquota** and **mmdefedquota** commands.
- Updated through normal allocation operations throughout the file system and when the **mmcheckquota** command is issued.
- Readable by the **mmlsquota** and **mmrepquota** commands.

The **.quota** files are read from the root directory when mounting a file system with quotas enabled. When these files are read, one of three possible actions take place:
- The files contain quota information and the user wants these files to be used.
- The files contain quota information, however, the user wants different files to be used.

  To specify the use of different files, the **mmcheckquota** command *must* be issued prior to the **mount** of the file system.
- The files do not contain quota information. In this case the **mount** fails and appropriate error messages are issued. For more information regarding **mount** failures, see *IBM Spectrum Scale: Problem Determination Guide*.

Enabling DMAPI

Whether or not the file system can be monitored and managed by the GPFS Data Management API (DMAPI) may be specified at file system creation by using the **-z** option on the **mmcrfs** command or changed at a later time by using the **-z** option on the **mmchfs** command.

The default is *not* to enable DMAPI for the file system.

For more information about DMAPI for GPFS, see *IBM Spectrum Scale: Command and Programming Reference*.

Verifying disk usage

The **-v** option controls whether the **mmcrfs** command checks whether the specified disks can safely be added to the file system.

The default (**-v yes**) is to perform the check and fail the command if any of the disks appear to belong to some other GPFS file system. You should override the default behavior and specify **-v no** only if the **mmcrfs** command rejects the disks and you are certain that *all* of the disks indeed do not belong to an active GPFS file system. An example for an appropriate use of **-v no** is the case where an **mmcrfs** command is interrupted for some reason and you are reissuing the command. Another example would be if you are reusing disks from an old GPFS file system that was not formally destroyed with the **mmdelfs** command.

Important: Using **mmcrfs -v no** on a disk that already belongs to a file system will corrupt that file system.

Changing the file system format to the latest level

You can change the file system format to the latest format supported by the currently-installed level of GPFS by issuing the **mmchfs** command with the **-V full** option or the **-V compat** option.

The **full** option enables all new functionality that requires different on-disk data structures. This may cause the file system to become permanently incompatible with earlier releases of GPFS. The **compat** option enables only changes that are backward compatible with the previous GPFS release. If all GPFS nodes that are accessing a file system (both local and remote) are running the latest level, then it is safe to use **full** option. Certain features may require you to run the **mmmigratefs** command to enable them.

For more information, see *File system format changes between versions of GPFS* in *IBM Spectrum Scale: Administration Guide*.

Enabling file system features

By default, new file systems are created with all currently available features enabled.

Since this may prevent clusters that are running earlier GPFS releases from accessing the file system, you can enable only the file system features that are compatible with the specified release by issuing the **mmcrfs** command with the **--version** *Version* option. For more information, see *mmcrfs command* in *IBM Spectrum Scale: Command and Programming Reference*.

Specifying whether the df command will report numbers based on quotas for the fileset

You can specify (when quotas are enforced for a fileset) whether the **df** command will report numbers based on the quotas for the fileset and not for the total file system.

To do so, use the **--filesetdf | --nofilesetdf** option on either the **mmchfs** command or the **mmcrfs** command.

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

Specifying the maximum number of files that can be created

The maximum number of files that can be created can be specified by using the **--inode-limit** option on the **mmcrfs** command and the **mmchfs** command.

Allowable values, which range from the current number of created inodes (determined by issuing the **mmdf** command with the **-F** option) through the maximum number of files that are supported, are constrained by the formula:

```
maximum number of files = (total file system space) / (inode size + subblock size)
```

You can determine the inode size (**-i**) and subblock size (**-f**) of a file system by running the **mmlsfs** command. The maximum number of files in a file system may be specified at file system creation by using the **--inode-limit** option on the **mmcrfs** command, or it may be increased at a later time by using **--inode-limit** on the **mmchfs** command. This value defaults to the size of the file system at creation divided by 1 MB and cannot exceed the architectural limit. When a file system is created, 4084 inodes are used by default; these inodes are used by GPFS for internal system files.

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

The **--inode-limit** option applies only to the root fileset. When there are multiple inode spaces, use the **--inode-space** option of the **mmchfileset** command to alter the inode limits of independent filesets. The **mmchfileset** command can also be used to modify the root inode space. The **--inode-space** option of the **mmlsfs** command shows the sum of all inode spaces.

Inodes are allocated when they are used. When a file is deleted, the inode is reused, but inodes are never deallocated. When setting the maximum number of inodes in a file system, there is the option to preallocate inodes. However, in most cases there is no need to preallocate inodes because, by default, inodes are allocated in sets as needed. If you do decide to preallocate inodes, be careful not to preallocate more inodes than will be used; otherwise, the allocated inodes will unnecessarily consume metadata space that cannot be reclaimed.

These options limit the maximum number of files that may actively exist within a file system. However, the maximum number of files in the file system may be restricted further by GPFS so the control structures associated with each file do not consume all of the file system space.

Further considerations when managing inodes:

1. For file systems that are supporting parallel file creates, as the total number of free inodes drops below 5% of the total number of inodes, there is the potential for slowdown in file system access. Take this into consideration when creating or changing your file system. Use the **mmdf** command to display the number of free inodes.
2. Excessively increasing the value for the maximum number of files may cause the allocation of too much disk space for control structures.

Controlling the order in which file systems are mounted

You can control the order in which the individual file systems are mounted at daemon startup or when using the **mmmount** command with one of the **all** keywords specified for the file system.

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

To do so, use the **--mount-priority** *Priority* option on the **mmcrfs**, the **mmchfs**, or the **mmremotefs** command.

The shared root file system for protocols must be mounted before other file systems that will be used to export protocol data.

A sample file system creation

To create a file system called **gpfs2** with the following properties:

- The disks for the file system that are listed in the file **/tmp/gpfs2dsk**
- Automatically mount the file system when the GPFS daemon starts (**-A yes**)
- Use a block size of 256 KB (**-B 256K**)
- Expect to mount it on 32 nodes (**-n 32**)
- Set both the default metadata replication and the maximum metadata replication to two (**-m 2 -M 2**)
- Set the default data replication to one and the maximum data replication to two (**-r 1 -R 2**)
- Use a default mount point of `/gpfs2` (**-T /gpfs2**)

Enter:

```
mmcrfs /dev/gpfs2 -F /tmp/gpfs2dsk -A yes -B 256K -n 32 -m 2 -M 2 -r 1 -R 2 -T /gpfs2
```

The system displays information similar to:

```
The following disks of gpfs2 will be formatted on node k194p03.tes.nnn.com:
    hd25n09: size 17796014 KB
    hd24n09: size 17796014 KB
    hd23n09: size 17796014 KB
Formatting file system ...
Disks up to size 59 GB can be added to storage pool system.
Creating Inode File
  56 % complete on Mon Mar  3 15:10:08 2014
 100 % complete on Mon Mar  3 15:10:11 2014
Creating Allocation Maps
Clearing Inode Allocation Map
Clearing Block Allocation Map
  44 % complete on Mon Mar  3 15:11:32 2014
  90 % complete on Mon Mar  3 15:11:37 2014
 100 % complete on Mon Mar  3 15:11:38 2014
Completed creation of file system /dev/gpfs2.
mmcrfs: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

To confirm the file system configuration, issue the command:

```
mmlsfs gpfs2
```

The system displays information similar to:

```
flag                value                    description
------------------- ------------------------ -----------------------------------
 -f                 262144                   Minimum fragment (subblock) size in bytes
 -i                 512                      Inode size in bytes
 -I                 32768                    Indirect block size in bytes
 -m                 2                        Default number of metadata replicas
 -M                 2                        Maximum number of metadata replicas
 -r                 1                        Default number of data replicas
 -R                 2                        Maximum number of data replicas
 -j                 scatter                  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                 none                     Quotas accounting enabled
                    none                     Quotas enforced
                    none                     Default quotas enabled
 --perfileset-quota yes                      Per-fileset quota enforcement
 --filesetdf        yes                      Fileset df enabled?
 -V                 15.01 (4.2.0.0)          File system version
 --create-time      Wed Jan 18 17:22:25 2017 File system creation time
 -z                 no                       Is DMAPI enabled?
 -L                 262144                   Logfile size
 -E                 yes                      Exact mtime mount option
 -S                 yes                      Suppress atime mount option
 -K                 whenpossible             Strict replica allocation option
 --fastea           yes                      Fast external attributes enabled?
 --encryption       no                       Encryption enabled?
 --inode-limit      2015232                  Maximum number of inodes
 --log-replicas     0                        Number of log replicas (max 2)
 --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                 gpfs1001nsd;gpfs1002nsd  Disks in file system
 -A                 yes                      Automatic mount option
 -o                 none                     Additional mount options
 -T                 /gpfs2                   Default mount point
 --mount-priority   0                        Mount priority
```

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

Fileset considerations for creating protocol data exports

You can create exports on the entire file system, on sub-directories of a file system, or on filesets.

A fileset is a file system object that enables you to manage data at a finer granularity than the file system. You can perform administrative operations such as defining quotas, creating snapshots, and defining file placement policies and rules, and specifying inode space values at the fileset level, especially when the fileset is independent.

In IBM Spectrum Scale, you can create exports even without filesets. Depending on your data management strategy, choose either of the following ways to create exports:

Create exports on the entire file system or on sub-directories of the file system

In this option, the export represents a large space. You can create independent filesets over the directories in this space and have finer control over the export directory paths. Universities and organizations that require a departmental multi-tenancy solution can choose this option.

Review the following example to better understand this option.

As a storage administrator of an organization, you want to create separate storage space for every department and user of the organization:

1. Export the root directory of the file system.

   ```
   mmnfs export add /gpfs/fs0
   ```

   Note: You can create a sub-directory in the root directory and export it. For example: `mmnfs export add /gpfs/fs0/home`

   For more information, see **mmnfs command** in *IBM Spectrum Scale: Command and Programming Reference*.
2. Create independent filesets in the root directory, linked to the subdirectory /gpfs/fs0/home.

   In the following example, it is assumed that there is a user user1 that is a part of the group group/HR.

   ```
   mmcrfileset fs0 hr_fileset --inode-space=new
   mmlinkfileset fs0 hr_fileset -J /gpfs/fs0/home/hr
   mmcrfileset fs0 user1_fileset --inode-space=new
   mmlinkfileset fs0 user1_fileset -J /gpfs/fs0/home/user1
   ```

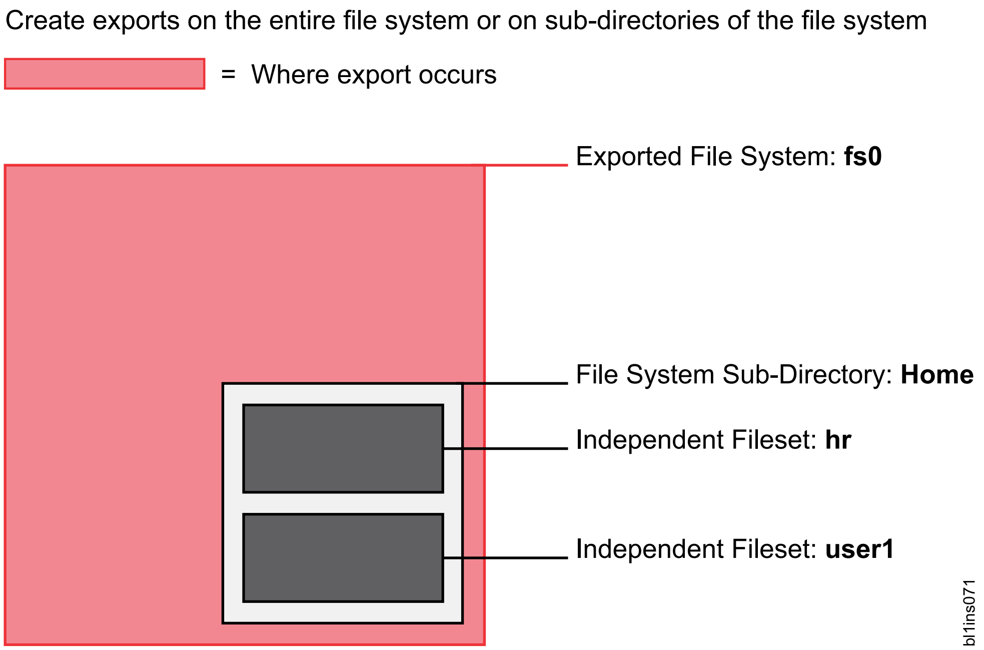
   For more information, see the following commands in the *IBM Spectrum Scale: Command and Programming Reference*.

   - **mmcrfileset command**
   - **mmlinkfileset command**
3. Similarly, create independent filesets for other departments and users.

   You can assign quota to each user and department via the independent filesets.

   The NFS and SMB clients can now mount the export and access their directories.

   Access to the data in the export directories is controlled via the group and user ACLs. In this case, only the users who are in group HR, which has group ACLs to read/write into the hr directory, can access the directory. The user user1 who is in the group group/HR can perform read/write to the user1 directory and the hr directory.


Create exports on the entire file system or on sub-directories of the file system
= Where export occurs
Exported File System: **fs0**
File System Sub-Directory: **Home**
Independent Fileset: **hr**
Independent Fileset: **user1**
bl1ins071

Create exports on independent filesets

In this option, the independent filesets represent discrete projects. One or more exports can be created on each fileset. You can apply the Information Lifecycle Management (ILM) policies over the filesets to automate the placement and management of file data. You can protect the export data by granting access permissions to specific workstations or IP addresses. Also, data in the exports can be preserved by the independent fileset's snapshot policy.

Review the following example to better understand this option.

You are a storage administrator of a private cloud hosting storage system that stores webcam data. You want to ensure that a webcam has access only to its storage directory. Also, you want the data analyzers to access all data so that they can look for activity and generate analytical reports.

1. Create an independent fileset `web_cam_data`.

   ```
   mmcrfileset fs0 web_cam_data --inode-space=new
   mmlinkfileset fs0 web_cam_data -J /gpfs/fs0/web_cam_data
   ```

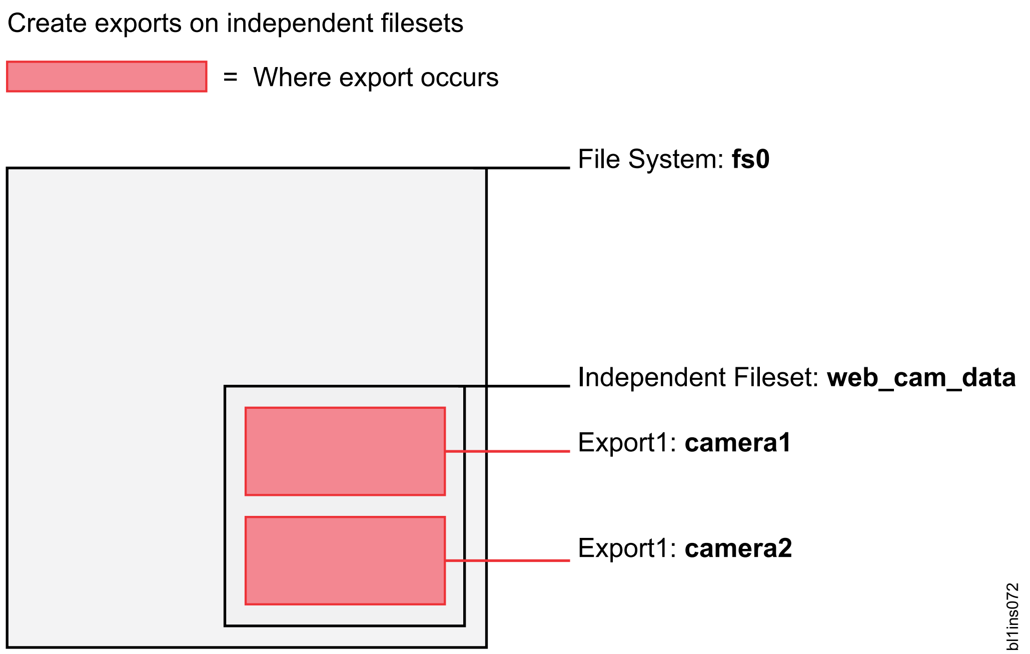
   Data from all webcams is stored in `/gpfs/fs0/web_cam_data`.

2. Create exports for both webcams.

   ```
   mkdir /gpfs/fs0/web_cam_data/camera1
   mkdir /gpfs/fs0/web_cam_data/camera2

   mmnfs export add "/gpfs/fs0/web_cam_data/camera1" \
   -c "198.51.100.2(Access_Type=RW);203.0.113.2(Access_Type=RO);203.0.113.3(Access_Type=RO)"
   mmnfs export add "/gpfs/fs0/web_cam_data/camera2" \
   -c "198.51.100.3(Access_Type=RW);203.0.113.2(Access_Type=RO);203.0.113.3(Access_Type=RO)"
   ```

The `webcam1` (IP: 198.51.100.2) mounts and records data to the `camera1` export and the `webcam2` (IP: 198.51.100.3) mounts and records data to the `camera2` export. The data analyzers (IP: 203.0.113.2 and 203.0.113.3) are given read access to both exports. Thus, the data is accessible only from the specified IP addresses.



Backup considerations for using IBM Spectrum Protect

If you are planning to use IBM Spectrum Protect to back up IBM Spectrum Scale file systems, some considerations apply.

Considerations for provisioning IBM Spectrum Protect servers to handle backup of IBM Spectrum Scale file systems

When backing up IBM Spectrum Scale file systems, the IBM Spectrum Protect server must be configured with adequate resources to handle a much larger load of client data, sessions, and mount points than typically encountered with single-user file systems such as workstations.

For cluster file system backup, a system architect must plan for multiple nodes (storage hosts) sending data in multiple streams, at high data rates to the server. Therefore, the IBM Spectrum Protect server must be equipped with the following:

- High bandwidth network connections to the IBM Spectrum Scale cluster
- Sufficient disk space organized into a IBM Spectrum Protect storage pool to be allocated to backup data from the cluster file system
- Sufficient number of mount points (or infinite for a disk class pool) accorded to the proxy client used for the cluster file system
- Sufficient number of IBM Spectrum Protect sessions permitted to the proxy client used for the cluster file system
- Sufficient number of physical tape drives to offload the backup data once it arrives into the disk pool

IBM Spectrum Protect data storage model

When using IBM Spectrum Protect for backing up IBM Spectrum Scale file systems, the IBM Spectrum Protect data storage architecture and its implications need to be considered.

In IBM Spectrum Scale, the following kinds of data are stored on disk:

- The file data that is the content stored in a file.
- The file metadata that includes all attributes related to the file. For example:

- – Create, access, and modify times
- – Size of the file, size occupied in file system, and number of blocks used
- – Inode information, owner user id, owning group id, and mode bits
- – POSIX rights or access control lists (ACLs)
- – Flags to indicate whether the file is immutable or mutable, read only, or append only
- – Extended attributes (EAs)

In IBM Spectrum Protect, the same file data and metadata is stored but the method of storing this data differs. The file content is stored in a IBM Spectrum Protect storage pool such as on disk or on tape while some of the metadata is stored in the IBM Spectrum Protect database. The primary reason for storing metadata in the IBM Spectrum Protect database is to provide fast access to information useful for backup requests.

However, not all metadata is stored in the IBM Spectrum Protect database. Access control lists (ACLs) and extended attributes (EAs) are stored with the file content in the storage pool (media depends on storage pool type). This has the following implications:

- When the ACL or EA of a file changes then the next backup job backs up the whole file again. This occurs because the file content, ACL, and EA are stored together in the IBM Spectrum Protect data pool, for example on tape and they need to be updated as one entity.

  Note: ACLs are inherited in the IBM Spectrum Scale file system. Therefore, an ACL on a top-level directory object can be inherited to all the descendant objects. ACL changes to a top-level directory object are therefore propagated down through the object tree hierarchy, rippling the change through all objects that inherited the original ACL. The number of files to be backed up increases even though nothing else in these files has changed. A surprisingly large backup workload can be induced by a seemingly small change to an ACL to a top level directory object.

  When IBM Spectrum Protect for Space Management capability is enabled, an ACL change such as this occurs to objects that are currently migrated to offline storage as well. These files will then need to be recalled during the next backup cycle to enable the updated ACL to be stored with the file data once again in their IBM Spectrum Protect backup storage pool.
- Renaming of a file also leads to a backup of the whole file because the IBM Spectrum Protect database is indexed by file object path name.

You can use the following approaches to mitigate the size of a backup workload when widely inherited ACLs are likely to be changed frequently.

- Avoid renaming directories that are close to the file system root.
- Avoid ACL and EA changes in migrated files as much as possible.
- Consider using the `skipacl` or `skipaclupdatecheck` options of the IBM Spectrum Protect client.

  Important: Be certain to note the implications of using these options by referring to *Clients options reference* in the *Backup-archive Client options and commands* section of the IBM Spectrum Protect documentation set on IBM Knowledge Center.

  Note: Using the `skipacl` option also omits EAs from the backup data store in the IBM Spectrum Protect backup pool. 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 either manually or automatically, by inheritance, to avoid an unauthorized user getting access to a file or a directory after it is restored.

How to identify backup and migration candidates

When using IBM Spectrum Protect for backing up IBM Spectrum Scale file systems, there are several choices for the method used to identify backup and migration candidates.

Limitations when using IBM Spectrum Protect Backup-Archive client to identify backup candidates

Using IBM Spectrum Protect Backup-Archive client to traverse IBM Spectrum Scale file systems to identify backup candidates does not scale well. For this reason, using the IBM Spectrum Scale **mmapplypolicy** engine is preferable because it is much faster to scan the file system for identifying backup candidates than traversing the file system.

Therefore, for processing backups on larger file systems, use the IBM Spectrum Scale command **mmbackup** instead of using the IBM Spectrum Protect Backup-Archive client commands such as **dsmc expire** or **dsmc selective** or **dsmc incremental** directly. Using the **mmbackup** command also provides the following benefits:

- Backup activities are run in parallel by using multiple IBM Spectrum Scale cluster nodes to send backup data in parallel to the IBM Spectrum Protect server.
- **mmbackup** creates a local shadow of the IBM Spectrum Protect database in the file system and uses it along with the policy engine to identify candidate files for backup. The IBM Spectrum Protect server does not need to be queried for this information saving time when calculating the backup candidate list.
  - **mmbackup** and its use of the policy engine can select candidates faster than the **dsmc** progressive incremental operation that is bounded by walk of the file system using the POSIX directory and file status reading functions.
  - Using **dsmc selective** with lists generated by **mmbackup** is also faster than using **dsmc incremental** even with similar lists generated by **mmbackup**.

  Note: It is recommended that scheduled backups of an IBM Spectrum Scale file system use **mmbackup** because **mmbackup** does not actively query the IBM Spectrum Protect server to calculate backup candidates. However, events such as file space deletion or file deletion executed on IBM Spectrum Protect server are not recognized until the user triggers a synchronization between the **mmbackup** shadow database and the IBM Spectrum Protect database.

The following table contains a detailed comparison of **mmbackup** and IBM Spectrum Protect Backup-Archive client backup commands:

Table 24. Comparison of **mmbackup** *and IBM Spectrum Protect Backup-Archive client backup commands*

	IBM Spectrum Scale policy-driven backup (mmbackup)	**IBM Spectrum Protect progressive incremental backup (dsmc incremental)**
Detects changes in files and sends a new copy of the file to the server.	Yes	Yes
Detects changes in metadata and updates the file metadata on the server or sends a new copy of the file to the server (for ACL/EA changes).	Yes	Yes
Detects directory move, copy, or rename functions, and sends a new copy of the file to the server.	Yes	Yes
Detects local file deletion and expires the file on the server.	Yes	Yes
Detects IBM Spectrum Protect file space deletion or node/policy changes, and sends a new copy of the file to the server.	No*	Yes

Table 24. Comparison of **mmbackup** *and IBM Spectrum Protect Backup-Archive client backup commands (continued)*

	IBM Spectrum Scale policy-driven backup (mmbackup)	**IBM Spectrum Protect progressive incremental backup (dsmc incremental)**
Detects file deletion from the IBM Spectrum Protect server and sends a new copy of the file to the server.	No*	Yes
Detects additions of new exclude rules and expires the file on the server.	Yes	Yes
Detects policy changes made to include rules and rebinds the file to the new storage pool.	No**	Yes
Detects copy mode and copy frequency configuration changes.	No*	Yes
Detects migration state changes (IBM Spectrum Protect for Space Management) and updates the server object.	Yes	Yes
Detects that a file wasn't processed successfully during a backup operation and attempts again at the next backup.	Yes	Yes
Supports IBM Spectrum Protect Virtual Mount Points to divide a file system into smaller segments to reduce database size and contention.	No	Yes
* The **mmbackup** command queries the IBM Spectrum Protect server only once at the time of the first backup. Changes that are performed on the IBM Spectrum Protect server by using the IBM Spectrum Protect administrative client cannot be detected by **mmbackup** processing. You must rebuild the **mmbackup** shadow database if the IBM Spectrum Protect server file space changes.		
** IBM Spectrum Protect includes rules with associated management class bindings that cannot be detected by **mmbackup** processing. Therefore, **mmbackup** processing does not rebind a file if a management class changes include rules.		

If you use IBM Spectrum Protect Backup-Archive client backup commands on file systems that are otherwise handled by using **mmbackup**, the shadow database maintained by **mmbackup** loses its synchronization with the IBM Spectrum Protect inventory. In such cases, you need to resynchronize with the IBM Spectrum Protect server which will inform **mmbackup** of the recent backup activities conducted with the **dsmc** command. Resynchronization might be a very time-consuming activity for large file systems with a high number of backed up items. To avoid these scenarios, use the **mmbackup** command only.

If you have used **dsmc selective** or **dsmc incremental** since starting to use **mmbackup** and need to manually trigger a synchronization between the **mmbackup** maintained shadow database and the IBM Spectrum Protect server:

- Use the **mmbackup --rebuild** if you need to do a synchronization only.
- Use the **mmbackup -q** if you need to do a synchronization followed by a backup of the corresponding file system.

Using the IBM Spectrum Protect for Space Management client to identify migration candidates

Using IBM Spectrum Protect for Space Management clients for traversing IBM Spectrum Scale file system to identify migration candidates does not scale well. The IBM Spectrum Protect **automigration** daemons consume space in the file system and also consume CPU resources. They do not have access to the internal structures of the file system in the way that the IBM Spectrum Scale **mmapplypolicy** command does, and so they cannot scale. Use the following steps instead:

1. Set the following environment variable before the installation of the IBM Spectrum Protect for Space Management client to prevent the **automigration** daemons from starting during the installation:

   ```
   export HSMINSTALLMODE=SCOUTFREE
   ```

   It is recommended to place this setting into the profile file of the root user.
2. Add the following option to your IBM Spectrum Protect client configuration file, `dsm.opt`, to prevent the **automigration** daemons from starting after every system reboot:

   ```
   HSMDISABLEAUTOMIGDAEMONS YES
   ```
3. Add the following option to your IBM Spectrum Protect client configuration file, `dsm.opt`, to ensure that the object ID is added to the inode, so that the file list based reconciliation (two-way-orphan-check) can be used:

   ```
   HSMEXTOBjidattr YES
   ```
4. While the **automigration** daemons are disabled, changes such as removal of migrated files are not automatically propagated to the IBM Spectrum Protect server. For housekeeping purposes, you must run the IBM Spectrum Protect reconciliation either manually or in a scheduled manner. For more information, see *Reconciling by using a GPFS policy* in the IBM Spectrum Protect for Space Management documentation on IBM Knowledge Center.

Comparison of snapshot based backups and backups from live system

Backing up large file systems can take many hours or even days. When using the IBM Spectrum Scale command **mmbackup**, time is needed for the following steps.

- Scanning the system to identify the objects that need to be backed up or expired.
- Expiring all objects removed from the system.
- Backing up all new or changed objects.

If the backup is run on the live file system while it is active, objects selected for the backup job can be backed up at different points in time. This can lead to issues when temporary or transient files that were present during the scan time are removed by the time the backup command tries to send them to the IBM Spectrum Protect server. The attempt to back up a file that is removed fails and the need to back this object up is still recorded in the shadow database.

Instead of backing up from a live file system, an alternative is to use snapshot based backups. Using the snapshot adds additional actions of creating or reusing a snapshot and removing it when the overall backup process completes. However, this approach provides several advantages because a snapshot is a point in time view of the file system that is read-only and it can be used for backing up the file system for as long as is necessary to complete the backup job. The advantages of following this approach are as follows:

- Transient or temporary files are backed up, provided they existed at the time the snapshot was taken.
- Protection against failures to back up due to server-side faults such as the IBM Spectrum Protect server running out of space. For example, if the database or storage pool becomes full or if the IBM Spectrum Protect server crashes, etc. In this case, a retry of the backup is possible for the point in time when the snapshot has been taken with no loss of function or backup.
- Retention of a backup within the system. Snapshots can be kept for a period of time providing an online backup copy of all files. This can protect against accidental deletions or modifications, and can be used to retrieve an earlier version of a file, etc.

- A means to fulfill a data protection policy even if the backup activity to IBM Spectrum Protect exceeds the nominal time window allotted. The snapshot can be kept for several days until backups are complete, and multiple snapshots can be kept until backup completes for each of them.

IBM Spectrum Scale provides the capability to create snapshots for a complete file system, known as global snapshots, and for independent filesets, known as fileset snapshots.

While snapshot based backups provide several advantages, the following considerations apply when using the snapshot capability:

- Snapshots might consume space; usually a snapshot used for backup is removed shortly after the backup operation finishes. Long lived snapshots retain their copy of the data blocks, taken from the file system's pool of free blocks. As they age, the snapshots consume more data blocks owing to the changes made in the read or write view of the file system.
- Snapshot deletion can take time depending on the number of changes that need to be handled while removing the snapshot. In general, the older a snapshot is, the more work it will require to delete it.
- Special consideration for use of IBM Spectrum Protect for Space Management:
  - When a migrated file stub that is already part of a snapshot is removed, a recall is initiated to keep the snapshot consistent. This is required because removal of the stub invalidates the offline references to the stored data. The recall is to fill blocks on disk and assign them to the snapshot view. Once the stub is removed from the file system and a reconcile process removes this file from the Space Management pool on the IBM Spectrum Protect server, there are no longer any references to the file data except the snapshot copy.

File system and fileset backups with IBM Spectrum Protect

Starting with IBM Spectrum Scale 4.1.1, **mmbackup** supports backup from independent filesets in addition to backup of the whole file system. Independent filesets have similar capabilities as a file system in terms of quota management, dependent filesets, snapshots, and having their own inode space.

Fileset **mmbackup** provides a finer granularity for backup purposes and permits the administrator to:

- Have different backup schedules for different filesets. For example, providing the flexibility that filesets containing more important data to be backed up more often than other filesets.
- Have a fileset dedicated for temporary or transient files or files that do not need to be backed up.
- Use file system backups in conjunction with fileset backup to implement a dual data protection scheme such that file system backup goes to one IBM Spectrum Protect server while fileset backup goes to a different IBM Spectrum Protect server.

Backups of file systems and independent filesets are controlled by using the **mmbackup** option `--scope`.

In the examples used in this topic, the following environment is assumed:

```
tsm server name            => tsm1,tsm2,...
file system device name    => gpfs0 or /dev/gpfs0
file system mountpoint     => /gpfs0

fileset names/junction path
---------------------------
depfset1,depfset2,...     /gpfs0/depfset1,/gpfs0/depfset2         for dependent filesets
indepfset1,indepfset2,... /gpfs0/indepfset1,/ibm/gpfs0/indepfset2 for indepdendent filesets
```

File system backup

File system backup protects the whole file system. The default value for option `--scope` is `filesystem` and thus it can be omitted.

To back up the `gpfs0` file system in its entirety to the IBM Spectrum Protect server named `tsm1`, use the following command:

```
mmbackup gpfs0 --tsm-servers tsm1
mmbackup gpfs0 --tsm-servers tsm1 --scope filesystem
mmbackup /dev/gpfs0 --tsm-servers tsm1
mmbackup /dev/gpfs0 --tsm-servers tsm1 --scope filesystem
mmbackup /gpfs0 --tsm-servers tsm1
mmbackup /gpfs0 --tsm-servers tsm1 --scope filesystem
```

This example shows the file system backup of `gpfs0` using the short or long device name (first 4 lines) or the mount point as a directory input (last 2 lines).

Independent fileset backup

Independent fileset backup protects all files that belong to the inode space of a single independent fileset. The backup of an independent fileset might include other dependent filesets and folders, but not nested independent filesets because each independent fileset has its own inode space.

The following information describes the **mmbackup** capability to support independent fileset. The examples with IBM Spectrum Protect show the potential/likely use of this capability.

To back up the independent fileset `indepfset1` of the file system `gpfs0` to IBM Spectrum Protect server named `tsm1`, use the following command:

```
mmbackup /gpfs0/indepfset1 --tsm-servers tsm1 --scope inodespace
```

Fileset backup is an additional option to existing IBM Spectrum Scale backup or data protection options. Therefore, starting with a file system in which the whole file system has already been backed up on a regular basis, administrators can also start protecting data with fileset backup on some or all filesets. Administrators may choose to utilize fileset backup at any time, with no limit. It is not required to back up the whole file system in either one mode or another unless the chosen data protection approach requires it. Also, administrators can choose different IBM Spectrum Protect servers for some or all fileset backups if it is desired to have these on separate servers. For example, a valid configuration may have one IBM Spectrum Protect server that only gets whole file system backups, and a different one for fileset backups.

Note: When mixing file system and fileset backup on the same IBM Spectrum Protect server, consider that a target file that is handled by both backup approaches gets backed up twice. Each backup activity consumes one storage version of the file. Thus, the same file version is stored twice on the IBM Spectrum Protect server.

If migrating to use only fileset backup to protect the whole file systems, ensure that all independent filesets are backed up and keep the backup of all filesets current. Remember to include a backup of the `root` fileset as well by using a command such as:

```
mmbackup /gpfs/gpfs0 --scope inode-space --tsm-servers tsm1
```

To verify the completeness of the data protection using fileset backup only, use the following steps:

1. Identify all independent filesets available in the file system by using the following command:

   `mmlsfileset` *device* `-L`

   Identify the independent filesets by the InodeSpace column. The value identifies the corresponding inode space or independent fileset while the first occurrence refers to the corresponding fileset name and fileset path.

   Note: Ensure that a backup is maintained of the fileset called `root` that is created when the file system is created and that can be seen as the first fileset of the file system.
2. For every identified independent fileset, verify that a shadow database exists as follows:
   a. Check for the file `.mmbackupShadow.<digit>.<TSMserverName>.fileset` in the fileset junction directory.

b. If the file exists, determine the time of last backup by looking at the header line of this file. The backup date is stored as last value of this line.

   For example:

   ```
   head -n1 /gpfs0/indepfset1/.mmbackupShadow.*.fileset
   %%mshadow0%%:00_BACKUP_FILES_41:1400:/gpfs0:mmbackup:1:Mon Apr 20 13:48:45 2015
   ```

   Where *Mon Apr 20 13:48:45 2015* in this example is the time of last backup taken for this fileset.

3. If any independent filesets are missing their corresponding `.mmbackupShadow.*` files, or if they exist but are older than the data protection limit for their backup time, then start or schedule a backup of these filesets.

   Note: This action needs to be done for every file system for which the fileset backup approach is chosen.

Note: Backup of a nested independent fileset is not supported. To work around this limitation, unlink the nested fileset and link it at another location in the root fileset prior to beginning to use fileset backup pervasively.

Considerations for using fileset backup with IBM Spectrum Protect

IBM Spectrum Protect stores backup data indexed by file or directory object path names in its database. However, file system objects are identified for HSM by their object ID extended attributes. In both cases, IBM Spectrum Protect is not aware of the fileset entity. Therefore, the fileset configuration of an IBM Spectrum Scale cluster is neither backed up nor can it be restored by IBM Spectrum Protect alone, unless separate action is taken.

IBM Spectrum Scale does provide commands such as **mmbackupconfig** and **mmrestoreconfig** to backup and restore file system configuration data.

Even though backups of independent filesets may be stored on a plurality of IBM Spectrum Protect servers, IBM Spectrum Protect for Space Management can only utilize one server per managed file system if **mmbackup** is used on that file system and sends the backup data to that same server. IBM Spectrum Protect HSM multiserver feature is not integrated with the **mmbackup** command.

If space management is enabled for a IBM Spectrum Scale file system, and a plurality of IBM Spectrum Protect servers are utilized for backup, the following limitations apply to the use of IBM Spectrum Protect for Space Management:

- The setting of `migrequiresbkup` cannot be utilized since the server for space management might not have access to the database for the server doing backup on a particular object.
- The *inline copy* feature that allows the backup of a migrated object through copying data internally from the HSM pool to the backup pool inside the IBM Spectrum Protect server cannot be utilized since migrated data may reside on a different server than the one performing the backup.

The following special considerations apply when using fileset backup with IBM Spectrum Protect:

- IBM Spectrum Protect Backup-Archive client does not restore fileset configuration information. Therefore, a full file system backup is not sufficient for disaster recovery when the whole file system structure needs to be recovered.
- Nested independent filesets cannot be backed up when using the fileset backups. Resolve the nesting before starting to protect the data with **mmbackup** on each fileset.
- The following operations are discouraged: fileset unlink, fileset junction path change via unlink and link operations, fileset deletion. These operations will incur a heavy load on the IBM Spectrum Protect server at the next backup activity due to the significant alteration of the path name space that result from these operations. If a fileset must be moved in the file system, be prepared for the extra time that might be required to run the next backup.
- If only utilizing fileset backup, be sure to include every fileset, including the root fileset to ensure complete data protection.

- When both file system and fileset backups use the same IBM Spectrum Protect server, new and changed files may be unintentionally backed up twice. This consumes two of the file versions stored in the IBM Spectrum Protect server with the same file data.

  Loss of data protection may occur if a fileset is unlinked or deleted for an extended period of time and the objects formerly contained in that fileset are allowed to expire from the backup sets.
- Do not move, rename, or relink a fileset, unless a new full backup of the fileset is expected to be made immediately after the restructuring occurs. IBM Spectrum Protect server only recognizes objects in backup only as long as they remain in their original path name from the root of the file tree (mount point).

The following special considerations apply when using fileset backup and IBM Spectrum Protect for Space Management:

- When using fileset backup to different IBM Spectrum Protect servers, the server setting **migrequiresbkup** must be set to NO to be able to migrate files.
- Automatic server selection for backup by **mmbackup** is not supported and therefore the IBM Spectrum Protect *multiple HSM server* feature is not integrated with **mmbackup** data protection of IBM Spectrum Scale. Using fileset backup to different IBM Spectrum Protect servers in conjunction with the *multiple HSM server* feature requires special handling and configuration because backup and space management rules must match so that files are handled by the same IBM Spectrum Protect server.

Considerations for backing up file systems that are HSM space managed with IBM Spectrum Protect for Space Management

When IBM Spectrum Protect for Space Management is used to migrate data to secondary storage pools (for example, tape volumes), this migrated data might need to be recalled to disk if the file becomes a backup candidate due to certain changes a user may apply to it.

Recall will automatically occur synchronously if the user attempts to change the file data. However, no synchronous recall occurs in the following scenarios:

- If a user changes the owner, group, mode, access control list, or extended attributes of the file.
- If the user renames the file or any of the parent directories of the file.

Any of these changes, however does make the file a candidate for the next **mmbackup** invocation. When **mmbackup** supplies the path to a migrated file to IBM Spectrum Protect for backup, synchronous recall will occur. This can lead to a "Recall Storm" in which tape library and IBM Spectrum Protect server resources become overburdened handling many simultaneous recall requests. Disk space is also consumed immediately by recalled data. These are undesirable outcomes and can interfere with the normal processing of the customer's workload.

Since **mmbackup** is able to discern that the data was migrated for such files, it will defer the backup, and instead append a record referring to the file's path name into a special file in the root of the file system, or fileset in the case of fileset-level backup. This list can then be used by the system administrator to schedule recall of the deferred files data to permit the backup to occur on the next invocation of the **mmbackup** command. By deferring the recall to the system administrator, **mmbackup** prevents unexpected, excessive tape library activity which might occur if many such migrated files were nominated for backup due to user actions such as renaming a high level directory or changing owner, group, or modes on many files.

The system administrator must schedule the recall for the migrated objects omitted by **mmbackup** according to the availability of tape and disk resources in the cluster. For information about optimized tape recall, see *Optimized tape recall processing* in the IBM Spectrum Protect for Space Management documentation in IBM Knowledge Center.

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

With the Quality of Service for I/O operations (QoS) capability, you can run a GPFS maintenance command without the risk of it dominating file system I/O performance and significantly delaying other tasks.

The GPFS *maintenance commands* are a group of about 20 potentially long-running, I/O-intensive commands that can generate hundreds or thousands of requests for I/O operations per second. This high demand for I/O can greatly slow down normal tasks that are competing for the same I/O resources. With QoS, you can assign an instance of a maintenance command to a QoS class that has a lesser I/O priority. Although the instance of the maintenance command now takes longer to run to completion, normal tasks have greater access to I/O resources and run more quickly. The maintenance commands are listed in the help topic for the *mmchqos command* in the *IBM Spectrum Scale: Command and Programming Reference*.

Note the following requirements and limitations:

- QoS requires the file system to be at V4.2.0.0 or later.
- 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 I/O operations per second (IOPS) but does not regulate them.
- When you change allocations, mount the file system, or re-enable QoS, a brief delay due to reconfiguration occurs before QoS starts applying allocations.

To set up QoS, you allocate shares of I/O operations per second (IOPS) to two QoS classes, **maintenance** and **other**. Each storage pool has these two classes. The **maintenance** class is the class in which the GPFS maintenance commands run by default. Typically you assign a smaller share of IOPS to this class so that the commands in it do not overwhelm the file system with I/O requests. The **other** class is the class to which all other processes belong. Typically you assign an unlimited number of IOPS to this class so that normal processes have greater access to I/O resources and finish more quickly.

When a process issues an I/O request, QoS first finds the QoS class to which the process belongs. It then finds whether the class has any I/O operations available for consumption. If so, QoS allows the I/O operation. If not, then QoS queues the request until more I/O operations become available for the class. For a particular storage pool, all instances of maintenance commands that are running concurrently compete for the IOPS that you allocated to the **maintenance** class of that storage pool. Similarly, all processes that belong to the **other** class compete for the IOPS that you allocated to the **other** class of that storage pool. If no processes in the **maintenance** class are running, processes in the **other** class can also compete for the maintenance IOPS.

If you do not configure a storage pool for QoS, then processes compete for I/O operations in that storage pool in the normal way without interference.

By default, for both the **maintenance** class and the **other** class, QoS divides the IOPS allocation equally among the nodes of the cluster that have the file system mounted. For example, you might assign 1000 IOPS to the **maintenance** class of storage pool SP1. If your allocation applies to all the nodes in the cluster (the default setting) and the cluster has five nodes, then the **maintenance** class of each node gets an allocation of 200 IOPS for storage pool SP1. At run time, on each node all processes that belong to the **maintenance** class and that access storage pool SP1 compete for the 200 IOPS.

You can also divide IOPS among a list of nodes, a user-defined node class, or the nodes of a remote cluster that is served by the file system.

To get started with QoS, follow the steps that are outlined in the topic *Setting the Quality of Service for I/O operations (QoS)* in *IBM Spectrum Scale: Administration Guide*.

Use the **mmchqos** command to allocate IOPS to the QoS classes **maintenance** and **other**. You can set up allocations for one pool or several pools in one call to the command. You can designate a single node, several nodes, or all the nodes in the current cluster to participate in the QoS operations. You can also designate the nodes in a remote cluster to participate. To preserve a particular configuration, you can set up a stanza file that the **mmchqos** command reads when it runs.

You can reset or change IOPS allocations at any time without unmounting the file system. You can also disable QoS at any time without losing your IOPS allocations. When you reenable QoS, it resumes applying the allocations.

Remember the following points:

- You can allocate shares of IOPS separately for each storage pool.
- QoS divides each IOPS allocation equally among the specified nodes that mounted the file system.
- Allocations persist across unmounting and remounting the file system.
- QoS stops applying allocations when you unmount the file system and resumes when you remount it.

Use the **mmlsqos** command to display the consumption of IOPS. The command displays data for subperiods within the period that you specify. You can configure the type of information that is displayed. You can display information for specified storage pools or for all storage pools. You can display information for various time periods, from 1 second up to about 15 minutes. You can display I/O performance for each QoS class separately or summed together. You can also display the I/O performance of each node separately or summed across the participating nodes.

Planning for extended attributes

IBM Spectrum Scale supports extended attributes for files, directories, block devices, pipes, and named sockets in a file system.

Extended file attributes are key-value pairs that can be set programmatically by the file system, by other middleware such as the Data Management API, by the operating system, or by users. The name of an extended attribute consists of a namespace name followed by a dot followed by an attribute name, as in the following extended attribute names:

```
gpfs.Encryption
user.swift.metadata
system.posix_acl_access
```

The public namespaces are *system*, *security*, *trusted*, and *user*. IBM Spectrum Scale has a few private namespaces, which include *gpfs* and *dmapi*.

You can access extended attributes in the public namespaces through operating system commands and functions. Linux has commands such as `setfattr` and `getfattr` and functions such as `setxattr()` and `getxattr()` for accessing extended attributes.

- In Linux, the following facts are relevant:
  - You must have root-level access to work with extended attributes in the *system*, *security*, and *trusted* namespaces.
  - To set an extended attribute for a file or file object (directory, block device, pipe, or named socket) you must have write access to the entity.
- In Windows, the following facts are relevant:
  - You must log on as an administrator to work with extended attributes in the *system*, *security*, and *trusted* namespaces.
  - Windows has file permissions for reading and writing extended attributes that are separate from its file permissions for reading or writing file data.
- In AIX, the following facts are relevant:
  - IBM Spectrum Scale does not support AIX commands or functions that access extended attributes.

However, the native operating system commands and functions that are described in the preceding paragraph cannot access the private namespaces of IBM Spectrum Scale:

- To access extended attributes in the private **gpfs** name space, you can make calls with the GPFS programming interfaces.
- To access extended attributes in the private **dmapi** namespace, you can make calls with the IBM Spectrum Scale Data Management API.

For more information, see the topic *gpfs_stat_x() subroutine* and *IBM Spectrum Scale Data Management API for GPFS information* in the *IBM Spectrum Scale: Command and Programming Reference*.

Note: For example, in Linux, the **cp** command with the **--preserve=all** parameter does not copy extended attributes that belong to the private IBM Spectrum Scale namespaces.

You can access the extended attributes in the IBM Spectrum Scale private namespaces only through IBM Spectrum Scale resources, which are described following. These resources can also access the extended attributes in the public namespaces. IBM Spectrum Scale supports these resources on all platforms, Linux, Windows, and AIX:

- The rules for automating file management include a set of functions for accessing, setting, and evaluating extended attributes. For more information, see *Extended attribute functions* in the *IBM Spectrum Scale: Administration Guide*.
- The **mmchattr** and **mmlsattr** commands provide parameters to create, set, get, or delete extended attributes. For more information, see *mmchattr command* and *mmlsattr command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

  Note: The **mmchattr** command does not set some DMAPI extended attributes because of their side effects. However, you can set these extended attributes through DMAPI. For more information, see *IBM Spectrum Scale Data Management API for GPFS information* in the *IBM Spectrum Scale: Command and Programming Reference* guide.
- The GPFS programming interfaces include a set of subroutines for getting and setting extended attributes. For more information, see *GPFS programming interfaces* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

Fast extended attributes are extended attributes that are stored in the inode of the file or file object so that they can be read or written quickly. Some features of IBM Spectrum Scale require support for fast extended attributes, such as DMAPI, file encryption, and file heat. To work with fast extended attributes, you use exactly the same methods that you use with regular extended attributes.

The following features in IBM Spectrum Scale also handle extended attributes:

- Limitations:
  - Extended attributes are not encrypted when a file is encrypted.
  - Extended attributes cannot be added, deleted, or modified in an immutable or an appendOnly file. This restriction does not apply to extended attributes in the **dmapi** or **gpfs** namespaces.
- Size:
  - To determine the maximum permitted size of extended attributes, see Q6.11 in IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).
  - Fast extended attributes are stored first in the inode of a file, then, if more space is needed, in a maximum of one overflow block. The overflow block is the same size as the current metadata block size. The maximum space in the overflow block for storing extended attributes is the minimum of 64 KB or the metadata block size.
- Administration commands:
  - The **mmafmconfig** command has parameters that enable or disable extended attributes or sparse file support in the AFM cache.

  - The **mmclone** command does not copy extended attributes.
  - See the administration commands in the item "Back up and restore" following.
- Encryption:
  - In encryption, file extension keys (FEKs) are stored in the `gpfs.Encryption` extended attribute. It is a good idea to create a file system with an inode size of 4 K or larger to accommodate FEKs that are encrypted several times. If you encounter an error message that says that the extended attributes are too large, you must either change the encryption policy so that the file key is wrapped fewer times, reduce the number of keys that are used to wrap a file key, or create a file system that has a larger inode size. For more information, see the following links:
    - *Encryption policies* in the *IBM Spectrum Scale: Administration Guide*
    - *Preparation for encryption* in the *IBM Spectrum Scale: Administration Guide*
    - Error message *6027-3470 [E]* in the *IBM Spectrum Scale: Problem Determination Guide*
- Back up and restore:
  - Scale Out Backup and Restore (SOBAR) preserves and restores extended attributes. For more information, see *Scale Out Backup and Restore (SOBAR)* in the *IBM Spectrum Scale: Administration Guide*.
  - The **mmbackup** and **mmrestorefs** commands preserve extended attributes. The **mmrestorefs** command has options to restore encryption attributes and to not restore external attributes. The IBM Spectrum Protect product has a setting in the `dsm.sys` configuration file that omits backing up a file if only an extended attribute is changed (`SKIPACLUPDATECHECK`). For more information, see *Options in the IBM Spectrum Protect configuration file dsm.opt* in the *IBM Spectrum Scale: Administration Guide*.
  - The **mmrestorefs** command has options to restore encryption attributes and to not restore external attributes.
- Disaster recovery
  - In outband disaster recovery, if you are transferring files that contain extended attributes, such as object data, you must use a method that transfers extended attributes, such IBM Spectrum Protect, cross-cluster mount, or AFM. For more information, see *Creating the outband disaster recovery setup* in the *IBM Spectrum Scale: Administration Guide*.
- Objects
  - In unified file and object access, the identity management modes **local_mode** and **unified_mode** have settings for retaining extended attributes. You can set values in the `object-server-sof.conf` configuration file to copy extended attributes, Windows attributes, and ACLs from an existing object. For more information, see the following links:
    - *Identity management modes for unified file and object access* in the *IBM Spectrum Scale: Administration Guide*
    - *Configuring authentication and ID mapping for file access* in the *IBM Spectrum Scale: Administration Guide*
- NFS v4
  - NFS v4 uses extended attributes to store some of its file attributes. For more information, see *Linux ACLs and extended attributes* in the *IBM Spectrum Scale: Administration Guide*.

Support for extended attributes began in GPFS v2.1. Support for fast extended attributes began in GPFS v3.4. In IBM Spectrum Scale v4.1 and after, fast extended attributes are enabled by default for a file system when the file system is created. For more information, see *File system format changes* in the *IBM Spectrum Scale: Administration Guide*.

If you are upgrading from IBM Spectrum Scale v3.3 to v3.4 or later, you must explicitly enable fast extended attributes in file systems that were created earlier than v3.4. After you upgrade IBM Spectrum Scale to the new level, enter the following command to enable extended attributes:

```
mmmigratefs <Device> --fastea
```

where *<Device>* is the device ID of the file system. To verify that extended attributes are enabled, issue the following command:

```
mmlsfs <Device> --fastea
```

For more information, see "Completing the upgrade to a new level of IBM Spectrum Scale" on page 418.

Planning for the Highly Available Write Cache feature (HAWC)

Learn about the Highly Available Write Cache feature (HAWC).

Components that HAWC interacts with

HAWC interacts with several fundamental components of IBM Spectrum Scale. You might want to review these components before you read about HAWC.

System storage pool

The *system storage pool* or *system pool* is a required storage pool that contains information that IBM Spectrum Scale uses to manage a file system. Each file system has only one system storage pool, which is automatically created when the file system is created. The system storage pool contains the following types of information:

- Control information (such as file system control structures, reserved files, directories, symbolic links, special devices)
- The metadata associated with regular files, including indirect blocks and extended attributes
- Regular file data, if the `usage=dataAndMetadata` option is set in the NSD stanza for a system storage pool NSD
- The file system recovery logs (default location)

System.log storage pool

The *system.log* storage pool is an optional dedicated storage pool that contains only the file system recovery logs. If you define this pool, then IBM Spectrum Scale uses it for all the file system recovery logs of the file system. Otherwise, the file system recovery logs are kept in the system storage pool. It is a good practice for the system.log pool to consist of storage media that is as fast as or faster than the storage media of the system storage pool. If the storage is nonvolatile, this pool can be used for the high-availability write cache (HAWC).

File system recovery logs

A *file system recovery log* is a write-ahead log or journal of I/O metadata that describes pending write operations for a node of a file system. In IBM Spectrum Scale, it is also sometimes referred to as the recovery log, the GPFS log, or the IBM Spectrum Scale log. IBM Spectrum Scale creates and maintains a separate recovery log for every node that mounts a file system. Recovery logs are stored in the system storage pool by default or in the system.log storage pool if one is defined. The recovery logs can be read by any node that mounts the file system. If a node is unexpectedly shut down while write operations are pending for one of its hard disks, IBM Spectrum Scale can read the recovery log for the failed node and restore the file system to a consistent state. The recovery can occur immediately, without having to wait for the failed node to return.

The recovery logs are also used by HAWC to temporarily store HAWC write data and metadata.

Page pool

The *page pool* is an area of pinned memory (memory that is never paged to disk) that contains file data and metadata associated with in-progress I/O operations. When IBM Spectrum Scale processes a file write operation, the first step is putting the write data and metadata for the write operation into the page pool. At an appropriate time, another thread writes the data to a hard disk and removes it from the page pool.

HAWC operation

The high-availability write cache is a disk-caching component that includes caching software and nonvolatile storage. HAWC also uses the file system recovery logs, in which the file system records metadata about its pending write operations. For HAWC purposes, the recovery logs must be located in nonvolatile storage.

When a file write operation arrives at a node, the first part of the processing is the same whether HAWC is active or not. The write data and metadata are copied into an entry in the page pool and the entry is added to a list of similar entries that are waiting for processing. When the entry is processed, the processing depends on whether HAWC is active.

Note: If the write operation is nonsynchronous, it returns to its caller after its write data and metadata are copied into the page pool entry. If the write operation is synchronous, it waits for a notification that the file data has been written to disk.

When HAWC is not active, the write data is copied from the page pool entry and written to the file on hard disk. If the write operation is synchronous, the system notifies the write operation that the write is successful and it returns to its caller.

When HAWC is active, the write data can take either of two paths:

- If the write operation is synchronous and the size of the file data is less than or equal to the write data threshold, HAWC copies the file data from the page pool entry into the recovery log, along with any I/O metadata that is required for recovery. The *write data threshold* variable is set by the **mmcrfs** command or the **mmchfs** command. Next HAWC notifies the original write operation that the file data is successfully written to hard disk. In fact, the file data is not written to hard disk yet, although it is preserved in the recovery log as a backup. HAWC then starts a write-behind thread that eventually writes the file data to the hard disk. When the data is safely written, HAWC purges the file data and I/O metadata from the recovery log, because it is no longer needed.
- If the write operation is not synchronous or if the size of the write data is greater than the write cache threshold, then the write data follows the same path that is followed when HAWC is not active. The system copies the write data from the page pool entry and writes it to hard disk. If the original write operation is synchronous, the system notifies it that the file data is safely written to the hard disk.

HAWC improves the performance of small synchronous write operations in two ways. First, it allows synchronous write operations to return the calling application as soon as the write data is written into the recovery log. The calling application does not have to wait for the much lengthier process of writing the data to hard disk. Second, the HAWC caching software can consolidate small sequential writes into one larger write. This consolidation eliminates all but one of the initial disk seeks that is required if the data is written as multiple writes.

The write-cache threshold variable can be adjusted by specifying a value for the **--write-cache-threshold** parameter of the **mmchfs** command. The valid range is 0 - 64 K in multiples of 4 K. You can also set this variable when you create the file system by specifying the same parameter in the **mmcrfs** command. Setting the write cache threshold to zero disables HAWC. You can update the write threshold variable at any time; the file system does not have to be mounted on the node.

HAWC storage scenarios

You can set up the HAWC storage in either of two configurations or scenarios. In the first scenario, the nonvolatile storage is located in a centralized fast storage device, such as a controller with SSDs:

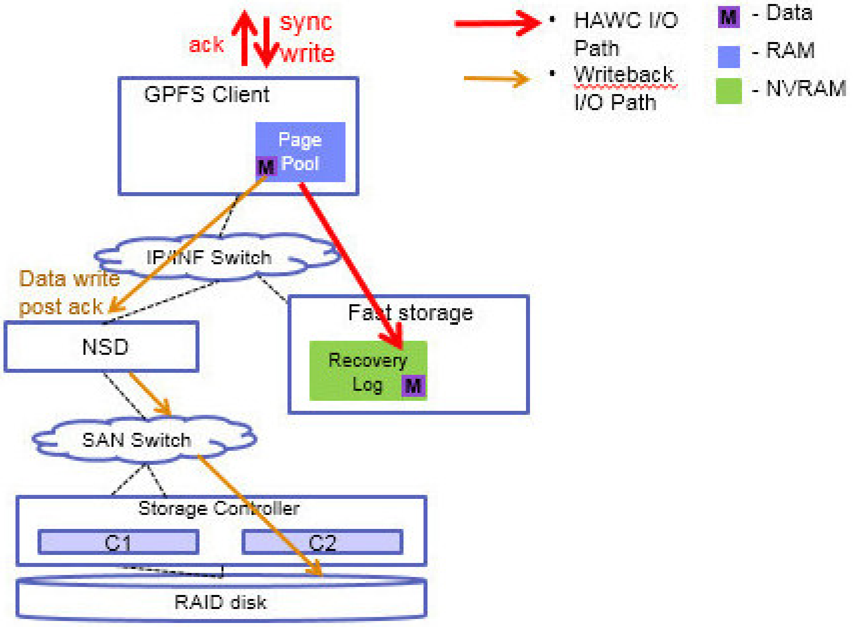


Figure 27. Shared fast storage

In this scenario, when a synchronous write operation arrives at a node, the file data and metadata are copied a page pool entry in the usual way. If the size of the file data is less than the write data threshold, HAWC copies the file data into the recovery log along with any I/O metadata that is required for recovery. Next, HAWC returns an acknowledgment to the write operation that indicates that the file data is successfully written to hard disk. HAWC then starts a write-behind thread that eventually writes the file data to the hard disk. When the data is safely written, HAWC purges the file data and I/O metadata for the write operation from the recovery log.

In the second scenario, the nonvolatile storage consists of multiple storage devices that are distributed across some or all of the nodes in the cluster:

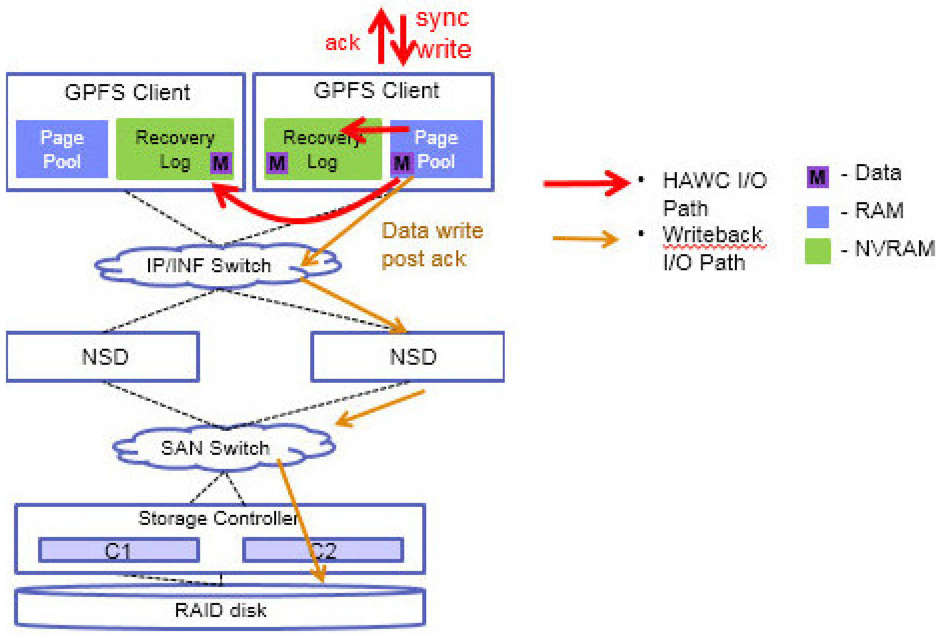


Figure 28. Distributed fast storage

Although the hardware configuration is different in the second scenario, the data flow is similar to the data flow of the first scenario. The synchronous write operation arrives at a node and the write data and I/O metadata are written into a page pool entry. If the size of the write data is smaller than the write storage threshold, HAWC copies the file data and relevant I/O metadata to the recovery log. The data is striped over the various disks that belong to the recovery log storage pool. HAWC returns a successful acknowledgment to the synchronous write operation and starts a write-behind thread that later writes the file data from the page pool entry to a hard disk. When the data is safely written, HAWC purges the file data and I/O metadata from the recovery log.

HAWC software configuration

After you install the nonvolatile storage devices, complete the following steps to configure IBM Spectrum Scale for HAWC. These steps assume that you are adding HAWC storage to an existing file system:

1. Stop the GPFS daemon on all the nodes of the cluster.
2. Create NSD stanzas for the nonvolatile storage devices. In the stanza, specify one storage pool for all the nonvolatile storage devices, which must be either the system pool or the system.log pool.
3. Run **mmcrnsd** to create the NSDs.
4. Run **mmaddisk** to add the NSDs to the file system and to create the system.log pool if necessary.
5. Start the GPFS daemons on all nodes.
6. Optionally, run the **mmchfs** command with the **-L** parameter to set the size of the recovery logs to a non-default value.
7. Optionally, run the **mmchfs** command with the **--log-replicas** parameter to set the number of replicas of the recovery log to a non-default value. This option is applicable only if the recovery logs are stored in the system.log pool.
8. To activate HAWC, run the **mmchfs** command with the **--write-cache-threshold** parameter set to a nonzero value.

HAWC is now active.

Planning for protocols

Authentication considerations

To enable read and write access to directories and files for the users on the IBM Spectrum Scale system, you must configure user authentication on the system. Only one user authentication method, and only one instance of that method, can be supported.

The following authentication services can be configured with the IBM Spectrum Scale system for file protocol access:

- Microsoft Active Directory (AD). For more information, see Table 26.
- Lightweight Directory Access Protocol (LDAP). For more information, see Table 25.
- Network Information Service (NIS) for NFS client access. For more information, see Table 27 on page 187.

The following authentication services can be configured with the IBM Spectrum Scale system for object access:

- Microsoft Active Directory (AD). For more information, see Table 29 on page 187.
- Lightweight Directory Access Protocol (LDAP). For more information, see Table 28 on page 187.
- Local authentication. For more information, see Table 30 on page 187.
- User defined. For more information, see Table 31 on page 187.

The following matrix gives a quick overview of the supported authentication configurations for both file and object access.

- ✓: Supported
- X: Not supported
- NA: Not applicable

File Protocol Authentication and ID mapping matrices

Table 25. Authentication support matrix when LDAP is the Authentication service being used

Authentication method	ID mapping method	SMB	SMB with Kerberos	NFSV3	NFSV3 with Kerberos	NFSV4	NFSV4 with Kerberos
LDAP with TLS	LDAP	✓	NA	✓	NA	✓	NA
LDAP with Kerberos	LDAP	✓	✓	✓	✓	✓	✓
LDAP with Kerberos and TLS	LDAP	✓	✓	✓	✓	✓	✓
LDAP without TLS and without Kerberos	LDAP	✓	NA	✓	NA	✓	NA

Table 26. Authentication support matrix when AD is the Authentication service being used

Authentication method	ID mapping method	SMB	SMB with Kerberos	NFSV3	NFSV3 with Kerberos	NFSV4	NFSV4 with Kerberos
AD	Automatic	✓	✓	X	X	X	X
AD	RFC2307	✓	✓	✓	✓	✓	✓
AD	LDAP	✓	✓	✓	X	X	X

Table 27. Authentication support matrix when NIS is the Authentication service being used

Authentication method	ID mapping method	SMB	SMB with Kerberos	NFSV3	NFSV3 with Kerberos	NFSV4	NFSV4 with Kerberos
NIS	NIS	NA	NA	✓	NA	✓	NA

Note: "Local" and "user-defined" authentications are not supported for File protocol.

Object Protocol Authentication Matrices

Table 28. Authentication support matrix when LDAP is the Authentication service being used

Authentication method	Object
LDAP with TLS	✓
LDAP with Kerberos	NA
LDAP with Kerberos and TLS	NA
LDAP with SSL	✓
LDAP without TLS and without Kerberos	✓

Table 29. Authentication support matrix when AD is the Authentication service being used

Authentication method	Object
AD	✓
AD with SSL	✓
AD with TLS	✓

Table 30. Authentication support matrix when "local" is the Authentication service being used

Authentication method	Object
Local	✓
Local (OpenStack Keystone)	✓
Local (OpenStack Keystone) with SSL	✓

Table 31. Authentication support matrix when "user defined" is the Authentication service being used

Authentication method	Object
User defined	✓

Note:

- NIS is not supported for Object protocol.
- When making use of Unified file and object access (serving same data with file as well as with object), select the appropriate authentication service. For more information, see the *Administering unified file and object access* topic in the *IBM Spectrum Scale: Administration Guide*.

For information on a complete list of supported authentication configurations for both file and Object protocols, see “General authentication support matrix” on page 196.

Unified Identity between Object & File : In this case, we need to make sure that the users get the same user UID and GID across POSIX, NFS, SMB, and Object. Therefore, only the following authentication mechanisms are supported:

- Object configured with AD and File with the same AD where the user/group ID is available on AD+RFC 2307

- Object configured with LDAP and File with the same LDAP where the user/group ID is available on LDAP

For more information, see the *Administering unified file and object access* topic in the *IBM Spectrum Scale: Administration Guide*.

The following diagram shows the high-level overview of the authentication configuration.

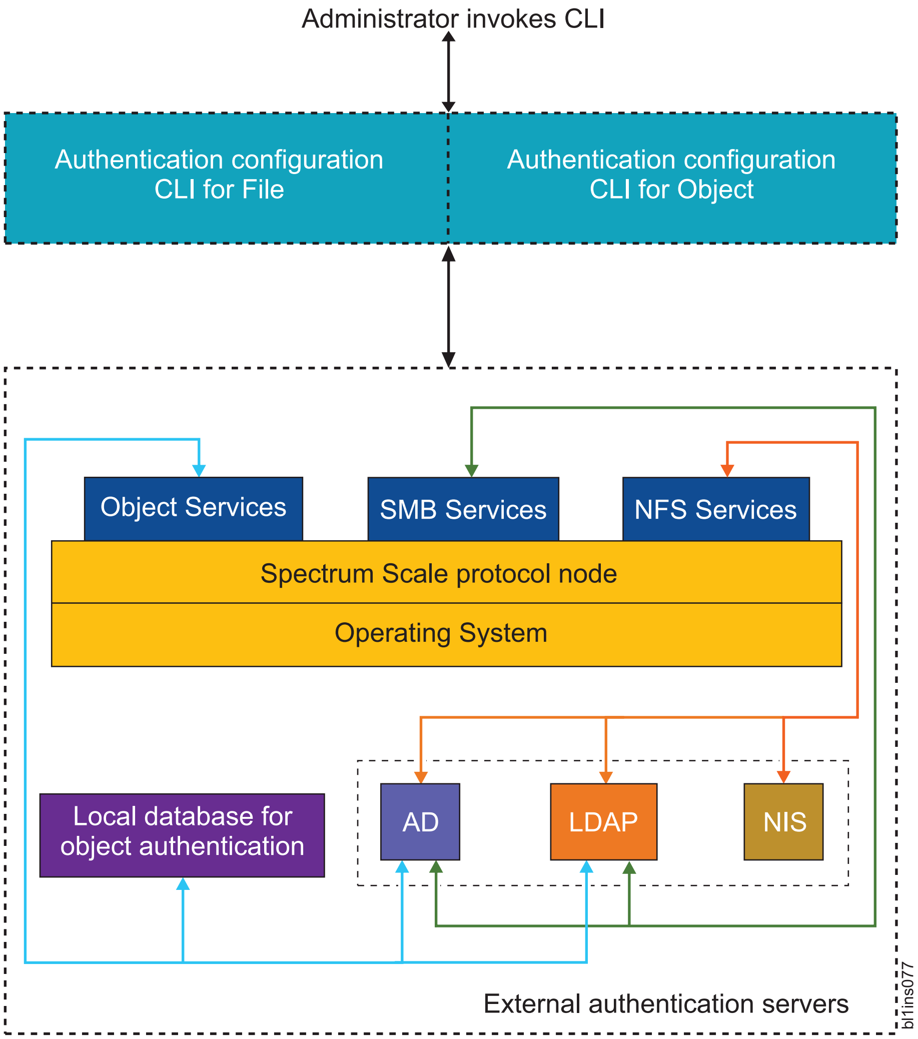

Administrator invokes CLI
Authentication configuration CLI for File
Authentication configuration CLI for Object
Object Services
SMB Services
NFS Services
Spectrum Scale protocol node
Operating System
Local database for object authentication
AD
LDAP
NIS
External authentication servers
bl1ins077

Figure 29. High-level overview of protocol user authentication

The authentication requests that are received from the client systems are handled by the corresponding services in the IBM Spectrum Scale system. For example, if a user needs to access the NFS data, the NFS services resolves the access request by interacting with the corresponding authentication and ID mapping servers.

Impact of enabling and disabling protocols on authentication

The following are the recommendations for enabling and disabling protocols.

- If authentication is already configured in the system, then the protocols that are enabled in the system to access data cannot be disabled, unless the system administrator cleans up the authentication configuration.

- If authentication is configured, a disabled protocol cannot be enabled. Hence, it is vital to plan properly to decide the protocols that you want to avail now and in near future and enable them correctly in advance.
- If authentication method chosen is AD, then authentication cannot be configured unless the SMB protocol is enabled and started.
- Protocols cannot be disabled unless authentication is unconfigured. Disabling of a protocol is a critical action and it might result in loss of access to the data hosted by that protocol or other protocols. After disabling the protocol, the authentication can be reconfigured with the same authentication servers and types so that the other enabled protocols can start functioning again.

  Note: Instead of disabling protocols, the system administrators might stop a protocol if they do not need it. Stopping a protocol is the recommended action instead of disabling the protocol. Stopping a protocol is as good as removing the protocol from the cluster but it does not have an impact on the access to the data. Users can retain the access when the protocol is started again. Use the **mmces stop -nfs -a** command to stop a protocol on the CES nodes. For more details on these options, see *mmces command* in *IBM Spectrum Scale: Command and Programming Reference*.
- The restrictions for enabling or disabling protocols are not applicable if user-defined authentication method is used for file or object access.
- The authentication configuration needs to be removed to enable a new protocol in the system. If other protocols are already enabled in the system, the system administrator must reconfigure the authentication with the same authentication servers and types so that the other enabled protocols can start functioning again.

You can also remove ID mappings, along with authentication, if you want to completely remove the authentication configuration. This results in permanent loss of access to the data.

Authentication and ID mapping for file access

The system supports an external authentication service to authenticate users on the system. Before you configure an authentication method, ensure that the external authentication service is set up correctly.

The following steps are involved in user authentication for file access:

1. User tries to connect to the IBM Spectrum Scale system by using their credentials.
2. The IBM Spectrum Scale system contacts the authentication server to validate the user.
3. The IBM Spectrum Scale system contacts the ID map server that provides UIDs and GIDs of the user and user group to verify the identity of the user.
4. If the user credentials are valid, the user gains access to the system.

The following diagram shows the high-level flow of authentication for File-based protocols.

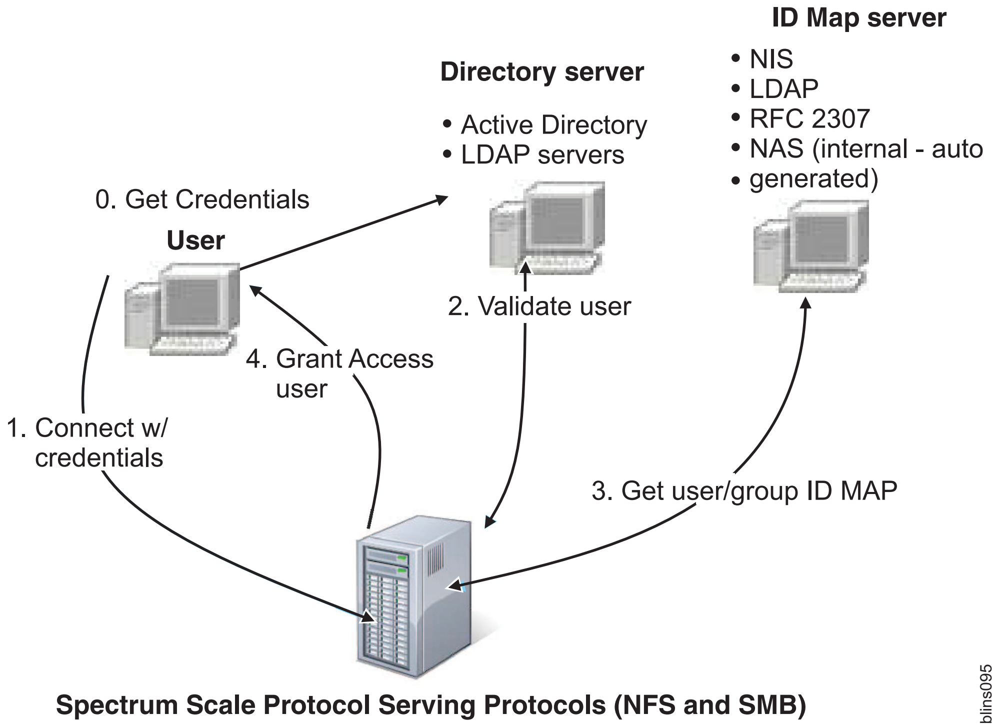


Figure 30. High-level flow of authentication for File protocols

ID mapping

The authentication of the user or groups of users is also associated with the identification of their unique identifiers. To support data access to Microsoft Windows clients (SMB protocol) and to allow interoperability, that is, to share data among UNIX and Windows clients (SMB and NFS protocols), the IBM Spectrum Scale system must map Windows SID to UNIX UID/GID. This process is referred to as ID mapping and the map is referred to as ID map. The ID mapping can be done either internally in the IBM Spectrum Scale system or in an external authentication server.

ID mapping is part of the user identification process in user authentication. The purpose of identification is to identify users and infrastructure components. Identification methods include unique user identifiers (IDs), keys, or fingerprints such as a public Secure Shell (SSH) key, and digital certificates such as a certificate of a web server.

UNIX based systems such as the IBM Spectrum Scale system use user names and user identifiers (UIDs) to represent users of the system. The user name is typically a human-readable sequence of alphanumeric characters and the UID is a positive integer value. When a user logs on to a UNIX system, the operating system looks up the UID and then uses this UID for further representation of the user. User names, UIDs, and the mapping of user names to UIDs are stored locally in the `/etc/passwd` file or on an external directory service such as Active Directory (AD), Lightweight Directory Access Protocol (LDAP), Keystone, or Network Information Service (NIS).

UNIX systems implement groups to maintain sets of users that have the same group permissions to access certain system resources. Similar to user names and UIDs, a UNIX system also maintains group names and group identifiers (GID). A UNIX user can be a member of one or more groups, where one group is the primary or default group. Group names, GIDs, the mapping of group names to GIDs, and the memberships of users to groups are stored locally in the `/etc/group` file or on an external directory service such as Active Directory, LDAP, Keystone, or NIS. The primary group of a user is stored in `/etc/passwd` or in an external directory service.

Windows systems reference all operating system entities as resources. For example, users, groups, computers, and so on are Windows resources. Each resource is represented by a security identifier (SID). Resource names and SIDs are stored locally in the Windows registry or in an external directory service such as Active Directory or LDAP. The following methods are used to map Windows SID to UNIX UID and GID:

- External ID mapping methods
  - RFC2307 when AD-based authentication is used
  - LDAP when LDAP-based authentication is used
- Internal ID mapping method
  - Automatic ID mapping when AD-based authentication is used

External ID mapping

A UID or GID of a user or group is created and stored in an external server such as Microsoft Active Directory, NIS server, or LDAP server.

External ID mapping is useful when user UID or group GID is preexisting in the environment. For example, if NFS client with UID and GID as 100 exists in the environment, and you want a certain share to be accessed by both SMB and NFS client, then you can use an external ID mapping server, assign UID/GID 100 to the SMB user, and thus, allow both SMB and NFS client to access same data.

Note: The external server administrator is responsible for creating or populating the UID/GID for the user/group in their respective servers.

The IBM Spectrum Scale system supports the following servers for external ID mapping:

- LDAP server, where the UID or GID is stored in a dedicated field in the user or group object on the LDAP server.
- AD server with RFC2307 schema extension defined. The UID or GID of a user or group that is defined in AD server is stored in a dedicated field of the user or group object.

The UID/GID defined in external server can be used by the IBM Spectrum Scale system.

LDAP ID mapping is supported only when the IBM Spectrum Scale is configured with LDAP authentication.

Internal ID mapping

The UID or GID of a user or group is created automatically by the IBM Spectrum Scale system and stored in the internal repositories.

When an external ID mapping server is not present in the environment or cannot be used, the IBM Spectrum Scale system uses its internal ID mapping method to create the UID/GID.

IBM Spectrum Scale supports an Automatic ID mapping method if AD-based authentication is used. The Automatic ID mapping method uses a reserved ID range to allocate an ID based on the following logic: A user or group in AD is identified by SID, which includes a component that is called RID. Whenever a user or group from an AD domain accesses IBM Spectrum Scale, a range is allocated per AD domain. The UID or GID is then allocated depending upon this range and the RID of the user/group.

Internal ID mapping cannot be used when the user UID or group GID is preexisting in the environment. However, while using internal ID mapping, if an NFS client wants to access data, then the NFS client must have a UID or GID that is identical to the one created by the IBM Spectrum Scale system.

Other supported authentication elements for file access

The system supports the following authentication elements as well:

- **Netgroups:** Groups of hosts are used to restrict access for mounting NFS exports on a set of hosts, and deny mounting on the remainder of the hosts. The IBM Spectrum Scale system supports only the netgroups that are stored in NIS and in Lightweight Directory Access Protocol (LDAP).
- **Kerberos:** Kerberos is a network authentication protocol that provides secured communication by ensuring passwords are not sent over the network to the system. The system supports Kerberos with both AD and LDAP-based authentication. When you configure AD-based authentication for any ID mapping method (automatic, RFC2307, LDAP), the Kerberos is enabled for the SMB access by default. However, Kerberos for NFS access is supported only for RFC2307 ID mapping method in AD-based authentication, and to enable NFS Kerberos access for AD-based authentication with RFC2307 ID mapping; you need to use the **--enable-nfs-kerberos** and **--unixmap-domains** options in the **`mmuserauth`** command.

  Kerberos is optional for both SMB and NFS access in LDAP. To enable Kerberos with LDAP, you need to integrate the system with MIT KDC.
- **Transport Level Security (TLS):** The TLS protocol is primarily used to increase the security and integrity of data that is sent over the network. These protocols are based on public key cryptography and use digital certificates based on X.509 for identification.

Caching user and user group information

Every UID and GID, whether generated automatically or through RFC2307 or NIS, is cached during a login attempt. For each successful user login attempt, the UID and GID are stored for seven days. For each failed user login attempt, the UID and GID are cached for two minutes.

When using an external ID mapping server, it is recommended that you not to change any previously created UID or GID. If you must change the UID or GID, plan this activity carefully, preferably before any data associated with the UID or GID exists on the IBM Spectrum Scale cluster. After the change, ensure that the cached entries are flushed out.

Caching user and user group information while using LDAP-based authentication

In LDAP-based user authentication, the **`mmuserauth service create`** command populates the LDAP bind user and bind password in its configuration file that is located at: `/etc/pam_ldap.conf`. Using this bind user and password, the *username* is looked up in the LDAP server with the configured *login_attribute*. This is the *uid* by default. As a result, *uid=<username>* is looked up in LDAP to fetch the DN for this user. If found, the DN and the *password* are used to perform an LDAP bind operation. The authentication fails if either the *username* is not found or if the bind operation fails.

Note: If LDAP is used for UNIX style login attempts, the *username* is also compared with what is returned from the LDAP server.

For SMB authentication, the ability to perform the LDAP bind with 'username' and 'password' are not available, since the password is in clear text. Therefore, the SMB client passes an NT hash for the password. The Samba server compares the NT hash from the client to the associated NT hash fetched from LDAP server. For this, the **`mmuserauth service create`** command configures the Samba registry with the LDAP bind user and password. The LDAP bind user and password are used to look up the Samba-related information from the LDAP server, such as the SID and the NT hash. If the NT hashes match for the bind user, then system access is granted.

User credentials, group ID mapping, and group membership caching: User credentials are not cached in the IBM Spectrum Scale system. The user and group ID mapping, and the group membership cache information, are stored in the SSSD component. The cache, local to each node, is for 90 minutes. There is no IBM Spectrum Scale native command to purge this cache. However, the sss_cache command (from the sssd-common package) can be used to refresh the cache.

SMB data caching: There is a 7 day cache period for the user/group SID within the Samba component of the IBM Spectrum Scale system. This cache is local to each node. There is no IBM Spectrum Scale native command to manage this cache. However, the 'net cache flush' command may be used, on a per node basis, to purge the cache. A negative cache entry persists for 2 minutes.

Netgroup caching: The netgroup information from LDAP is also cached for 90 minutes with the SSSD component. The sss_cache command can be used to refresh the cache.

Caching user credentials while using NIS-based authentication for file access

In IBM Spectrum Scale, an NIS server may be used only for users and groups, UID/GID, and group membership lookups for NFS access. In addition, the netgroups defined in NIS are honored as NFS client attributes within the NFS export configuration. Users and groups, UID and GID information, names, and group membership are cached within the SSSD component. This cache, local to each node, is for 90 minutes. There is no IBM Spectrum Scale native command to purge this cache.

The **sss_cache** command from the sssd-common package may be used to refresh the cache. The sss_cache utility only marks the entries in the respective database as expired. If the NIS server is online and available, the expired entries are refreshed immediately. If the NIS server is not available, the old cached entries that are still marked expired become valid. These values are refreshed once the server is available.

The netgroup information from the NIS is also cached for 90 minutes within the SSSD component. The sss_cache command can be used to refresh the cache.

Caching user and group ID mapping and group membership details while using AD-based authentication

The Samba component in the IBM Spectrum Scale system uses the SMB protocol NTLM and Kerberos authentication for user authentication. User and group SID to UID and GID mapping information is cached within the Samba component. This cache, local to each node, is maintained for seven days. There is no IBM Spectrum Scale native command to manage this cache. However, the **net cache flush** command may be used to refresh the cache. The negative cache entry persists for 2 minutes.

The group membership cache, local to each IBM Spectrum Scale protocol node, lies within the Samba component. For an authenticated user, its group membership cache is valid for the lifetime of that session. The group membership is only refreshed on a new authentication request. Therefore, if an SMB tree connect is requested on an existing session, the group cache is not refreshed. So, if there are group membership changes made on the AD server, all of the existing sessions for that user must be disconnected in order to obtain a new authentication request and subsequent cache refresh for the user. If there is no new authentication request made on behalf of the user, a simple user and group information look-up is requested. For example, if you issue the **id** command on the protocol node, the winbind component in IBM Spectrum Scale searches the database to check whether that user's cached information exists from a previous authentication request. If an appropriate entry is found, it is returned. Otherwise, the winbind fetches the mapping information from the AD server and caches it for five minutes.

Note: If a user was authenticated in the past, its group membership cache information is within the database and valid for the session lifetime. Winbind keeps referring to this information and will never try to fetch new information from the AD server. To force winbind to fetch new information, you need to make an authentication request, on behalf of the user, to the node in the same way as you would connect from a CIFS client.

Authentication for object access

The OpenStack identity service that is enabled in the system confirms an incoming request by validating a set of credentials that are supplied by the user. The identity management consists of both authentication and authorization processes.

In the authentication and authorization process, an object user is identified in the IBM Spectrum Scale system by the attributes such as user ID, password, project ID, role, and domain ID with Keystone API V3. The keystone server that is used for the OpenStack identity service that is configured on the IBM Spectrum Scale system manages the user authentication requests for object access with the help of either an external or internal authentication server for the user management. An internal or an external keystone server can be configured to manage the identity service. You can use the following authentication methods for object access either by using an internal keystone server shipped IBM Spectrum Scale or by using an external keystone server:

- External authentication. The keystone server interacts with the following external servers for the user management:
  - AD
  - LDAP
- Internal authentication. The keystone server interacts with an internal database to manage users and authentication requests.

The user can select the authentication method based on their requirement. For example, if the enterprise deployment already consists of AD and the users in the AD need to access object data, the customer would use AD as the backend authentication server.

When the authentication method selected is either AD or LDAP, the user management operations such as creating a user, deleting a user are the responsibility of the AD and LDAP administrator. If local authentication is selected for object access, the user management operations must be managed by the keystone server administrator. The authorization tasks such as defining user roles, creating projects, associating a user with a project are managed by the keystone administrator. The keystone server administration can be done either by using Keystone V3 REST API or by using OpenStack python-based client that is called openstackclient.

The following diagram depicts how the authentication requests are handled when IBM Spectrum Scale is with an internal keystone server and external AD or LDAP authentication server.

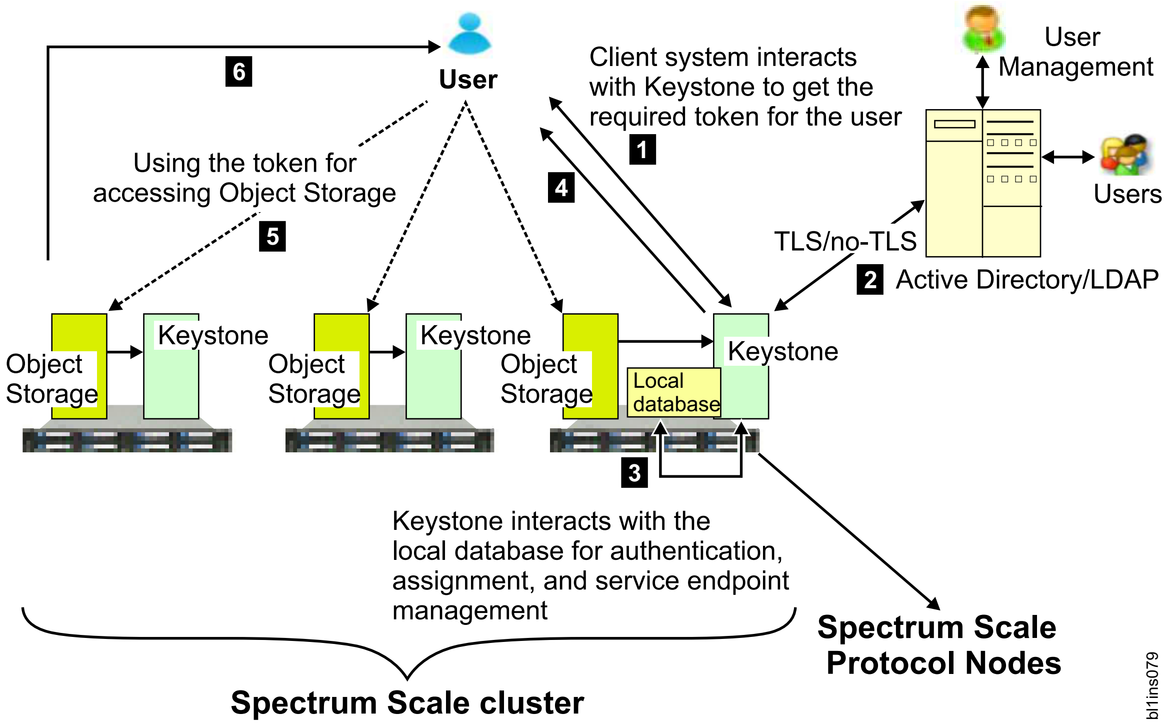


Figure 31. IBM Spectrum Scale integration with internal Keystone server and external AD or LDAP authentication server

The following list provides the authentication process for object access:

1. The user raises the access request to get access to the object data.
2. The keystone server communicates with the authentication server such as AD, LDAP, or a local database. The keystone server interacts with the authentication server for authentication, authorization, and service end-point management.
3. If the user details are valid, the keystone server interacts with the local database to determine the user roles and issues a token to grant access to the user.
4. The OpenStack identity service offers token-based authentication for object access. When user credentials are validated, the identity service issues an authentication token, which the user provides in subsequent requests. That is, the access request also includes the token that is granted in step 3. The token is an alphanumeric string of text that is used to access OpenStack APIs and resources.
5. The authenticated user contacts the object storage to access the data that is stored in it.
6. The object storage grants permission to the user to work on the data based on the associated project ID and user role.

Each object user is part of a project. A project is used to group or isolate resources. Each user needs to be defined with the set of user rights and privileges to perform a specific set of operations on the resources of the project to which it belongs to.

The Identity service also tracks and manages the access to the OpenStack services that are installed on the system. It provides one or more endpoints that can be used to access resources and perform authorized operations. Endpoints are network-accessible addresses (URLs) that can be used to access the service. When the user is authenticated, the keystone server provides a list of services and a token to the user to access the services. For example, if the user is authenticated to access the Object Storage service, the keystone server provides the token to access the service. The Object Storage service then verifies the token and fulfills the request.

To achieve high availability, Object Storage and Keystone are activated on all protocol nodes. The internal database that is required to validate the user is installed on the shared root file system.

Depending upon the configuration, the keystone server needs to interact with AD, LDAP, or the internal database for user authentication. The Keystone internal database is also used for assigning users to projects with a specific role for controlling access to projects. It also holds the definition of OpenStack services and the endpoints for those services.

Each node configured with object storage is configured to interact with the Keystone server.

Deleting authentication and ID mapping

You can choose to delete an authentication method and the current ID mappings that are configured in the system. This may result in a complete loss of data access. Before you delete an ID mapping, determine how you will maintain data access if needed.

It is not possible to delete both an authentication method and the current ID mappings at the same time. If you need to delete the current ID mappings, you need to first delete the authentication method that is configured in the system. If only the authentication settings are deleted, you can return to the existing setting by reconfiguring authentication. There is no such recovery method for a deleted ID mapping. Typically, IDs are deleted in test systems or as part of a service operation.

For more information on how to delete authentication and ID mappings, see *Deleting authentication and ID mapping configuration* in *IBM Spectrum Scale: Administration Guide*.

General authentication support matrix

The following matrix gives a quick overview of the supported authentication configurations for both file and object access.

Table 32. General authentication support matrix

Authentication method	**ID mapping method**	**SMB**	**SMB with Kerberos**	**NFSV3**	**NFSV3 with Kerberos**	**NFSV4**	**NFSV4 with Kerberos**	**Object**
User-defined	User-defined	NA	NA	NA	NA	NA	NA	✓
LDAP with TLS	LDAP	✓	NA	✓	NA	✓	NA	✓
LDAP with Kerberos	LDAP	✓	✓	✓	✓	✓	✓	NA
LDAP with Kerberos and TLS	LDAP	✓	✓	✓	✓	✓	✓	NA
LDAP without TLS and without Kerberos	LDAP	✓	NA	✓	NA	✓	NA	✓
AD	Automatic	✓	✓	X	X	X	X	✓
AD	RFC2307	✓	✓	✓	✓	✓	✓	✓
AD	LDAP	✓	✓	✓	X	X	X	✓
NIS	NIS	NA	NA	✓	NA	✓	NA	NA
Local	None	NA	NA	NA	NA	NA	NA	✓

Note:

- The ID mapping option that is given in this table is only applicable for file access. Ignore the ID mapping details that are listed in the table if you are looking for the supported configurations for object access.
- In the User-defined mode, the customer is free to choose the authentication and ID mapping methods for file and object and manage on their own. That is, the authentication needs to be configured by the administrator outside of the IBM Spectrum Scale commands and ensure that it is common and consistent across the cluster.
- If LDAP-based authentication is used, ACL management for SMB is not supported.

Planning for NFS

There are a number of decisions that must be made before deploying NFS in an IBM Spectrum Scale environment. The following is an outline of these decision points.

The IBM Spectrum Scale for NFS architecture is shown in Figure 32.

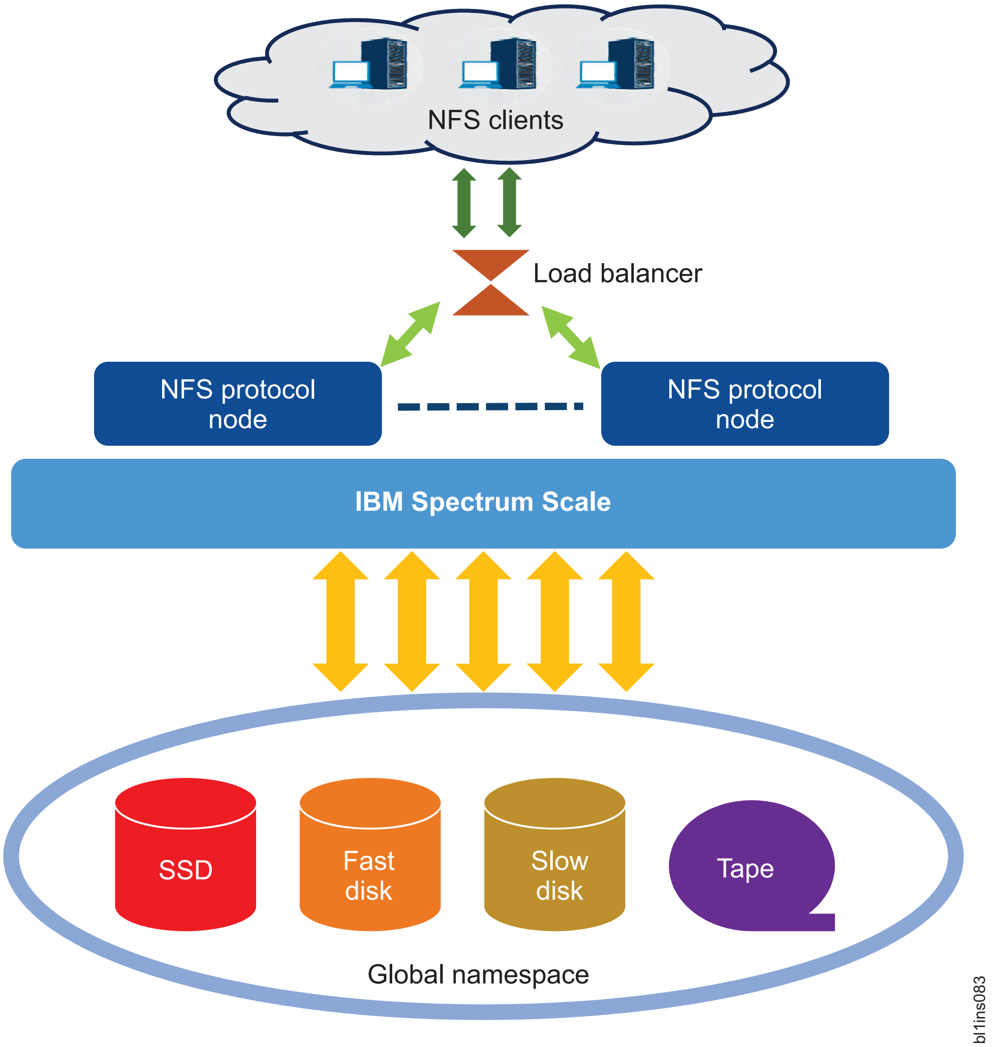


Figure 32. IBM Spectrum Scale for NFS architecture

Each IBM Spectrum Scale protocol node that has NFS enabled, runs the NFS service, other protocol services (SMB and Object, if enabled), and the IBM Spectrum Scale client. NFS users make requests through an NFS client to perform an NFS operation on a mounted NFS file system, for example, to read or to write a file. The request is routed to an NFS protocol node typically by a load balancer or by using DNS round robin. The NFS server on the protocol node handles the request against the backend IBM Spectrum Scale storage file system. The request completion status is then returned to the user through the NFS client.

File system considerations for the NFS protocol

Ensure that all IBM Spectrum Scale file systems that export data via NFS are mounted with the **syncnfs** option to prevent clients from running into data integrity issues during failover.

Linux operating systems

On Linux operating systems, the IBM Spectrum Scale **syncnfs** setting is the default mount option.

AIX operating system

On AIX, the IBM Spectrum Scale **nosyncnfs** setting is the default mount option. Enter the following command to change the setting to **syncnfs**:

```
mmchfs <fileSystem> -o syncnfs
```

Enter the following command to verify:

```
mmlsfs <fileSystem> -o
```

For more information about mount options, see *Mount options specific to IBM Spectrum Scale* in *IBM Spectrum Scale: Command and Programming Reference*.

For fileset considerations for the NFS protocol, see “Fileset considerations for creating protocol data exports” on page 166.

NFS supported clients

IBM Spectrum Scale has tested some clients that can be used for accessing IBM Spectrum Scale data by using NFS protocols.

IBM support can help resolve NFS client / server issues when these can be recreated on one of the NFS clients listed here as tested.

Table 33. Tested NFS clients

Operating System	PPC LE	PPC BE	X86_64	X86_32
RHEL6	✓	✓	✓	✓
RHEL7	✓	✓	✓	✓
SLES12	X	✓	✓	X
AIX6	X	✓	X	X
AIX7	X	✓	X	X

Note: Other operating systems and versions might work, but were not tested.

You should always:

- Install the latest fix level for the operating system
- Run an operating system still supported by the vendor
- Keep IBM Spectrum Scale on the latest fix level

Explicitly not supported

Microsoft Server 2008 NFS client

Considerations for NFS clients

When using NFS clients in an IBM Spectrum Scale environment, the following considerations apply.

- If you mount the same NFS export on one client from two different IBM Spectrum Scale NFS protocol nodes, data corruption might occur.
- IBM Spectrum Scale 4.1.1 and later releases allow concurrent access to the same file data using SMB, NFS, and native POSIX access. For concurrent access to the same data, some limitations apply. For more information, see *Multiprotocol export considerations* in *IBM Spectrum Scale: Administration Guide*.
- 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.
- It is recommended to use the CES IP address of the IBM Spectrum Scale system to mount the NFS export on an NFS client.
  - You can use the **mmces address list** command to view the CES IPs.
  - You can use the **mmlscluster --ces** command to determine which nodes host which CES IPs.
- For items that could affect NFS access, see the *Authentication limitation* topic in the *IBM Spectrum Scale: Administration Guide*.

Planning for SMB

This section describes the steps to be taken before using the SMB service in IBM Spectrum Scale.

The SMB support for IBM Spectrum Scale 4.1.1 and later versions allows clients to access the GPFS file system using SMB clients. The protocol node runs the SMB, NFS, and Object services. The SMB service on the protocol node provides file serving to SMB clients. Requests are routed to an SMB protocol node, typically by using DNS round robin. The SMB server on the protocol node handles these requests by issuing calls to the IBM Spectrum Scale file system.

Note: The SMB service is based on a clustering component CTDB. It is recommended to have the CTDB network traffic on some dedicated private network. The CTDB traffic is using the same network interface as the GPFS daemon. It can be configured or changed through the **--daemon-interface** option in the **mmaddnode** and **mmchnode** commands. The recommendation is to ensure that this is on a private network that is not visible outside the cluster.

SMB file serving

IBM Spectrum Scale provides SMB file serving to SMB clients. This topic describes how to manage SMB file serving in IBM Spectrum Scale.

Thousands of clients can concurrently access the SMB shares in an IBM Spectrum Scale system. The system configuration must be able to support a large number of planned SMB connections. For more information on creating protocol data exports, see “Fileset considerations for creating protocol data exports” on page 166.

SMB connections

Each IBM Spectrum Scale protocol node is capable of handling a large number of SMB connections.

The number of concurrent SMB connections that a protocol node can handle is dependent on the following factors:

- Number of processors
- Amount of memory installed in the protocol node
- The IO workload. This depends on the following factors:
  - Number of concurrent SMB connections
  - Lifetime and frequency of SMB connections
  - Overhead due to metadata operations
  - Frequency of operations on each SMB connections

  - Concurrent access to the same files
- The storage configuration and advanced functions that are configured to run while serving high number of SMB connections

SMB fail-over scenarios and upgrade

When you are planning an IBM Spectrum Scale system configuration that has a high number of SMB connections, you must consider the impact that a fail-over can have on the performance of the system.

In the event that an IBM Spectrum Scale protocol node fails, the IP addresses hosted by that protocol node are located to another IBM Spectrum Scale protocol node. SMB clients have to re-connect to one of the remaining CES nodes. They can use the same IP address after the fail-over is hosted by another CES node. These remaining protocol nodes handle all the SMB connections.

Therefore, when you plan an IBM Spectrum Scale system that has a high number of SMB connections, some buffer in terms of number of SMB connections must be factored into the overall system configuration. This will prevent a system overload during a fail-over scenario, thus reducing any adverse effects to the system performance.

A similar consideration applies to SMB upgrades. The SMB upgrade happens in two phases. During the first phase of the upgrade process, the first half of the nodes are updated. The remaining nodes handle the SMB connections of the node being updated. Once the first half of the nodes are updated, the upgrade moves to the second phase. In the second phase, SMB will be shut down completely. This is done in order to update the SMB code on all of the remaining protocol nodes concurrently. This results in a brief outage of the SMB service.

SMB limitations

IBM Spectrum Scale can host a maximum of 1,000 SMB shares. There must be less than 3,000 SMB connections per protocol node and less than 20,000 SMB connections across all protocol nodes.

IBM Spectrum Scale 4.1.1 and later versions allow concurrent access to the same data file through SMB and NFS, and through native POSIX access.

No support of SID history.

For more details on these options, see the *mmsmb command* in the *IBM Spectrum Scale: Command and Programming Reference*.

For more information on SMB client access and limitations, see *Multiprotocol export considerations* in the *IBM Spectrum Scale: Administration Guide*.

SMB client limitations

Windows: No access for Windows XP or Windows 2003 clients, as SMB protocol version 1 is not supported by IBM Spectrum Scale.

Linux: The Linux kernel SMB client defaults to SMB protocol version 1. However, IBM Spectrum Scale does not support this protocol version. Use the `version` option with the kernel SMB client to set a higher protocol version:

```
mount.cifs //fsc_serv/ral /media/ss –o user=aduser1,pass=password1,dom=W2K8DOM05,vers=2.0
```

Note: Check with the vendor of your client operating system about the support provided for SMB2 or SMB3 protocols.

Mac Operating System: To avoid potential file ID inconsistencies between the client and the server, it is recommended that you set the file ID to off.

[...] you set the file ID to off.

From the Mac client command line, create a new file called `/etc/nsmb.conf`.

In the file, add:

```
[default]
file_ids_off=yes
```

Windows: SMB 1 is not supported, thus no access for Windows XP clients.

The GPFS SMB users trying authentication via IBM Spectrum Scale CLI or GUI: The GPFS SMB server is using the special notation "<username>%<password>" to specify user name and password by using a single string. As all user names are interpreted by using this notation, it is not possible to have user names containing '%' characters.

Note: This limitation does not apply for SMB protocol access.

Linux: SMB1 is not supported, so make sure to use the version option with the kernel SMB client:

```
mount.cifs //fsc_serv/ral /media/ss -o user=aduser1,pass=password1,dom=W2K8DOM05,vers=2.0
```

```
[default]
file_ids_off=yes
```

SMB share limitations

Consider all the limitations and support restrictions before you create an SMB share.

SMB share limitations include the following:

- 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 **`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.
- Concurrent access to the same files or directories from many SMB connections on multiple nodes can result in scalability problems.
- 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.
- IBM Spectrum Scale can act as an MS-DFS target but does not currently allow any redirects.
- 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.

- No support of dynamic ACLs and SACLs.
- FLUSH requests are a no-op.
- SMB Direct (SMB over RDMA) is not supported.
- Multi-channel for SMB is not supported.
- Durable handles are not supported.
- Resilient® handles are not supported.
- Persistent handles are not supported.
- Storing arbitrary data in Security Descriptors is not supported.
- Pre-allocating space larger than the file size is not supported. A check for the available disk space is used instead when processing an allocation request from an SMB client.
- The Witness notification protocol is not supported.
- SMB protocol version 1 and older dialects are not supported.
- Continuous Availability (CA) SMB exports are not supported.
- Scaleout SMB exports are not supported.
- Creating snapshots from an SMB client is not supported.
- Application instance ID is not supported.
- FSCTL_SET_ZERO_DATA control code to deallocate space in an existing file is not supported.
- Setting compression from an SMB client or querying the compression state from an SMB client is not supported.
- All CES nodes need to be the same hardware architecture (x86 versus Power) and the same endianness (little endian versus big endian).
- Rolling code upgrade is not supported for SMB. The SMB service needs to be stopped on all CES nodes for the upgrade to ensure that the same SMB software level is used on all CES nodes.
- The **AAPL create** context used by Mac OS clients is not supported.
- The number of open files for each SMB connection is limited to 16384.
- No support of SMB 3.1.

SMB node limitations

IBM Spectrum Scale allows a maximum of 16 nodes in a CES cluster.

Planning for object storage deployment

There are a number of decisions that must be made before beginning the object storage deployment in an IBM Spectrum Scale environment. The following is an outline of these decision points.

The IBM Spectrum Scale for object storage architecture is shown in Figure 33 on page 203.

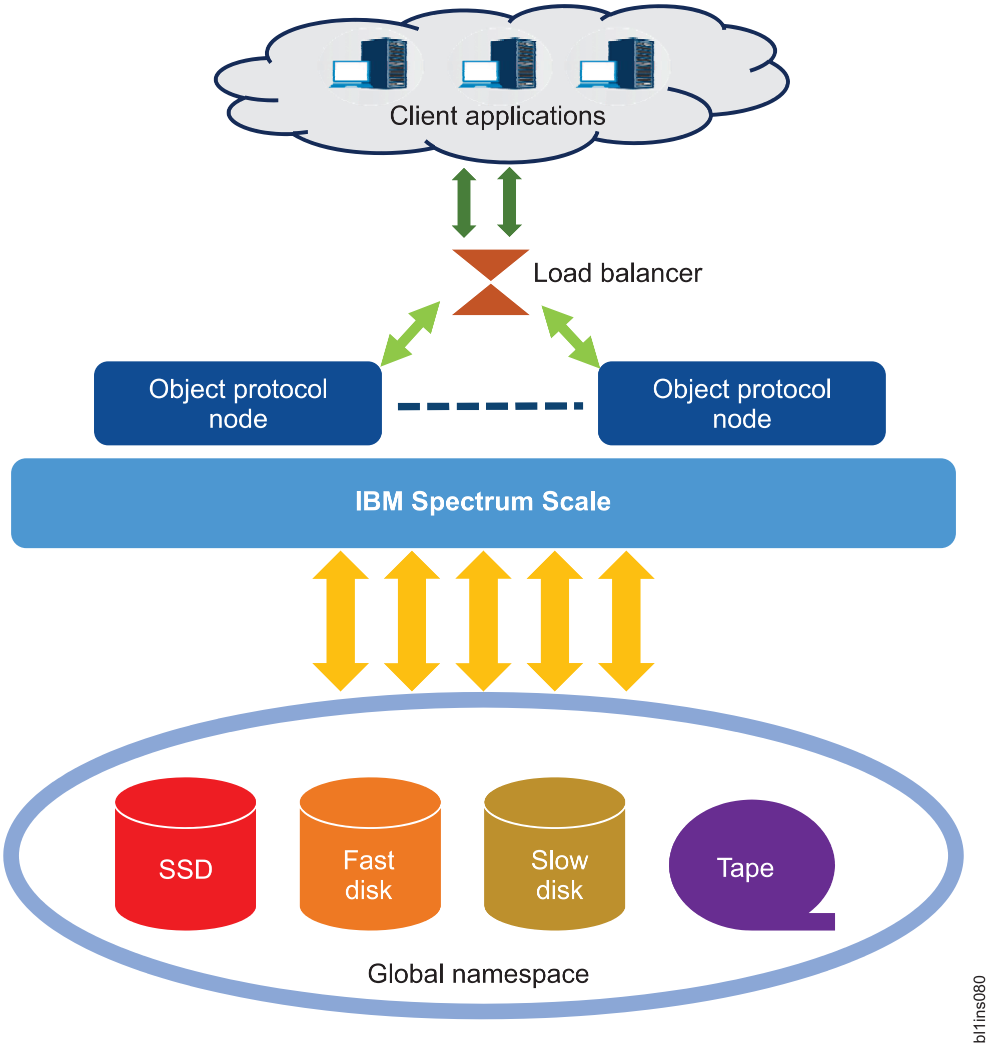


Figure 33. IBM Spectrum Scale for object storage architecture

For every object PUT request, the following inodes are created:

- a hashes directory for a PUT request of every new object
- a target object file
- the parent directory of the hash directory if it doesn't already exist
- the partition directory unless it already exists
- a `hashes.pkl` file
- a temporary `.lock` file

Each protocol node runs all OpenStack Swift object services, the Keystone identity service, and the IBM Spectrum Scale client. Client applications make requests to perform object operations such as uploading an object, downloading an object, and deleting an object. The request is routed to a protocol node typically by a load balancer or by using DNS round robin. The protocol node implements that request by creating, retrieving, or deleting the object on the backend IBM Spectrum Scale storage. The request completion status is then returned to the client application. Each client request must include an authentication token. Typically, client applications first request a token from the Keystone service, and then provide that token in the subsequent object requests, until the token expires. At that point, a new token can be requested.

Client applications make requests to perform operations such as uploading an object, downloading an object, and deleting an object. Account and container information can also be updated, viewed, and removed by client applications. For more information on the API available to clients, see the OpenStack Swift API documentation here: http://developer.openstack.org/api-ref/object-storage/index.html

For more information on OpenStack Swift, see OpenStack Swift documentation.

For more information on OpenStack Keystone as it is used with Swift, see Keystone Auth section in the OpenStack Swift documentation.

Load balancing

All IBM Spectrum Scale protocol nodes are all active and provide a front-end for the entire object store. Requests can come in directly to the protocol nodes, but it is common to use a load balancer to distribute requests evenly across the nodes.

Any web load balancer product can be used, or DNS Round Robin can be used. In either case, there is an IP address and a corresponding **cluster host name** that is used to identify the object storage cluster. This is the address clients communicate with, and the load balancer (or DNS) routes client requests to a particular protocol node.

Cluster host name

The cluster host name is required during the installation process.

It is the **endpoint** parameter in the **spectrumscale config object** command and it is the **ces_hostname** parameter in the **mmobj swift base** command. For more information, see *mmobj command* and *spectrumscale command* in *IBM Spectrum Scale: Command and Programming Reference*. Additionally, the cluster host name might be used in your load balancer or DNS round robin configuration. It is the fully qualified name that clients send their object requests to.

Authentication method

IBM Spectrum Scale for object storage supports a flexible array of configuration options for authentication.

If you already have Keystone deployed in your environment, you can configure IBM Spectrum Scale for object storage to use the existing or external Keystone. If you do configure Keystone as part your IBM Spectrum Scale cluster, you can manage the user information locally, or you can integrate it with a Microsoft Active Directory or LDAP system.

Backup and disaster recovery strategy

For information on backup and disaster recovery strategy for your object storage, see the *Protocols cluster disaster recovery* section in the *IBM Spectrum Scale: Administration Guide*.

SELinux considerations

To simplify the configuration of the IBM Spectrum Scale for object storage environment, the installation process detects whether SELinux is enabled or not. If SELinux is enabled, the installation process performs steps so that the object services and the database software running on the protocol nodes can interact with the required file system and system resources.

The `openstack-selinux` package is installed automatically when the `spectrum-scale-object` RPM is installed. This configures the object services for SELinux.

If the installer detects that SELinux is enabled, it does the following steps:

1. Ensures that the Postgres database can access the Keystone database directory on the CES shared root file system:

```
semanage fcontext -a -t postgresql_db_t "<keystone db directory>(/.*)?"
semanage fcontext -a -t postgresql_log_t "<keystone db directory>/log(/.*)?"
restorecon -R "<keystone db directory>"
```

2. Ensures that object processes can access the object data fileset:

   ```
   semanage fcontext -a -t swift_data_t "<object fileset directory>(/.*)?"
   restorecon -R <object fileset directory>/*
   ```

Attention: If SELinux is disabled during installation of IBM Spectrum Scale for object storage, enabling SELinux after installation is not supported.

SELinux packages required for IBM Spectrum Scale for object storage

IBM Spectrum Scale for object storage requires the following SELinux packages to be installed:

- `selinux-policy-base` at 3.13.1-23 or higher
- `selinux-policy-targeted` at 3.12.1-153 or higher

Eventual consistency model

An eventual consistency model is used when uploading or deleting object data.

According to the CAP theorem, in distributed storage systems, a system can guarantee two of the following three claims:

1. Consistency
2. Availability
3. Partition tolerance

With Swift and IBM Spectrum Scale for object storage, the availability and partition tolerance claims are chosen. Therefore, in some cases, it is possible to get an inconsistent listing of object store contents. The most common case is consistency between container listing databases and objects on disk. In the case of a normal object upload, the object data is committed to the storage and the container listing database is updated. Under heavy load, it is possible that the container listing database update does not complete immediately. In that case, the update is made asynchronously by the object-updater service. In the time between the original object commit and when the object-updater executes, you will not see the new object in the container listing, even though the object is safely committed to storage. A similar situation can occur with object deletes.

Planning for unified file and object access

Unified file and object access allows use cases where you can access data using object as well as file interfaces. Before using unified file and object access, you need to plan for a number of aspects.

Planning for identity management modes for unified file and object access:

As you plan for unified file and object access, it is important to understand the identity management modes for unified file and object access.

Based on the identity management mode that you plan to use, you need to plan for the authentication mechanism to be configured with file and object. In an existing IBM Spectrum Scale setup, where an authentication mechanism is already set up, a suitable identity management mode needs to be chosen for unified file and object access.

For more information, see *Identity management modes for unified file and object access* in *IBM Spectrum Scale: Administration Guide*.

Authentication planning for unified file and object access:

Planning for a suitable authentication mechanism for a unified file and object access setup requires an understanding of the Identity management modes for unified file and object access and the authentication setups supported with these modes.

For more information, see *Authentication in unified file and object access* and *Identity management modes for unified file and object access* in *IBM Spectrum Scale: Administration Guide*.

ibmobjectizer service schedule planning:

The unified file and object access feature allows accessing data ingested through file to be accessed from object. This access is enabled by a process called objectization. Objectization is done by the **ibmobjectizer** service that runs periodically and converts the file data located under unified file and object access enabled filesets to make it available for object access.

It is important to understand the amount and frequency of file data that is expected to be ingested and objectized because the objectization service needs to be scheduled accordingly by the administrator.

For information about the **ibmobjectizer** service, see *The objectizer process*, *Setting up the objectizer service interval*, and *Configuration files for IBM Spectrum Scale for object storage* in *IBM Spectrum Scale: Administration Guide*.

Prerequisites for unified file and object access:

To enable unified file and object access, you must enable the `file-access` object capability.

For more information see "Object capabilities" on page 35.

For unified file and object access, the authentication prerequisites depend on the identity management mode for unified file and object access that you plan to use. It is important to decide the mode before configuring unified file and object access, although it is possible to move from one mode to another. For more information on changing identity management modes for unified file and object access, see *Configuring authentication and setting identity management modes for unified file and object access* in *IBM Spectrum Scale: Administration Guide*.

For more information, see *Authentication in unified file and object access* and *Identity management modes for unified file and object access* in *IBM Spectrum Scale: Administration Guide*.

Unified file and object access is deployed as a storage policy for object storage. Therefore, it is important to understand the concept of a storage policy for object storage and how to associate containers with storage policies. For more information, see "Storage policies for object storage" on page 32.

Planning for multi-region object deployment

Use the following information to plan your multi-region object deployment.

Enabling multi-region enables the primary rings to be multi-region and all data to be stored in all regions. Storage policies can be used to limit objects to certain regions.

For information on enabling multi-region object deployment, see "Enabling multi-region object deployment initially" on page 315.

For information on adding a new region to a multi-region object deployment environment, see *Adding a region in a multi-region object deployment* in *IBM Spectrum Scale: Administration Guide*.

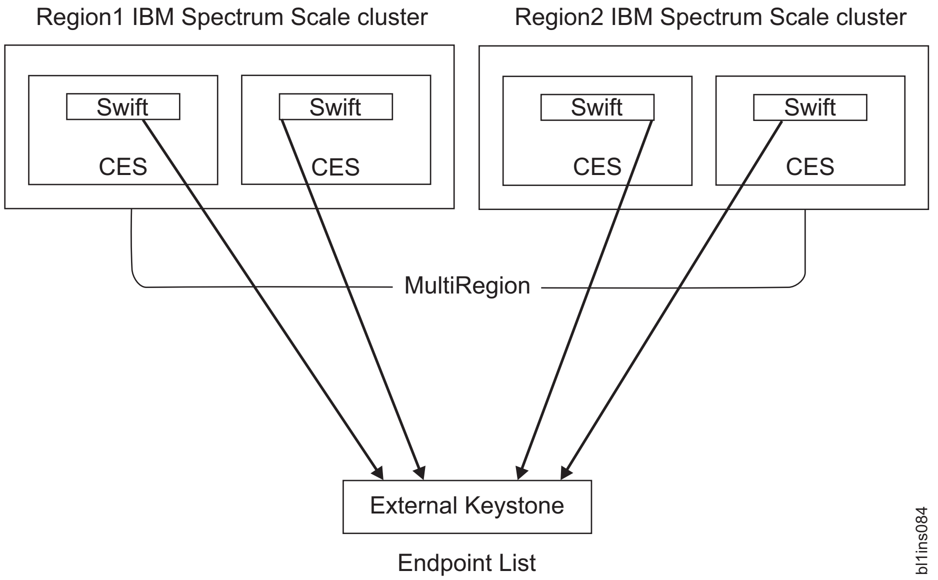
Authentication considerations for multi-region object deployment:

In a multi-region object deployment environment, all regions must use the same Keystone service.

The keystone service can be a local keystone installed with the object deployment or it can be an independent service. Subsequent clusters that join the environment must specify an external keystone server during installation.

The following two methods can be used for object authentication configuration with a multi-region setup:

- By using the external keystone



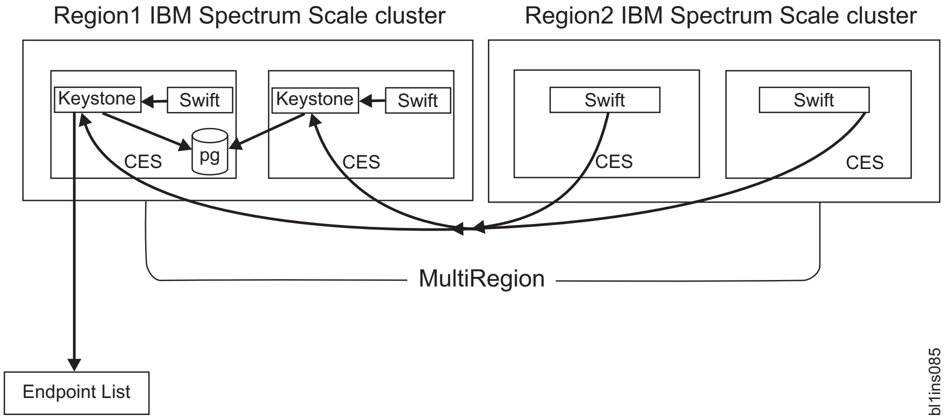
```
+------+-----------+--------------+--------------+---------+-----------+-------------------------------------------+
| ID   | Region    | Service Name | Service Type | Enabled | Interface | URL                                       |
+------+-----------+--------------+--------------+---------+-----------+-------------------------------------------+
| e310 | RegionOne | swift        | object-store | True    | public    | http://region1:8080/v1/AUTH_%(tenant_id)s |
| 1679 | RegionOne | swift        | object-store | True    | internal  | http://region1:8080/v1/AUTH_%(tenant_id)s |
| c458 | RegionOne | swift        | object-store | True    | admin     | http://region1:8080                       |
| 8a01 | Region2   | swift        | object-store | True    | public    | http://region2:8080/v1/AUTH_%(tenant_id)s |
| b821 | Region2   | swift        | object-store | True    | internal  | http://region2:8080/v1/AUTH_%(tenant_id)s |
| 5188 | Region2   | swift        | object-store | True    | admin     | http://region2:8080                       |
+------+-----------+--------------+--------------+---------+-----------+-------------------------------------------+
```

Note: The external keystone and HA must be managed and configured by the customers.

1. On all the participant clusters of the multi-region setup, configure the external keystone with the **`spectrumscale auth object external`** command.
2. Set the **`keystone_url`** and **`configure_remote_keystone`** properties.
3. For manual installation, use the **`mmobj swift base`** command with the **`--remote-keystone-url`** and **`--configure-remote-keystone`** arguments.

Note: The installer can automatically create these endpoints if the option to configure the remote keystone is used during installation and **`–configure-remote-keystone`** is specified.

- By using the keystone installed on one of the IBM Spectrum Scale clusters



```
+------+-----------+--------------+--------------+---------+-----------+------------------------------------------+
| ID   | Region    | Service Name | Service Type | Enabled | Interface | URL                                      |
+------+-----------+--------------+--------------+---------+-----------+------------------------------------------+
| e310 | RegionOne | swift        | object-store | True    | public    | http://region1:8080/v1/AUTH_%(tenant_id)s |
| 1679 | RegionOne | swift        | object-store | True    | internal  | http://region1:8080/v1/AUTH_%(tenant_id)s |
| c458 | RegionOne | swift        | object-store | True    | admin     | http://region1:8080                      |
| 8a01 | Region2   | swift        | object-store | True    | public    | http://region2:8080/v1/AUTH_%(tenant_id)s |
| b821 | Region2   | swift        | object-store | True    | internal  | http://region2:8080/v1/AUTH_%(tenant_id)s |
| 5188 | Region2   | swift        | object-store | True    | admin     | http://region2:8080                      |
+------+-----------+--------------+--------------+---------+-----------+------------------------------------------+
```

Note: If the region1 cluster stops functioning, the complete multi-region setup will be unusable because the keystone service is not available.

1. On the first cluster of multi-region setup, configure local Keystone with the **`spectrumscale auth object local|ad|ldap`** command by using the installation toolkit.
2. For manual installation, use the **`mmobj swift base`** command with the **`--local-keystone`** arguments for configuring with keystone with local authentication type.
3. For configuring the object authentication with ad | ldap, use **`mmuserauth service create|delete`** command after **`mmobj swift base`** with **`--local-keystone`**.
4. On the second and third clusters of the multi-region setup, configure the external keystone with the **`spectrumscale auth object external`** command.
5. Set the **`keystone_url`** and **`configure_remote_keystone`** properties.
6. For manual installation, use the **`mmobj swift base`** command with the **`--remote-keystone-url`** and **`--configure-remote-keystone`** arguments.

Note: The installer can automatically create these endpoints if the option to configure remote keystone is used during installation if **`--configure-remote-keystone`** is specified.

Data protection considerations for multi-region object deployment: Each cluster must maintain appropriate backups. A multi-region cluster can be more resilient to failures because if a cluster loses multi-region objects or a single cluster fails, the lost objects are restored through the normal Swift replication behavior.

Since all regions use the same Keystone service, ensure the Keystone data is backed up appropriately.

Network considerations for multi-region object deployment:

To communicate with nodes in other regions, the Swift services connect to the network addresses as defined in the ring files, which are the CES IP addresses. This means that every node must be able to connect to all CES IPs in all regions.

Ensure that the network routing is properly set up to enable the connections. Ensure that the necessary firewall configuration is done to allow connection to the object, account, and container servers (typically ports 6200, 6201, and 6202) on all nodes in all regions. Swift uses **rsync** to replicate data between regions, so the **rsync** port 873 also needs to be opened between the regions.

Monitoring and callbacks in an multi-region object deployment setup: Within a region, the monitoring of multi-region ring files and distribution to nodes is dependent upon the existing monitoring framework and the work done to enable storage policies in the monitoring layer. For the distribution of ring and configuration changes to other regions, the administrator needs to perform those steps manually using the **mmobj multiregion** command. For more information, see *mmobj command* in *IBM Spectrum Scale: Command and Programming Reference*.

Performance considerations for multi-region object deployment: Before objects are replicated from one cluster to another, there is a delay because of the replicator run time and the network load.

Each region's access to the Keystone server affects the performance of authentication capabilities within its region.

Planning for Cloud services

Proper planning is one of the most important activities that you need to carry out for successful implementation of Cloud services on the IBM Spectrum Scale cluster.

Hardware requirements for Cloud services

You must meet certain hardware requirements to be able to install and use Cloud services on the IBM Spectrum Scale cluster.

For running Cloud services, adhere to the recommended CPU and memory requirements for a CES node even when installing Cloud services on an NSD node. For more information, see http://www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html.

The hardware requirements for Cloud services are:

- Any standard x86 64-bit servers or Power Linux nodes that run supported Linux distributions.
- The minimum size that is required for the `/var/MCStore` folder is 12 GB.

Note: For better performance, it is recommended to have a minimum of 2 CPU socket server of the latest Intel variety with at least 128 GB of memory.

A high CPU count promotes better cloud tiering throughput because although object storage can be slow in I/O operations per thread, object storage can support many threads. Use sixteen or more CPUs when you select your hardware.

Cloud tiering services demand a large amount of memory, which is why the minimum recommended memory size is 128 GB. Memory size requirements increase if the number of files increases, as you add files on the cloud means you must increase the memory that is on your system. It is recommended that you use 10 - 20 times as much memory that is required so the Cloud services can cache its directory database data.

Software requirements for Cloud services

You must meet certain software requirements to be able to install and use Cloud services on the IBM Spectrum Scale cluster.

Note: Cloud services is shipped along with IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only. Therefore, IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition is required for the installation.

The software requirements for Cloud services are

- Cloud services core server package is installed on nodes that run RHEL 7.2 Linux distribution and above. Cloud services core server package can also be installed on nodes that run RHEL 7.2 on Power8 Little Endian.
- The client/binary package can be installed on other Linux distributions such as SLES 11.2, Ubuntu 14.x, and RHEL 7.2 and above.
- It is required to use the following Linux version:
  - RHEL 7.2 (Maipo)
  - Kernel version: Cloud services require the minimum kernel revision to be at 3.10.0-327.22.2.el7.x86_64. For more information, see IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).
- The following packages must be installed on the Transparent cloud tiering core server nodes:
  - Redhat-release-server
  - Unzip
  - Tar
  - bc - An arbitrary precision calculator language
  - SQLite 3.7 and above
  - GPFS base RPMs
  - GPFS RPMs required for a CES node
  - GPFS Advanced Edition RPMs
  - gpfs.ext-5.0.0-0-0.x86_64 (for client nodes)
- For the delete functionality to work in Swift, verify that the Bulk middleware to be in pipeline, as follows:

```
vim /etc/swift/proxy-server.conf
[pipeline:main] pipeline = healthcheck cache bulk authtoken keystone proxy-server
```

Network considerations for Cloud services

This topic describes the network considerations that you need to follow for installing Cloud services on your IBM Spectrum Scale cluster.

Note:

- Network Time Protocol (NTP) must be correctly set on all nodes where the Cloud services are installed. If correct date and time are not set, some of the cloud storage providers will refuse to authenticate your requests.
- Cloud services always use the GPFS cluster (daemon) network and not the CES network.

Cloud storage supports storing cold data on a public cloud object storage service, such as IBM Softlayer or Amazon S3. When using a public cloud, the connection is over a WAN.

Cloud services for both Transparent cloud tiering and Cloud data Sharing can consume a considerable amount of WAN bandwidth. Customers need to ensure that there is sufficient WAN bandwidth for migrations and recalls along with any other WAN usage that they might require. When Cloud services run, an administrator can monitor WAN usage and other running processes in the IBM Spectrum Scale performance monitor. For a list of metrics, see the *Cloud services* section in the *List of Metrics* topic in the *IBM Spectrum Scale: Administration Guide*.

To avoid any conflict of Cloud services I/O traffic and internal IBM Spectrum Scale cluster I/O traffic, it is recommended that the Cloud services node is configured with two or more NICs. One NIC card is designated as the target for all Transparent cloud tiering I/O, and the other one is designated for Cloud services I/O traffic.

After the NIC card is set up and connected to the WAN, it is recommended to run any external network speed test, such as speedtest.net, to ensure that the I/O bandwidth to the external network meets your expectations.

If customers choose to configure a private cloud instance, they need to ensure that there is sufficient bandwidth between the IBM Spectrum Scale cluster and the private cloud to support migrating and recalling files. It is preferred to have a 10 GB network for higher throughput.

Cluster node considerations for Cloud services

This topic describes the cluster considerations that need to be followed before you install Cloud services on the IBM Spectrum Scale cluster.

You can install the core Cloud services packages on a maximum of 4 CES or NSD (or a combination of both) nodes that you want as members of your Cloud services node class. The client package is installed on all other nodes within the same cluster. Cloud data transfer (migration, recall, import, or export) occurs only from the Cloud services nodes. However, you can initiate a cloud data transfer from either a server or a client node.

Do not use a mixed node cluster configuration in which there are nodes on the cluster that do not allow the installation of clients. For example, Windows nodes do not allow the installation of clients because there is no Transparent cloud tiering Windows client. These configurations:

- Do not support transparent data recalling
- Cannot respond to requests that initiate from nodes on which a client is not installed
- Cannot efficiently perform deletions

Cloud services is supported for use only in IBM Spectrum Scale clusters and on any associated remotely mounted clients, with Intel x86 Linux and Power LE Linux nodes. Use of Windows, PowerPC Big Endian, Linux on Z or AIX nodes that access file systems, where Cloud services is used (either within the cluster or via a remotely mounted client), is untested and unsupported. Inclusion of such nodes is allowed only if there is no possibility that they will ever access Cloud services; for example, such nodes are tested and supported for PowerPC Big Endian nodes in Elastic Storage Server (ESS) BE & LE servers, where no user applications are supported on Elastic Storage Server (ESS).

IBM Cloud Object Storage considerations

This topic describes about the points that you need to consider before you use IBM Cloud Object Storage as the object storage provider.

Note: Cloud services 3.7.3.2 or above is required for the Cloud services functionality. If you are running an older versions of IBM Cloud Object Storage, please contact IBM support for upgrading your version.

Before you begin, ensure that an access pool is created and available on the IBM Cloud Object Storage system. The access pool must have 'Cloud Storage Object' configured as the API type.

Before you configure Cloud services with IBM Cloud Object Storage as the object storage provider, ensure that the following settings are done through IBM Cloud Object Storage dsNet Manager GUI. If these settings are not correctly set, the cloud account configuration or data migration will fail.

- During the Cloud services setup process, two vaults are created on the IBM Cloud Object Storage system. One of the vaults is used for storing data and the other one is used for storing metadata. The data vault that is created by Cloud services contains the “container-prefix” appended with a unique ID.

  Note: IBM Cloud Object Storage endpoint URL (“cloud_url” command line option) should not have this container or vault name.

- In order for Cloud services to be able to create the vault, the user (whose access key is specified in the **mmcloudgateway account create** configuration command) should have "Vault Provisioner" role assigned through the dsNet Manager UI.

  To create vaults, go to **dsNet Manager** UI > **Security** tab and click **user** > **Roles** > **Add "Vault Provisioner"** role.

  Ensure that either "Create Only" or "Create and Delete" permission is selected under **Administration** > **Provisioning API configuration**. This enables Cloud services to create vaults by using the IBM Cloud Object Storage vault provisioning APIs.

  Note: Delete access is not required for the "Vault Provisioner".

  Note: To specify the container name of an existing container, use the **--container-name** parameter of the **mmcloudgateway** command.
- For IBM Cloud Object Storage deployed on public cloud and with container mode enabled, user has to contact the cloud object storage admin to obtain the accessor IP address/hostname, S3 credentials, and provisioning code (location) for the container vault.
- To create vaults through the provisioning API, IBM Cloud Object Storage uses provisioning templates. You can provide a provisioning template to the Cloud services in two ways:
  - **Using default template**: A default vault template can be set on the dsNet Manager UI. Go to **dsNet Manager** UI > **Configure** tab > **Storage pools** > **Create Vault template**. Then, **dsNet system** > **Default Vault template** and then select the newly created template. The vault that is created by Cloud services uses the default template.

    Note: The default template on the IBM Cloud Object Storage system should have index enabled.
  - **Using Provisioning code**: If the default vault template is not set or not preferred, then a provisioning code can be used. Ensure that during the creation of the vault template, a unique user-defined provisioning code is specified. The provisioning code is different from the name.

    To look up the provisioning code, go to dsNet Manager UI > **Configure** tab > **Storage pools** > **Vault Templates** > select the template, and look for "Provisioning Code"). Use the **--location** option in the **mmcloudgateway account create** command to specify the provisioning code. Using this mechanism, the vault template configured for Cloud services is used for vault provisioning.

    Note: If there is no default provisioning template set on the IBM Cloud Object Storage system and a provisioning code is not specified to the **mmcloudgateway account create** command, then the command will fail. If the provisioning code specified is not found on the IBM Cloud Object Storage system, then the command will fail.

The following settings are recommended when you create a vault template on IBM Cloud Object Storage dedicated for Transparent cloud tiering.

Table 34. Recommended settings when creating a vault template on IBM Cloud Object Storage

Configuration	Recommended Values	Comment
Width	As per IBM Cloud Object Storage documented configuration for production	
Threshold	As per IBM Cloud Object Storage documented configuration for production	
WriteThreshold	As per IBM Cloud Object Storage documented configuration for production	

Table 34. Recommended settings when creating a vault template on IBM Cloud Object Storage (continued)

Configuration	Recommended Values	Comment
Alert Level	As per IBM Cloud Object Storage documented configuration for production	
Alert Level	As per IBM Cloud Object Storage documented configuration for production	
SecureSlice Technology	Disabled	The Cloud services performs the client-side encryption of all data before the data is stored on the object storage. Hence encryption is disabled on the vault.
SecureSliceAlgorithm	Not applicable since SecureSlice is disabled.	
Versioning	Disabled	Transparent cloud tiering has built-in versioning capability, hence IBM Cloud Object Storage versioning can be disabled. For the Cloud Data Sharing service, versioning may or may not be turned off depending on the needs for retaining versioning on the data.
DeleteRestricted	Yes/No	Gateway does not attempt to delete the vaults, hence this setting can be set to Yes or No.
Name Index	Disabled	Disabling this setting can result in improved Vault performance.
Recovery Listing	Enabled	When Name Index is disabled, this setting is enabled to be able to conduct a disaster recovery if needed.

Essentially, IBM Cloud Object Storage needs two provisioning templates. One of them is used for storing data and the other one is used for metadata/book keeping. For performance reasons, the vault used for storing data should have **Name Index** disabled and for searchability reasons, the other vault should have index enabled. The second provisioning template should have **Name Index** enabled and the rest of the settings are the same as above. This vault provisioning template must be set as default (Click the **Configure** tab and scroll down to see the option to set the default template). Pass the provisioning code ('demo' in the example) of the first vault provisioning template to the **mmcloudgateway** command during account creation by using the **--location** parameter.

Firewall recommendations for Cloud services

This topic describes the firewall recommendations that you need to follow to be able to implement Cloud services on your cluster.

Port that can be used is 8085 TCP

- Enables connections to Cloud services nodes from all IBM Spectrum Scale nodes on this port. All communications from non-cluster nodes on this port can be blocked.
- Cloud services nodes are required to communicate with the configured Object storage provider. Typically, this communication occurs over HTTPS (443) or HTTP (80). Contact your Object storage provider for more details.
- The internal port that is used by Cloud services can be changed from 8085 to any other port by using the **mmcloudgateway config** command.

Table 35. Port requirements

Port number	Protocol	Service name	Component involved in communication
8085	TCP	Transparent cloud tiering	Intra cluster
Object storage provider dependent	TCP	Transparent cloud tiering	Transparent cloud tiering connection to Object storage provider on the external network. Typically HTTPS (443) or HTTP (80)

Note: For firewall recommendations for other components such as performance monitoring tool, protocol access, and GUI, see the *Securing the IBM Spectrum Scale system using firewall* topic in the *IBM Spectrum Scale: Administration Guide*.

Performance considerations

While default configurations on the IBM Spectrum Scale and Cloud services work for the migration and recall of data, some adjustments might be required for optimized performance. This topic addresses some of such adjustments that will likely have the most significant effect on performance.

IBM Spectrum Scale cluster hardware

Memory, CPU, storage, and network play a significant role in determining how well the cloud tiering is going to perform.

Memory: Cloud services requires at least 2 GB RAM for the heap memory, and theIBM Spectrum Scale page pool should also be reasonably high. It is recommended to have a minimum of 64 GB RAM for page pool on the Gateway nodes.

CPU: Cloud services can run more threads to transfer data if the system has more CPU cores available. For better performance, it is recommended to have a minimum of 24 or 32 CPU cores (the more, the better) on the Cloud services gateway nodes.

Storage: The read/write performance from the IBM Spectrum Scale file system is one of the major factors that determine the migrate/recall performance. To achieve desirable performance, the read/write throughput should be more than the available network bandwidth to the cloud. Some of the general recommendations are:

- Have flash disks for storing metadata on the file system.
- If possible, have a separate flash-based file system to keep the Cloud services database. This will ensure that the internal cloud service operations that can span the entire database (like reconcile) run in a reasonable amount of time.

  Note: Adding flash storage helps particularly with scaling up the number of files that are supported by Cloud services and is apparent when you get to the hundreds of millions of files migrated to the cloud. So, this recommendation on flash storage only applies if you are scaling to those kinds of numbers.

Network: It is recommended to use 10 GB Ethernet for the network interfaces that are used to transmit data to the cloud. In case of on-premise object store, the object store also should have 10 GB Ethernet. It is recommended to have a dedicated external links to the cloud storage and not to use these interfaces for intra-cluster communication.

Software: Cloud services uses the Data Management Application programming Interface (DMAPI) of the IBM Spectrum Scale to read and write the data from the file system. As Cloud services can migrate data only as fast as it might read from the file system, it is important to ensure that the read/write performance of the file system is good. Set the following key configuration parameters by using the `mmchconfig` command:

- **pagepool** – 16 GB
- **maxGeneralThreads** – 1280
- **dmapiWorkerThreads** – 64
- **maxFilesToCache** – 100k
- **workerThreads** – 1024

Another important IBM Spectrum Scale configuration is the file system block size. It is recommended to have the block size greater than theCloud services slice size (default 512 KB). This will ensure that you do not read from the disk more than once to fill in a single slice.

Cloud services configurations

Cloud services has configurable parameters, which can be adjusted based on the environment:

- `connector.server.migrate-threadpool-size` – This determines the number of internal threads that accept the migrate requests from the command line. The default value is 32, and is ideal for most of the situations where you have up to 32 CPU cores. If you have a gateway node with more than 32 CPU cores, it is advisable to increase the number to match the number of CPU cores that you have, so Cloud services can process more migrate requests at once.

  Note: This recommendation is for Intel servers. For Power 8 LE servers, you can efficiently run two threads per CPU core, so adjust your thread count accordingly.
- `connector.server.recall-threadpool-size` – Normally you do not adjust this parameter. This determines the number of internal threads that accept recall requests from the command line. If you are sure of your workload, you can adjust the migrate and recall thread pool size. For example, if you have a very cold archive application and expect more migrates and a very limited number of recalls, then you could increase the migrate thread pool size and reduce the recall thread pool size.

For more information, see the *Tuning Cloud services parameters* topic in the *IBM Spectrum Scale: Administration Guide*.

Recommendations on number of Gateway nodes: While increasing the number of gateway nodes to migrate data can increase the total performance, it is to be remembered that we have to find an optimum limit. For example, if the network link from each node to the cloud is 10 Gb and overall read throughput of the file system is 2 GB/s on cluster, just two gateway nodes could achieve the maximum throughput possible. Depending on the file system read/write throughput and the network link, the user has to make an informed decision on what should be the optimal number of gateway nodes.

Recommendation on number of Cloud Storage Access Points (CSAPs): If you are going to a public cloud service such as the public IBM Cloud Object Storage (COS) service, there is no need to add in multiple access points since these services have a built-in load balancer. For on-premise object storage, you may choose to add in multiple Cloud Storage Access Points (CSAP) if you are not using a load balancer as follows:

- If there are multiple gateway nodes in the cluster, using more than one CSAP for a Cloud Account is a good idea. The number of requests that can be processed by a single Cloud Storage Access Point is limited, and multiple nodes hitting the same access point can cause the performance to drop. A general guide line is to have about as many CSAPs as the number of gateway nodes.

Mixing migrate and recall workloads

Cloud services internally shares threads that carry out an operation on each slice of data, as well as threads that carry out the interactions with cloud object storage. Hence, running a huge workload of migrates and recalls simultaneously, would affect the total turn-around of each activity, compared to running migrates and recalls separately. Cloud services expects migrates to work in batches, where policy identifies a set of cold files to be moved to a cloud storage tier. Recalls, however, are expected to be accessed on demand and are given higher priority than migrates. It is wise to schedule migration policies where possible during off-peak hours to avoid recall performance impacts.

Note: Bulk policy-based recall operations are supported, but not expected to be the common method for recalling files. Since recalls are generally expected to be on demand, recalls are prioritized ahead of migrates.

Distributing files using policies

IBM Spectrum Scale policies can be used to distribute the migrations and recalls to all the nodes in the node class. To effectively use the policy, refer to the following points:

- The **mmapplypolicy** takes two parameters for the number of parallel threads processing the files (-m) and number of files processed per thread (-b).
  - –m has a default value of 24. If you have more CPU cores available, it is advisable to adjust the –m value accordingly
  - –b has a default value of 100. You can increase the number if you want to process more files in one iteration.
  - Plan the –m and –b carefully, so that all the nodes are given equal number of files. For example, if you have 5000 files to migrate and 4 nodes with Cloud services installed, the default values of –m 24 and –b 100 will try to distribute 2400 files to each node. There will not be enough files to be distributed to all nodes, and two nodes will be idle.
- -N parameter of the **mmapplypolicy** will accept the node class name, so the policy will distribute the files to all the nodes in the node class.

Using multiple file system or file sets for the same node group

A customer can configure multiple file systems and multiple file sets for a single node group, and run migrates in parallel for all of them. If they are looking for high performance, it is recommended that they configure the policy in a manner that, migrates run on a single file system or file set at a time. Cloud services design uses a single server process, and resources such as memory, networking and threading are shared between the containers. Running parallel migrates can split the resource, and hence the effective throughput of a single container can be impacted.

Maintenance activities

It is recommended to run the maintenance activities such as reconcile, deletes during a specific time window where migrates are not happening on the system. Reconcile on a large database can lock the database for a long time, hence creating a large backlog for the entries to be inserted to the DB. It is best to avoid this. The maintenance activities for each configured file system or file set can be run simultaneously on different nodes because the databases for these are not shared. Running multiple maintenance activities on the same node at the same time is not recommended due to memory resource constraints.

Container spillover

For achieving better performance and ease of maintenance, it is recommended to keep no more than 100 million entries per container. After that, the customer will must create a new container for the same file system/fileset and continue migrations. This will ensure that a new database is created for the new container, and the container cloud database size does not grow too large. Operation such as reconcile for a 100 million file database can take a few hours. Growing the size much beyond that results in maintenance operations taking too long since the duration of these operations relative to database size is non-linear.

Object storage configurations

IBM Cloud Object Storage

For on-premise configurations, it is recommended to have multiple accessors setup, so that a load balancer or Cloud services itself will have no single point of failure in its cloud services with COS. For public IBM COS service, there is a built-in load balancer – no additional CSAP or load balancer work is necessary. For public IBM COS service, there is a built-in load balancer, so no additional CSAP or load balancer work is necessary.

Swift

For on-premise configurations, it is recommended to have multiple accessors setup, so that a load balancer or Cloud services itself will have no single point of failure in its cloud services with Swift.

Amazon S3

Use the appropriate supported Amazon S3 region while configuring the cloud account. Load balancing or multiple CSAPs are not necessary as Amazon S3 has a built-in load balancer.

Security considerations

You can integrate Cloud services with IBM Security Key Lifecycle Manager (ISKLM) to provide security to the data that is stored on the cloud storage tier, or you can use the native key manager provided with the Cloud services.

Transparent cloud tiering: For information on integration of ISKLM with Transparent cloud tiering, see *Configuring Cloud services with SKLM* in *IBM Spectrum Scale: Administration Guide.*.

Note: Ensure that you back up your security keys. Data encrypted by using the Cloud services cannot be recovered, if security keys are lost.

Cloud data sharing: Cloud data sharing currently supports importing any cloud data, assuming it is not read in an encrypted format. Cloud data sharing supports encrypted data only if it was exported by Cloud data sharing and the encryption keys are shared between the importer and the exporter. For this reason, it is recommended that the native Cloud services and Cloud data sharing encryption are used to provide encryption at rest. A secure connection should be used to transfer data between IBM Spectrum Scale and the cloud. Thus, data is stored encrypted on IBM Spectrum Scale, and stored encrypted on the cloud, and is secured by the connection when transferring between the two systems.

Backup considerations for Transparent cloud tiering

You must adhere to some guidelines for backing up files that need to be migrated to a cloud storage tier.

In many cases, Transparent cloud tiering (cooler data) might be run along with file system backups (for hot and warm data) for data protection. Transparent cloud tiering should not be used as a replacement for a valid backup strategy. Files that are migrated with Transparent cloud tiering might be lost if accidentally deleted or if there is a complete site failure. File system backups can protect from these types of events.

When you work with Transparent cloud tiering file system backups, it is important that a file is backed up before it is sent to a cloud storage. A file that is not backed up is recalled from the cloud storage tier when a backup program is run. This can cause unnecessary recall traffic, and also cause the backup to take longer to run. Recalling data also involves cost if you are using a public cloud provider.

To ensure that files are not migrated before they are backed up, it is best to define a migration policy that explicitly excludes files that are recently modified. For example, if nightly backups are run, at minimum, a migration policy should exclude files that are modified within the past day, so they can be backed up before they are migrated. In most cases, it is best to wait longer to avoid recalls if a backup window is missed for some reason. For example, if you run a nightly backup, it is advisable to exclude files that are modified in the past 3 days. If there is an issue with running backups for a long period, it might be necessary to disable a migration policy until the problem with backups is resolved. This avoids many recalls during the next successful backup.

The following is a sample migration policy and it should be tuned as per the backup frequency that is configured for a IBM Spectrum Scale file system.

Note: The amount of time to wait to migrate a file should be greater than the frequency of file system backups.

For information on backing up configuration data, see *Backing up the Cloud services configuration* topic in *IBM Spectrum Scale: Administration Guide*.

For information on backing up the cloud database, see *Backing up the Cloud data sharing database to the cloud* topic in *IBM Spectrum Scale: Administration Guide*.

You can exclude recently modified files with a policy statement such as the following:

```
WHERE (DAYS(CURRENT_TIMESTAMP) - DAYS(MODIFICATION_TIME)) > 5
```

This statement can also be added for manual application of the policy for a directory.

```
/* Define a cloud pool, using the 'mmcloudgateway' command to perform migrations to the pool */
   RULE EXTERNAL POOL 'cloudpool' EXEC '/usr/lpp/mmfs/bin/mmcloudgateway files' OPTS '-F'

/* Define a migrate rule, in this example we will migrate from the system pool */
   RULE 'MigrateToCloud' MIGRATE FROM POOL 'system'
/* Define the threshold to use, in this example we will migrate when the pool is 85% full,
   and attempt to migrate files until the pool is 75% full */
   THRESHOLD (85, 75)
/* Next, define a weight for the file to determine which files to migrate first */
/* Here we will use the last access time for the file by subtracting it from the current time,
   giving a higher weight to older files */
   WEIGHT(CURRENT_TIMESTAMP - ACCESS_TIME)
/* Define that we want to move to the cloud pool */
   TO POOL ('cloudpool')
/* We don't want to migrate files that are smaller than 8k in size, because there are little to
   no space savings due to file overhead */
   WHERE (KB_ALLOCATED > 8)
/* Do not migrate Cloud services internal files, which reside in the .mcstore directory */
   AND NOT(PATH_NAME LIKE '%/.mcstore/%'
/* Ignore files that have been modified in the past 5 days. This needs to be customized based on
   the backup frequency. */
   AND (DAYS(CURRENT_TIMESTAMP) - DAYS(MODIFICATION_TIME)) > 5
```

Quota support for tiering

Transparent cloud tiering supports container-level quotas in ways that are described here.

IBM Spectrum Scale does not account for data that Transparent cloud tiering has migrated to the Cloud for quota support.

- Files in non-resident states do not count against quota as defined by IBM Spectrum Scale.

Object storage generally supports quotas and might be used to provide quota support.

- For example, IBM Cloud Object Storage supports hard quotas at the granularity of a container.

General guidelines for quota support:

- Use fileset support – this provides the level of granularity you need for cloud storage quotas to be useful.
- Set the desired quota for each fileset by using the object storage quota setting for the container.
- On container spillover, the size for each container in the fileset will need to be accounted for.
  - Since all but the most recent container will no longer be growing, you can set the quota for the most recent (and active) container based on the remaining capacity after subtracting for the current size of the older containers.
  - If older containers have a significant number of files deleted from them, the quota for the active container might need to be periodically adjusted upwards.

Note: For more information on setting quota at a container level, see documentation for the specific cloud account that you use in your environment.

Firewall recommendations

The IBM Spectrum Scale system administrator is supposed to follow certain recommendations to set up firewall to secure the IBM Spectrum Scale system from unauthorized access.

For more information firewall recommendations, see *Securing the IBM Spectrum Scale system using firewall* in *IBM Spectrum Scale: Administration Guide*.

Considerations for GPFS applications

Application design should take into consideration the exceptions to Open Group technical standards with regard to the **stat()** system call and NFS V4 ACLs. Also, a technique to determine whether a file system is controlled by GPFS has been provided.

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

- *Exceptions to Open Group technical standards*
- *Determining if a file system is controlled by GPFS*
- *GPFS exceptions and limitation to NFS V4 ACLs*
- *Considerations for GPFS applications*

Chapter 3. Steps for establishing and starting your IBM Spectrum Scale cluster

There are several steps you must perform to establish and start your IBM Spectrum Scale cluster. This topic provides the information you need for performing those steps.

The installation toolkit, available on a few Linux distributions, automates many of the steps listed below.

You can install IBM Spectrum Scale and deploy protocols either manually or by using the installation toolkit. This topic provides the information you need for establishing and starting your IBM Spectrum Scale cluster manually. If you have already installed IBM Spectrum Scale with the installation toolkit, these steps have already been completed.

Follow these steps to establish your IBM Spectrum Scale cluster:

1. Review supported hardware, software, and limits by reviewing the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) for the latest recommendations on establishing a IBM Spectrum Scale cluster.
2. Install the IBM Spectrum Scale licensed program on your system:
   - For existing systems, see Chapter 15, “Upgrading,” on page 375.
   - For new systems:
     - For your Linux nodes, see Chapter 4, “Installing IBM Spectrum Scale on Linux nodes and deploying protocols,” on page 223.
     - For your AIX nodes, see Chapter 5, “Installing IBM Spectrum Scale on AIX nodes,” on page 319.
     - For your Windows nodes, see Chapter 6, “Installing IBM Spectrum Scale on Windows nodes,” on page 323.
3. Decide which nodes in your system will be quorum nodes (see “Quorum” on page 136).
4. Create your GPFS cluster by issuing the **mmcrcluster** command. See “GPFS cluster creation considerations” on page 141.
5. Use the **mmchlicense** command to assign an appropriate GPFS license to each of the nodes in the cluster. See “IBM Spectrum Scale license designation” on page 129 for more information.

If you use the installation toolkit to install IBM Spectrum Scale, following steps 2 through 5, and optionally, step 6 are completed by the installation toolkit.

After your IBM Spectrum Scale cluster has been established:

1. Ensure you have configured and tuned your system according to the values suggested in the *Configuring and tuning your system for GPFS* topic in *IBM Spectrum Scale: Administration Guide*.
2. Start IBM Spectrum Scale by issuing the **mmstartup** command. For more information, see **mmstartup command** in *IBM Spectrum Scale: Command and Programming Reference*.
3. Create new disks for use in your file systems by issuing the **mmcrnsd** command. See “Network Shared Disk (NSD) creation considerations” on page 147.
4. Create new file systems by issuing the **mmcrfs** command. See “File system creation considerations” on page 151.
5. Mount your file systems by issuing the **mmmount** command.
6. As an optional step, you can also create a temporary directory (**/tmp/mmfs**) to collect problem determination data. The **/tmp/mmfs** directory can be a symbolic link to another location if more space can be found there. If you decide to do so, the temporary directory should *not* be placed in an IBM Spectrum Scale file system, as it might not be available if IBM Spectrum Scale fails.

If a problem should occur, IBM Spectrum Scale might write 200 MB or more of problem determination data into **/tmp/mmfs**. These files must be manually removed when any problem determination is complete. This should be done promptly so that a **NOSPACE** condition is not encountered if another failure occurs. An alternate path can be specified by issuing the **mmchconfig dataStructureDump** command.

Chapter 4. Installing IBM Spectrum Scale on Linux nodes and deploying protocols

Use this information for installing IBM Spectrum Scale on Linux nodes and deploying protocols.

Before installing IBM Spectrum Scale, you must review "Planning for GPFS" on page 133 and the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

Installing IBM Spectrum Scale without ensuring that the prerequisites listed in "Hardware requirements" on page 133, "Software requirements" on page 134, and "Installation prerequisites" on page 224 are satisfied can lead to undesired results.

There are three methods available for installing IBM Spectrum Scale on Linux nodes.

Installation method	**Description**
Manual	You can install the GPFS software packages using the **rpm** command (for SLES and Red Hat Enterprise Linux) or the **dpkg** command (for Ubuntu Linux). For more information, see "Manually installing the IBM Spectrum Scale software packages on Linux nodes" on page 233.
Installation toolkit	On Red Hat Enterprise Linux 7.x, Ubuntu 16.04 LTS, and SLES 12, you can install GPFS and deploy protocols using the installation toolkit. **Note:** Object protocol is not supported on SLES 12. For more information, see "Installing IBM Spectrum Scale on Linux nodes with the installation toolkit" on page 258. For information on limitations if you are using the installation toolkit, see "Limitations of the installation toolkit" on page 263.
Installation GUI	You can use the installation GUI to install IBM Spectrum Scale system. However, the scope of installation related activities that can be done using the installation GUI is limited. For more information, see "Installing IBM Spectrum Scale by using the graphical user interface (GUI)" on page 312.

Deciding whether to install IBM Spectrum Scale and deploy protocols manually or with the installation toolkit

On supported Linux distributions, you can install IBM Spectrum Scale and deploy protocols either manually or using the installation toolkit.

Why would I want to use the installation toolkit?

While planning your installation, consider the advantages provided by the installation toolkit. The installation toolkit:

1. Simplifies IBM Spectrum Scale cluster creation
2. Automatically creates NSD and file system stanza files
3. Creates new NSDs, file systems, and adds new nodes
4. Has a single package manager for all components of a release
5. Deploys Object, SMB, NFS by automatically pulling in all pre-requisites
6. Configures file and Object authentication during deployment
7. Consistently sets and syncs time on all nodes
8. Deploys the IBM Spectrum Scale GUI
9. Configures Performance Monitoring consistently across all nodes
10. Simplifies upgrade with a single command to upgrade all components on all nodes.

Before deciding which method you want to use, review the following topics:

- "Installation prerequisites"
- "IBM Spectrum Scale packaging overview" on page 230
- "Understanding the installation toolkit options" on page 261
- "Limitations of the installation toolkit" on page 263
- "Using the installation toolkit to perform installation tasks: Explanations and examples" on page 272
- "Manually installing the IBM Spectrum Scale software packages on Linux nodes" on page 233

Installation prerequisites

These are the several prerequisites for installing IBM Spectrum Scale including those for installing protocols and performance monitoring.

Required packages for supported Linux distributions

IBM Spectrum Scale Cluster Export Services (CES) requires the following packages to be installed:

- `ethtool`
- `nfs-utils` (on Ubuntu: `nfs-common`)
- `rpcbind`
- `psmisc`
- `iputils` (on Ubuntu: `iputils-arping`)

For the list of packages that must be installed on the system before you can install IBM Spectrum Scale, see "Software requirements" on page 134.

Prerequisites for the installation toolkit

For information on prerequisites for using the installation toolkit, see "Preparing to use the installation toolkit" on page 270.

Requirements for file authentication

Before configuring file authentication, ensure that prerequisites are met to avoid installation failure. For information on prerequisites for configuring file authentication, see *Configuring authentication and ID mapping for file access* in *IBM Spectrum Scale: Administration Guide*.

For Active Directory (AD) authentication considerations, see *Integrating with AD server* in *IBM Spectrum Scale: Administration Guide*.

For LDAP considerations, see the *Integrating with LDAP server* topic in *IBM Spectrum Scale: Administration Guide*.

Required packages for IBM Spectrum Scale for object storage

IBM Spectrum Scale for object storage requires the following SELinux packages to be installed:

- selinux-policy-base at 3.13.1-23 or higher
- selinux-policy-targeted at 3.12.1-153 or higher

Attention: If SELinux is disabled during installation of IBM Spectrum Scale for object storage, enabling SELinux after installation is not supported.

Requirements for IBM Spectrum Scale object storage on Ubuntu 16.04 LTS

- The postgresql version must be 9.5 or later.
- A postgresql cluster named Keystone is created during the configuration process. Do not migrate the Keystone cluster to a newer postgresql version.
- Object storage is not integrated with AppArmor.

During installation, the following conflicting packages are removed if they were previously installed:

- keystone
- python-keystone
- swift
- python-swift
- swift-object
- swift-container
- swift-object-expirer
- swift-proxy
- python-keystonemiddleware
- python-keystoneauth1
- python-keystoneclient
- python-openstackclient
- python-swiftclient

Cleanup required if you previously used the Object Redpaper configuration process

If you have already configured the system with Openstack software to use GPFS (following instructions from the Object Red paper) be sure to follow the Object cleanup steps in "Cleanup procedures required if reinstalling with the installation toolkit" on page 366 before you start any installation activity.

Requirements for the management GUI

- PostgreSQL server of the Linux distribution must be installed on the node on which the GUI is to be installed.
- IBM Spectrum Scale Java™ Runtime Environment (JRE) must be installed (gpfs.java RPM) on the node on which the GUI is to be installed.
- IBM Spectrum Scale performance collector (gpfs.pmcollector RPM) must be installed on the node on which the GUI is to be installed.

Required packages for file audit logging

The following packages must be installed on all nodes before you can enable file audit logging:

- gpfs.java
- openssl

For more information, see "Enabling and configuring file audit logging using the installation toolkit" on page 287.

GPFS 4.2.0 or later required on protocol and quorum nodes

If you want to run protocols in a mixed-version cluster, the protocol nodes and all of the quorum nodes must run 4.2.0 or later; other nodes can run an older version.

Note: For the Object protocol, the GPFS software is not required on the node that hosts the Keystone identity service if it is deployed on a separate node from the GPFS protocol nodes.

Additional GPFS requirements

- If you want to run protocols, Cluster Configuration Repository (CCR) must be available.
- **mmchconfig release=LATEST** must be run.
- A GPFS cluster and a file system are required. If this infrastructure does not already exist, you must install the GPFS software, create a GPFS cluster and create a file system. You can use the installation toolkit to do these tasks. For more information, see "Using the installation toolkit to perform installation tasks: Explanations and examples" on page 272.
- The GPFS file system must be mounted on all GPFS protocol nodes.
- It is strongly recommended to configure the file system to only support NFSv4 ACLs. You can use the installation toolkit to do this task also if you use the installation toolkit to install GPFS. For more information, see "Using the installation toolkit to perform installation tasks: Explanations and examples" on page 272.

  Alternatively, you can use the **-k nfs4** parameter for **mmcrfs**. NFSv4 ACLs are a requirement for ACL usage with the SMB and NFS protocols. For more information, see **mmcrfs** examples in the *IBM Spectrum Scale: Command and Programming Reference*.
- Quotas are not enabled by default in GPFS file systems but are recommended for use with SMB and NFS protocols. For more information about quota management, see *Enabling and disabling GPFS quota management* in *IBM Spectrum Scale: Administration Guide*.

Creation of a file system or fileset or path for a CES shared root, and creation of an object fileset

The installation toolkit uses a shared root storage area to install the protocols on each node. This storage is also used by NFS and object protocols to maintain system data associated with the cluster integration we provide. This storage can be a subdirectory in an existing file system or it can be a file system on its own. Once this option is set, changing it will requires a restart of GPFS.

You can use the installation toolkit to set up this CES shared root storage area if you use the toolkit for GPFS installation and file system creation. For more information, see "Using the installation toolkit to perform installation tasks: Explanations and examples" on page 272.

However, if you want to set up shared root before launching the installation toolkit, the following steps can be used:

1. Create a file system or fileset for shared root. Size must be at least 4 GB.
2. Use the following command:

   mmchconfig cesSharedRoot=*path_to_the_filesystem/fileset_created_in_step_1*

For Object, the installation toolkit creates an independent fileset in the GPFS file system that you name.

SSH and network setup

This is a general GPFS prerequisite, not specific to protocols other than protocol-specific ports that need to be open on the firewall.

The node used for initiating installation of SMB, NFS, and/or Object protocols using the installation toolkit must be able to communicate through an internal or external network with all protocol nodes to be installed. All nodes also require SSH keys to be set up so that the installation toolkit can run remote commands without any prompts. Examples of prompts include a prompt for a remote node's password or a prompt for a yes-or-no question. No prompts should occur when using SSH among any cluster nodes to and from each other, and to and from the node designated for installation.

For information on ports required by IBM Spectrum Scale, see *Securing the IBM Spectrum Scale system using firewall* and *GPFS port usage* in *IBM Spectrum Scale: Administration Guide*.

For examples of how to open firewall ports on different operating systems, see *Examples of how to open firewall ports* in *IBM Spectrum Scale: Administration Guide*.

Repository setup

This is a general GPFS prerequisite, not specific to protocols.

The installation toolkit contains all necessary code for installation; however, there may be base operating system packages required as prerequisites. In order to satisfy any prerequisites, it might be necessary for your nodes to have access to a DVD repository or an external repository accessible by network. Repositories containing IBM Spectrum Scale dependencies include the following x86_64 example:

```
rhel-x86_64-server-7 Red Hat Enterprise Linux Server
```

For more information about Yum repositories, see Configuring Yum and Yum Repositories (https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/System_Administrators_Guide/sec-Configuring_Yum_and_Yum_Repositories.html).

For information on setting up repositories on SLES nodes, see Managing Software Repositories and Services(https://www.suse.com/documentation/sles-12/book_sle_deployment/data/sec_y2_sw_instsource.html).

For information on setting up repositories on Ubuntu nodes, see Repositories/Ubuntu(https://help.ubuntu.com/community/Repositories/Ubuntu).

Important: You must disable all configured EPEL repositories on all nodes added to the installation toolkit before proceeding with installation, deployment, or upgrade.

NTP setup

It is recommended that Network Time Protocol (NTP) be configured on all nodes in your system to ensure that the clocks of all of the nodes are synchronized. Clocks that are not synchronized will cause debugging issues and authentication problems with the protocols.

If you are using the installation toolkit for IBM Spectrum Scale installation, then no prior NTP configuration is required other than that the NTP package is installed and the NTP daemon (ntpd) is not running on all nodes.
Before setting up NTP on a node running on SLES 12 SP1 or earlier, you must run the **logprof** command, save the updated local profiles, and then reboot the node. This step is not required for nodes running on SLES 12 SP2.
For example:

```
# logprof
Reading log entries from /var/log/messages.
Updating AppArmor profiles in /etc/apparmor.d.
Enforce-mode changes:

Profile:  /usr/sbin/avahi-daemon
Path:     /run/nscd/group
Mode:     r
Severity: unknown

[1 - /run/nscd/group]

(A)llow / [(D)eny] / (G)lob / Glob w/(E)xt / (N)ew / Abo(r)t / (F)inish / (O)pts
Adding /run/nscd/group r to profile.

Profile:  /usr/sbin/avahi-daemon
Path:     /run/nscd/passwd
Mode:     r
Severity: unknown
```

```
[1 - /run/nscd/passwd]

(A)llow / [(D)eny] / (G)lob / Glob w/(E)xt / (N)ew / Abo(r)t / (F)inish / (O)pts
Adding /run/nscd/passwd r to profile.

= Changed Local Profiles =

The following local profiles were changed.  Would you like to save them?
[1 - /usr/sbin/avahi-daemon]

(S)ave Changes / [(V)iew Changes] / Abo(r)t
Writing updated profile for /usr/sbin/avahi-daemon.
```

Reboot the node for the changes to take effect.

Use the **config ntp** commands listed in Understanding the spectrumscale installation toolkit options, Table 1 to set NTP servers and then NTP will automatically be configured at install time. For example:

```
/usr/lpp/mmfs/5.0.0.0/installer/spectrumscale config ntp -e on -s 198.51.100.2,198.51.100.4
[ WARN ] The NTP package must already be installed and full bidirectional access to the UDP
port 123 must be allowed.
[ WARN ] If NTP is already running on any of your nodes, NTP setup will be skipped. To stop
NTP run 'service ntpd stop'.
[ INFO ] Setting NTP to on
[ INFO ] Setting Upstream NTP Servers to 198.51.100.2,198.51.100.4
```

However, if you are manually installing IBM Spectrum Scale or would prefer to manually configure NTP, see the following example of enabling NTP on a node.

On Red Hat Enterprise Linux nodes

```
# yum install -y ntp
# ntpdate <NTP_server_IP>
# systemctl enable ntpd
# systemctl start ntpd
# timedatectl list-timezones
# timedatectl set-timezone
# systemctl enable ntpd
```

On SLES 12 nodes

```
# zypper install -y ntp
# ntpdate <NTP_server_IP>
# systemctl enable ntpd
# systemctl start ntpd
# timedatectl list-timezones
# timedatectl set-timezone
# systemctl enable ntpd
```

On SLES 12 nodes, if the **systemctl start ntp** command hangs, use the resolution documented in Novell Knowledgebase Document 7015867.

On Ubuntu 16.04 LTS nodes

```
# apt-get install -y ntp
# ntpdate <NTP_server_IP>
# systemctl enable ntpd
# systemctl start ntpd
# timedatectl list-timezones
# timedatectl set-timezone
# systemctl enable ntpd
# systemctl status ntpd
```

For information about NTP configuration on Ubuntu nodes, see Ubuntu documentation about NTP.

Required packages for performance monitoring

This is a general GPFS prerequisite, not specific to protocols.

Ensure that the following packages are installed on the system before the installation of performance monitoring:

- boost-regex on Red Hat Enterprise Linux
- libboost_regex on SLES
- libboost-regex-dev on Ubuntu Linux

Required packages for mmprotocoltrace

Some of the advanced tracing components of the **mmprotocoltrace** command require specific packages to be installed on all nodes that need to participate in tracing related operations.

- To enable network tracing with **mmprotocoltrace**, you need to install the wireshark package.
- To enable the syscalls-tracing for SMB, you need to install the strace package.

Collection of core dump data

For information about changing configuration to enable collection of core dump data from protocol nodes, see *Configuration changes required on protocol nodes to collect core dump data* in *IBM Spectrum Scale: Problem Determination Guide*.

Preparing the environment on Linux nodes

Before proceeding with installation, prepare your environment by following the suggestions in the following sections.

Add the GPFS bin directory to your shell PATH

Ensure that the PATH environment variable for the root user on each node includes **/usr/lpp/mmfs/bin**. (This is not required for the operation of GPFS, but it can simplify administration.)

Other suggestions for cluster administration

GPFS commands operate on all nodes required to perform tasks. When you are administering a cluster, it may be useful to have a more general form of running commands on all of the nodes. One suggested way to do this is to use an OS utility like **dsh** or **pdsh** that can execute commands on all nodes in the cluster. For example, you can use **dsh** to check the kernel version of each node in your cluster:

```
# dsh uname -opr
Node01: 2.6.18-128.1.14.e15 x86_64 GNU/Linux
Node02: 2.6.18-128.1.14.e15 x86_64 GNU/Linux
```

Once you have **dsh** set up, you can use it to install GPFS on a large cluster. For details about setting up **dsh** or a similar utility, review the documentation for the utility.

Verify that prerequisite software is installed

Before installing GPFS, it is necessary to verify that you have the correct levels of the prerequisite software installed on each node in the cluster. If the correct level of prerequisite software is *not* installed, see the appropriate installation manual before proceeding with your GPFS installation.

For the most up-to-date list of prerequisite software, see the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

The FAQ contains the latest information about the following:

- Supported Linux distributions and kernel levels
- Recommended or required RPM levels
- Software recommendations
- Configuration information

Before proceeding, see also “GPFS and network communication” on page 17.

IBM Spectrum Scale packaging overview

The IBM Spectrum Scale self-extracting package consists of several components.

Note: The installation toolkit and NFS, SMB, and object related packages are available only in the protocols installation packages.

Components	**Description**
Installation toolkit and license	The components necessary to begin the protocol installation.
GPFS	GPFS version 5.0.x, included in this self-extracting package. It also includes call home and file audit logging packages.
NFS (Ganesha)	The packages for Ganesha, the open-source, user-space implementation of the NFS protocol. They can be installed, enabled, and configured using the installation toolkit.
SMB (Samba)	The packages for Samba, the open-source implementation of the SMB protocol. They can be installed and enabled using the installation toolkit.
Object (Swift and Keystone)	The Swift and Keystone packages necessary for enablement of Object services. They can be installed, enabled, and configured using the installation toolkit.
Performance monitoring tool	The packages necessary for performance monitoring by GPFS (including protocols). They can be installed and enabled using the installation toolkit.

Preparing to install the GPFS software on Linux nodes

Follow the steps in this topic in the specified order to install the GPFS software.

This procedure installs GPFS on one node at a time:

1. “Accepting the electronic license agreement on Linux nodes”
2. “Extracting the IBM Spectrum Scale software on Linux nodes” on page 231
3. “Extracting IBM Spectrum Scale patches (update SLES or Red Hat Enterprise Linux RPMs or Ubuntu Linux packages)” on page 232
4. “Installing the IBM Spectrum Scale man pages on Linux nodes” on page 233
5. “Deciding whether to install IBM Spectrum Scale and deploy protocols manually or with the installation toolkit” on page 223

   In this step, you can choose to install GPFS and protocols either manually or using the **spectrumscale** installation toolkit.
6. “Verifying the IBM Spectrum Scale installation on SLES and Red Hat Enterprise Linux nodes” on page 245

   or

   “Verifying the IBM Spectrum Scale installation on Ubuntu Linux nodes” on page 245

Accepting the electronic license agreement on Linux nodes

The GPFS software license agreement is shipped with the GPFS software and is viewable electronically. When you extract the GPFS software, you are asked whether or not you accept the license.

The electronic license agreement must be accepted before software installation can continue. Read the software agreement carefully before accepting the license. See "Extracting the IBM Spectrum Scale software on Linux nodes."

Extracting the IBM Spectrum Scale software on Linux nodes

The IBM Spectrum Scale software is delivered in a self-extracting archive.

The self-extracting image contains the following.

- The IBM Spectrum Scale product installation SLES and Red Hat Enterprise Linux RPMs and Ubuntu Linux packages
- The License Acceptance Process (LAP) Tool

  The LAP Tool is invoked for acceptance of the IBM Spectrum Scale license agreements. The license agreements must be accepted to obtain access to the IBM Spectrum Scale product installation images.
- A version of the Java Runtime Environment (JRE) necessary to run the LAP Tool

1. Copy the self-extracting product image from the DVD to a local directory, specifying the correct version of the product for your hardware platform and Linux distribution. For example:
   ```
   cp /media/dvd/Spectrum_Scale_Standard-5.0.0.0-x86_64-Linux-install/
   /tmp/Linux/Spectrum_Scale_Standard-5.0.0-0-x86_64-Linux-install
   ```
2. Verify that the self-extracting program has executable permissions, for example:
   ```
   # ls -l /tmp/Spectrum_Scale_Standard-5.0.0-0-x86_64-Linux-install
   ```

   The system displays information similar to the following:
   ```
   -rwxr-xr-x l root root 673341648 Nov 10 13:53 /tmp/Spectrum_Scale_Standard-5.0.0-0-x86_64-Linux-install
   ```
   If it is not executable, you can make it executable by issuing the **chmod +x** command.

   Note: To get a list of all packages in the self-extracting installation image, you can issue the following command (specifying the **--manifest** option):
   ```
   Spectrum_Scale_Standard-5.0.0-0-x86_64-Linux-install --manifest
   ```

   The system displays output similar to the following:
   ```
   manifest
   sw,rpm,gpfs.base-5.0.0-0.x86_64.rpm,
   Thu 16 Feb 2017 10:40:41 AM MST,md5sum:99e940728f73649520533c06ae14b941
   sw,rpm,gpfs.adv-5.0.0-0.x86_64.rpm,
   Thu 16 Feb 2017 10:46:43 AM MST,md5sum:8c7a635ee2381b075e7960813bef3c47
   sw,rpm,gpfs.callhome-ecc-client-5.0.0-0.000.noarch.rpm,
   Thu 16 Feb 2017 12:22:35 PM MST,md5sum:9939576c9212650916e65092dd0e2c2e
   sw,rpm,gpfs.gskit-8.0.50-79.x86_64.rpm,
   Thu 16 Feb 2017 10:45:15 AM MST,md5sum:2e844aa5351dd87c234f5273ef9f3673
   sw,rpm,gpfs.gss.pmsensors-5.0.0-0.el7.x86_64.rpm,
   Wed 15 Feb 2017 09:56:29 AM MST,md5sum:3f4c084e93f3bd0d7cdf4adb8fe65448
   sw,rpm,gpfs.gss.pmcollector-5.0.0-0.el7.x86_64.rpm,
   Wed 15 Feb 2017 09:56:30 AM MST,md5sum:5fd3d21c935fea27d3b8c9b97b8c49e2
   sw,rpm,gpfs.gss.pmsensors-5.0.0-0.sles12.x86_64.rpm,
   Wed 15 Feb 2017 09:56:34 AM MST,md5sum:abab4976a83814877a140bab5e96c12f
   sw,rpm,gpfs.gss.pmcollector-5.0.0-0.sles12.x86_64.rpm,
   Wed 15 Feb 2017 09:56:31 AM MST,md5sum:48ae6aedf41c6681c98bc167dc1babdf
   sw,rpm,gpfs.gpl-5.0.0-0.noarch.rpm,
   Thu 16 Feb 2017 10:52:11 AM MST,md5sum:5989f71ace5dd4e7ca92f030de451819
   sw,rpm,gpfs.gui-5.0.0-0_170213182617.noarch.rpm,
   Mon 13 Feb 2017 10:34:37 AM MST,md5sum:e562910893e468cf9790865093623129
   sw,rpm,gpfs.msg.en_US-5.0.0-0.noarch.rpm,
   Thu 16 Feb 2017 10:44:37 AM MST,md5sum:6fe50b1bf9265d6dab322e0a472266ec
   sw,rpm,gpfs.callhome-5.0.0-0.000.sles12.noarch.rpm,
   Thu 16 Feb 2017 12:21:00 PM MST,md5sum:0308af742b0e2e3469cb8016c83c972a
   sw,rpm,gpfs.ext-5.0.0-0.x86_64.rpm,
   Thu 16 Feb 2017 10:44:48 AM MST,md5sum:d95f6b2470d95747afba7d3ee48cd570
   sw,rpm,gpfs.java-5.0.0-0.x86_64.rpm,
   ```

```
Tue 22 Nov 2016 08:38:17 AM MST,md5sum:03b8d22c87a145c168dddeef16f44ca2
sw,rpm,gpfs.callhome-5.0.0-0.000.el7.noarch.rpm,
Thu 16 Feb 2017 12:20:44 PM MST,md5sum:de55de4bcf87f7ec2406010ae9b88882
```

(The remaining packages in the installation image are displayed.)

3. Invoke the self-extracting image that you copied from the DVD by using a command such as `Spectrum-Scale-Standard-5.0.0.0-x86_64-Linux-install` and accept the license agreement:
   a. By default, the LAP Tool, JRE, and GPFS installation images are extracted to the target directory `/usr/lpp/mmfs/5.0.0.0`
   b. The license agreement files on the media can be viewed in text-only mode or graphics mode:
      - Text-only is the default mode. When run the output explains how to accept the agreement:
        ```
        <...Last few lines of output...>
        Press Enter to continue viewing the license agreement, or
        enter "1" to accept the agreement, "2" to decline it, "3"
        to print it, "4" to read non-IBM terms, or "99" to go back
        to the previous screen.
        ```
      - To view the files in graphics mode, invoke `Spectrum-Scale-Standard-5.0.0.0-x86_64-Linux-install`

        Using the graphics-mode installation requires a window manager to be configured.
   c. You can use the **--silent** option to accept the license agreement automatically.
   d. Use the **--help** option to obtain usage information from the self-extracting archive.

Upon license agreement acceptance, the IBM Spectrum Scale product installation images are placed in the extraction target directory (`/usr/lpp/mmfs/5.0.0.0`). This directory contains the GPFS packages according to the target Linux distribution that are applicable for the IBM Spectrum Scale edition that you installed (Standard, Advanced, or Data Management). For more information, see “Location of extracted packages” on page 235.

Note: For this release, the IBM Global Security Kit (GSKit) version for RPMs and Ubuntu Linux packages must be at least 8.0.50-79 or higher.

In this directory there is a **license** subdirectory that contains license agreements in multiple languages. To view which languages are provided, issue the following command:

```
ls /usr/lpp/mmfs/5.0.0.0/license
```

The system displays information similar to the following:

```
Chinese_TW.txt French.txt  Japanese.txt       notices.txt   Slovenian.txt
Chinese.txt    German.txt  Korean.txt         Polish.txt    Spanish.txt
Czech.txt      Greek.txt   Lithuanian.txt     Portuguese.txt Status.dat
English.txt    Indonesian.txt   Italian.txt non_ibm_license.txt Russian.txt   Turkish.txt
```

The license agreement remains available in the extraction target directory under the **license** subdirectory for future access. The license files are written using operating system-specific code pages. This enables you to view the license in English and in the local language configured on your machine. The other languages are not guaranteed to be viewable.

Extracting IBM Spectrum Scale patches (update SLES or Red Hat Enterprise Linux RPMs or Ubuntu Linux packages)

Typically when you install an IBM Spectrum Scale system there are patches available. It is recommended that you always look for the latest patches when installing or updating an IBM Spectrum Scale node.

Note: For information about restrictions pertaining to Ubuntu Linux, see the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

IBM Spectrum Scale patches (update SLES and Red Hat Enterprise Linux RPMs and Ubuntu Linux packages) are available from the IBM Support Portal: Downloads for General Parallel File System(www.ibm.com/support/entry/portal/Downloads/Software/Cluster_software/General_Parallel_File_System).

IBM Spectrum Scale patches (update SLES and Red Hat Enterprise Linux RPMs and Ubuntu Linux packages) are available from Fix Central http://www-933.ibm.com/support/fixcentral/i

The updated SLES and Red Hat Enterprise Linux RPMs and Ubuntu Linux packages are distributed as self-extracting base software.

If you are applying a patch during the initial installation of IBM Spectrum Scale on a node, you only need to build the portability layer once after the base and update SLES or Red Hat Enterprise Linux RPMs or Ubuntu Linux packages are installed.

Installing the IBM Spectrum Scale man pages on Linux nodes

In order to use the IBM Spectrum Scale man pages, the **gpfs.docs** package (rpm or deb package) must be installed.

Once you have installed the **gpfs.docs** package, the IBM Spectrum Scale man pages are located at **/usr/share/man/**.

You do not need to install the **gpfs.docs** package on all nodes if man pages are not desired (for example, if local file system space on the node is minimal).

If you are using the installation toolkit on supported Linux distributions, it automatically installs the associated man pages when it runs `spectrumscale install` to create a new GPFS cluster.

Manually installing the IBM Spectrum Scale software packages on Linux nodes

Use this information to manually install the IBM Spectrum Scale software on Linux nodes.

The IBM Spectrum Scale software packages are installed using the **rpm** command (for SLES and Red Hat Enterprise Linux) or the **dpkg** command (for Ubuntu Linux).

Required packages

The following packages are required for SLES and Red Hat Enterprise Linux:

- `gpfs.base-5.0.*.rpm`
- `gpfs.gpl-5.0.*.noarch.rpm`
- `gpfs.msg.en_US-5.0.*.noarch.rpm`
- `gpfs.gskit-8.0.50.*.rpm`
- `gpfs.license*.rpm`
- `gpfs.ext-5.0.*.rpm`
- | `gpfs.compression-5.0.*.rpm`
- `gpfs.crypto-5.0.*.rpm` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only)
- `gpfs.adv-5.0.*.rpm` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only)

The following packages are required for Ubuntu Linux:

- `gpfs.base_5.0.*.deb`
- `gpfs.gpl_5.0.*all.deb`

- `gpfs.msg.en-us_5.0.*all.deb`
- `gpfs.gskit_8.0.50.*.deb`
- `gpfs.license*.deb`
- `gpfs.ext_5.0.*.deb`
- `gpfs.compression_5.0.*.deb`
- `gpfs.crypto_5.0.*.deb` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only)
- `gpfs.adv_5.0.*.deb` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only)

Note: For this release, the IBM Global Security Kit (GSKit) version for RPMs and Ubuntu Linux packages must be at least 8.0.50-79 or later.

Optional packages for SLES 12 and Red Hat Enterprise Linux

Note: You can also use the installation toolkit to install and configure packages on supported Linux distributions. For more information, see "Overview of the installation toolkit" on page 258. The `gpfs.docs-5.0.*noarch.rpm` package is optional for SLES and Red Hat Enterprise Linux.

- `gpfs.java-5.0.*.rpm`
- `gpfs.callhome-ecc-client-5.0.*.rpm`
- `gpfs.callhome-5.0.*.rpm`
- `gpfs.gui-5.0.*.rpm`
- `gpfs.kafka-5.0.*.x86_64.rpm` (Red Hat Enterprise Linux x86_64 only)
- `gpfs.librdkafka-5.0.*.x86_64.rpm` (Red Hat Enterprise Linux x86_64 only)

For information about manually installing call home and management GUI packages, see Chapter 12, "Installing call home," on page 359 and "Manually installing IBM Spectrum Scale management GUI" on page 252.

The following performance monitoring packages are also optional for SLES and Red Hat Enterprise Linux.

- `gpfs.gss.pmcollector-5.0.*.rpm`
- `gpfs.gss.pmsensors-5.0.*.rpm`

The following protocols packages are available for Red Hat Enterprise Linux 7.x (x86_64-linux and PPC64-linux platforms):

- `nfs-ganesha-2.5.*.el7.*.rpm`
- `nfs-ganesha-gpfs-2.5.*.el7.*.rpm`
- `nfs-ganesha-utils-2.5.*.el7.*.rpm`
- `gpfs.smb-4.6.*_gpfs_*.el7.*.rpm`
- `spectrum-scale-object-5.0.*.noarch.rpm`
- `gpfs.scst*.el7.*.rpm`
- `gpfs.pm-ganesha-5.0.*.el7.*.rpm`
- `pmswift_5.0.*.noarch.rpm`
- `gpfs.protocols-support-5.0.*.rpm`

The following protocols packages are available for SLES 12:

- `nfs-ganesha-2.5.*.sles12.x86_64.rpm`
- `nfs-ganesha-gpfs-2.5.*.sles12.x86_64.rpm`
- `nfs-ganesha-utils-2.5.*.sles12.x86_64.rpm`
- `gpfs.smb-4.6.*_gpfs_*.sles12.x86_64.rpm`

- gpfs.pm-ganesha-5.0.*.sles12.x86_64.rpm
- gpfs.protocols-support-5.0.*.rpm

For information on manually installing packages including protocols packages on Red Hat Enterprise Linux 7.x, SLES 12, and Ubuntu 16.04 LTS nodes, see "Manually installing IBM Spectrum Scale and deploying protocols on Linux nodes" on page 239.

For information about manually installing performance monitoring packages, see "Manually installing the Performance Monitoring tool" on page 250.

Optional packages for Ubuntu Linux

The gpfs.docs_5.0.*all.deb package is optional for Ubuntu Linux.

The following performance monitoring packages are also optional for Ubuntu Linux (x86_64-linux only).

- gpfs.gss.pmcollector_5.0.*.deb
- gpfs.gss.pmsensors_5.0.*.deb

The following protocols packages are available for Ubuntu 16.04 LTS:

- nfs-ganesha_2.5.*amd64.deb
- nfs-ganesha_gpfs-2.5.*amd64.deb
- nfs-ganesha_utils-2.5.*amd64.deb
- gpfs.smb_4.6.*gpfs*amd64.deb
- spectrum-scale-object_5.0.*.deb
- gpfs.pm-ganesha-5.0.*amd64.deb
- pmswift_5.0.*.deb
- gpfs.protocols-support_5.0.*.deb

The following packages are required (and provided) only on the Elastic Storage Server (ESS):

- gpfs.gnr.base-1.0.0-0.ppc64.rpm
- gpfs.gnr-5.0.*.ppc64.rpm
- gpfs.gss.firmware-5.0.*.ppc64.rpm

For more information about Elastic Storage Server (ESS), see Elastic Storage Server (ESS) documentation on IBM Knowledge Center.

The gpfs.gnr-3.5.*.ppc64.rpm package is required only if IBM Spectrum Scale RAID on Power 775 is used. For more information about IBM Spectrum Scale RAID, see *IBM Spectrum Scale RAID: Administration* in Elastic Storage Server (ESS) documentation on IBM Knowledge Center.

Location of extracted packages

The installation images are extracted to following component specific directories.

Base GPFS packages

Red Hat Enterprise Linux or SLES: gpfs_rpms

Ubuntu: gpfs_debs

NFS packages

Red Hat Enterprise Linux or SLES: ganesha_rpms

Ubuntu: ganesha_debs

SMB packages

Red Hat Enterprise Linux or SLES: smb_rpms

Ubuntu: smb_debs

Object packages

Red Hat Enterprise Linux: object_rpms

Ubuntu: object_debs

Performance monitoring packages

Red Hat Enterprise Linux or SLES: zimon_rpms

Ubuntu: zimon_debs

Depending on the target Linux distribution, the RPMs are extracted in following paths where *_rpms or *_debs is the component specific directory.

- Red Hat Enterprise Linux 7.x: /usr/lpp/mmfs/5.0.x.x/*_rpms/rhel7
- Ubuntu 16.04 LTS: /usr/lpp/mmfs/5.0.x.x/*_debs/ubuntu16
- SLES 12: /usr/lpp/mmfs/5.0.x.x/*_rpms/sles12

Note: Object protocol is not supported on SLES 12.

Installation of packages for SLES or Red Hat Enterprise Linux

To install all of the required GPFS SLES or Red Hat Enterprise Linux RPMs for the IBM Spectrum Scale Standard Edition, change the directory to where the installation image is extracted. For example, issue this command:

```
cd /usr/lpp/mmfs/5.0.0.0/gpfs_rpms
```

Then, issue this command:

```
rpm -ivh gpfs.base*.rpm gpfs.gpl*rpm gpfs.license.std*.rpm gpfs.gskit*rpm gpfs.msg*rpm
gpfs.ext*rpm gpfs.compression*rpm
```

To install all of the required GPFS SLES or Red Hat Enterprise Linux RPMs for the IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition, change to the directory where the installation image is extracted. For example, enter the following command:

```
cd /usr/lpp/mmfs/5.0.0.0/gpfs_rpms
```

Then, issue the following command for Advanced Edition:

```
rpm -ivh gpfs.base*.rpm gpfs.gpl*rpm gpfs.license.adv*.rpm gpfs.gskit*rpm
gpfs.msg*rpm gpfs.ext*rpm gpfs.compression*rpm gpfs.adv*rpm gpfs.crypto*rpm
```

or issue the following command for Data Management Edition:

```
rpm -ivh gpfs.base*.rpm gpfs.gpl*rpm gpfs.license.dm*.rpm gpfs.gskit*rpm
gpfs.msg*rpm gpfs.ext*rpm gpfs.compression*rpm gpfs.adv*rpm gpfs.crypto*rpm
```

To install the optional gpfs.doc package for GPFS SLES or Red Hat Enterprise Linux issue the following command:

```
cd /usr/lpp/mmfs/5.0.0.0/gpfs_rpms
rpm -ivh gpfs.docs*.rpm
```

Installation of packages for Ubuntu Linux

To install all of the GPFS Ubuntu Linux packages for the IBM Spectrum Scale Standard edition, change the directory to where the installation image is extracted. For example, issue this command:

```
cd /usr/lpp/mmfs/5.0.0.0/gpfs_debs
```

then, issue this command:

```
dpkg -i gpfs.base*deb gpfs.gpl*deb gpfs.license.std*.deb gpfs.gskit*deb gpfs.msg*deb
gpfs.ext*deb gpfs.compression*deb
```

To install all of the GPFS Ubuntu Linux packages for IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition, change the directory to where the installation image is extracted. For example, issue this command:

```
cd /usr/lpp/mmfs/5.0.0.0/gpfs_debs
```

then, issue this command for Advanced Edition:

```
dpkg -i gpfs.base*deb gpfs.gpl*deb gpfs.license.adv*.deb gpfs.gskit*deb
gpfs.msg*deb gpfs.ext*deb gpfs.compression*deb gpfs.adv*deb gpfs.crypto*deb
```

or, issue this command for Data Management Edition:

```
dpkg -i gpfs.base*deb gpfs.gpl*deb gpfs.license.dm*.deb gpfs.gskit*deb
gpfs.msg*deb gpfs.ext*deb gpfs.compression*deb gpfs.adv*deb gpfs.crypto*deb
```

To install optional gpfs.doc package for GPFS Ubuntu Linux package, change the directory to where the installation image is extracted and install the package. For example, issue this command:

```
dpkg -i /usr/lpp/mmfs/5.0.0.0/gpfs_debs/gpfs.doc*deb
```

To install the GPFS GUI, see “Manually installing IBM Spectrum Scale management GUI” on page 252.

To optionally install the call home feature, see Chapter 12, “Installing call home,” on page 359.

See also the following topics:

“Manually installing the Performance Monitoring tool” on page 250

“Object protocol further configuration” on page 314

Building the GPFS portability layer on Linux nodes

Before starting GPFS, you must build and install the GPFS portability layer.

The GPFS portability layer is a loadable kernel module that allows the GPFS daemon to interact with the operating system.

Note: The GPFS kernel module should be updated every time the Linux kernel is updated. Updating the GPFS kernel module after a Linux kernel update requires rebuilding and installing a new version of the module.

To build the GPFS portability layer on Linux nodes, do the following:

1. Check for the following before building the portability layer:
   - Updates to the portability layer at the IBM Support Portal: Downloads for General Parallel File System(www.ibm.com/support/entry/portal/Downloads/Software/Cluster_software/General_Parallel_File_System).
   - The latest kernel levels supported in the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).
2. Build your GPFS portability layer in one of the following ways:
   - Using the **mmbuildgpl** command (recommended) for nodes running GPFS 4.1.0.4 or later. For more information, see *mmbuildgpl command* in *IBM Spectrum Scale: Command and Programming Reference*.
   - Using the **Autoconfig** tool.
   - Using the directions in **/usr/lpp/mmfs/src/README**.

Using the mmbuildgpl command to build the GPFS portability layer on Linux nodes

Starting with GPFS 4.1.0.4, you can use the **mmbuildgpl** command to simplify the build process.

To build the GPFS portability layer using **mmbuildgpl**, enter the following command:

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

Each kernel module is specific to a Linux version and platform. If you have multiple nodes running exactly the same operating system level on the same platform, and only some of these nodes have a compiler available, you can build the kernel module on one node, then create an installable package that contains the binary module for ease of distribution.

If you choose to generate an installable package for portability layer binaries, perform the following additional step:

```
/usr/lpp/mmfs/bin/mmbuildgpl --build-package
```

When the command finishes, it displays the location of the generated package similar to:

```
Wrote: /root/rpmbuild/RPMS/x86_64/gpfs.gplbin-3.10.0-229.el7.x86_64-5.0.0-0.x86_64.rpm
```

or

```
Wrote: /tmp/deb/gpfs.gplbin_5.0.0-0_amd64.deb
```

You can then copy the generated package to other machines for deployment. The generated package can *only* be deployed to machines with identical architecture, distribution level, Linux kernel, and IBM Spectrum Scale maintenance level. If the `gpfs.gplbin` package is installed, installing the `gpfs.gpl` package is not required.

Note: During the package generation, temporary files are written to the **/tmp/rpm** or **/tmp/deb** directory, so be sure there is sufficient space available. By default, the generated package goes to **/usr/src/packages/RPMS/<arch>** for SUSE Linux Enterprise Server, **/usr/src/redhat/RPMS/<arch>** for Red Hat Enterprise Linux, and **/tmp/deb** for Ubuntu Linux.

Using the Autoconfig tool to build the GPFS portability layer on Linux nodes

To simplify the build process, GPFS provides an automatic configuration tool called **Autoconfig**.

The following example shows the commands required to build the GPFS portability layer using the **Autoconfig** tool:

```
cd /usr/lpp/mmfs/src
make Autoconfig
make World
make InstallImages
```

Each kernel module is specific to a Linux version and platform. If you have multiple nodes running exactly the same operating system level on the same platform, or if you have a compiler available on only one node, you can build the kernel module on one node, then create an installable package that contains the binary module for ease of distribution.

If you choose to generate an installable package for portability layer binaries, perform the following additional step:

`make rpm` for SLES and RHEL Linux

`make deb` for Ubuntu Linux

When the command finishes, it displays the location of the generated package similar to:

```
Wrote: /root/rpmbuild/RPMS/x86_64/gpfs.gplbin-3.10.0-229.el7.x86_64-5.0.0-0.x86_64.rpm
```

or

```
Wrote: /tmp/deb/gpfs.gplbin_5.0.0-0_amd64.deb
```

You can then copy the generated package to other machines for deployment. The generated package can *only* be deployed to machines with identical architecture, distribution level, Linux kernel, and IBM Spectrum Scale maintenance level. If the `gpfs.gplbin` package is installed, installing the `gpfs.gpl` package is not required.

Note: During the package generation, temporary files are written to the **/tmp/rpm** or **/tmp/deb** directory, so be sure there is sufficient space available. By default, the generated package goes to **/usr/src/packages/RPMS/<arch>** for SUSE Linux Enterprise Server, **/usr/src/redhat/RPMS/<arch>** for Red Hat Enterprise Linux, and **/tmp/deb** for Ubuntu Linux.

Manually installing IBM Spectrum Scale and deploying protocols on Linux nodes

Use this information to manually install IBM Spectrum Scale and deploying protocols on systems running on Red Hat Enterprise Linux 7.x, SLES 12, and Ubuntu 16.04 LTS.

These prerequisites must be met before installing IBM Spectrum Scale on Linux systems.

- Repository must be set up.
  - For information on setting up repository on Red Hat Enterprise Linux 7.x, see Red Hat Enterprise Linux 7 *System Administrator's Guide*.
  - For information on setting up repository on SLES 12, see SLES 12 *Deployment Guide*.
  - For information on setting up repository on Ubuntu 16.04 LTS, see Ubuntu 16.04 *Server Guide*.
- All prerequisite packages must be installed. For a list of prerequisite packages, see "Software requirements" on page 134 and "Installation prerequisites" on page 224.
- Passwordless SSH must be set up between all nodes in the cluster. For more information, see *Problems due to missing prerequisites* in *IBM Spectrum Scale: Problem Determination Guide*.
- Firewall configuration must be according to your requirements. It is recommended that firewalls are in place to secure all nodes. For more information, see *Securing the IBM Spectrum Scale system using firewall* in *IBM Spectrum Scale: Administration Guide*.

  To check the status of the firewall on Red Hat Enterprise Linux 7.x, issue the following command:

  ```
  systemctl status firewalld
  ```

  To check the status of the firewall on SLES 12, issue the following command:

  ```
  sudo /sbin/rcSuSEfirewall2 status
  ```

  To check the status of the firewall on Ubuntu 16.04 LTS, issue the following command:

  ```
  sudo ufw status
  ```

  For examples of how to open firewall ports on different operating systems, see *Examples of how to open firewall ports* in *IBM Spectrum Scale: Administration Guide*.
- Every node must have a non-loopback IP address assigned. In some scenarios, a freshly installed node might have its host name pointing to 127.0.0.1 in `/etc/hosts`. 127.0.0.1 is a loopback IP address and it is not sufficient for multi-node IBM Spectrum Scale cluster creation. In these cases, each node needs a static IP with connectivity to the other nodes.
- Optionally, the bash profile can be updated to allow easier access to IBM Spectrum Scale commands.
  - Verify that the PATH environment variable for the root user on each node includes `/usr/lpp/mmfs/bin` and the required tool directories. This allows a user to execute IBM Spectrum Scale commands without having to first change directory to `/usr/lpp/mmfs/bin`.

– Export the *WCOLL* variable used by **mmdsh**. In the following example, /nodes is a manually created file, listing line by line, each node within the cluster using the nodes' FQDN. Once IBM Spectrum Scale is installed, this configuration allows the **mmdsh** command to execute commands on multiple nodes simultaneously.

Example:

```
# cat ~/.bash_profile
# .bash_profile
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
  . ~/.bashrc
fi

######
# User specific environment and startup programs
######
######
# Specifics for GPFS testing
######
export PATH=$PATH:$HOME/bin:/usr/lpp/mmfs/bin
export WCOLL=/nodes
```

Log out and then log in again for the changes in the bash profile to take effect.

Installing IBM Spectrum Scale packages on Linux systems

Use this information to manually install IBM Spectrum Scale packages on systems running on Red Hat Enterprise Linux 7.x, SLES 12, and Ubuntu 16.04 LTS.

1. Download and extract IBM Spectrum Scale packages, and then accept the licensing agreement. For more information, see "Extracting the IBM Spectrum Scale software on Linux nodes" on page 231 and "Accepting the electronic license agreement on Linux nodes" on page 230.

   For information on the location of extracted packages, see "Location of extracted packages" on page 235.

2. Install the IBM Spectrum Scale RPMs, including the Advanced Edition packages, by issuing one of the following commands depending on the operating system.

   - On Red Hat Enterprise Linux 7.x and SLES 12, issue the following command.

     ```
     # rpm -ivh gpfs.base*.rpm gpfs.gpl*rpm gpfs.license*rpm gpfs.gskit*rpm
     gpfs.msg*rpm gpfs.ext*rpm gpfs.compression*rpm gpfs.adv*rpm gpfs.crypto*rpm
     ```

   - On Ubuntu 16.04 LTS, issue the following command.

     ```
     # dpkg -i gpfs.base*deb gpfs.gpl*deb gpfs.license.dm*.deb gpfs.gskit*deb
     gpfs.msg*deb gpfs.ext*deb gpfs.compression*deb gpfs.adv*deb gpfs.crypto*deb
     ```

   A sample output is as follows.

   ```
   Preparing... ########################################### [100%]
   1:gpfs.base ########################################### [ 11%]
   2:gpfs.ext ########################################### [ 22%]
   3:gpfs.compression ########################################### [ 27%]
   4:gpfs.adv ########################################### [ 33%]
   5:gpfs.crypto ########################################### [ 44%]
   6:gpfs.gpl ########################################### [ 56%]
   7:gpfs.license.adv ########################################### [ 67%]
   8:gpfs.msg.en_US ########################################### [ 78%]
   9:gpfs.gskit ########################################### [ 89%]
   10:gpfs.base-debuginfo ########################################### [100%]
   ```

3. Build the portability layer using the following command.

   ```
   # /usr/lpp/mmfs/bin/mmbuildgpl
   ```

   A sample output is as follows. The output will vary depending on the operating system.

   ```
   ---------------------------------------------------------
   mmbuildgpl: Building GPL module begins at Thu Mar 24 15:18:23 CST 2016.
   ---------------------------------------------------------
   Verifying Kernel Header...
     kernel version = 31228004 (3.12.28-4-default, 3.12.28-4)
   ```

```
  module include dir = /lib/modules/3.12.28-4-default/build/include
  module build dir   = /lib/modules/3.12.28-4-default/build
  kernel source dir  = /usr/src/linux-3.12.28-4/include
  Found valid kernel header file under /lib/modules/3.12.28-4-default/build/include
Verifying Compiler...
  make is present at /usr/bin/make
  cpp is present at /usr/bin/cpp
  gcc is present at /usr/bin/gcc
  g++ is present at /usr/bin/g++
  ld is present at /usr/bin/ld
Verifying Additional System Headers...
  Verifying linux-glibc-devel is installed ...
    Command: /bin/rpm -q linux-glibc-devel
    The required package linux-glibc-devel is installed
make World ...
make InstallImages ...
--------------------------------------------------------
mmbuildgpl: Building GPL module completed successfully at Thu Mar 24 15:18:31 CST 2016.
--------------------------------------------------------
```

You might need to install prerequisite packages, if the portability layer cannot be built due to missing prerequisite packages. For a list of prerequisite packages, see “Software requirements” on page 134.

If the repository setup and the kernel configuration are correct, you can install the prerequisite packages by issuing one of the following commands depending on the operating system:

Red Hat Enterprise Linux 7.x:

```
# /usr/bin/yum install -y package_name1 package_name2 ... package_nameN
```

SLES 12:

```
# /usr/bin/zypper install -y package_name1 package_name2 ... package_nameN
```

Ubuntu 16.04 LTS:

```
# /usr/bin/apt-get install -y package_name1 package_name2 ... package_nameN
```

Once IBM Spectrum Scale is built on all nodes, you can create the cluster. It is recommended to have an odd number of quorum nodes and that your NSD nodes be designated as quorum nodes

4. Create the cluster using the following command.

   ```
   # /usr/lpp/mmfs/bin/mmcrcluster -N nodeslist --ccr-enable -r /usr/bin/ssh -R /usr/bin/scp -C cluster1.spectrum
   ```

   In this command example, `nodeslist` is a file that contains a list of nodes and node designations to be added to the cluster and its contents are as follows:

   ```
   node1:quorum
   node2
   node3
   node4:quorum-manager
   node5:quorum-manager
   ```

   Running the **mmcrcluster** command generates output similar to the following:

   ```
   mmcrcluster: Performing preliminary node verification ...
   mmcrcluster: Processing quorum and other critical nodes ...
   mmcrcluster: Processing the rest of the nodes ...
   mmcrcluster: Finalizing the cluster data structures ...
   mmcrcluster: Command successfully completed
   mmcrcluster: Propagating the cluster configuration data to all
   affected nodes.  This is an asynchronous process.
   # Thu Mar 24 15:33:06 CST 2016: mmcommon pushSdr_async: mmsdrfs propagation started
   Thu Mar 24 15:33:10 CST 2016: mmcommon pushSdr_async: mmsdrfs propagation completed; mmdsh rc=0
   ```

5. Accept proper role licenses for the nodes using the following commands.

   a. Accept the server licenses for the applicable nodes.

      ```
      # /usr/lpp/mmfs/bin/mmchlicense server --accept -N node1,node4,node5
      ```

      The system displays output similar to the following:

```
The following nodes will be designated as possessing server licenses:
    node1
    node4
    node5
mmchlicense: Command successfully completed
mmchlicense: Warning: Not all nodes have proper GPFS license designations.
    Use the mmchlicense command to designate licenses as needed.
mmchlicense: Propagating the cluster configuration data to all
affected nodes.  This is an asynchronous process.
# Thu Mar 24 15:37:59 CST 2016: mmcommon pushSdr_async: mmsdrfs propagation started
Thu Mar 24 15:38:01 CST 2016: mmcommon pushSdr_async: mmsdrfs propagation completed; mmdsh rc=0
```

b. Accept the client licenses for the applicable nodes.

```
# /usr/lpp/mmfs/bin/mmchlicense client --accept -N node2,node3
```

The system displays output similar to the following:

```
The following nodes will be designated as possessing client licenses:
    node2
    node3
mmchlicense: Command successfully completed
mmchlicense: Propagating the cluster configuration data to all
affected nodes.  This is an asynchronous process.
# Thu Mar 24 15:38:26 CST 2016: mmcommon pushSdr_async: mmsdrfs propagation started
Thu Mar 24 15:38:28 CST 2016: mmcommon pushSdr_async: mmsdrfs propagation completed; mmdsh rc=0
```

IBM Spectrum Scale cluster is now created. You can view the configuration information of the cluster using the following command.

```
# /usr/lpp/mmfs/bin/mmlscluster
```

The system displays output similar to the following:

```
GPFS cluster information
========================
  GPFS cluster name:         cluster1.spectrum
  GPFS cluster id:           9933771118354342448
  GPFS UID domain:           cluster1.spectrum
  Remote shell command:      /usr/bin/ssh
  Remote file copy command:  /usr/bin/scp
  Repository type:           CCR
GPFS cluster configuration servers:
-----------------------------------
  Primary server:    node1 (not in use)
  Secondary server:  (none)
 Node  Daemon node name  IP address    Admin node name  Designation
-------------------------------------------------------------------
   1   node1            203.0.113.11  node1           quorum
   2   node4            203.0.113.14  node4           quorum-manager
   3   node5            203.0.113.15  node5           quorum-manager
   4   node2            203.0.113.12  node2
   5   node3            203.0.113.13  node3
```

6. Start the GPFS daemons and the cluster using the following command.

   ```
   # /usr/lpp/mmfs/bin/mmstartup -N NodesList
   ```

   NodeList is the list of nodes on which the daemons and the cluster are to be started.

   You can use the **mmgetstate -a** *NodeList* command to verify that the GPFS software is running on these nodes.

Creating NSDs and file systems as part of installing IBM Spectrum Scale on Linux systems

Use this information to create NSDS and file systems for installing IBM Spectrum Scale on Red Hat Enterprise Linux 7.x, SLES 12, and Ubuntu 16.04 LTS systems.

For information about NSD creation considerations, see "Network Shared Disk (NSD) creation considerations" on page 147.

1. Create NSDs as follows.

a. Identify available physical disks in the /dev directory.

b. Create the NSD stanza file to be used with the **mmcrnsd** command.

   For example, consider you have 2 disks, /dev/sdc and /dev/sdd, and your node name is exnode1 and it is running Linux. In this case, your NSD stanza file contents are similar to the following:

```
%nsd:
   device=/dev/sdc
   nsd=nsd1
   servers=exnode1
   usage=dataAndMetadata
   failureGroup=-1
   pool=system
%nsd:
   device=/dev/sdd
   nsd=nsd2
   servers=exnode1
   usage=dataAndMetadata
   failureGroup=-1
```

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

c. Create the NSDs using the following command.

```
# mmcrsnd –F NSD_Stanza_Filename
```

2. Create a GPFS file system using the following command.

```
# mmcrfs fs1 -F NSD_Stanza_Filename -k nfs4
```

   Attention: Ensure that you do not use the same NSD stanza files that was used to create NSDs. Using the same NSD stanza file is not desirable because it results in all NSDs getting associated with one file system. If this file system gets corrupted, you can no longer access it and because all your NSDs are associated with this file system, your cluster becomes unusable.

Installing Cluster Export Services as part of installing IBM Spectrum Scale on Linux systems

Use this information to install Cluster Export Services (CES) on Red Hat Enterprise Linux 7.x, SLES 12, and Ubuntu 16.04 LTS systems.

Important: If you plan to install and configure CES, all protocols available on your platform must be installed. Installing a subset of available protocols is not supported.

1. Unmount GPFS file systems and stop GPFS on all nodes using the following command.

```
# mmshutdown -a
```

2. Configure the CES shared root file system on one of the available file systems using the following command.

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

3. Start GPFS on all nodes in the cluster using the following command.

```
# mmstartup -a
```

4. Enable CES on the required nodes using the following command.

```
# mmchnode ––ces-enable –N prnode1,prnode2,prnode3
```

5. Add the IP addresses of the protocol nodes to CES using the following commands.

```
# mmces address add --ces-ip 198.51.100.2
# mmces address add --node prnode1 --ces-ip 198.51.100.2
```

6. Verify the CES configuration using the following commands.

```
# mmlscluster --ces
# mmces address list
```

At this point, there are no services enabled. You can verify that using the **mmces services list --all** command. The system displays output similar to the following.

```
No CES services are enabled.
```

7. Download and extract IBM Spectrum Scale protocol packages, and then accept the licensing agreement. For more information, see “Extracting the IBM Spectrum Scale software on Linux nodes” on page 231 and “Accepting the electronic license agreement on Linux nodes” on page 230.

   For information on the location of extracted packages, see “Location of extracted packages” on page 235.
8. Install IBM Spectrum Scale NFS packages by issuing one of the following commands depending on the operating system.
   - On Red Hat Enterprise Linux 7.x and SLES 12, issue the following command.
     ```
     # rpm -ivh NFS_Package_Name1 NFS_Package_Name2 ... NFS_Package_NameN
     ```
   - On Ubuntu 16.04 LTS, issue the following command.
     ```
     # dpkg -i NFS_Package_Name1 NFS_Package_Name2 ... NFS_Package_NameN
     ```

   For a list of packages for the current IBM Spectrum Scale release, see “Manually installing the IBM Spectrum Scale software packages on Linux nodes” on page 233.
9. Remove conflicting Samba packages that might have been installed on each CES node by issuing one of the following commands depending on the operating system.
   - On Red Hat Enterprise Linux 7.x and SLES 12, issue the following command.
     ```
     # mmdsh -N all rpm -e samba-common --nodeps
     # mmdsh -N all rpm -e samba-client
     ```
   - On Ubuntu 16.04 LTS, issue the following command.
     ```
     # mmdsh -N all dpkg -P samba-common
     # mmdsh -N all dpkg -P samba-client
     ```
10. Install IBM Spectrum Scale SMB package by issuing one of the following commands depending on the operating system.
    - On Red Hat Enterprise Linux 7.x and SLES 12, issue the following command.
      ```
      # rpm -ivh gpfs.smb*.rpm
      ```
    - On Ubuntu 16.04 LTS, issue the following command.
      ```
      # dpkg -i gpfs.smb*.deb
      ```

    For a list of packages for the current IBM Spectrum Scale release, see “Manually installing the IBM Spectrum Scale software packages on Linux nodes” on page 233.
11. Install IBM Spectrum Scale for object storage package using one of these procedures depending on the operating system.
    - “Manually installing IBM Spectrum Scale for object storage on Red Hat Enterprise Linux 7.x nodes” on page 247.
    - “Manually installing IBM Spectrum Scale for object storage on Ubuntu 16.04 LTS nodes” on page 248.
12. If you want to use the CES BLOCK service that is available only on Red Hat Enterprise Linux 7.x, install the package using the following command.
    ```
    # rpm -ivh BLOCK_Service_Package_Name
    ```

    For a list of packages for the current IBM Spectrum Scale release, see “Manually installing the IBM Spectrum Scale software packages on Linux nodes” on page 233.

    For an overview of the CES BLOCK service, see “CES iSCSI support” on page 39.

    For information on configuring the CES BLOCK service, see *Configuring and enabling the BLOCK service* in *IBM Spectrum Scale: Administration Guide*.

Enabling NFS, SMB, and object on Linux systems

Use this information to enable NFS, SMB, and object on Linux systems after installing the packages.

Before you begin enabling NFS, SMB, and object, you must have installed the corresponding IBM Spectrum Scale packages.

- Enable NFS as follows.
  1. Mask and stop the kernel NFS service on all nodes where CES NFS needs to be installed using the following commands.
     ```
     # systemctl mask nfs-server.service
     # systemctl daemon-reload
     # systemctl stop nfs
     ```
  2. Enable the NFS services using the following command.
     ```
     # mmces service enable NFS
     ```
     If you get the `/sbin/rpc.statd: not found [No such file or directory]` error, try the following workaround.

     Run the following commands on every protocol node.
     ```
     # ln -s /usr/sbin/rpc.statd /sbin/rpc.statd
     # ln -s /sbin/rpcinfo /usr/sbin/rpcinfo
     ```
     After you have run these commands, try enabling the NFS services again.
- Enable the SMB services using the following command.
  ```
  # mmces service enable SMB
  ```
- Enable object on Red Hat Enterprise Linux 7.x and Ubuntu 16.04 LTS nodes. For more information, see "Manually installing IBM Spectrum Scale for object storage on Red Hat Enterprise Linux 7.x nodes" on page 247 and "Manually installing IBM Spectrum Scale for object storage on Ubuntu 16.04 LTS nodes" on page 248.

Verifying the IBM Spectrum Scale installation on Ubuntu Linux nodes

You can verify the installation of the IBM Spectrum Scale Ubuntu Linux packages on each node.

To check that the software has been successfully installed, use the **dpkg** command:

```
dpkg -l | grep gpfs
```

The system returns output similar to the following:

```
ii gpfs.base 5.0.0-0        GPFS File Manager
ii gpfs.docs 5.0.0-0        GPFS Server Man Pages and Documentation
ii gpfs.gpl 5.0.0-0         GPFS Open Source Modules
ii gpfs.gskit 8.0.50-79     GPFS GSKit Cryptography Runtime
ii gpfs.msg.en-us 5.0.0-0   GPFS Server Messages - U.S. English
ii gpfs.ext 5.0.0-0         GPFS Extended Features
ii gpfs.compression 5.0.0-0  IBM Spectrum Scale Compression Libraries
ii gpfs.license.std-5.0.0-0  IBM Spectrum Scale Standard Edition ITLM files
```

If you have IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition installed, you should also see the following line in the output:

```
ii gpfs.crypto 5.0.0-0      GPFS Cryptographic Subsystem
ii gpfs.adv 5.0.0-0         GPFS Advanced Features
ii gpfs.license.dm-5.0.0-0   IBM Spectrum Scale Advanced Edition ITLM files
or
ii gpfs.license.adv-5.0.0-0  IBM Spectrum Scale Data Management Edition ITLM files
```

Verifying the IBM Spectrum Scale installation on SLES and Red Hat Enterprise Linux nodes

You can verify the installation of the IBM Spectrum Scale SLES or Red Hat Enterprise Linux RPMs on each node.

To check that the software has been successfully installed, use the **rpm** command:

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

The system returns output similar to the following:

```
gpfs.docs-5.0.0-0
gpfs.base-5.0.0-0
gpfs.msg.en_US-5.0.0-0
gpfs.ext-5.0.0-0
gpfs.compression-5.0.0-0
gpfs.gpl-5.0.0-0
gpfs.gskit-8.0.50-79
gpfs.license.std-5.0.0-0
```

If you have IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition installed, you should also see the following in the output:

```
gpfs.crypto-5.0.0-0
gpfs.adv-5.0.0-0
gpfs.license.dm-5.0.0-0 or gpfs.license.adv-5.0.0-0
```

For installations that include IBM Spectrum Scale RAID, you should also see the following in the output:

```
gpfs.gnr-5.0.0-0
```

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

To run the system health check for verifying IBM Spectrum Scale installation, issue the following command:

```
mmhealth node show -N all
```

For Linux on Z: Changing the kernel settings

In order for GPFS to run on Linux on Z, the kernel settings need to be changed.

Before starting GPFS, perform the following steps on each Linux on Z node.

On SLES 12:

1. In the /etc/default/grub file, add the following:

   ```
   GRUB_CMDLINE_LINUX_DEFAULT=" hvc_iucv=8 TERM=dumb osamedium=eth instmode=ftp x
   crashkernel=206M-:103M cio_ignore=all,!ipldev,!condev vmalloc=4096G "
   ```

2. Run the following command:

   ```
   grub2-mkconfig -o /boot/grub2/grub.cfg
   ```

3. Reboot the node.

On SLES11 and Red Hat Enterprise Linux:

1. In the /etc/zipl.conf file, add vmalloc=4096G user_mode=home as shown in the following example:

   ```
   (10:25:41) dvtc1a:~ # cat /etc/zipl.conf
   # Modified by YaST2. Last modification on Mon May 19 09:39:04 EDT 2014
   [defaultboot]
   defaultmenu = menu

   ###Don't change this comment - YaST2 identifier: Original name: linux###
   [SLES11_SP3_2]
       image = /boot/image-3.0.101-0.15-default
       target = /boot/zipl
       ramdisk = /boot/initrd-3.0.101-0.15-default,0x2000000
       parameters = "root=/dev/mapper/mpatha_part2 hvc_iucv=8 TERM=dumb
    resume=/dev/mapper/mpatha_part1 crashkernel=258M-:129M vmalloc=4096G
   user_mode=home"
   ```

   Note: For SUSE Linux Enterprise Server (SLES) 11 and Red Hat Enterprise Linux 6, user_mode=home is optional. For Red Hat Enterprise Linux 7.1 and later releases, this parameter is not required.

2. Run the **zipl** command.

   Note: For information about the **zipl** command, see the topic about the initial program loader for IBM Z (-zipl) in Device Drivers, Features, and Commands (www.ibm.com/support/knowledgecenter/api/content/linuxonibm/liaaf/lnz_r_dd.html) in the Linux on Z library overview.
3. Reboot the node.

Note: For more detailed information about installation and startup of GPFS on IBM Z, see the "Getting started with Elastic Storage for Linux on Z based on GPFS technology" white paper, available on the Welcome Page for IBM Spectrum Scale in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/ibmspectrumscale_welcome.html).

Manually installing IBM Spectrum Scale for object storage on Red Hat Enterprise Linux 7.x nodes

IBM Spectrum Scale for object storage is typically installed using the installation toolkit. If you do not want to use the installation toolkit, use the following steps to manually install IBM Spectrum Scale for object storage.

For information about prerequisites, see "Software requirements" on page 134 and "Installation prerequisites" on page 224.

Before you begin manually installing IBM Spectrum Scale for object storage, complete the following prerequisite tasks.

1. Create protocol nodes for object service. For more information, see *Configuring CES protocol service IP addresses* in *IBM Spectrum Scale: Administration Guide*.
2. Add at least one CES IP address.

Manually install the object protocol and then enable object services as follows.

1. On all protocol nodes, install the `spectrum-scale-object` package and its associated dependencies by issuing the following command.

   ```
   yum -y install spectrum-scale-object
   ```

   The package is created in the `/usr/lpp/mmfs/5.0.0.0/object_rpms/rhel7` directory by expanding the `Spectrum_Scale_Protocols` installation image. For more information about extracting an installation image, see "Extracting the IBM Spectrum Scale software on Linux nodes" on page 231.

   Note: The path `/usr/lpp/mmfs/5.0.0.0/object_rpms/rhel7`, depends upon the release version.

   You might need to create the **yum** repository before using this command. To create the **yum** repository, create an entry similar to the following in the `/etc/yum.repos.d` directory:

   ```
   [spectrum_scale]
                   name=spectrum_scale
                   baseurl=file:///usr/lpp/mmfs/5.0.0.0/object_rpms/rhel7
                   enabled=1
                   gpgcheck=0
   ```

   Note: The path `file:///usr/lpp/mmfs/5.0.0.0/object_rpms/rhel7`, in the above example depends upon the release version.
2. From one of the protocol nodes, install the object protocol by using the **mmobj swift base** command.

   ```
   # mmobj swift base -g /gpfs/fs1 -o object_fileset  \
   --cluster-hostname protocol-cluster.example.net  --local-keystone --admin-password passw0rd
   ```

   A sample output is as follows.

```
mmobj swift base: Validating execution environment.
mmobj swift base: Performing SELinux configuration.
mmobj swift base: Creating fileset /dev/fs1 object_fileset.
mmobj swift base: Configuring Keystone server in /gpfs/fs1/object/keystone.
mmobj swift base: Creating postgres database.
mmobj swift base: Validating Keystone environment.
mmobj swift base: Validating Swift values in Keystone.
mmobj swift base: Configuring Swift services.
mmobj swift base: Uploading configuration changes to the CCR.
mmobj swift base: Configuration complete.
```

After the initial install is complete, the object protocol needs to be enabled across all the protocol nodes.

3. Complete the configuration and start object services on all protocol nodes by using the **mmces service enable** command.

   ```
   # mmces service enable OBJ
   ```

4. List the protocols enabled in the IBM Spectrum Scale cluster by using the **mmces service list** command. List a verbose output of object services running on the local node using the -v flag.

   ```
   # mmces service list -v
   ```

   A sample output is as follows.

   ```
   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
   ```

Manually installing IBM Spectrum Scale for object storage on Ubuntu 16.04 LTS nodes

IBM Spectrum Scale for object storage is typically installed using the installation toolkit. If you do not want to use the installation toolkit, use the following steps to manually install IBM Spectrum Scale for object storage on Ubuntu 16.04 LTS nodes.

For information about prerequisites, see “Software requirements” on page 134 and “Installation prerequisites” on page 224.

Before you begin manually installing IBM Spectrum Scale for object storage, complete the following prerequisite tasks.

1. After GPFS is configured, create protocol nodes for object service. For more information, see *Configuring CES protocol service IP addresses* in *IBM Spectrum Scale: Administration Guide*.
2. Add at least one CES IP address.

Manually install and configure the object protocol and then enable object services as follows.

1. On all protocol nodes, install the spectrum-scale-object package and its associated dependencies by issuing the following command.

   ```
   apt-get -o Dpkg::Options::="--force-overwrite" --allow-unauthenticated install -y spectrum-scale-object
   ```

The package is created in the /usr/lpp/mmfs/5.0.0.0/object_debs/ubuntu16 directory by expanding the Spectrum_Scale_Protocols installation image. For more information about extracting an installation image, see “Extracting the IBM Spectrum Scale software on Linux nodes” on page 231.

You might need to create the local APT repository before using the **apt-get** command. To create the APT repository, use these steps:

a. Create a file named Packages.gz.

```
# cd /usr/lpp/mmfs/5.0.0.0/object_debs/ubuntu16
# dpkg-scanpackages . /dev/null | gzip -9c > Packages.gz
```

b. Add the following entry in the /etc/apt/sources.list file.

```
deb file:/usr/lpp/mmfs/5.0.0.0/object_debs/ubuntu16 ./
```

c. Update the repository.

```
# apt-get update
```

Note: The path /usr/lpp/mmfs/5.0.0.0/object_debs/ubuntu16, depends upon the release version.

2. From one of the protocol nodes, configure the object protocol by issuing the following command.

```
# mmobj swift base -g /ibm/fs1 -o object_fileset --cluster-hostname protocol-cluster.example.net \
--local-keystone --admin-password Passw0rd --admin-user keystone
```

A sample output is as follows.

```
mmobj swift base: Validating execution environment.
  mmobj swift base: Creating fileset gpfs0 object_fileset.
  mmobj swift base: Configuring Keystone server in /ibm/fs1/object/keystone.
  mmobj swift base: Creating postgres database.
  mmobj swift base: Setting cluster attribute object_database_node=198.51.100.13.
  mmobj swift base: Validating Keystone environment.
  mmobj swift base: Validating Swift values in Keystone.
  mmobj swift base: Configuring Swift services.
  mmobj swift base: Setting cluster attribute object_singleton_node=198.51.100.13.
  mmobj swift base: Uploading configuration changes to the CCR.
  mmobj swift base: Configuration complete.
```

After the initial installation is complete, the object protocol needs to be enabled across all the protocol nodes.

3. Complete the configuration and start object services on all protocol nodes by issuing the following command.

```
# mmces service enable OBJ
```

4. Issue the **mmces service list** command to list the protocols enabled in the IBM Spectrum Scale cluster. 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
```

A sample output is as follows.

```
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
```

Manually installing the Performance Monitoring tool

The Performance Monitoring tool is automatically installed by the installation toolkit. You can also install the Performance Monitoring tool manually for the releases that are not supported by the installation toolkit.

For more information, see "Understanding the installation toolkit options" on page 261. For more information about the Performance Monitoring tool, see *Configuring the Performance Monitoring tool* in *IBM Spectrum Scale: Administration Guide*.

During an upgrade to the tool or in case of any issues, you can manually install the tool by performing the following steps:

1. Download the installation images and install the Performance Monitoring packages, which are extracted to following directories depending on the target Linux distribution.
   - RHEL 7.x: `/usr/lpp/mmfs/5.0.0.0/zimon_rpms/rhel7`
   - SLES 12: `/usr/lpp/mmfs/5.0.0.0/zimon_rpms/sles12`
   - Ubuntu 16: `/usr/lpp/mmfs/5.0.0.0/zimon_rpms/ubuntu16`
   - Other supported distributions: `/usr/lpp/mmfs/5.0.0.0/zimon_rpms`

   Performance Monitoring packages

   ```
   gpfs.gss.pmsensors_version-release.os.target_arch.file_format
   gpfs.gss.pmcollector_version-release.os.target_arch.file_format
   gpfs.pm-ganesha_version-release.os.target_arch.file_format
   ```

   Then use your operating system's native package management mechanism.

   For example:
   - To install 5.0.*-0 sensors on a Red Hat 7 x86_64 node use the following command:
     ```
     rpm -ivh gpfs.gss.pmsensors-5.0.*-0.el7.x86_64.rpm
     ```
   - To install 5.0.*-0 sensors on a SLES 12 x86_64 node use the following command:
     ```
     rpm -ivh gpfs.gss.pmsensors-5.0.*.sles12.x86_64.rpm
     ```
   - To install 5.0.*-0 sensors on a Ubuntu amd64 node, use the following command:
     ```
     dpkg -i gpfs.gss.pmsensors-5.0.*-0.U*amd64.deb
     ```
   - To install a 5.0.*-0 collector on a Red Hat 7 PPC64LE node use the following command:
     ```
     rpm -ivh gpfs.gss.pmcollector-5.0.*-0.el7.ppc64le.rpm
     ```
   - To install a 5.0.*-0 Ganesha sensors use the following command:
     ```
     rpm -ivh gpfs.pm-ganesha-5.0.*.el7.x86_64.rpm
     ```
2. A single collector can easily support up to 400 sensor nodes. The collector can be any node on the system. All sensors will report to this node. Select any node in the system to be the collector node. For information on configuring the sensors and collectors, see the *Configuring the performance monitoring tool* section in the *IBM Spectrum Scale: Problem Determination Guide*.
   - To enable performance monitoring for Object, install the `pmswift` RPM:
     ```
     rpm -ivh pmswift-<version>-<release>.noarch.rpm
     ```
     where `<version>` is equal to or greater than 4.2 and `<release>` is equal to or greater than 0.

     The installation of the `pmswift` RPM also copies SWIFT related sensors configuration files, namely, `SwiftAccount.cfg`, `SwiftContainer.cfg`, `SwiftObject.cfg` and `SwiftProxy.cfg` to the Performance Monitoring tool's installation directory, `/opt/IBM/zimon/`. The pmswift rpm converts the operational metrics for Object into a form that is usable by the Performance Monitoring tool.

     After installation of the `pmswift` RPM, the following steps must be carried out:
     a. Edit the Object configuration files for all Object servers that reside in cluster configuration repository (CCR), using the following command:
        ```
        /usr/local/pmswift/bin/pmswift-config-swift set
        ```

CCR will then propagate modified configuration files to /etc/swift/ directory on all the protocol nodes within the cluster. The modified configuration files are:

- account - *.conf
- container - *.conf
- object - *.conf
- proxy - *.conf

b. Use the /usr/local/pmswift/bin/pmswift-config-zimon set command to edit the sensors configuration information stored in the CCR. This adds the SWIFT related following sensors entries:

```
{
        # SwiftAccount operational metrics
        name = "SwiftAccount"
        period = 1
        type = "generic"
},
{
        # SwiftContainer operational metrics
        name = "SwiftContainer"
        period = 1
        type = "generic"
},
{
        # SwiftObject operational metrics
        name = "SwiftObject"
        period = 1
        type = "generic"
},
{
        # SwiftProxy operational metrics
        name = "SwiftProxy"
        period = 1
        type = "generic"
},
```

These entries are then automatically propagated to the ZIMonSensors.cfg file in /opt/IBM/zimon on all the nodes in the cluster.

c. Start the **pmswiftd.service** using the following command:

```
systemctl start pmswiftd.service
```

d. Start/restart the **pmsensors.service** using the following command:

```
systemctl start|restart pmsensors.service
```

For more information on how to manually upgrade pmswift, see the “Manually upgrading pmswift” on page 395 topic.

3. If the protocol sensors are enabled on a GPFS-only node, you will see an error regarding them being unavailable, however, the other sensors will continue running.
4. Start the sensors on each node using the **systemctl start pmsensors.service** command.
5. On the collector nodes, start the collector, using the **systemctl start pmcollector.service** command.
6. To ensure that sensors and collectors are restarted after the node reboots, you can enable them using the following commands:

Sensors

To disable sensors, use the **systemctl disable pmsensors.service** command.

To enable sensors, use the **systemctl enable pmsensors.service** command.

Collector

To disable the collector, use the **systemctl disable pmcollector.service** command.

To enable the collector, use the **systemctl enable pmcollector.service** command.

The collector node will start gathering all the requested metrics.

| **Important:** Once the RPM is installed, the pmsensor and pmcollector services are activated automatically.

Note: Although you can enable sensors on every node in a system, with the increase in number of nodes, the metric collection work for the collector also increases. It is recommended to ensure that collection of metrics does not increase above 1000000 metrics per second.
By default, the installation toolkit enables sensors on each protocol node (CES node) but not on the other GPFS nodes (non-CES nodes).

Metrics can be retrieved from any node in the system using the **mmperfmon query** command. For more information, see *mmperfmon command* in *IBM Spectrum Scale: Command and Programming Reference*.

For more information about the Performance Monitoring tool, see *Configuring the Performance Monitoring tool* in *IBM Spectrum Scale: Problem Determination Guide*.

Manually installing IBM Spectrum Scale management GUI

The management GUI provides an easy way for the users to configure, manage, and monitor the IBM Spectrum Scale system.

You can install the management GUI by using the following methods:

- Installing management GUI by using the installation GUI. For more information on how to install the management GUI by using the installation GUI, see "Installing IBM Spectrum Scale by using the graphical user interface (GUI)" on page 312.
- Installing management GUI by using the installation toolkit. For more information on installing the management GUI by using the installation toolkit, see "Installing IBM Spectrum Scale management GUI by using the installation toolkit" on page 291.
- Manual installation of the management GUI. The following sections provides the details of how to manually install the management GUI.

Prerequisites

The prerequisites that are applicable for installing the IBM Spectrum Scale system through CLI is applicable for installation through GUI as well. For more information on the prerequisites for installation, see "Installation prerequisites" on page 224.

The Installation package that is part of the IBM Spectrum Scale GUI package is necessary for the installation. You need to extract this package to start the installation. The performance tool packages are also required to enable the performance monitoring tool that is integrated into the GUI. The following packages are required for performance monitoring tools in GUI:

- The performance tool collector package. This package is placed only on the collector nodes.
- The performance tool sensor package. This package is applicable for the sensor nodes, if not already installed.

Note: The GUI must be a homogeneous stack. That is, all packages must be of the same release. For example, do not mix the 5.0.0 GUI rpm with a 4.2.3 base rpm. However, GUI PTFs and efixes can usually be applied without having to install the corresponding PTF or efix of the base package. This is helpful if you just want to get rid of a GUI issue without changing anything on the base layer.
The following table lists the IBM Spectrum Scale GUI and performance tool package that are required for different platforms.

Table 36. GUI packages required for each platform

GUI Platform	**Package name**
RHEL 7.x x86	`gpfs.gui-5.0.0-0.noarch.rpm` `gpfs.java-5.0.0-0.x86_64.rpm`
RHEL 7.x ppc64 (big endian)	`gpfs.gui-5.0.0-0.noarch.rpm` `gpfs.java-5.0.0-0.ppc64.rpm`
RHEL 7.x ppc64le (little endian)	`gpfs.gui-5.0.0-0.noarch.rpm` `gpfs.java-5.0.0-0.ppc64le.rpm`
SLES12 x86	`gpfs.gui-5.0.0-0.noarch.rpm` `gpfs.java-5.0.0-0.x86_64.rpm`
SLES12 ppc64le (little endian)	`gpfs.gui-5.0.0-0.noarch.rpm` `gpfs.java-5.0.0-0.ppc64le.rpm`

Ubuntu 16	`gpfs.gui_5.0.0-0_all.deb`
	`gpfs.java_5.0.0-0_amd64.deb`
	`gpfs.java_5.0.0-0_powerpc.deb`
	`gpfs.java_5.0.0-0_ppc64le.deb`
	`gpfs.java_5.0.0-0_ppc64le.deb`

Performance monitoring tool platform	**Performance monitoring tool rpms**
RHEL 7.x x86	`gpfs.gss.pmcollector-5.0.0-0.el7.x86_64.rpm` `gpfs.gss.pmsensors-5.0.0-0.el7.x86_64.rpm`
RHEL 7 s390x	`gpfs.gss.pmsensors-5.0.0-0.el7.s390x.rpm` `gpfs.gss.pmcollector-5.0.0-0.el7.s390x.rpm`
RHEL 7.x ppc64	`gpfs.gss.pmcollector-5.0.0-0.el7.ppc64.rpm` `gpfs.gss.pmsensors-5.0.0-0.el7.ppc64.rpm`
RHEL 7.x ppc64 LE	`gpfs.gss.pmcollector-5.0.0-0.el7.ppc64le.rpm` `gpfs.gss.pmsensors-5.0.0-0.el7.ppc64le.rpm`
SLES12 x86	`gpfs.gss.pmcollector-5.0.0-0.SLES12.x86_64.rpm` `gpfs.gss.pmsensors-5.0.0-0.SLES12.X86_64.rpm`
SLES12 SP1 s390x	`gpfs.gss.pmsensors-5.0.0-0.SLES12.1.s390x.rpm` `gpfs.gss.pmcollector-5.0.0-0.SLES12.1.s390x.rpm`
SLES12 ppc64	`gpfs.gss.pmcollector-5.0.0-0.SLES12.ppc64.rpm` `gpfs.gss.pmsensors-5.0.0-0.SLES12.ppc64.rpm`
SLES12 ppc64 LE	`gpfs.gss.pmcollector-5.0.0-0.SLES12.ppc64le.rpm` `gpfs.gss.pmsensors-5.0.0-0.SLES12.ppc64le.rpm`
Ubuntu sensor packages	`gpfs.gss.pmsensors_5.0.0-0.U16.04_amd64.deb`
Ubuntu collector packages	`gpfs.gss.pmcollector_5.0.0-0.U16.04_amd64.deb`

Note: Before you start the installation, ensure that the rpm files that are specific to the platform are placed on the installer node.

Ensure that the performance tool collector runs on the same node as the GUI.

Yum repository setup

You can use yum repository to manually install the GUI rpm files. This is the preferred way of GUI installation as yum checks the dependencies and automatically installs missing platform dependencies like the postgres module, which is required but not included in the package.

Installation steps

You can install the management GUI either using the package manager (yum or zypper commands) or by installing the packages individually.

Installing management GUI by using package manager (yum or zypper commands)

It is recommended to use this method as the package manager checks the dependencies and automatically installs missing platform dependencies. Issue the following commands to install management GUI:

Red Hat Enterprise Linux

```
yum install gpfs.gss.pmsensors-5.0.0-0.el7.<arch>.rpm
yum install gpfs.gss.pmcollector-5.0.0-0.el7.<arch>.rpm
yum install gpfs.java-5.0.0-0.<arch>.rpm
yum install gpfs.gui-5.0.0-0.noarch.rpm
```

SLES

```
zypper install gpfs.gss.pmsensors-5.0.0-0.SLES12.<arch>.rpm
zypper install gpfs.gss.pmcollector-5.0.0-0.SLES12.<arch>.rpm
zypper install gpfs.java-5.0.0-0.<arch>.rpm
zypper install gpfs.gui-5.0.0-0.noarch.rpm
```

Installing management GUI by using RPM

Issue the following commands for both RHEL and SLES platforms:

```
rpm -ivh gpfs.java-5.0.0-0.<arch>.rpm
rpm -ivh gpfs.gss.pmsensors-5.0.0-0.el7.<arch>.rpm
rpm -ivh gpfs.gss.pmcollector-5.0.0-0.el7.<arch>.rpm
rpm -ivh gpfs.gui-5.0.0-0.noarch.rpm
```

Installing management GUI on or Ubuntu by using dpkg and apt-get

Issue the following commands for Ubuntu platforms:

```
dpkg -i gpfs.java_5.0.0-0_<arch>.deb
dpkg -i gpfs.gss.pmsensors_5.0.0-0.<kernel>_<arch>.deb
dpkg -i gpfs.gss.pmcollector_5.0.0-0.<kernel>_<arch>.deb
apt-get install postgresql
dpkg -i gpfs.gui_5.0.0-0_all.deb
```

The sensor package must be installed on any additional node that you want to monitor. All sensors must point to the collector node.

Start the GUI

Start the GUI by issuing the **`systemctl start gpfsgui`** command.

Note: The default user name and password to access the IBM Spectrum Scale management GUI is `admin` and `admin001` respectively. Due to security reasons, it is recommended to change the default password after the first login.

Enabling performance tools in management GUI

The performance tool consists of sensors that are installed on all nodes that need to be monitored. It also consists of one or more collectors that receive data from the sensors. The GUI expects that a collector runs on a GUI node. The GUI queries the collectors for performance and capacity data. The following steps use the automated approach to configure and maintain performance data collection by using the **mmperfmon** CLI command. Manually editing the `/opt/IBM/zimon/ZIMonSensors.cfg` file is not compatible with this configuration mode.

1. Install the necessary software packages. Install the collector software package, `gpfs.gss.pmcollector`, on all GUI nodes. Install the sensor software packages, `gpfs.gss.pmsensors`, on all nodes, which are supposed to send the performance data. While mixed software levels for sensors are accepted, it is required that all collectors run at the same software level.
2. Initialize the performance collection. Use the **mmperfmon config generate --collectors [node list]** command to create an initial performance collection setup on the selected nodes. The GUI nodes must be configured as collector nodes. Depending on the installation type, this configuration might be already completed before. However, verify the existing configuration.
3. Enable nodes for performance collection. You can enable nodes to collect performance data by issuing the **mmchnode --perfmon -N [SENSOR_NODE_LIST]** command. **[SENSOR_NODE_LIST]** is a comma-separated list of sensor nodes' host names or IP addresses and you can also use a node class. Depending on the type of installation, nodes might have already been configured for performance collection.
4. Configure peer configuration for the collectors. The collector configuration is stored in the `/opt/IBM/zimon/ZIMonCollector.cfg` file. This file defines collector peer configuration and the aggregation rules. If you are using only a single collector, you can skip this step. Restart the pmcollector service after making changes to the configuration file. The GUI must have access to all data from each GUI node. For more information on how to configure a system to use multiple collectors, see *Configuring multiple collectors* in *IBM Spectrum Scale: Administration Guide*.
5. Configure aggregation configuration for the collectors. The collector configuration is stored in the `/opt/IBM/zimon/ZIMonCollector.cfg` file. The performance collection tool has already predefined rules on how data is aggregated once data gets older. By default, four aggregation domains are created: a raw domain that stores the metrics uncompressed, a first aggregation domain that aggregates data to 30-second averages, a second aggregation domain that stores data in 15-minute averages, and a third aggregation domain that stores data in 6-hour averages.

   In addition to the aggregation that is done by the performance collector, the GUI might request aggregated data depending on the zoom level of the chart. For details on configuring aggregation, see *Configuring the collector* in *IBM Spectrum Scale: Administration Guide*.
6. Configure the sensors. Several GUI pages display performance data that is collected with the help of performance monitoring tools. If data is not collected, the GUI shows the error messages like "No Data Available" or "Objects not found" in the performance charts. Installation using the spectrumscale installation toolkit manages the default performance monitoring installation and configuration. The GUI contextual help that is available on various pages shows performance metric information. The GUI context-sensitive help also lists the sensor names. Use **mmperfmon config show** command to verify the sensor configuration. Use the **mmperfmon config update** command to adjust the sensor configuration to match your needs. For more information on configuring sensors, see *Configuring the sensor* in *IBM Spectrum Scale: Administration Guide*.

   The local file `/opt/IBM/zimon/ZIMonSensors.cfg` can be different on every node and it can be changed whenever there is a configuration change. Therefore, this file must not be edited manually when using the automated configuration mode. During distribution of the sensor configuration, the restrict clause is evaluated and the period for all sensors is set to `0` in the `/opt/IBM/zimon/ZIMonSensors.cfg` file on those nodes that did not match the restrict clause. You can check the local file to confirm that a restrict clause worked as intended.

Capacity-related sensors need manual steps for initialization. It is necessary to configure a single node where the capacity collection must occur. You can do this by using a restrict clause. The restrict clause expects that the node name must be the admin node name as shown by the **mmlscluster** command. Alternative host names that are known to DNS ignored in the restrict clause.

To collect file system and disk level capacity data, update the sensor configuration as shown in the following example, where the IBM Spectrum Scale node name of the GUI node is gui_node:

```
mmperfmon config update GPFSDiskCap.restrict=gui_node GPFSDiskCap.period=86400
```

If this sensor is disabled, the GUI does not show any capacity data. The recommended period is 86400, which means once per day. As this sensor runs **mmdf**, it is not recommended to use a value less than 10800 (every 3 hours) for *GPFSDiskCap.period*.

To show fileset capacity information, it is necessary to enable quota for all file systems where fileset capacity must be monitored. For information on enabling quota, see the **-q** option in *mmchfs command* and *mmcheckquota command* in *IBM Spectrum Scale: Command and Programming Reference*.

To update the sensor configuration to trigger an hourly collection of capacity-based fileset capacity information, issue the **mmperfmon** command as shown in the following example:

```
mmperfmon config update GPFSFilesetQuota.restrict=gui_node GPFSFilesetQuota.period=3600
```

Checking GUI and performance tool status

Issue the **systemctl status gpfsgui** command to know the GUI status as shown in the following example:

```
systemctl status gpfsgui.service
gpfsgui.service - IBM_GPFS_GUI Administration GUI
Loaded: loaded (/usr/lib/systemd/system/gpfsgui.service; disabled)
Active: active (running) since Fri 2015-04-17 09:50:03 CEST; 2h 37min ago
Process: 28141 ExecStopPost=/usr/lpp/mmfs/gui/bin/cfgmantraclient unregister (code=exited, s
tatus=0/SUCCESS)
Process: 29120 ExecStartPre=/usr/lpp/mmfs/gui/bin/check4pgsql (code=exited, status=0/SUCCESS)
Main PID: 29148 (java)
Status: "GSS/GPFS GUI started"
CGroup: /system.slice/gpfsgui.service
≪─29148 /opt/ibm/wlp/java/jre/bin/java -XX:MaxPermSize=256m -Dcom.ibm.gpfs.platform=GPFS
-Dcom.ibm.gpfs.vendor=IBM -Djava.library.path=/opt/ibm/wlp/usr/servers/gpfsgui/lib/
-javaagent:/opt/ibm/wlp/bin/tools/ws-javaagent.jar -jar /opt/ibm/wlp/bin/tools/ws-server.jar gpfsgui
--clean

Apr 17 09:50:03 server-21.localnet.com java[29148]: Available memory in the JVM: 484MB
Apr 17 09:50:03 server.localnet.com java[29148]: Max memory that the JVM will attempt to use: 512MB
Apr 17 09:50:03 server.localnet.com java[29148]: Number of processors available to JVM: 2
Apr 17 09:50:03 server.localnet.com java[29148]: Backend started.
Apr 17 09:50:03 server.localnet.com java[29148]: CLI started.
Apr 17 09:50:03 server.localnet.com java[29148]: Context initialized.
Apr 17 09:50:03 server.localnet.com systemd[1]: Started IBM_GPFS_GUI Administration GUI.
Apr 17 09:50:04 server.localnet.com java[29148]: [AUDIT ] CWWKZ0001I: Application /
started in 6.459 seconds.
Apr 17 09:50:04 server.localnet.com java[29148]: [AUDIT ] CWWKF0012I: The server
installed the following features: [jdbc-4.0, ssl-1.0, localConnector-1.0, appSecurity-2.0,
jsp-2.2, servlet-3.0, jndi-1.0, usr:FsccUserRepo, distributedMap-1.0].
Apr 17 09:50:04 server-21.localnet.com java[29148]: [AUDIT ] CWWKF0011I: ==> When you see
the service was started anything should be OK !
```

Issue the **systemctl status pmcollector** and **systemctl status pmsensors** commands to know the status of the performance tool.

You can also check whether the performance tool backend can receive data by using the GUI or alternative by using a command line performance tool that is called *zc*, which is available in `/opt/IBM/zimon` folder. For example:

```
echo "get metrics mem_active, cpu_idle, gpfs_ns_read_ops last 10 bucket_size 1" | ./zc 127.0.0.1
Result example:
1: server-21.localnet.com|Memory|mem_active
2: server-22.localnet.com|Memory|mem_active
3: server-23.localnet.com|Memory|mem_active
4: server-21.localnet.com|CPU|cpu_idle
5: server-22.localnet.com|CPU|cpu_idle
6: server-23.localnet.com|CPU|cpu_idle
7: server-21.localnet.com|GPFSNode|gpfs_ns_read_ops
8: server-22.localnet.com|GPFSNode|gpfs_ns_read_ops
9: server-23.localnet.com|GPFSNode|gpfs_ns_read_ops
Row Timestamp mem_active mem_active mem_active cpu_idle cpu_idle cpu_idle gpfs_ns_read_ops
gpfs_ns_read_ops gpfs_ns_read_ops
1 2015-05-20 18:16:33 756424 686420 382672 99.000000 100.000000 95.980000 0 0 0
2 2015-05-20 18:16:34 756424 686420 382672 100.000000 100.000000 99.500000 0 0 0
3 2015-05-20 18:16:35 756424 686420 382672 100.000000 99.500000 100.000000 0 0 6
4 2015-05-20 18:16:36 756424 686420 382672 99.500000 100.000000 100.000000 0 0 0
5 2015-05-20 18:16:37 756424 686520 382672 100.000000 98.510000 100.000000 0 0 0
6 2015-05-20 18:16:38 774456 686448 384684 73.000000 100.000000 96.520000 0 0 0
7 2015-05-20 18:16:39 784092 686420 382888 86.360000 100.000000 52.760000 0 0 0
8 2015-05-20 18:16:40 786004 697712 382688 46.000000 52.760000 100.000000 0 0 0
9 2015-05-20 18:16:41 756632 686560 382688 57.580000 69.000000 100.000000 0 0 0
10 2015-05-20 18:16:42 756460 686436 382688 99.500000 100.000000 100.000000 0 0 0
```

Node classes used for the management GUI

The IBM Spectrum Scale management GUI automatically creates the following node classes during installation:

- `GUI_SERVERS`: Contains all nodes with a server license and all the GUI nodes
- `GUI_MGMT_SERVERS`: Contains all GUI nodes

Each node on which the GUI services are started is added to these node classes.

For information about removing nodes from these node classes, see “Removing nodes from management GUI-related node class” on page 372.

For information about node classes, see *Specifying nodes as input to GPFS commands* in *IBM Spectrum Scale: Administration Guide*.

Related concepts:

“Uninstalling the IBM Spectrum Scale management GUI” on page 371
Do the following to uninstall management GUI and remove the performance monitoring components that are installed for the GUI:

“Introduction to IBM Spectrum Scale GUI” on page 94
The IBM Spectrum Scale management GUI provides an easy way to configure and manage various features that are available with the IBM Spectrum Scale system.

Root privilege considerations for IBM Spectrum Scale management GUI

The IBM Spectrum Scale GUI WebSphere Java process no longer runs as “root” but as a user named “scalemgmt”. This provides improved security because web applications running as root are vulnerable to security threats. The *scalemgmt* user is set up as a system account with no login privileges.

The GUI and GUI user still need root privileges in the backend to perform the following tasks:

- Monitoring and managing the cluster by issuing the IBM Spectrum Scale CLI commands.
- Bind to the privileged ports such as 80 (http) and 443 (https) to work with the GUI.

The system automatically performs the required configuration. It does not require any configuration to be done by the user.

Enabling the scalemgmt user to monitor and manage the system through GUI

As root privileges are not available to the GUI user, the system enables the scalemgmt user to run the CLI commands through sudo. The GUI installation adds the file `/etc/sudoers.d/scalemgmt_sudoers`, which allows the scalemgmt user to run commands that match the wildcard “`/usr/lpp/mmfs/bin/mm`”.

Binding to the GUI privileged ports

A non-root process cannot bind privileged ports such as 80 and 443. So, the GUI process now runs on port 47443 and 47080 and uses *iptables* rules to forward port 443 to 47443 and 80 to 47080. The GUI is only available in the browser on port 443 through HTTPS. However, versions prior to 4.2.3 used port 80 to send events to the GUI, so to stay backwards compatible in clusters that still contain nodes with older versions, the GUI still accepts events on port 80.

To reach the GUI from outside, not only port 443 and 80 but also port 47443 and 47080 must be opened to the outside. This is because the port forwarding redirects port 443 to 47443 and 80 to 47080 before evaluating the rules of the iptables INPUT chain.

The iptables rules necessary for the port forwarding are automatically checked every time the GUI is started through the **`systemctl start gpfsgui`** command. The user does not have to configure anything manually.

Installing IBM Spectrum Scale on Linux nodes with the installation toolkit

Use this information to install IBM Spectrum Scale using the installation toolkit. The information to use the installation toolkit for tasks such as deploying protocols on existing clusters and adding nodes to an existing installation is also documented.

Overview of the installation toolkit

The installation toolkit automates the steps required to install GPFS, deploy protocols, and install updates and patches.

| For a release-wise list of features available in the installation toolkit, see Table 37 on page 260.

When using the installation toolkit, you provide environmental information and the toolkit installs, configures, and deploys the optimal configuration, dynamically creating a cluster definition file.

This install toolkit enables you to do the following:

- Install and configure GPFS.
- Add GPFS nodes to an existing cluster.
- Deploy and configure SMB, NFS, Object, and performance monitoring tools on top of GPFS.
- | Perform verification prior to installing, deploying, or upgrading. It includes checking whether | passwordless SSH is set up correctly.
- Configure authentication services for protocols.
- | Enable and configure call home and file audit logging functions.
- Upgrade GPFS and all protocols and install patches.

Installation and configuration are driven through commands.

In the self-extracting package, the installation toolkit is in this location:

`*location_extracted_to*/5.0.x.x/installer`

Using the installation toolkit is driven from the **spectrumscale** executable in the installer directory, and this can optionally be added to the path.

Note: The installation toolkit installs the Chef configuration management tool, a Python-based tool wrapped around Opscode Chef technologies, enabling configuration management and deployment at scale. For more information, see Apache license information (www.apache.org/licenses/LICENSE-2.0).

The installation toolkit operation consists of four phases:

1. User input using **`spectrumscale`** commands
   a. All user input is recorded into a cluster definition file file in `/usr/lpp/mmfs/5.0.x.x/installer/configuration`
   b. Review the cluster definition file to make sure that it accurately reflects your cluster configuration
   c. As you input your cluster configuration, remember that you can have the toolkit act on parts of the cluster by simply not telling it about nodes that may have incompatible OS levels, architectures, etc.
2. A **`spectrumscale install`** phase
   a. Installation acts upon all nodes defined in the cluster definition file.
   b. GPFS and performance monitoring packages are installed.
   c. Call home and file audit logging packages might be installed.
   d. GPFS portability layer is created.
   e. GPFS is started.
   f. A GPFS cluster is created.
   g. Licenses are applied.
   h. GUI nodes might be created and the GUI might be started upon these nodes.
   i. NTP, performance monitoring, GPFS ephemeral ports, and cluster profile might be configured.
   j. NSDs might be created.

   Note: File systems are not created during installation.
3. A **`spectrumscale deploy`** phase
   a. Deployment acts upon all nodes defined into the cluster definition file.
   b. File systems are configured. It is possible to only configure file systems during the deploy phase if you do not want protocols.
   c. SMB, NFS, and Object protocol packages are copied to all protocol nodes.
   d. SMB, NFS, and Object services may be started.
   e. Authentication might be configured.
   f. Licenses are applied.
   g. GUI nodes might be created and the GUI might be started upon these nodes.
   h. NTP, performance monitoring, call home, file audit logging, GPFS ephemeral ports, and cluster profile might be configured.
   i. NSDs might be created.
4. A **`spectrumscale upgrade`** phase
   a. Upgrade acts upon all nodes input into the cluster definition file.
   b. All installed/deployed components are upgraded
   c. Upgrades are sequential with multiple passes. For more information, see “IBM Spectrum Scale installation toolkit - upgrade process flow” on page 413.

d. The IBM Spectrum Scale GUI might be upgraded and restarted upon upgrade

Note: The upgrade phase does not allow new function to be enabled. In the upgrade phase, the required packages are upgraded, but adding functionality must be done either before or after the upgrade.

For information about command options available with the **spectrumscale** command, see the **spectrumscale** command description in the *IBM Spectrum Scale: Command and Programming Reference*.

The following table lists the features available in the installation toolkit in the reverse chronological order of releases.

Table 37. Installation toolkit: List of features

Release	Features
5.0.0.x	• Extended operating system support – Ubuntu 16.04.0, 16.04.1, 16.04.2, 16.04.3 on x86_64 – RHEL 7.4 on x86_64, PPC64, and PPC64LE – SLES 12 SP3 on x86_64 • Improved deployment integration with Elastic Storage Server: The installation toolkit has the ability to detect ESS nodes (EMS and I/O) and it ensures validation of permitted operations when adding protocol or client nodes to a cluster containing ESS. • File audit logging installation and configuration • Call home configuration • Cumulative object upgrade support • Enhanced network connectivity pre-checks including passwordless SSH validation from the admin node • Updated file system default block size for more likely best performance defaults
4.2.3.x	• Extended Operating System support (SLES 12 SP1, SP2 on x86_64) • Enhanced problem determination: Enhanced pre-checks including checks for base OS repository setup, base software requirements, authentication prerequisites and error reporting to console for faster problem resolution • Config populate: Installation toolkit configuration retrieval from existing clusters that queries all required cluster topology and state (allowing for seamless transition to further operations with the installation toolkit usage on clusters created without it). • Installation and upgrade of `gpfs.adv`, `gpfs.crypto`, and license packages • Cluster upgrade enhancements (Extended upgrade pre-checks and supports the Spectrum scale upgrade when in an LTFS environment) • Support for up to 7 secondary NSD servers
4.2.2.x	• Enhanced operating system support – RHEL 7.3 on x86_64, PPC64, PPC64LE – RHEL 7.1, RHEL7.2 on PPC64LE – SLES 12 on x86_64 (no Object) – RHEL 6.8 on x86_64 and PPC64 (no CES) • Extended platform architecture support (PPC64LE with RHEL 7.1 and 7.2) • Heterogeneous OS clusters (a single cluster can now contain any combination of supported operating systems)

Table 37. Installation toolkit: List of features (continued)

Release	Features
4.2.1.x	• Enhanced operating system support (RHEL 7.2 CES protocol support on x86_64 and PPC64) • Guided installation and deployment • Logging and debugging and separate error log • Enhanced pre-checks • Support for Central Chef server • Shared Java support
4.2.0.x	• All upgrades involving CES protocols must upgrade to 4.2.0.0 before continuing on. • Management GUI installation and configuration during installation, deployment, and node addition • Installation GUI for basic cluster configuration (no NSD or file system or Protocols) • Additional Object configuration • Build package now Includes full packages for upgrades • Non-RHEL 7.x awareness
4.1.1 **[Initial release of the installation toolkit]**	• Stacked CES + NSD node support (not ESS) • Cluster installation and creation • NSD creation during installation • Client / Server license configuration • File system creation during deployment • Operating system support (RHEL 7.0 and RHEL 7.1 on x86_64) • Protocols Quick Overview Guide • Single node cluster support • Install Toolkit portability (Newer toolkits can be used with older code) • Authentication configuration during deployment – File: LDAP, AD, NIS, none – Object: LDAP, AD, local, ext • Performance monitoring re-configuration • Ability to add nodes, protocols, NSDs, file systems, authentication to existing configurations

Understanding the installation toolkit options

Use the following information to understand how to work with the installation toolkit, and toolkit options.

When you use the installation toolkit to install GPFS, the procedure comprises two stages:

1. Using a series of **spectrumscale** commands to add node and NSD specifications and configuration properties to the cluster definition file.
2. Using the **spectrumscale install** command to install GPFS on the nodes specified in the cluster definition file and to apply the configuration options to the cluster.

When you use the installation toolkit to deploy protocols, the procedure comprises two similar stages:

1. Using a series of **spectrumscale** commands to specify environment details, to identify which protocols are to be enabled, and to define protocol-specific properties in the cluster definition file.

2. Using the **spectrumscale deploy** command to deploy protocols as specified in the cluster definition file.

If you already have an existing GPFS cluster, with GPFS started and at least one file system for the CES shared file system, you can just define protocol nodes in the cluster definition and then deploy protocols.

See "Using the installation toolkit to perform installation tasks: Explanations and examples" on page 272 for information on how to use the command options listed in Table 38 to perform the installation and deployment. After you review the examples, you can tailor them according to your own needs and then proceed to implement them.

Table 38. **spectrumscale** *command options for installing GPFS and deploying protocols*

spectrumscale command option	Purpose
node add	Adds node specifications to the cluster definition file.
node delete	Removes node specifications from the cluster definition file
nsd add	Adds NSD specifications to the cluster definition file.
nsd clear	Clears all NSDs.
nsd delete	Removes a single NSD.
nsd list	Lists all NSDs currently in the configuration.
nsd modify	Modifies an NSD.
nsd balance	Balances the NSD preferred node between the primary and secondary nodes.
filesystem list	Lists all file systems that currently have NSDs assigned to them.
filesystem modify	Changes the block size and mount point of a file system.
auth file	Configures file authentication on protocols in the cluster definition file.
auth object	Specifies Object authentication on protocols in the cluster definition file.
config gpfs	Adds GPFS-specific properties to the cluster definition file.
install	Installs GPFS on the configured nodes, creates a cluster, and creates NSDs.
config protocols	Provides details about the GPFS environment to be used during protocol deployment.
config perfmon	Configures performance monitoring settings
config object	Defines object-specific properties to be applied during deployment.
deploy	Creates file systems and deploys protocols on your configured nodes.
config ntp	Configures NTP settings
callhome	Configures call home settings
fileauditlogging	Configures file audit logging settings
upgrade	Upgrades the various components of the installation.

The list of **spectrumscale** command options listed is not exhaustive. For all command options available with the **spectrumscale** command and for more information about these command options, see the **spectrumscale** command description in the *IBM Spectrum Scale: Command and Programming Reference*.

Required cleanup if you are reinstalling with the installation toolkit

If you are reinstalling with the installation toolkit, you must perform some cleanup procedures first. See "Cleanup procedures required if reinstalling with the installation toolkit" on page 366.

Limitations of the installation toolkit

Before using the installation toolkit to install IBM Spectrum Scale and deploy protocols, review the following limitations and workrounds, if any.

Table 39. Functions not supported by the installation toolkit

Function	**Description**	**Workaround, if any**
All-at-once upgrade	The installation toolkit does not support a single outage, all-at-once upgrade.	Use the manual upgrade procedure.
CES groups	The installation toolkit does not support the configuration of CES groups. This causes protocol deployments to fail in multi-network environments. This might cause issues with upgrades as well.	
Clusters larger than 16 nodes	The installation toolkit does not restrict the number of nodes in a cluster in which the toolkit can be used. However, it is currently designed as a single node server so as the number of nodes increases, the bandwidth to the installer node decreases and the latency goes up. This might cause issues in implementations with more than 16 nodes. **Note:** If you want to use the installation toolkit in a cluster larger than 16 nodes, contact scale@us.ibm.com.	
Clusters without passwordless SSH between all nodes	If clusters are setup in an `AdminCentral=True` configuration, which is a widely used configuration, the installation toolkit and protocols might not function correctly.	Set up passwordless SSH between all nodes in the cluster and to the nodes themselves using FQDN, IP address, and host name.
Compression	The installation toolkit does not configure file system compression.	After installation, configure the compression function manually. For more information, see *File compression* in *IBM Spectrum Scale: Administration Guide*.

Table 39. Functions not supported by the installation toolkit (continued)

Function	**Description**	**Workaround, if any**
Concurrent upgrade	The installation toolkit does not support concurrent upgrade. You must plan for an outage depending on your setup. This is a brief outage to prevent mixed code versions from running at the same time and it is also required in case of a manual upgrade. Although the upgrade is non-concurrent, data is typically still accessible during the upgrade window. Access might be lost and need to be re-established multiple times due to how the upgrade procedure is executed among nodes.	
Configuration change	The installation toolkit does not support changing existing settings and node designations. Although, it can be used to add nodes, NSDS, or file systems to an existing cluster. The installation does not support authentication reconfiguration. It does not use the authentication section of the cluster definition file during upgrade.	If you want to change the authentication method, see If you want to change the authentication method, see *Modifying authentication method* in *IBM Spectrum Scale: Administration Guide*.
Customer designation of sensor or collector nodes	The installation toolkit does not support customer designation of sensor or collector nodes. The installation toolkit automatically sets up sensor or collectors without allowing the user to choose which nodes will have these functions.	1. Use the **./spectrumscale config perfmon -d** flag to disable performance monitoring and the **-r** flag to reconfigure performance monitoring, if required. 2. Follow up with manual configuration of performance monitoring.
Custom profiles	The installation toolkit allows a user to choose between two GPFS profiles during installation prior to cluster creation.	Obtain the sample profiles that the installation toolkit uses and edit them directly. The profiles are located in the `/usr/lpp/mmfs/profiles` directory.
Disabling or uninstalling protocols and uninstalling GPFS	The installation toolkit does not support disabling or uninstalling protocols and uninstalling GPFS on an existing GPFS cluster.	Use the manual procedures. • For information about removing exports, see *Managing protocol data exports* in *IBM Spectrum Scale: Administration Guide*. • For information about disabling protocol services, see *Managing protocol services* in *IBM Spectrum Scale: Administration Guide*. • For information about uninstalling GPFS, see Chapter 14, “Steps to permanently uninstall GPFS and/or Protocols,” on page 365.

Table 39. Functions not supported by the installation toolkit (continued)

Function	Description	Workaround, if any
Encryption	The installation toolkit does not support encrypted file systems. Therefore, installation and deployment using the installation toolkit do not work if the CES shared root file system or any other file system that the installation toolkit works with is encrypted.	
EPEL and OpenStack repositories	The installation toolkit does not support the configuration of protocols when either EPEL or OpenStack repositories are configured	Remove or disable these repositories before using the installation toolkit for installation, deployment, or upgrade.
File system DMAPI flag set to Yes (-z)	The file system DMAPI flag is used in IBM Spectrum Scale for IBM Spectrum Protect for Space Management and policy management, attaching with an IBM Spectrum Protect server, and with IBM Spectrum Archive. The installation toolkit cannot be used for installation or deployment in clusters where the DMAPI flag is set to `Yes` for any of the file systems. For upgrade in clusters where the DMAPI flag is set to `Yes` for any of the file systems, the installation toolkit can be used with some additional steps.	Use the manual procedures for installation and deployment. For more information, see "Manually installing the IBM Spectrum Scale software packages on Linux nodes" on page 233. For information on using the installation toolkit for upgrade in clusters where the DMAPI flag is set to `Yes` for any of the file systems, see "Upgrading IBM Spectrum Scale on IBM Spectrum Archive Enterprise Edition (EE) nodes using the installation toolkit" on page 411.
FPO configuration for disks	The installation toolkit does not support the extra stanza file flags required for FPO setup.	Do one of the following: 1. Create NSDs using the installation toolkit. 2. Manually edit the NSD stanza file afterwards. The installation toolkit places the NSD stanza file in the `/usr/lpp/mmfs` directory. 3. Use **`mmchnsd`** to do the changes. **OR** 1. Create the cluster using the installation toolkit. 2. Deploy protocols on the protocol nodes using the installation toolkit. 3. Manually create the NSDs for the FPO setup.
Host-based SSH authentication	The installation toolkit does not support host-based SSH authentication. It supports only key-based SSH authentication.	Either set up key-based SSH authentication temporarily for use with the toolkit, or follow the manual steps in "Manually installing the IBM Spectrum Scale software packages on Linux nodes" on page 233.

Table 39. Functions not supported by the installation toolkit (continued)

Function	Description	Workaround, if any
IPv6	The installation toolkit supports only IPv4 address as input. Protocols also have this limitation.	
iSCSI BLOCK	The installation toolkit does not install or configure the iSCSI BLOCK function.	Use the manual procedure. For more information, see "Installing Cluster Export Services as part of installing IBM Spectrum Scale on Linux systems" on page 243.
Multiple CES networks	The installation toolkit does not support deployments with multiple CES networks. Attempting deployment in this configuration has a high probability of failure. This is because when the CES address pool has multiple subnets, the command for adding CES address assigns an IP to a CES node that cannot handle that subnet of address, which causes the deployment failure.	
Multiple clusters	The installation toolkit does not support multiple clusters being defined in the cluster definition.	
Multi-region object deployment	For a multi-region object deployment, the installation toolkit only sets up the region number not the replication. For information about setting up multi-region object deployment, see "Enabling multi-region object deployment initially" on page 315.	
NFS or SMB exports configuration	The installation toolkit does not configure any exports on SMB or NFS.	Use the manual procedure. For information about configuring Cluster Export Services and creating exports, see *Configuring Cluster Export Services* and *Managing protocol data exports* in *IBM Spectrum Scale: Administration Guide*.
Node function addition during upgrade	The installation toolkit does not support designating node functionality during upgrade.	To add a function to the cluster or a node, designate this new function using the installation toolkit and proceed with an installation or a deployment. Perform this action either before or after an upgrade.
NSD balancing with pools or failure groups	The installation toolkit NSD balance function only balances NSDs evenly across primary and secondary NSD servers. It does not take into account pools or failure groups.	Add nodes using the **./spectrumscale nsd add** command that have the required balanced values already configured manually across failure groups, pools, and primary and secondary NSD servers.

Table 39. Functions not supported by the installation toolkit (continued)

Function	Description	Workaround, if any
NSD SAN attachment during initial installation	The installation toolkit cannot be used for NSD SAN attachment during initial installation because when adding NSDs using the installation toolkit, a primary and an optional comma-separated list of secondary NSD servers must be designated.	1. Create NSDs using the installation toolkit. 2. Manually edit the NSD stanza file afterwards. 3. Use **mmchnsd** to do the changes.
NTP server setup	The installation toolkit can be used to configure NTP clients on supported operating systems but not NTP servers.	
Package managers other than `yum`, `zypper`, or `apt--get`	The installation toolkit requires the use of `yum` (RHEL), `zypper` (SLES), and `apt--get` (Ubuntu) package managers to function.	
PPC and x86_64 mix	The installation toolkit does not support mixed CPU architecture configurations.	Use the installation toolkit on a subset of nodes that are supported and then manually install, deploy, or upgrade on the remaining nodes.
PPC LE and PPC BE mix	The installation toolkit does not support mixed endian configurations.	Use the installation toolkit on a subset of nodes that are supported and then manually install, deploy, or upgrade on the remaining nodes.
Quorum or manager configuration after cluster installation	The installation toolkit allows a user to add `-m` (to specify a manager node) and `-q` (to specify a quorum node) flags to various nodes as they are added. If the proposed configuration does not match the existing configuration, the installation toolkit does nothing to change it.	Manually change node roles by using the **mmchnode** command.
Remote mounted file systems	• Object protocol deployment using the installation toolkit fails in case of remotely mounted file systems. This occurs because the Object component must both list and create filesets, which is not allowed on a remotely mounted file system. • NFS and SMB deployments on remote mounted file systems work using the installation toolkit. • The set up of a CES shared root file system when the file system is remotely mounted works using the installation toolkit.	
Repository proxy	The installation toolkit does not support proxy setups when working with repositories. `yum repolist` must not have any failed repos and it must be clean.	Ensure that there are no stale or failed repositories and that only the base OS repositories are enabled during any installation toolkit activities such as installation, deployment, or upgrade.

Table 39. Functions not supported by the installation toolkit (continued)

Function	Description	Workaround, if any
RPMs that place dependencies upon GPFS RPMs and GPFS settings	In an environment where some RPMs have dependencies on base GPFS RPMs or GPFS settings, the installation toolkit cannot be used for installation or upgrade.	
Running **mmchconfig release=LATEST** to complete an upgrade	The installation toolkit does not run **mmchconfig release=LATEST** after an upgrade. This is to give users time to verify an upgrade success and decide if the code level upgrade should be finalized.	Use **mmchconfig release=LATEST** after an upgrade using the installation toolkit to finalize the upgrade across the cluster.
Separate admin and daemon network	The installation toolkit does not support separate admin and daemon network.	
Sudo user	The installation toolkit does not function correctly unless run as root. Running as sudo or as another user does not work.	
Support for AIX, Debian, PowerKVM, Windows, Linux on Z	The installation toolkit does not support AIX, Debian, PowerKVM, Windows, or Linux on Z operating systems. If these operating systems are installed on any cluster nodes, do not add these nodes to the installation toolkit.	Use the installation toolkit on a subset of nodes that are supported and then manually perform installation, deployment, or upgrade on the remaining nodes. For information about manual upgrade, see Chapter 15, "Upgrading," on page 375.
Tie-Breaker NSD configuration	The installation toolkit does not configure tie-breaker disks.	Manually set the tie-breaker configuration as required using **mmchconfig** after completing installation using the toolkit.
Transparent cloud tiering	The installation toolkit does not install, configure, or upgrade Transparent cloud tiering.	Use the manual procedures. For more information, see Chapter 7, "Installing Cloud services on IBM Spectrum Scale nodes," on page 337.
Unique NSD device configuration	The installation toolkit relies upon a user having already configured and run the **nsddevices** sample script provided within a GPFS installation. The **mmcrnsd** and **mmchnsd** commands require running of the **nsddevices** script beforehand. Therefore, the installation toolkit will fail if this has not been done by the user.	
Upgrade in clusters in which the entire protocol stack is not installed	The installation toolkit does not support upgrade in clusters in which the entire protocols stack was not originally installed.	Use the manual upgrade procedure. For information about manual upgrade, see Chapter 15, "Upgrading," on page 375.

Table 39. Functions not supported by the installation toolkit (continued)

Function	Description	Workaround, if any
Upgrade while skipping over versions	The installation toolkit does not support skipping over major or minor versions of IBM Spectrum Scale releases when upgrading. For example, If an IBM Spectrum Scale cluster is at version 4.1.1.x, you cannot use the installation toolkit to upgrade it directly to version 5.0.x.	Do an initial upgrade from version 4.1.1.x to 4.2.x and then subsequently do another upgrade from 4.2.x to 5.0.x.

Related concepts:

"Limitations of config populate option of the installation toolkit" on page 292
You can use the **config populate** option of the installation toolkit to populate the cluster definition file with the current cluster state.

Mixed operating system support with the installation toolkit

You can use the installation toolkit to install GPFS and deploy protocols in a cluster that contains nodes that are running on different operating systems. Several validations are done when you use the **spectrumscale** command in a mixed operating system cluster.

The following operating systems are supported with the installation toolkit in a mixed operating system cluster.

Table 40. Operating systems supported with the installation toolkit in a mixed cluster

CPU architecture	Operating system
x86_64	• Red Hat Enterprise Linux 7.1, 7.2, and 7.3 • Ubuntu 16.04 LTS • SLES 12

Note: The object protocol is not supported on SLES 12.

Important:

- In a mixed operating system cluster, all protocol nodes must be running on the same operating system. However, nodes that are running on different minor versions of Red Hat Enterprise Linux 7.x (7.1, 7.2, or 7.3) can be designated as protocol nodes in the same cluster.
- In a mixed operating system cluster, all nodes must have the same CPU architecture and endianness.

The installation toolkit performs the required validations in a mixed operating system cluster to prevent users from attempting any configurations that are not supported. These validations include the following.

Table 41. Validations by the installation toolkit in a mixed operating system cluster

Operation that is attempted with the installation toolkit	Validations done
Node addition by using the **spectrumscale node add** *NodeName* command	Check whether the operating system of the target node is supported. If it is not supported, an error occurs.

Table 41. Validations by the installation toolkit in a mixed operating system cluster (continued)

Operation that is attempted with the installation toolkit	Validations done
Protocol node designation by using the **spectrumscale node add** *NodeName* **-p** command	• Check whether the operating system of the target node is SLES 12. If it is SLES 12, a warning is generated that states: Object protocol is not supported on SLES 12. • Check whether all protocol nodes being added are running on the same operating system. If you attempt to add protocol nodes running on different operating systems, then that operation is not allowed. For example, if you have added a protocol node running on Red Hat Enterprise Linux 7.x and then you try to add another protocol node running on SLES 12, then that operation is not allowed.
Protocol enablement by using the **spectrumscale enable** *Protocol* command	Check whether the protocol being enabled is supported on the operating system that is running on the node. If the protocol is not supported, an error occurs and you cannot continue with the installation and deployment. For example, if you try to enable object on a node running on SLES 12, an error occurs.

Preparing to use the installation toolkit

Before using the installation toolkit, complete the following preparatory steps on all the nodes on which you plan to use the installation toolkit.

1. Ensure that the following requirements are met.

   Operating systems supported with the installation toolkit

   The installation toolkit is supported on the following operating systems.

   - Red Hat Enterprise Linux 7.1, 7.2, 7.3, and 7.4 operating systems on the x86_64 architecture.
   - SLES 12 operating system on the x86_64 architecture.
   - Ubuntu 16.04 LTS operating system on the x86_64 architecture.
   - Red Hat Enterprise Linux 7.1, 7.2, 7.3, and 7.4 operating systems on the PPC64 architecture.
   - Red Hat Enterprise Linux 7.1, 7.2, 7.3, and 7.4 operating systems on the PPC64LE architecture.

   For information about how the installation toolkit can be used in a cluster that has nodes with mixed operating systems, see “Mixed operating system support with the installation toolkit” on page 269.

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

   Required packages for the installation toolkit

   The installation toolkit requires the following package:

   - python-2.7

   Root access for the installation toolkit

   The installation toolkit must be run as the root user. On Ubuntu 16.04 LTS, root login is disabled by default. To use the installation toolkit on Ubuntu 16.04 LTS, you must enable the root login.

   Call home information required for the installation toolkit

The installation toolkit requires call home data to work properly. You must have the *customer name*, the *customer id*, the *customer email*, and the *customer country code* available before installing and configuring call home with the installation toolkit.

Disable auto updates on Ubuntu 16.04 LTS nodes

On Ubuntu 16.04 LTS nodes, auto updates or unattended upgrades must be disabled, and kernel auto updates upon node reboot must also be disabled. This is to ensure that any updates that are not supported with the installation toolkit do not get installed.

For information on prerequisites for protocols and performance monitoring, see "Installation prerequisites" on page 224.

2. Set up passwordless SSH as follows.
   - From the admin node to the other nodes in the cluster.
   - From protocol nodes to other nodes in the cluster.
   - From every protocol node to rest of the protocol nodes in the cluster.

   The installation toolkit performs verification during the precheck phase to ensure that passwordless SSH is set up correctly. This verification includes:
   - Check whether passwordless SSH is set up between all admin nodes and all the other nodes in the cluster. If this check fails, a fatal error occurs.
   - Check whether passwordless SSH is set up between all protocol nodes and all the other nodes in the cluster. If this check fails, a warning is displayed.
   - Check whether passwordless SSH is set up between all protocol nodes in the cluster. If this check fails, a fatal error occurs.
3. Set up repositories depending on your operating system.
   - These package managers must be set up depending on the operating system.
     - Red Hat Enterprise Linux 7.1, 7.2, 7.3, 7.4: yum repository must be set up on all nodes in the cluster
     - SLES 12: zypper repository must be set up on all nodes in the cluster
     - Ubuntu 16.04 LTS: apt repository must be set up on all nodes in the cluster
   - Configure repositories depending on whether you have internet connection or not. However, ensure that base operating system packages are available and EPEL repositories are disabled.
4. Ensure that SELinux is disabled.

   Note: If you cannot disable SELinux for some reason, it must be in the permissive mode. However, there are a lot of log entries that are generated in the permissive mode. Therefore, disabling SELinux is recommended.
5. Ensure that the ports required for installation are open.

   Note: The installation toolkit checks if the firewall daemon is running and displays a warning if it is running. If the required ports are open, you can ignore the warning.
   For information about the ports that need to be open, see *Securing the IBM Spectrum Scale system using firewall* in *IBM Spectrum Scale: Administration Guide*.
6. Ensure that the required kernel packages are installed.

   For more information, see "Software requirements" on page 134.

   Note: You might have to force the installation of a specific version of these packages because a package of a version newer than the corresponding kernel might get picked up by default.
7. Ensure that networking is set up in one of the following ways.
   - DNS is configured such that all host names, either short or long, are resolvable.
   - All host names are resolvable in the `/etc/hosts` file. The host entries in the `/etc/hosts` file must be in the following order:

```
<IP address> <Fully qualified domain name> <Short name>
```

8. Obtain the IBM Spectrum Scale self-extracting protocols installation package.

   Note:

   The installation toolkit is available only in the protocols packages.

Using the installation toolkit to perform installation tasks: Explanations and examples

Use these explanations and examples of installation toolkit options and tasks to perform installation and deployment by using the installation toolkit.

After you have determined how you want your own system to be configured, and after you have reviewed the **spectrumscale** command description in the *Command reference* section *IBM Spectrum Scale: Command and Programming Reference*, you can tailor these examples to do the following:

- Set up your install node.
- Add node and NSD specifications, file system information, and GPFS configuration properties to your cluster definition file, and then install GPFS and configure your cluster according to the information in that file.
- Specify environment details, identify which protocols are to be enabled, and define protocol-specific properties in the cluster definition file, and then deploy protocols on your system.

For information on using the installation toolkit for the installation and the deployment of protocol and client nodes in a cluster containing ESS, see "ESS awareness with the installation toolkit" on page 294.

Before using the installation toolkit for installation and deployment, ensure that the preparatory steps are completed. For more information, see "Preparing to use the installation toolkit" on page 270.

Using the installation toolkit to perform installation and deployment involves doing these tasks.

1. "Setting up the installer node"
2. "Defining the cluster topology for the installation toolkit" on page 273
3. "Installing GPFS and creating a GPFS cluster" on page 277
4. "Deploying protocols" on page 279
5. "Setting up authentication" on page 284

For more installation toolkit related information, see the following topics.

- "Installation of performance monitoring tool using the installation toolkit" on page 286
- "Enabling and configuring file audit logging using the installation toolkit" on page 287
- "Enabling and configuring call home using the installation toolkit" on page 289
- "Installing IBM Spectrum Scale management GUI by using the installation toolkit" on page 291
- "Logging and debugging for installation toolkit" on page 286

Setting up the installer node

The first step in using the installation toolkit for installing IBM Spectrum Scale and deploying protocols is to configure your installer node.

A good candidate for your installer node is the GPFS `admin` node, because a prerequisite of this node is that it must be able to communicate with all other nodes without the need to provide a password. Also, SSH connections between the `admin` node and all other nodes must be set up to suppress prompting. Therefore, no prompts should exist when using ssh among any cluster nodes to and from each other, and to and from the server.

First, find the installation toolkit by changing directories to where it was extracted (the default extraction path for 5.0.0.0 is as follows. This path varies depending on the code level).

```
cd /usr/lpp/mmfs/5.0.0.0/installer
```

To configure the installer node, issue the following command:

```
./spectrumscale setup -s InstallNodeIP -i SSHIdentity
```

The **-s** argument identifies the IP that nodes will use to retrieve their configuration. This IP will be one associated with a device on the installer node. (This is automatically validated during the setup phase.)

Optionally, you can specify a private SSH key to be used to communicate with nodes in the cluster definition file, using the **-i** argument.

In an Elastic Storage Server (ESS) cluster, if you want to use the installation toolkit to install GPFS and deploy protocols you must specify the setup type as `ess` while setting up the installer node as follows.

```
./spectrumscale setup -s InstallNodeIP -i SSHIdentity -st ess
```

For more information, see “ESS awareness with the installation toolkit” on page 294.

As part of the setup process, the appropriate Chef packages will be deployed on the installer node.

Defining the cluster topology for the installation toolkit

Use these instructions to set up the cluster definition file prior to installing GPFS and deploying protocols.

- “Defining the cluster topology for the installation toolkit”
- “Adding and configuring NSD server nodes in the cluster definition file” on page 275
- “Defining file systems” on page 276

Adding node definitions to the cluster definition file

GPFS node definitions are added to the cluster definition file through the **spectrumscale node add** command.

1. If the install toolkit is being used from a location outside of any of the nodes to be installed, a GPFS Admin Node is required. The Admin Node will be used to run GPFS cluster-wide commands.

   To specify a GPFS Admin Node in the cluster definition file, use the **-a** argument:

   ```
   ./spectrumscale node add gpfsnode1 -a
   ```

   If no GPFS Admin Node is specified in the cluster definition file, the node the install toolkit is running on is automatically designated as the Admin Node. If GUI nodes are to be installed, each GUI node must also be marked as an Admin node.

   The role of an Admin node with regards to the installation toolkit is to serve as the coordinator of the installation, deployment, and upgrade. This node will also act as a central repository for all Spectrum Scale rpms. For larger clusters, it is important to have an Admin node with plenty of network bandwidth to all other nodes in the cluster.

2. To add GPFS Client nodes to the cluster definition file, provide no arguments:

   ```
   ./spectrumscale node add gpfsnode1
   ```

3. To add GPFS manager nodes to the cluster definition file, use the **-m** argument:

   ```
   ./spectrumscale node add gpfsnode2 -m
   ```

   If no manager nodes are added to the cluster definition, the install toolkit will automatically designate manager nodes using the following algorithm:

   a. First, all protocol nodes in the cluster definition will be designated as manager nodes.

   b. If there are no protocol nodes, all NSD nodes in the cluster definition will be designated as manager nodes.
   c. If there are no NSD nodes, all nodes in the cluster definition will be designated as manager nodes.

4. GPFS quorum nodes are added to the cluster definition using the **-q** argument.

   ```
   ./spectrumscale node add gpfsnode3 -q
   ```

   If no quorum nodes are added to the cluster definition, the install toolkit will automatically designate quorum nodes using the following algorithm:

   a. If the number of nodes in the cluster definition is less than 4, all nodes will be designated as quorum nodes.
   b. If the number of nodes in the cluster definition is between 4 and 9 inclusive, 3 nodes will be designated as quorum nodes.
   c. If the number of nodes in the cluster definition is between 10 and 18 inclusive, 5 nodes will be designated as quorum nodes.
   d. If the number of nodes in the cluster definition is greater than 18, 7 nodes will be designated as quorum nodes.

   This algorithm will preferentially select NSD nodes as quorum nodes. If the number of NSD nodes is less than the number of quorum nodes to be designated then any other nodes will be selected until the number of quorum nodes is satisfied.

5. GPFS NSD servers are added to the cluster definition using the **-n** argument.

   ```
   ./spectrumscale node add gpfsnode4 -n
   ```

6. GPFS Graphical User Interface servers are added to the cluster definition using the **-g** argument.

   ```
   ./spectrumscale node add gpfsnode3 -g
   ```

   A GUI server must also be an admin node. Use the -a flag:

   ```
   ./spectrumscale node add gpfsnode3 -a
   ```

   If no nodes have been specified as management GUI servers, then the GUI will not be installed. It is recommended to have at least 2 management GUI interface servers and a maximum of 3 for redundancy.

7. To display a list of all nodes in the cluster definition file, use the **./spectrumscale node list** command. For example:

   ```
   ./spectrumscale node list
   ```

   ```
   [ INFO  ] List of nodes in current configuration:
   [ INFO  ] [Installer Node]
   [ INFO  ] 9.168.100.1
   [ INFO  ]
   [ INFO  ] [Cluster Name]
   [ INFO  ] gpfscluster01
   [ INFO  ]
   [ INFO  ] GPFS Node         Admin  Quorum  Manager  NSD Server  Protocol GUI Server  OS     Arch
   [ INFO  ] gpfsnode1           X       X                            X                 rhel7  x86_64
   [ INFO  ] gpfsnode2                                                X                 rhel7  x86_64
   [ INFO  ] gpfsnode3                   X       X         X                            rhel7  x86_64
   [ INFO  ] gpfsnode4                   X       X         X                            rhel7  x86_64
   ```

   Starting with IBM Spectrum Scale release 4.2.2, the output of the **spectrumscale node list** command also includes CPU architecture and operating system of the nodes.

   If you are upgrading to IBM Spectrum Scale release 4.2.2 or later, ensure that the operating system and architecture fields in the cluster definition file are updated. These fields are automatically updated during the upgrade precheck. You can also update the operating system and CPU architecture fields in the cluster definition file by issuing the **spectrumscale config update** command. For more information, see "Upgrading IBM Spectrum Scale components with the installation toolkit" on page 407.

- If you want to enable and configure call home by using the installation toolkit, you must add one of the nodes as the call home node as follows.

  ```
  ./spectrumscale node add -c CallHomeNode
  ```

- If you want to use the installation toolkit to install GPFS™ and deploy protocols in an Elastic Storage Server (ESS) cluster, you must add the EMS node of the ESS system as follows.

  ```
  ./spectrumscale node add -e EMSNode
  ```

  For more information, see "ESS awareness with the installation toolkit" on page 294.
- For information on adding nodes to an existing installation, see "Adding nodes, NSDs, or file systems to an existing installation" on page 308.

Adding and configuring NSD server nodes in the cluster definition file

Note: A CES shared root file system is required for protocols deployment with IBM Spectrum Scale.

1. To configure NSDs, you must first have added your NSD server nodes to the configuration:

   ```
   ./spectrumscale node add -n nsdserver1
   ./spectrumscale node add -n nsdserver2
   ```

2. Once NSD server nodes are in the configuration, you can add NSDs to the configuration.

   ```
   ./spectrumscale nsd add /dev/sdb -p nsdserver1 -s nsdserver2
   ```

   The installation toolkit supports standalone NSDs which connect to a single NSD server or shared NSDs which connect to both a primary and secondary NSD server.

   When adding a standalone NSD, skip the secondary NSD server parameter.

   When adding a shared NSD, it is important to know the device name on the node which is to become the primary NSD server. It is not necessary to know the device name on the secondary NSD server because the device will be looked up using its UUID.

   Note: Although it is not necessary to know the device name on the secondary NSD server, it may be helpful to create a consistent mapping of device names if you are using multipath. For more information, see "NSD disk discovery" on page 25.

   Here is an example of adding a shared NSD to the configuration by specifying the device name on the primary server along with the primary and secondary servers.

3. The name of the NSD will be automatically generated based on the NSD server names. This can be changed after the NSD has been added by using the modify command and supplying a new name to the **-n** flag; the new name must be unique:

   ```
   ./spectrumscale nsd modify nsd_old_name -n nsd_new_name
   ```

4. It is possible to view all NSDs currently in the configuration using the list command:

   ```
   ./spectrumscale nsd list
   ```

5. To remove a single NSD from the configuration, supply the name of the NSD to the delete command:

   ```
   ./spectrumscale nsd delete nsdserver1_nsdserver2_1
   ```

6. To clear all NSDs and start from scratch, use the clear command:

   ```
   ./spectrumscale nsd clear
   ```

7. Where multiple devices are connected to the same pair of NSD servers, they can be added in bulk either by providing a list of all devices, or by using wild cards:

   ```
   ./spectrumscale nsd add -p nsdserver1 -s nsdserver2 /dev/dm-1 /dev/dm-2 /dev/dm-3
   ```

   or

   ```
   ./spectrumscale nsd add -p nsdserver1 -s nsdserver2 "/dev/dm-*"
   ```

   A connection will be made to the primary server to expand any wild cards and check that all devices are present. When using wild cards, it is important to ensure that they are properly escaped, as otherwise they may be expanded locally by your shell. If any devices listed cannot be located on the primary server, a warning will be displayed, but the command will continue to add all other NSDs.

8. When adding NSDs, it is good practice to have them distributed such that each pair of NSD servers is equally loaded. This is usually done by using one server as a primary for half of the NSDs, and

the other server as primary for the remainder. To simplify this process, it is possible to add all NSDs at once, then later use the balance command to switch the primary and secondary servers on some of the NSDs, as required. A connection will be made to the original secondary server to look up device names on that node automatically. To automatically balance a pair of NSD servers, you must specify one of the nodes in that pair:

```
./spectrumscale nsd add "/dev/dm-*" -p serverA -s serverB
```

```
[ INFO  ] Connecting to serverA to check devices and expand wildcards.
[ INFO  ] Adding NSD serverA_serverB_1 on serverA using device /dev/dm-0.
[ INFO  ] Adding NSD serverA_serverB_2 on serverA using device /dev/dm-1.
$ ./spectrumscale nsd list
[ INFO  ] Name                       FS      Size(GB) Usage   FG Pool    Device    Servers
[ INFO  ] serverA_serverB_1 Default 13      Default 1  Default /dev/dm-0 serverA,serverB
[ INFO  ] serverA_serverB_2 Default 1       Default 1  Default /dev/dm-1 serverA,serverB
./spectrumscale nsd balance --node serverB
./spectrumscale nsd list
```

```
[ INFO  ] Name                       FS      Size(GB) Usage   FG Pool    Device    Servers
[ INFO  ] serverA_serverB_1 Default 13      Default 1  Default /dev/dm-0 serverB,serverA
[ INFO  ] serverA_serverB_2 Default 1       Default 1  Default /dev/dm-1 serverA,serverB
```

9. Ordinarily a connection will be made to the primary NSD server when adding an NSD. This is done to check device names and so that details such as the disk size can be determined, but is not vital. If it is not feasible to have a connection to the nodes while adding NSDs to the configuration, these connections can be disabled using the **--no-check** flag. Extra care is needed to manually check the configuration when using this flag.

   ```
   ./spectrumscale nsd add /dev/sda -p nsdserver1 -s nsdserver2 --no-check
   ```

10. You can set the failure group, file system, pool, and usage of an NSD in two ways:
    - using the add command to set them for multiple new NSDs at once
    - using the modify command to modify one NSD at a time

    ```
    ./spectrumscale nsd add "/dev/dm-*" -p nsdserver1 -s nsdserver2 \
    -po pool1 -u dataOnly -fg 1 -fs filesystem_1
    ./spectrumscale nsd modify nsdserver1_nsdserver2_1 -u metadataOnly -fs filesystem_1
    ```

For information on adding NSDs to an existing installation, see “Adding nodes, NSDs, or file systems to an existing installation” on page 308.

Defining file systems

File systems are defined implicitly with the NSD configuration and will only be created, at the time of deployment, if there are NSDs assigned to them.

Note: Be aware that if you do not configure file systems, the installation toolkit will automatically create a file system named `scale0` mounted at `/ibm/scale0/`.

Note: A CES shared root file system is required for protocols deployment with IBM Spectrum Scale.

1. To specify a file system, use the **`nsd add`** or **`nsd modify`** command to set the file system property of the NSD:

   ```
   ./spectrumscale nsd add "/dev/dm-*" -p server1 -s server2 -fs filesystem_1
   [ INFO  ] The installer will create the new file system filesystem_1 if it does not already exist.
   ```

   ```
   ./spectrumscale nsd modify server1_server2_1 -fs filesystem_2
   [ INFO  ] The installer will create the new file system filesystem_2 if it does not already exist.
   ```

2. To list all file systems that currently have NSDs assigned to them, use the **`list`** command. This will also display file system properties including the block size and mount point:

   ```
   ./spectrumscale filesystem list
   [ INFO  ] Name         BlockSize   Mountpoint           NSDs Assigned
   [ INFO  ] filesystem_1 Default     /ibm/filesystem_1     3
   [ INFO  ] filesystem_2 Default     /ibm/filesystem_2     1
   ```

3. To alter the block size and mount point from their default values, use the modify command:

```
./spectrumscale filesystem modify filesystem_1 -B 1M -m /gpfs/gpfs0
```

Important: NSDs will be created when the **./spectrumscale install** command is issued. The file system will be created when the **./spectrumscale deploy** command is issued.

It is not possible to directly rename or delete a file system; this is instead done by reassigning the NSDs to a different file system using the **nsd modify** command.

At this point, the cluster definition file file will have:

- Nodes and node types defined
- NSDs optionally defined
- File systems optionally defined

To proceed with the GPFS installation, go to the next step: "Installing GPFS and creating a GPFS cluster."

For information on adding file systems to an existing installation, see "Adding nodes, NSDs, or file systems to an existing installation" on page 308.

Installing GPFS and creating a GPFS cluster

After setting up your install node, you can use the installation toolkit to define nodes, NSDs, and file systems in the cluster definition file, and install GPFS according to the information in that file.

Steps for installing GPFS using the installation toolkit

After defining nodes in the cluster definition file (see "Defining the cluster topology for the installation toolkit" on page 273), you can set additional GPFS configuration information in that file. You use the **spectrumscale config gpfs** command to do this.

1. To specify a cluster name in the cluster definition, use the **-c** argument:

   ```
   ./spectrumscale config gpfs -c gpfscluster01.my.domain.name.com
   ```

   In this example, `gpfscluster01.my.domain.name.com` is the cluster name.

   If no cluster name is specified, the GPFS Admin Node name is used as the cluster name. If the user-provided name contains periods, it is assumed to be a fully qualified domain name. If the cluster name is not a fully qualified domain name, the cluster name domain name will be inherited from the Admin Node domain name.
2. To specify a profile to be set on cluster creation in the cluster definition use the **-p** argument:

   ```
   ./spectrumscale config gpfs -p randomio
   ```

   The valid values for -p option are default (for **gpfsProtocolDefaults** profile) and random I/O (for **gpfsProtocolRandomIO** profile). The profiles are based on workload type : sequential I/O (**gpfsProtocolDefaults**) or random I/O (**gpfsProtocolRandomIO**). The defined profiles above can be used to provide initial default tunables/settings for a cluster. If additional tunable changes are required, see **mmchconfig** command and the **mmcrcluster** command in *IBM Spectrum Scale: Command and Programming Reference*.

   If no profile is specified in the cluster definition, the **gpfsProtocolDefaults** profile will be automatically set on cluster creation.
3. To specify the remote shell binary to be used by GPFS, use the **-r** argument:

   ```
   ./spectrumscale config gpfs -r /usr/bin/ssh
   ```

   If no remote shell is specified in the cluster definition, `/usr/bin/ssh` will be used as default.
4. To specify the remote file copy binary to be used by GPFS use the **-rc** argument:

   ```
   ./spectrumscale config gpfs -rc /usr/bin/scp
   ```

   If no remote file copy binary is specified in the cluster definition, `/usr/bin/scp` will be used a default.
5. To specify an ephemeral port range to be set on all GPFS nodes use the **-e** argument:

   ```
   ./spectrumscale config gpfs -e 70000-80000
   ```

For information about the ephemeral port range, see *GPFS port usage* in *IBM Spectrum Scale: Administration Guide*.

If no port range is specified in the cluster definition, 60000-61000 will be used as default.

6. To view the current GPFS configuration settings, issue the following command:

   ./spectrumscale config gpfs --list

   ```
   [ INFO  ] No changes made. Current settings are as follows:
   [ INFO  ] GPFS cluster name is gpfscluster01
   [ INFO  ] GPFS profile is default
   [ INFO  ] Remote shell command is /usr/bin/ssh
   [ INFO  ] Remote file copy command is /usr/bin/scp
   [ INFO  ] GPFS Daemon communication port range is 60000-61000
   ```

To perform environment checks prior to running the install, use **./spectrumscale install** with the **-pr** argument:

```
./spectrumscale install -pr
```

This is not required, however, as **install** with no arguments will also run this.

Understanding what the installation toolkit does during a spectrumscale install

- If the installation toolkit is being used to install GPFS on all nodes, create a new GPFS cluster, and create NSDs, it will automatically perform the steps outlined below.
- If the installation toolkit is being used to add nodes to an existing GPFS cluster and/or create new NSDs, it will automatically perform the steps outlined below.
- If all nodes in the cluster definition file are in a cluster, then the installation toolkit will automatically perform the steps outlined below.

To add nodes to an existing GPFS cluster, at least one node in the cluster definition must belong to the cluster where the nodes not in a cluster are to be added. The GPFS cluster name in the cluster definition must also exactly match the cluster name outputted by **mmlscluster**.

Once the **spectrumscale install** command has been issued, the toolkit undergoes these flows:

To install GPFS on all nodes, create a new GPFS cluster, and create NSDs

- Run pre-install environment checks
- Install the GPFS packages on all nodes
- Build the GPFS portability layer on all nodes
- Install and configure performance monitoring tools
- Create a GPFS cluster
- Configure licenses
- Set ephemeral port range
- Create NSDs (if any are defined in the cluster definition)
- Run post-install environment checks

To add nodes to an existing GPFS cluster and create any new NSDs

- Run pre-install environment checks
- Install the GPFS packages on nodes to be added to the cluster
- Install and configure performance monitoring tools on nodes to be added to the cluster
- Add nodes to the GPFS cluster
- Configure licenses
- Create NSDs (if any new NSDs are defined in the cluster definition)
- Run post-install environment checks

If all nodes in the cluster definition are in a cluster

- Run pre-install environment checks
- Skip all steps until NSD creation
- Create NSDs (if any new NSDs are defined in the cluster definition)
- Run post-install environment checks

Although not part of GPFS, NTP configuration on every node is useful for cluster operation and future debugging. You can use the **./spectrumscale ntp** options for configuring NTP on all nodes at installation time. For example:

```
/usr/lpp/mmfs/5.0.0.0/installer/spectrumscale config ntp -e on -s 198.51.100.2,198.51.100.4
[ WARN ] The NTP package must already be installed and full bidirectional access to the UDP
port 123 must be allowed.
[ WARN ] If NTP is already running on any of your nodes, NTP setup will be skipped.
To stop NTP run 'service ntpd stop'.
[ INFO ] Setting NTP to on
[ INFO ] Setting Upstream NTP Servers to 198.51.100.2,198.51.100.4
```

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

What to do next

Upon completion of the installation, you will have an active GPFS cluster. Within the cluster, NSDs may have been created, performance monitoring will have been configured, and all product licenses will have been accepted. File systems will be fully created in the next step: deployment.

Install can be re-run in the future to:

- add NSD server nodes
- add GPFS client nodes
- add GUI nodes
- add NSDs
- define new file systems for use with deploy

Deploying protocols

Use this information to deploy protocols in an IBM Spectrum Scale cluster using the installation toolkit.

Deployment of protocol services is performed on a subset of the cluster nodes that have been designated as protocol nodes using the **./spectrumscale node add node_name -p** command. Protocol nodes have an additional set of packages installed that allow them to run the NFS, SMB and Object protocol services.

Data is served through these protocols from a pool of addresses designated as Export IP addresses or CES “public” IP addresses using **./spectrumscale config protocols -e IP1,IP2,IP3...** or added manually using the **mmces address add** command. The allocation of addresses in this pool is managed by the cluster, and IP addresses are automatically migrated to other available protocol nodes in the event of a node failure.

Before deploying protocols, there must be a GPFS cluster that has GPFS started and it has at least one file system for the CES shared root file system.

Notes: Here are a few considerations for deploying protocols.

1. All the protocol nodes must be running the supported operating systems, and the protocol nodes must be all Power (in big endian mode) or all Intel. Although the other nodes in the cluster could be on other platforms and operating systems.

   For information about supported operating systems for protocol nodes and their required minimum kernel levels, see IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html)

2. The packages for all protocols are installed on every node designated as a protocol node; this is done even if a service is not enabled in your configuration.
3. Services are enabled and disabled cluster wide; this means that every protocol node serves all enabled protocols.
4. If SMB is enabled, the number of protocol nodes is limited to 16 nodes.
5. If your protocol node has Red Hat Enterprise Linux 7.3 installed, there might be an NFS service already running on the node that can cause issues with the installation of IBM Spectrum Scale NFS packages. To avoid these issues, before starting the deployment, you must do the following:
   a. Stop the NFS service using the **systemctl stop nfs.service** command.
   b. Disable the NFS service using the **systemctl disable nfs.service** command.

      This command ensures that this change is persistent after system reboot.
6. The installation toolkit does not support adding protocol nodes to an existing ESS cluster prior to ESS version 3.5.

Deploying protocols involves doing the following sub-tasks.

1. "Defining a shared file system for protocols"
2. "Adding protocol nodes to the cluster definition file"
3. "Enabling NFS and SMB" on page 281
4. "Configuring object" on page 281
5. "Adding export IPs" on page 282
6. "Running the spectrumscale deploy command" on page 283

Defining a shared file system for protocols

To use protocol services, a shared file system (CES shared root file system) must be defined. If the installation toolkit is used to install GPFS, NSDs can be created at this time and, if associated with a file system, the file system is then be created during deployment. If GPFS has already been configured, the shared file system can be specified manually or by re-running the **spectrumscale install** command to assign an existing NSD to the file system. If re-running **spectrumscale install**, be sure that your NSD servers are compatible with the installation toolkit and contained within the `clusterdefinition.txt` file.

Note: Be aware that if you do not configure a protocol's shared file system, the installation toolkit automatically creates one for you named `ces_shared` mounted at `/ibm/ces_shared`. This works only if you have created at least one NSD and that NSD is not already assigned to a file system.

The **./spectrumscale config protocols** command can be used to define the shared file system (**-f**) and mount point (**-m**):

```
usage: spectrumscale config protocols [-h] [-l] [-f FILESYSTEM]
                                      [-m MOUNTPOINT]
```

For example: **./spectrumscale config protocols -f cesshared -m /gpfs/cesshared**.

To view the current settings, issue this command:

```
$ ./spectrumscale config protocols --list
```

```
[ INFO  ] No changes made. Current settings are as follows:
[ INFO  ] Shared File System Name is cesshared
[ INFO  ] Shared File System Mountpoint is /gpfs/cesshared
```

Adding protocol nodes to the cluster definition file

To deploy protocols on nodes in your cluster, they must be added to the cluster definition file as protocol nodes.

Issue the following command to designate a node as a protocol node:

```
./spectrumscale node add NODE_IP -p
```

Enabling NFS and SMB

To enable or disable a set of protocols, the **./spectrumscale enable** and **./spectrumscale disable** commands should be used. For example:

```
./spectrumscale enable smb nfs
```

```
[ INFO  ] Enabling SMB on all protocol nodes.
[ INFO  ] Enabling NFS on all protocol nodes.
```

The current list of enabled protocols is shown as part of the **spectrumscale node list** command output; for example:

```
./spectrumscale node list
```

```
[ INFO  ] List of nodes in current configuration:
[ INFO  ] [Installer Node]
[ INFO  ] 9.71.18.169
[ INFO  ]
[ INFO  ] [Cluster Name]
[ INFO  ] ESDev1
[ INFO  ]
[ INFO  ] [Protocols]
[ INFO  ] Object : Disabled
[ INFO  ] SMB : Enabled
[ INFO  ] NFS : Enabled
[ INFO  ]
[ INFO  ] GPFS Node              Admin  Quorum  Manager  NSD Server  Protocol  GUI Server  OS     Arch
[ INFO  ] ESDev1-GPFS1             X       X       X                    X                  rhel7  x86_64
[ INFO  ] ESDev1-GPFS2                             X                    X                  rhel7  x86_64
[ INFO  ] ESDev1-GPFS3                             X                    X                  rhel7  x86_64
[ INFO  ] ESDev1-GPFS4             X       X       X          X                            rhel7  x86_64
[ INFO  ] ESDev1-GPFS5             X       X       X          X                            rhel7  x86_64
```

Starting with IBM Spectrum Scale release 4.2.2, the output of the **spectrumscale node list** command includes CPU architecture and operating system of the nodes.

If you are upgrading to IBM Spectrum Scale release 4.2.2 or later, ensure that the operating system and architecture fields in the cluster definition file are updated. These fields are automatically updated during the upgrade precheck. You can also update the operating system and CPU architecture fields in the cluster definition file by issuing the **spectrumscale config update** command. For more information, see "Upgrading IBM Spectrum Scale components with the installation toolkit" on page 407.

Configuring object

If the object protocol is enabled, further protocol-specific configuration is required; these options are configured using the **spectrumscale config object** command, which has the following parameters:

```
usage: spectrumscale config object [-h] [-l] [-f FILESYSTEM] [-m MOUNTPOINT]
                                   [-e ENDPOINT] [-o OBJECTBASE]
                                   [-i INODEALLOCATION] [-t ADMINTOKEN]
                                   [-au ADMINUSER] [-ap ADMINPASSWORD]
                                   [-SU SWIFTUSER] [-sp SWIFTPASSWORD]
                                   [-dp DATABASEPASSWORD]
                                   [-mr MULTIREGION] [-rn REGIONNUMBER]
                                   [-s3 {on,off}]
```

The object protocol requires a dedicated fileset as its back-end storage; this fileset is defined using the **--filesystem** (**-f**), **--mountpoint**(**-m**) and **--objectbase** (**-o**) flags to define the file system, mount point, and fileset respectively.

The **--endpoint**(**-e**) option specifies the host name that is used for access to the file store. This should be a round-robin DNS entry that maps to all CES IP addresses; this distributes the load of all keystone and object traffic that is routed to this host name. Therefore, the endpoint is an IP address in a DNS or in a load balancer that maps to a group of export IPs (that is, CES IPs that were assigned on the protocol nodes) .

The following user name and password options specify the credentials used for the creation of an admin user within Keystone for object and container access. The system prompts for these during **`spectrumscale deploy`** pre-check and **`spectrumscale deploy`** if they have not already been configured by **`spectrumscale`**. The following example shows how to configure these options to associate user names and passwords:

```
./spectrumscale config object -au -admin -ap -dp
```

The **-ADMINUSER**(**-au**) option specifies the admin user name. This credential is for the Keystone administrator. This user can be local or on remote authentication server based on authentication type used.

The **-ADMINPASSWORD**(**-ap**) option specifies the password for the admin user.

Note: You are prompted to enter a Secret Encryption Key which is used to securely store the password. Choose a memorable pass phrase which you are prompted for each time you enter the password.

The **-SWIFTUSER**(**-su**) option specifies the Swift user name. The **-ADMINUSER**(**-au**) option specifies the admin user name. This credential is for the Swift services administrator. All Swift services are run in this user's context. This user can be local or on remote authentication server based on authentication type used.

The **-SWIFTPASSWORD**(**-sp**) option specifies the password for the Swift user.

Note: You are prompted to enter a Secret Encryption Key which is used to securely store the password. Choose a memorable pass phrase which you are prompted for each time you enter the password.

The **-DATABASEPASSWORD**(**-dp**) option specifies the password for the object database.

Note: You are prompted to enter a Secret Encryption Key which is used to securely store the password. Choose a memorable pass phrase which you are prompted for each time you enter the password

The **-MULTIREGION**(**-mr**) option enables the multi-region object deployment feature. The **-REGIONNUMBER**(**-rn**) option specifies the region number.

The **-s3** option specifies whether the S3 (Amazon Simple Storage Service) API should be enabled.

-t ADMINTOKEN sets the `admin_token` property in the `keystone.conf` file which allows access to Keystone by token value rather than user/password. When installing with a local Keystone, by default the installer dynamically creates the `admin_token` used during initial configuration and deletes it when done. If set explicitly with `-t`, `admin_token` is not deleted from `keystone.conf` when done. The admin token can also be used when setting up a remote Keystone server if that server has `admin_token` defined.

Attention: If SELinux is disabled during installation of IBM Spectrum Scale for object storage, enabling SELinux after installation is not supported.

Adding export IPs

Note: This is mandatory for protocol deployment.

Export IPs or CES "public" IPs are used to export data via the protocols (NFS, SMB, Object). File and Object clients use these public IPs to access data on GPFS file systems. Export IPs are shared between all protocols and are organized in a public IP pool (there can be fewer public IPs than protocol nodes).

Note: Export IPs must have an associated host name and reverse DNS lookup must be configured for each.

1. To add Export IPs to your cluster, issue either this command:

   ./spectrumscale config protocols --export-ip-pool *EXPORT_IP_POOL*

   Or this command:

   ./spectrumscale config protocols -e *EXPORT_IP_POOL*

   Within these commands, `EXPORT_IP_POOL` is a comma-separated list of IP addresses.
2. To view the current configuration, issue the following command:

   ./spectrumscale node list

   To view the CES shared root and the IP pool, issue the following command:

   ./spectrumscale config protocols -l

   To view the object configuration, run the following command:

   ./spectrumscale config object -l

Running the spectrumscale deploy command

After adding the previously-described protocol-related definition and configuration information to the cluster definition file you can deploy the protocols specified in that file.

To perform deploy checks prior to deploying, use the **spectrumscale deploy** command with the **--pr** argument:

```
./spectrumscale deploy --pr
```

This is not required, however, because **spectrumscale deploy** with no argument also runs this.

You can also use the **mmnetverify** command to identify any network problems before doing the deployment. For more information, see *mmnetverify command* in *IBM Spectrum Scale: Command and Programming Reference*.

Use the following command to deploy protocols:

```
spectrumscale deploy
```

Note: You are prompted for the Secret Encryption Key that you provided while configuring object and/or authentication unless you disabled prompting.

This does the following:

- Performs pre-deploy checks.
- Creates file systems and deploys protocols as specified in the cluster definition file.
- Performs post-deploy checks.

You can explicitly specify the **--precheck** (**-pr**) option to perform a dry run of pre-deploy checks without starting the deployment. Alternatively, you can specify the **--postcheck** (**-po**) option to perform a dry run of post-deploy checks without starting the deployment. These options are mutually exclusive.

After a successful deployment, you can verify the cluster and CES configuration by running this command:

```
$ /usr/lpp/mmfs/bin/mmlscluster --ces
```

What to do next

Upon completion of the tasks described in this topic, you have deployed additional functionality to an active GPFS cluster. This additional functionality may consist of file systems, protocol nodes, specific protocols, and authentication. Although authentication can be deployed at the same time as protocols, we have separated the instructions for conceptual purposes. Continue with the setting up authentication step if you have not yet set up authentication. See the topic "Setting up authentication" for instructions on how to set up authentication.

You can rerun the **spectrumscale deploy** command in the future to do the following:
- add file systems
- add protocol nodes
- enable additional protocols
- configure and enable authentication

Setting up authentication

Use these instructions to set up authentication for protocols by using the installation toolkit.

Setting authentication configuration settings in the cluster definition file

You need to set up authentication methods for both the file and object users. If the object protocol is enabled on the protocol nodes, the installer automatically sets up the default local object authentication for object access. If you plan to configure object authentication with an external Keystone server and you are using the installation toolkit, do not configure external Keystone with the installation toolkit. For more information, see *Configuring object authentication with external Keystone server* in *IBM Spectrum Scale: Administration Guide*. To make any more changes for authentication, issue the **spectrumscale auth** command as shown in the following example:

```
./spectrumscale auth -h
usage: spectrumscale auth [-h] {commitsettings,file,object} ...
```

If the cluster has two separate servers that control different node groups, run this command separately for object and file. Run the **spectrumscale auth** command with the data access method that you want to use, either file or object, and the authentication type.

Authentication prerequisites:

There are a few additional prerequisites needed if you wish to configure authentication.

The following packages must be installed on all protocol nodes before running **./spectrumscale deploy**

If Object authentication is required:
- `openldap-clients`

If file authentication is required:
- `sssd`
- `ypbind`
- `openldap-clients`

To set up object authentication by using the installation toolkit:

Note: If you plan to configure object authentication with an external Keystone server and you are using the installation toolkit, do not configure external Keystone with the installation toolkit. For more information, see *Configuring object authentication with external Keystone server* in *IBM Spectrum Scale: Administration Guide.*

Object **auth** has two extra options to enable https or to enable pki. If you wish to set up either of these, you can include them in the command and you will be prompted in the next step to give the paths to the certificates required.

1. To set up object authentication, run this command:

   ```
   ./spectrumscale auth object [-h] [--https] [--pki] {local,external,ldap,ad}
   ```

   This will automatically open a template file for you to fill with the required **auth** settings. For more information about these settings, see *mmuserauth command* in *IBM Spectrum Scale: Command and Programming Reference*.
2. Save the file and close it, and the settings will automatically be loaded for the installer to set up object authentication after protocols have been enabled.

   If this **auth** command has been run, authentication will automatically be enabled by the installer.

   Note: Using unusual characters or white space in settings will require you to enter the setting in single quotes (' '). For example:

   ```
   unixmap_domains = 'testdomain(1000-3000)'
   bind_username = 'My User'
   ```
3. If required, configure file authentication by following the steps provided in the next section.
4. Issue the **./spectrumscale deploy -pr** command to initiate a pre-check to make sure that the cluster is in a good state for deployment.
5. After the successful pre-check, issue the **./spectrumscale deploy** command to deploy the new authentication configuration.

To set up file authentication by using the installation toolkit:

1. To set up file authentication, run this command:

   ```
   ./spectrumscale auth file [-h] {ldap,ad,nis,none}
   ```

   This will automatically open a template file for you to fill with the required authentication settings. For more information about these settings, see *mmuserauth command* in *IBM Spectrum Scale: Command and Programming Reference*.
2. Save the file and close it; the settings will automatically be loaded for the installer to set up file authentication after protocols have been enabled.

   Note: Using unusual characters or white space in settings will require you to enter the setting in single quotes (' '). For example:

   ```
   unixmap_domains = 'testdomain(1000-3000)'
   bind_username = 'My User'
   ```
3. If required, configure object authentication by following the steps that are explained in the previous section.
4. Issue the **./spectrumscale deploy -pr** command to initiate a pre-check to make sure that the cluster is in a good state for deployment.
5. After the successful pre-check, issue the **./spectrumscale deploy** command to deploy the new authentication configuration.

To clear authentication settings listed in the install toolkit:

To clear authentication settings in the install toolkit, run this command:

```
./spectrumscale auth clear
```

This does not clear or change a live and running authentication configuration. The **./spectrumscale auth clear** command just clears the authentication settings from the `clusterdefinition.txt` file that is used by the installation toolkit during the deployment.

Note: If the installation toolkit is used to set up object support and file support (especially SMB) with AD or LDAP Authentication, the authentication setup might cause a temporary monitoring failure and trigger an IP failover. This might lead to an error message similar to the following when configuring object : `"mmcesobjcrbase: No CES IP addresses are assigned to this node."`

If the **spectrumscale** installer failed because of this problem, do the following:

1. Check the cluster state by running the **`mmlscluster --ces`** command, and wait till the failed state of all nodes is cleared (`flags=none`).
2. Rebalance the IP addresses by issuing this command: **`mmces address move --rebalance`**
3. Rerun the installation toolkit to complete the object setup.

Installation of performance monitoring tool using the installation toolkit

By default, the performance monitoring tool gets installed when IBM Spectrum Scale is installed on the system using the **./spectrumscale install** command.

For more information on the performance monitoring tool, see *Performance Monitoring tool overview* in *IBM Spectrum Scale: Problem Determination Guide*.

The following configuration occurs on the system after the installation:

- Sensors are installed and started on every node.
- The collector is installed and configured on two nodes in all cases apart from the following:
  - One node if there is a one-node cluster.
  - Three nodes when there are three GUI servers specified in the cluster definition.
- The proxies for the installed protocols are started. In the case of NFS and SMB, the proxy is installed with the code. In the case of object, the proxy is installed as a separate step as the final part of the installation.

During the installation toolkit prechecks if it is found that the collector nodes require reconfiguring then the **`./spectrumscale config perfmon -r on`** command needs to be run to enable the installation toolkit to perform this reconfiguration.

Note: A reconfiguration results in the removal of an existing collector and it might move the collector to different nodes and reset sensor data. Custom sensors and data might be erased. To disable reconfiguration, run the **`./spectrumscale config perfmon -r off`** command.

If reconfiguration is disabled then the installation toolkit performs all installation tasks except for those related to performance monitoring and the management GUI, if GUI servers are specified.

For information on how to install the performance monitoring tool manually, see “Manually installing the Performance Monitoring tool” on page 250.

For information on configuring the performance monitoring tool, see *Configuring the Performance Monitoring tool* in *IBM Spectrum Scale: Problem Determination Guide*.

Logging and debugging for installation toolkit

The installation toolkit reports progress to the console, and certain options also log output to a file.

Console output

The install toolkit reports progress to the console. By default only information level messages are displayed. If more verbose output is required the **-v** flag can be passed to the **spectrumscale** command.

Note: this flag must be the first argument passed to the **spectrumscale** command; for example:

```
./spectrumscale -v install --precheck
```

Log files

In addition to console output, the **install** and **deploy** functions of the installation toolkit will log verbose output to the `logs` subdirectory of the toolkit (the full log path will be reported to the console when logging begins). The log file will always contain verbose output even if the **-v** flag is not specified on the command line.

Install and deploy checks

The **install** and **deploy** functions each have a set of checks performed before and after their main action. These checks can be triggered separately by using either the **--precheck** or **--postcheck** flag.

It is safe to run the pre- and post-checks multiple times, as they will interrogate the cluster but will not make any changes.

Note that, as with the full **install** and **deploy** functions, the pre- and post-checks will create a detailed log file that can be used to provide further detail about errors encountered.

Gathering support information

If an unresolvable error is encountered when using the toolkit, a support package can be gathered using the **installer.snap.py** script that is available in the same directory as the installation toolkit.

Note: The **installer.snap.py** script must be run from the toolkit directory.

The **installer.snap.py** script gathers information from each of the nodes defined in the configuration and packages it such that it is ready for submission to the IBM Support Center. It creates a tar file with the prefix `installer_snap` in the `/tmp` directory.

The script gathers the following information:

- Cluster configuration file
- Install toolkit logs
- Information from the local node of the protocol services
- A GPFS snap file from each node defined in the configuration
- Basic environment information from the local node including:
  - Current OS and kernel version
  - System message log (`dmesg`)
  - File system usage
  - Network configuration
  - Current processes

Collection of core dump data

For information about changing configuration to enable collection of core dump data from protocol nodes, see *Configuration changes required on protocol nodes to collect core dump data* in *IBM Spectrum Scale: Problem Determination Guide*.

Enabling and configuring file audit logging using the installation toolkit

You can use the installation toolkit to enable and configure the file audit logging function in the cluster definition file. After enabling this function at the cluster level, you must enable it on file systems.

Based on the file audit logging configuration options specified in the cluster definition file using the installation toolkit, the function is enabled and configured in the cluster accordingly during the deployment.

For information on required packages for file audit logging, see “Requirements and limitations for file audit logging” on page 361 and “Installation prerequisites” on page 224.

Prerequisites for file audit logging include:

- A minimum of three nodes are required to enable file audit logging. If you have fewer than three protocol nodes, or you want to specify a different set of nodes to use for file audit logging that are not protocol nodes, you must perform the following step after the installation completes but before initiating the deployment procedure.

  Important: This step appears differently in the installation toolkit. Disregard that version and perform the following step instead:

  ```
  mmmsgqueue enable -N NodeList
  ```

  If you have three or more protocol nodes, this step is not required.
- A file system must be specified in the cluster definition file before you can enable file audit logging.

You can configure the file audit logging related options in the cluster definition file by using the installation toolkit as follows.

By default, file audit logging is disabled in the cluster definition file.

- To enable file audit logging in the cluster definition file, issue the following command before doing installation or deployment with the installation toolkit:

  ```
  ./spectrumscale fileauditlogging enable
  ```

  After enabling, file audit logging packages (`gpfs.kafka` and `gpfs.librdkafka`) are installed on all nodes in the cluster specified to the installation toolkit during the installation.
- To disable file audit logging in the cluster definition file, issue the following command:

  ```
  ./spectrumscale fileauditlogging disable
  ```
- To list the file audit logging configuration in the cluster definition file, issue the following command:

  ```
  ./spectrumscale fileauditlogging list
  ```

You can verify whether file audit logging is enabled in the cluster definition file by viewing the output of the **./spectrumscale node list** command:

```
[ INFO  ] List of nodes in current configuration:
[ INFO  ] [Installer Node]
[ INFO  ] 198.51.100.15
[ INFO  ] Setup Type: SpectrumScale
[ INFO  ]
[ INFO  ] [Cluster Name]
[ INFO  ] ESDev1
[ INFO  ]
[ INFO  ] [Protocols]
[ INFO  ] Object : Disabled
[ INFO  ] SMB : Enabled
[ INFO  ] NFS : Enabled
[ INFO  ]
[ INFO  ] File Audit logging : Disabled
[ INFO  ]
[ INFO  ] GPFS Node              Admin  Quorum  Manager  NSD Server  Protocol  GUI Server  OS     Arch
[ INFO  ] ESDev1-GPFS1             X       X       X                    X                  rhel7  x86_64
[ INFO  ] ESDev1-GPFS2                             X                    X                  rhel7  x86_64
[ INFO  ] ESDev1-GPFS3                             X                    X                  rhel7  x86_64
[ INFO  ] ESDev1-GPFS4             X       X       X          X                            rhel7  x86_64
[ INFO  ] ESDev1-GPFS5             X       X       X          X                            rhel7  x86_64
```

After enabling the file audit logging function in the cluster definition file, you must enable it on file systems on which you want to enable file audit logging.

- To enable file audit logging on a file system in the cluster definition file, issue the following command:

```
./spectrumscale filesystem modify --fileauditloggingenable FileSystemName
```

You can also specify the retention period and log fileset name with this command. For example, to specify a retention period of 180 days and to specify the log fileset name `testlog`, issue the following command:

```
./spectrumscale filesystem modify --fileauditloggingenable --retention 180 --logfileset testlog FileSystemName
```

- To disable file audit logging on a file system in the cluster definition file, issue the following command:

```
./spectrumscale filesystem modify --fileauditloggingdisable FileSystemName
```

Note: These file audit logging configuration-related changes become effective after the deployment procedure, initiated with **`./spectrumscale deploy`**, is completed.

Related concepts:

"Manually installing file audit logging" on page 362
Use this information to manually install file audit logging on your system.

Enabling and configuring call home using the installation toolkit

You can use the installation toolkit to enable and configure the call home function in the cluster definition file.

Based on the call home configuration options specified in the cluster definition file using the installation toolkit, the call home function is enabled and configured in the cluster accordingly during the installation. By default, the call home function is enabled and the call home packages (`gpfs.callhome` and `gpfs.callhome-ecc-client`) are installed on all nodes in the cluster. If you plan to use the call home function, you must configure call home with the mandatory parameters by using the installation toolkit. For more information, see this command example.

The installation toolkit does not support all operating systems and platforms supported by call home. The installation toolkit makes the following exceptions:

- The installation toolkit does not support Linux on Z.
- The installation toolkit only supports SLES and Ubuntu on x86, while call home also supports these operating systems on Power.

In an IBM Spectrum Scale cluster, a call home setup comprises a call home group that consists of a call home node and multiple call home child nodes. The call home node collects data from the call home child nodes and uploads it to the IBM Support Center. Multiple call home groups can be configured within a cluster. The data gathering and upload can be configured individually on each group. You can use the groups to reflect logical units in the cluster. For example, you can create a group for all protocol nodes and another group for all non-protocol nodes. Call home groups help in distributing the data-gather and data-upload workload to prevent bottlenecks. You can create groups of any size between one and the number of nodes in you cluster. The larger the group is the higher is the workload on the call home node. It is recommended to limit the group size to 32 nodes. Larger groups are also possible but it might result in performance issues.

The call home node needs to have network connectivity to the IBM Support Center because the call home node needs to upload data to IBM Support Center for all the child nodes in the call home group. The call home node must be able to reach `https://esupport.ibm.com` either directly or using the proxy, which is specified in the call home configuration.

For more information about the call home function, see *Monitoring the IBM Spectrum Scale system by using call home* in *IBM Spectrum Scale: Problem Determination Guide*.

You can specify the call home node using the installation toolkit by issuing the following command:

```
./spectrumscale node add -c CallHomeNode
```

Note: This step is not mandatory. If a call home node is not specified, the installation toolkit assigns one of the nodes in the cluster as the call home node.
You can specify more than one node with this command if you want to configure multiple call home groups.

To configure call home options in the cluster definition file by using the installation toolkit, use the **./spectrumscale callhome** command as follows.

- To enable the call home in the cluster definition file by using the installation toolkit, issue the following command.

  ```
  ./spectrumscale callhome enable
  ```

  A sample output is as follows.

  ```
  [ INFO  ] Enabling the callhome.
  [ INFO  ] Configuration updated.
  ```

  Note: By default, call home is listed as enabled in the cluster definition file.
- To disable the call home in the cluster definition file by using the installation toolkit, issue the following command.

  ```
  ./spectrumscale callhome disable
  ```

  A sample output is as follows.

  ```
  [ INFO  ] Disabling the callhome.
  [ INFO  ] Configuration updated.
  ```

  The call home function is enabled by default in the cluster definition file. If you disable it in the cluster definition file, when you issue the **./spectrumscale install** command, the call home packages are installed on the nodes but no configuration is done.
- To specify the call home configuration settings in the cluster definition file by using the installation toolkit, issue the **./spectrumscale callhome config** command. The following command example shows configuring the mandatory call home parameters:

  ```
  ./spectrumscale callhome config -n username -i 456123 -e username@example.com -cn US
  ```

  A sample output is as follows.

  ```
  [ INFO  ] Setting Customer name to username
  [ INFO  ] Setting Customer id to 456123
  [ INFO  ] Setting Customer email to username@example.com
  [ INFO  ] Setting Customer country to US
  ```

  You are then prompted to accept or decline the support information collection message.

  ```
  By accepting this request, you agree to allow IBM and its subsidiaries to store and use
  your contact information and your support information anywhere they do business worldwide. For more
  information, please refer to the Program license agreement and documentation. If you agree, please
  respond with 'accept' for acceptance, else with 'not accepted' to decline: accept
  ```

  If you accept, the following output is displayed.

  ```
  [ INFO  ] License is accepted so the call home will be configured, if it is enabled.
  [ INFO  ] Configuration is updated.
  ```

  Note: You can specify the `-a` parameter with **./spectrumscale callhome config** to accept the support information collection message in advance.
- To clear the call home configuration specified in the cluster definition file by using the installation toolkit, issue the **./spectrumscale callhome clear** command. For example:

  ```
  ./spectrumscale callhome clear -i -e -cn
  ```

  A sample output is as follows.

```
[ INFO  ] Clearing Customer id
[ INFO  ] Clearing Customer email
[ INFO  ] Clearing Customer country
```

- To schedule the call home data collection in the cluster definition file by using the installation toolkit, issue the **./spectrumscale callhome schedule** command. For example:

  ```
  ./spectrumscale callhome schedule -d
  ```

  A sample output is as follows.

  ```
  [ INFO  ] Setting scheduling to daily.
  [ INFO  ] Configuration is updated.
  ```

  If call home data collection is scheduled daily, data uploads are by default executed at 02:xx AM each day. If call home data collection is scheduled weekly, data uploads are by default executed at 03:xx AM each Sunday. In both cases, xx is a random number from 00 to 59.

- To list the call home configuration settings in the cluster definition file by using the installation toolkit, issue the following command.

  ```
  ./spectrumscale callhome list
  ```

  A sample output is as follows.

  ```
  [ INFO  ] Current settings are as follows:
  [ INFO  ] Callhome is enabled.
  [ INFO  ] Schedule is daily.
  [ INFO  ] Setting Customer name to username
  [ INFO  ] Setting Customer id to 456123
  [ INFO  ] Setting Customer email to username@example.com
  [ INFO  ] Setting Customer country to US
  [ WARN  ] No value for Proxy ip in clusterdefinition file.
  [ WARN  ] No value for Proxy port in clusterdefinition file.
  [ WARN  ] No value for Proxy user in clusterdefinition file.
  [ WARN  ] No value for Proxy password in clusterdefinition file.
  ```

Note: These call home configuration-related changes become effective after the installation procedure, initiated with **./spectrumscale install**, is completed.

For detailed information on the **./spectrumscale callhome** command options, see **spectrumscale command** in *IBM Spectrum Scale: Command and Programming Reference*.

Installing IBM Spectrum Scale management GUI by using the installation toolkit

Use the following information for installing the IBM Spectrum Scale management GUI using the installation toolkit.

The IBM Spectrum Scale is only installed when nodes have been specified as GUI servers in the cluster definition with the -g option:

```
./spectrumscale node add gpfsnode3 -g
```

The IBM Spectrum Scale GUI requires passwordless ssh to all other nodes in the cluster. Therefore, the GUI server node must be added as an admin node using the -a flag:

```
./spectrumscale node add gpfsnode3 -a
```

If no nodes have been specified as GUI servers, then the GUI will not be installed. It is recommended to have at least 2 GUI interface servers and a maximum of 3 for redundancy.

The GUI will be installed on specified GUI servers when you run the **./spectrumscale install** command and on protocol nodes also specified as GUI servers during the **./spectrumscale deploy** command.

At the end of a successful GPFS installation or protocol deployment, you can access the GUI through a web browser with the following node address: `https://<GUI server IP or hostname>`.

Note: The default user name and password to access the IBM Spectrum Scale management GUI is `admin` and `admin001` respectively. Due to security reasons, it is recommended to change the default password after the first login.

Populating cluster definition file with current cluster state using the installation toolkit

You can use the installation toolkit to populate the cluster definition file with the current cluster state before proceeding with upgrade.

Before using this functionality, review its limitations. For more information, see "Limitations of config populate option of the installation toolkit."

In the following scenarios, you might need to update the cluster definition file with the current cluster state:

- A manually created cluster in which you want to use the installation toolkit
- A cluster created using the installation toolkit in which manual changes were done

Use the installation toolkit to populate the cluster definition file as follows.

1. Define the installer node by issuing the **./spectrumscale setup -s** *InstallNode* command.
2. Repopulate the cluster definition file with the current cluster state by issuing the **./spectrumscale config populate --node** *Node* command.

   Note: This command creates the backup of the existing cluster definition file and it creates a new cluster definition file from scratch.

   In a cluster containing ESS, you must specify the EMS node with the `config populate` command. For example:

   ```
   ./spectrumscale config populate --node EMSNode
   ```

In case you are performing an upgrade, proceed with the rest of the upgrade steps. For more information, see "Upgrading IBM Spectrum Scale components with the installation toolkit" on page 407.

Limitations of config populate option of the installation toolkit

You can use the **config populate** option of the installation toolkit to populate the cluster definition file with the current cluster state.

This functionality has the following limitations.

Table 42. Limitations of the config populate functionality

Function	Description
Call home	The config populate functionality does not support retrieving call home configuration information. If call home is enabled in the cluster, use the **./spectrumscale callhome enable**, **./spectrumscale callhome config**, and **./spectrumscale callhome schedule** commands to update the information in the cluster definition file.
CES Groups	The config populate functionality does not detect CES groups. This might cause issues with CES stretch clusters
Clusters larger than 16 nodes	The config populate functionality does not support more than 16 nodes. For more information, see "Limitations of the installation toolkit" on page 263.

Table 42. Limitations of the config populate functionality (continued)

Function	**Description**
Custom GPFS profiles	The config populate functionality does not detect custom profiles. This might not be an issue in most scenarios except when you plan to use the config populate functionality to back up a cluster's configuration for a future new installation.
File audit logging	The config populate functionality does not support retrieving file audit logging configuration information. If file audit logging is enabled in the cluster, use the **`./spectrumscale fileauditlogging enable`** and **`./spectrumscale filesystem modify --fileauditloggingenable`** *FileSystemName* commands to update the information in the cluster definition file.
FPO	The config populate functionality does not detect FPO NSD stanza information. However, the existing FPO NSD stanza information is not affected.
Greater than 3 GUI nodes	The config populate functionality adds in GUI nodes properly if the number of GUI nodes is less than or equal to 3. However, if there are more than 3 GUI nodes, the config populate functionality does not add them. This might leave you with GUI nodes that are not at the latest level of code that have to be manually upgraded, and other unexpected side effects.
Greater than 3 performance monitoring collector nodes	The config populate functionality does not support more than 3 performance monitoring collector nodes. If GUI nodes are detected, they are also assigned as the performance monitoring collector nodes.
Highly-available write cache (HAWC)	The config populate functionality does not detect HAWC information. However, the existing HAWC information is not affected.
Heterogeneous cluster	The config populate functionality cannot be used in a cluster that contains nodes with mixed CPU architectures or endianness. The installation toolkit can still be used in these type of clusters, but only on the subset of nodes that have the same CPU architecture and endianness.
IBM Spectrum Protect for Space Management and IBM Spectrum Archive Enterprise Edition (EE)	The config populate functionality does not detect IBM Spectrum Protect for Space Management and IBM Spectrum Archive Enterprise Edition (EE) information. The installation toolkit provides some DMAPI detection and gives recommendations on how to proceed in case issues are encountered. For example: DMAPI holds an FS open for one of these functions, thereby causing upgrades to fail. In this case, the installation toolkit shows a hint that identifies the node and that it might need to be rebooted to proceed with the upgrade.

Table 42. Limitations of the config populate functionality (continued)

Function	Description
IBM Spectrum Scale release 4.2.0.2	In the IBM Spectrum Scale release 4.2.0.2, although the config populate functionality can be used to populate the cluster definition file with the current cluster state, not all parameters are properly updated. For example, CES IPs are not properly detected and you might need to add them manually by using the following command: **./spectrumscale config protocols -e***ces_ip1*,*ces_ip2*,.... Review your cluster configuration after running **./spectrumscale config populate** to ensure that all desired values are populated.
Local read-only cache (LROC)	The config populate functionality does not detect LROC information. However, the existing LROC information is not affected.
Non-federated performance monitoring setup	The config populate functionality does not support non-federated performance monitoring setup. The installation toolkit converts all performance monitoring configuration to federated mode unless specified to ignore performance monitoring completely.
NSD SAN attachment	The config populate functionality does not support retrieval of the NSD information in case of an NSD SAN attachment.
Transparent cloud tiering	The config populate functionality does not detect Transparent cloud tiering related information.
Windows, AIX, Z series	The config populate functionality does not support these operating systems. If any of these operating systems are detected on a node and if that node has the same CPU architecture and endianness as the node from which the installation toolkit is running, the config populate functionality shows a warning, skips the node, and continues populating information from other supported nodes.

Related concepts:

"Limitations of the installation toolkit" on page 263
Before using the installation toolkit to install IBM Spectrum Scale and deploy protocols, review the following limitations and workrounds, if any.

ESS awareness with the installation toolkit

You can use the installation toolkit to install IBM Spectrum Scale and deploy protocols in a cluster containing Elastic Storage Server (ESS).

When using the installation toolkit in a cluster containing ESS, use the following high-level steps:

1. Add protocol nodes in the ESS cluster by issuing the following command.

   ```
   ./spectrumscale node add NodeName -p
   ```

   You can add other types of nodes such as client nodes, NSD servers, and so on depending on your requirements. For more information, see "Defining the cluster topology for the installation toolkit" on page 273.
2. Specify one of the newly added protocol nodes as the installer node and specify the setup type by issuing the following command.

   ```
   ./spectrumscale setup -s NodeIP -i SSHIdentity -st ess
   ```

The installer node is the node on which the installation toolkit is extracted and from where the installation toolkit command, **spectrumscale**, is initiated.

3. Specify the EMS node of the ESS system to the installation toolkit by issuing the following command.

   ```
   ./spectrumscale node add NodeName -e
   ```

   This node is also automatically specified as the admin node. The admin node, which must be the EMS node in an ESS configuration, is the node that has access to all other nodes to perform configuration during the installation.

   You can use the `config populate` option of the installation toolkit to populate the cluster definition file with the current configuration of the EMS node, and the protocol and the client nodes in the cluster. In a cluster containing ESS, you must specify the EMS node with the `config populate` command. For example:

   ```
   ./spectrumscale config populate --node EMSNode
   ```

   For more information, see "Populating cluster definition file with current cluster state using the installation toolkit" on page 292.

4. Proceed with specifying other configuration options, installing, and deploying by using the installation toolkit. For more information, see "Defining the cluster topology for the installation toolkit" on page 273, "Installing GPFS and creating a GPFS cluster" on page 277, and "Deploying protocols" on page 279.

Several validations are done when you use the installation toolkit in an ESS cluster to ensure that functions that are configured on ESS EMS and I/O server nodes are not affected. Additionally, some considerations apply when you are using the installation toolkit in an ESS cluster. These validations and considerations are as follows.

- Adding ESS EMS node and I/O server nodes as protocol nodes is not allowed.
- Adding GUI nodes in the ESS cluster by using the installation toolkit is not allowed. It is assumed that the EMS node is running a GUI setup and the ability to add more GUI nodes by using the installation toolkit is disabled to not affect the existing configuration. During upgrade, the installation toolkit does not alter an existing configuration that contains one or more GUI setups. The upgrade of any of these components is done outside of the ESS EMS and I/O server nodes, and their configuration is not changed.
- Adding I/O server nodes as NSD nodes by using the installation toolkit is not allowed.
- Creating NSDs and file systems from disks within the ESS I/O server nodes by using the installation toolkit is not allowed.
- Specifying a cluster name by using the installation toolkit is not allowed. The existing cluster name is detected and used by the toolkit.
- CES shared root file system must be mounted on all protocol nodes.
- The installation toolkit installs a Chef RPM on the ESS nodes but does no other configuration changes. The Chef RPM can be safely removed, if needed.
- Only performance monitoring sensor packages are installed on all nodes in an ESS cluster other than the ESS EMS and I/O server nodes. They are configured to point to the existing collector on the EMS node. Apart from package installation, the installation toolkit adds protocol-specific sensors to the overall cluster performance monitoring configuration during the installation and deployment phases. During upgrade, the installation toolkit does not alter an existing configuration that might have one or more collectors or sensors. The upgrade of any of these components is done outside of the ESS EMS and I/O server nodes, and their configuration is not changed.

Related concepts:

"Using the installation toolkit to perform installation tasks: Explanations and examples" on page 272
Use these explanations and examples of installation toolkit options and tasks to perform installation and deployment by using the installation toolkit.

Configuration of an IBM Spectrum Scale stretch cluster in an export services environment: a sample use case

This page describes a stretch cluster with NFS, SMB, and Object protocols that are enabled, installed, and deployed using the installation toolkit.

Overview of the stretch cluster use case

A single GPFS cluster is defined over three geographically separate sites: two production sites and a tiebreaker site. One or more file systems are created, mounted, and accessed concurrently from the two active production sites that are connected over a reliable WAN network.

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.

With two copies of the data in separate locations, if one site has an unrecoverable disaster, you can recover from a single site with no data loss. Data from two separate sites can share namespace and be accessed by either site. CES groups are enabled to control traffic to the local site. For more information, see *Synchronous mirroring with GPFS replication* in *IBM Spectrum Scale: Administration Guide*.

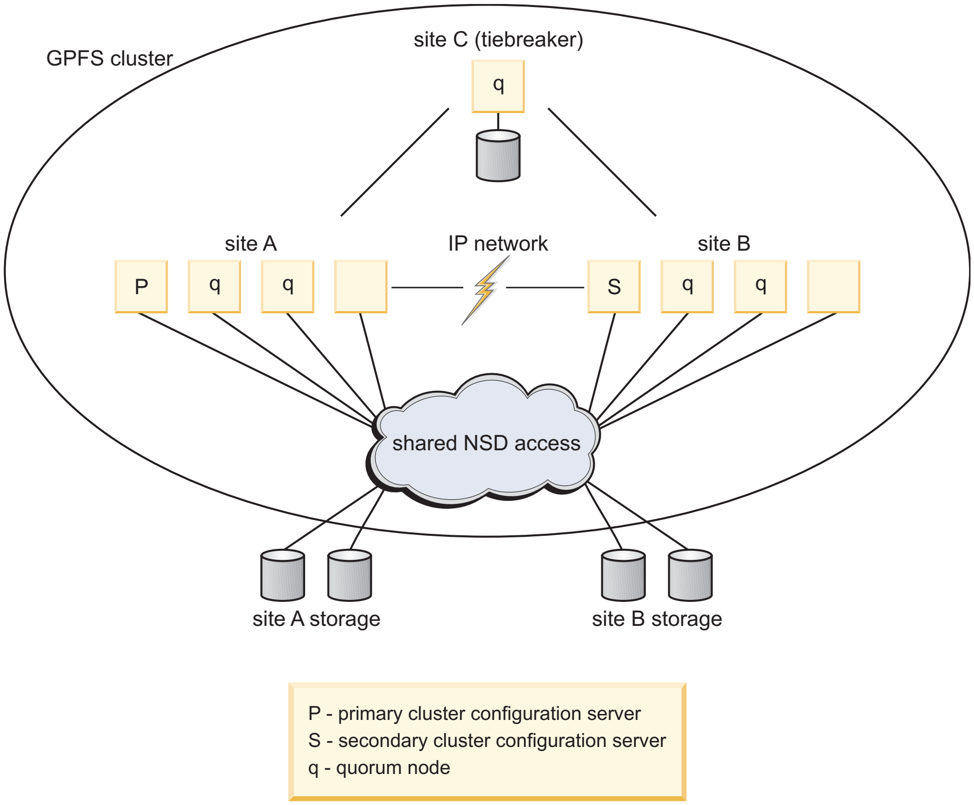


Figure 34. Synchronous mirroring with GPFS replication

About the tested use case

The stretch cluster in this use case was configured as active-active, meaning that clients can read and write data from either site of the cluster. For more information, see *Synchronous mirroring with GPFS replication* in *IBM Spectrum Scale: Administration Guide*. You can also configure an active-passive stretch cluster. For more information, see *An active-passive cluster* in *IBM Spectrum Scale: Administration Guide*. You can replicate data with GPFS in two ways: synchronous replication and asynchronous replication with Active File Management (AFM). Because there are some differences between the two options, you need to understand both options in order to choose the solution that best fits your use case.

Synchronous replication

Synchronous replication ensures that:

- your data is always available
- you can read and write in both locations
- you do not have to perform recovery actions from applications except for changing the IP address/hostname.

Because data is synchronously replicated, the application gets consistent data and no data is lost during failover or failback. Synchronous replication requires a reliable high-speed, low latency network between sites, and it has a performance impact on the application when the failback occurs.

Asynchronous replication with AFM

Asynchronous replication with AFM can work on a high latency network. Because data is asynchronously replicated, all updates might not be replicated when a failure occurs. Therefore, the application needs to be able to tolerate data loss and to run recovery operations to fail back. This setup is usually two separate GPFS clusters instead of one cluster at multiple geographical sites.

This use case covers the synchronous replication solution only. The use case shows how in an active-active stretch cluster when one site's storage is offline, you can still read from and write to a replica of the data to the site that is online.

Limitations of a stretch cluster that uses GPFS synchronous replication

Consider the following limitations of a stretch cluster with NFS, SMB, or Object for synchronously replicating data:

1. The IBM Spectrum Scale installation toolkit cannot deploy protocols if the CES networks across the two sites cannot communicate. For more information, see *Limitations* in *IBM Spectrum Scale: Big Data and Analytics Guide*.
2. If the Object protocol and the CES networks are separate and cannot communicate across the two sites, then object can use only one site to read and write data. For guidance on setup, refer to *Configuration of object for isolated node and network groups* in *IBM Spectrum Scale: Administration Guide*.
3. If your implementation requires you to set up Spectrum Scale IBM Spectrum Scale for object to one site, you will not have a seamless failover if you lose all of the protocol nodes on that site. You need to change the object ring configuration so that it points back to the CES Group that is available on the other site. For details, see *Configuration of object for isolated node and network groups* in *IBM Spectrum Scale: Administration Guide*.
4. When you have object enabled on one site and that entire site goes down unexpectedly, you might have to recover your endpoints manually since you can no longer ping them. In this case, refer to the steps provided in the Openstack Install Guide under Keystone Services https://docs.openstack.org/mitaka/install-guide-rdo/keystone-services.html and Keystone Install https://docs.openstack.org/mitaka/install-guide-rdo/keystone-install.html.

Note: A deployment has a high chance of failure if the CES networks at each site cannot communicate with each other. For more information, see *Limitations of the spectrumscale toolkit* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*. For this use case, the cluster was deployed with protocols with separate VLAN'd networks on each site; however, those networks are able to communicate with each other.

Using the spectrumscale installation toolkit to install a stretch cluster

When you set up a stretch cluster, it is important to understand the physical setup of the storage and how it maps from each site to each file system and failure group. Figure 2 shows the tested configuration where each site (A and B) has storage that is only seen by the NSD servers in that site.

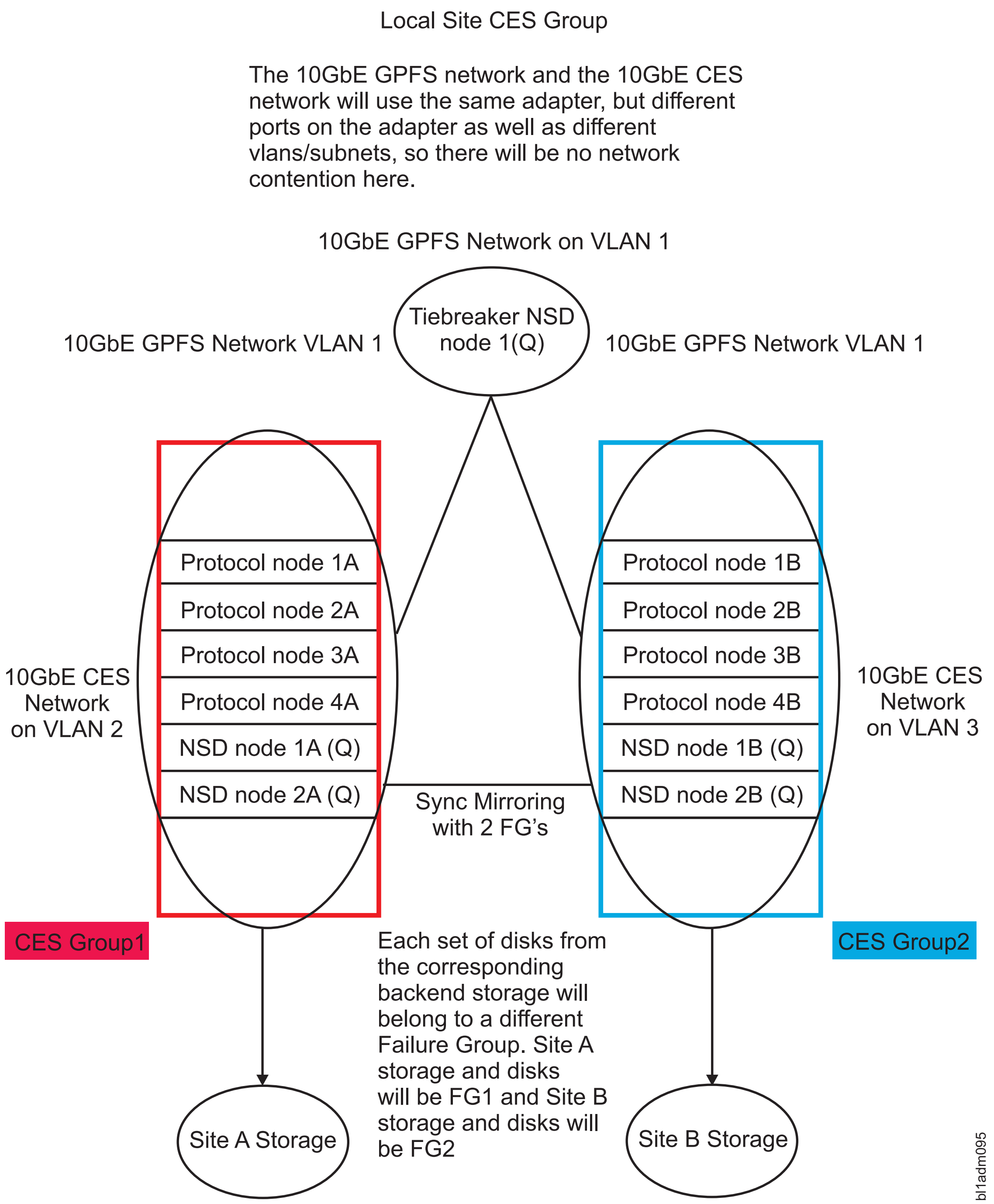

Local Site CES Group
The 10GbE GPFS network and the 10GbE CES network will use the same adapter, but different ports on the adapter as well as different vlans/subnets, so there will be no network contention here.
10GbE GPFS Network on VLAN 1
10GbE GPFS Network VLAN 1
Tiebreaker NSD node 1(Q)
10GbE GPFS Network VLAN 1
Protocol node 1A
Protocol node 2A
Protocol node 3A
Protocol node 4A
NSD node 1A (Q)
NSD node 2A (Q)
10GbE CES Network on VLAN 2
Protocol node 1B
Protocol node 2B
Protocol node 3B
Protocol node 4B
NSD node 1B (Q)
NSD node 2B (Q)
10GbE CES Network on VLAN 3
Sync Mirroring with 2 FG's
CES Group1
CES Group2
Each set of disks from the corresponding backend storage will belong to a different Failure Group. Site A storage and disks will be FG1 and Site B storage and disks will be FG2
Site A Storage
Site B Storage
bl1adm095

Figure 35. Local Site CES Group

For this use case example, the installation toolkit was used to install the IBM Spectrum Scale software. You can find the installation toolkit by changing directories to where it was extracted (the default 5.0.x.x extraction path follows. This path might vary depending on the code level).

```
cd /usr/lpp/mmfs/5.0.x.x/installer
```

Use these instructions to install and deploy a stretch cluster.

1. Designate a setup node by issuing the following command:

   ```
   ./spectrumscale setup -s InstallNodeIP
   ```

The setup node is used to run all of the toolkit commands and to specify the protocol and NSD nodes.

2. Specify the protocol and NSD nodes by issuing the following commands:

   ```
   ./spectrumscale node add protocol1A -a -p -g
   ./spectrumscale node add protocol2A -a -p -g
   ./spectrumscale node add protocol3A -p
   ./spectrumscale node add protocol4A -p
   ./spectrumscale node add protocol1B -p
   ./spectrumscale node add protocol2B -p
   ./spectrumscale node add protocol3B -p
   ./spectrumscale node add protocol4B -p
   ./spectrumscale node add nsd1A -n -q
   ./spectrumscale node addnsd2A -n -q
   ./spectrumscale node add nsd1B -n -q
   ./spectrumscale node add nsd2B -n -q
   ./spectrumscale node add nsd3C -n -q
   ```

   The **-s** argument identifies the IP address that nodes use to retrieve their configuration. This IP address is one associated with a device on the installation node. (The IP address is automatically validated during the setup phase.)

   The **-q** argument indicates the quorum nodes that are to be configured in the cluster. To keep the cluster accessible during a failure, configure most of the quorum nodes to have GPFS active. In this use case, there are five quorum nodes, therefore three must be active to keep the cluster accessible. These nodes were chosen specifically because they are the least likely to become inaccessible at the same time. Because nsd1A and nsd2A are at one site, nsd1B and nsd2B are at a second site, and nsd3C is at a third site, the likelihood of more than three going down at a time is minimal.

   No manager nodes were specified with the **-m** argument, but by default, if no **-m** argument is specified, the installation toolkit automatically sets the protocol nodes to manager nodes, leaving an even balance across both sites.

   The GUI node designations are specified with the **-g** argument to be on protocol nodes that reside on the same site, but you can choose to have a single GUI, two GUIs on one site, or two GUIs on different sites. In this case, two GUIs were tested on a single site.

3. Define NSD mappings to physical disks and assign those NSDs to failure groups and file systems. The following example NSDs are designated as dataAndMetadata; however, if you have the capacity (disk space and disk speed), set up Metadata disks on SSDs for the best performance.

   ```
   ./spectrumscale nsd add -p nsd1A -s nsd2A -u dataAndMetadata -fs ces -fg 2 /dev/mapper/lun_8
   ./spectrumscale nsd add -p nsd1B  -s nsd2B  -u dataAndMetadata -fs ces -fg 1 /dev/mapper/lun_1

   ./spectrumscale nsd add -p nsd1B -s nsd2B -u dataAndMetadata -fs gpfs0 -fg 2 /dev/mapper/lun_6
   ./spectrumscale nsd add -p nsd2B -s nsd1B -u dataAndMetadata -fs gpfs0 -fg 2 /dev/mapper/lun_4
   ./spectrumscale nsd add -p nsd1B -s nsd2B -u dataAndMetadata -fs gpfs0 -fg 2 /dev/mapper/lun_10
   ./spectrumscale nsd add -p nsd2B -s nsd1B -u dataAndMetadata -fs gpfs0 -fg 2 /dev/mapper/lun_24
   ./spectrumscale nsd add -p nsd2A -s nsd1A -u dataAndMetadata -fs gpfs0 -fg 1 /dev/mapper/lun_2
   ./spectrumscale nsd add -p nsd1A -s nsd2A -u dataAndMetadata -fs gpfs0 -fg 1 /dev/mapper/lun_3
   ./spectrumscale nsd add -p nsd2A -s nsd1A -u dataAndMetadata -fs gpfs0 -fg 1 /dev/mapper/lun_4
   ./spectrumscale nsd add -p nsd1A -s nsd2A -u dataAndMetadata -fs gpfs0 -fg 1 /dev/mapper/lun_5

   ./spectrumscale nsd add -p nsd3C -u descOnly -fs gpfs0 -fg 3 /dev/sda
   ./spectrumscale nsd add -p nsd3C -u descOnly -fs ces -fg 3 /dev/sdb
   ```

   Each file system, ces or gpfs0, has multiple disks that have primary and secondary servers at each site. This ensures that the file system stays online when an entire site goes down. With multiple primary and secondary servers for each disk and failure group that is local to each site, the GPFS replication keeps the data up to date across both sites. A disk with a primary and secondary server on site A belongs to failure group 1, and a disk with a primary and secondary server on site B belongs to failure group 2. This enables the two-way replication across the failure groups, meaning that one replica of data is kept at each site. The nsd3C node is known as the tiebreaker node. The physical disks that reside on that node /dev/sda and /dev/sdbare are designated as 'descOnly' disks and are local to that node and are their own failure group. The descOnly argument indicates that the disk contains no file data or metadata. it is used solely to keep a copy of the file system descriptor. It is recommended to have that tiebreaker node in a separate geographical location than the other two sites.

4. Set up the file system characteristics for two-way replication on both the ces and gpfs0 file systems by issuing the following command:

   ```
   ./spectrumscale filesystem modify -r 2 -mr 2 ces
   ./spectrumscale filesystem modify -r 2 -mr 2 gpfs0
   ```

   This sets the metadata and data replication to 2.
5. Designate file system paths for protocols and for object by issuing the following commands:

   ```
   ./spectrumscale config protocols -f ces -m /ibm/ces
   ./spectrumscaleconfig object -f gpfs0 -m /ibm/gpfs0
   ```

6. Set the cluster name by issuing the following command:

   ```
   ./spectrumscale configgpfs -c gumby.tuc.stglabs.ibm.com
   ```

7. Install the stretch cluster by issuing the following command:

   ```
   ./spectrumscale install --precheck
   ./spectrumscale install
   ```

8. Set up the IP lists by issuing the following command:

   ```
   ./spectrumscale config protocols -e
   10.18.52.30,10.18.52.31,10.18.52.32,10.18.52.33,10.18.60.30,10.18.60.31,10.18.60.32,10.18.60.33
   ./spectrumscale filesystem list
   ```

9. Enable the protocols by issuing the following commands:

   ```
   ./spectrumscale enable nfs
   ./spectrumscale enable smb
   ./spectrumscale enable object
   ```

10. Configure object by issuing the following commands:

    ```
    ./spectrumscale config object -o Object_Fileset
    ./spectrumscale config object --adminpassword
    ./spectrumscale config object --databasepassword
    ```

11. Configure authentication by issuing the following command:

    ```
    ./spectrumscale auth file ad
    ./spectrumscale node list
    ```

12. Deploy the stretch cluster by issuing the following commands:

    ```
    ./spectrumscale deploy --precheck
    ./spectrumscale deploy
    ```

13. After the deployment completes, check the AD setup and status.

    For the use case, the same AD server was on both sites, but you can use any authentication type in a stretch cluster that is supported on a single-site IBM Spectrum Scale cluster. Note that because a stretch cluster is still one cluster, more than one authentication method per site is not supported.

    To check the status of the cluster's authentication issue either of these commands **mmuserauth service list** or **mmuserauth service check --server-reachability**.

    Issue the **mmuserauth service list** command. The system displays information similar to the following:

    ```
    FILE access configuration : AD
    PARAMETERS               VALUES
    -------------------------------------------------
    ENABLE_NFS_KERBEROS      false
    SERVERS                  10.18.2.1
    USER_NAME                Administrator
    NETBIOS_NAME             stretch_cluster
    IDMAP_ROLE               master
    IDMAP_RANGE              10000-1000000
    IDMAP_RANGE_SIZE         10000
    UNIXMAP_DOMAINS          DOMAIN1(10000000-299999999)
    LDAPMAP_DOMAINS          none
    ```

```
OBJECT access configuration : LOCAL
PARAMETERS              VALUES
-------------------------------------------------
ENABLE_KS_SSL           false
ENABLE_KS_CASIGNING     false
KS_ADMIN_USER           admin
```

Issue the **mmuserauth service check --server-reachability** command. The system displays information similar to the following:

```
Userauth file check on node: protocol1A
         Checking nsswitch file: OK
AD servers status
          NETLOGON connection: OK
          Domain join status: OK
          Machine password status: OK
Service 'gpfs-winbind' status: OK

Userauth object check on node: protocol1A
          Checking keystone.conf: OK
          Checking wsgi-keystone.conf: OK
          Checking /etc/keystone/ssl/certs/signing_cert.pem: OK
          Checking /etc/keystone/ssl/private/signing_key.pem: OK
          Checking /etc/keystone/ssl/certs/signing_cacert.pem: OK

Service 'httpd' status: OK
```

Possible steps to convert an IBM Spectrum Scale cluster to a stretch cluster

1. Add the nodes from the second and third sites to the original cluster either manually or by using the spectrumscale toolkit.
2. Create the tiebreaker disks on the third site.
3. If replicating an existing file system, use the **mmchfs** command to set the replicas of data and metadata blocks to 2. If you are creating a new file system, ensure that the replication factor is set to 2 when it is created. For details, see the section Using the spectrumscale installation toolkit to install a stretch cluster.
4. Restripe your file system by issuing the **mmrestripefs <filesystem> -R** command.
5. Enable CES on the protocol nodes that you have added to the configuration.
6. Create CES groups on both sites.

Configuring the stretch cluster

1. Set up some basic tuning parameters.

   For the use case, the following tuning parameters were used to improve the performance and reliability of the cluster. Tuning parameters will vary significantly depending on the hardware resources in your environment.

   ```
   mmchconfig readReplicaPolicy=fastest
   mmchconfig unmountOnDiskFail=yes -N nsd3
   mmchconfig workerThreads=1024 -N cesNode
   mmchconfig -ipagepool=43G -N protocol1A
   mmchconfig -ipagepool=31G -N protocol2A
   mmchconfig pagepool=48G -N protocol3A
   mmchconfig pagepool=48G -N protocol4A
   mmchconfig pagepool=48G -N protocol1B
   mmchconfig pagepool=48G -N protocol2B
   mmchconfig pagepool=48G -N protocol3B
   mmchconfig pagepool=48G -N protocol4B
   mmchconfig pagepool=12G -N nsd1A
   mmchconfig pagepool=16G -N nsd1B
   mmchconfig pagepool=12G -N nsd2B
   mmchconfig pagepool=12G -N nsd3C
   mmchconfig maxFilesToCache=2M
   mmchconfig maxMBpS=5000 -N cesNodes
   ```

For details on each parameter, see *Parameters for performance tuning and optimization* in *IBM Spectrum Scale: Administration Guide*. The use case was tested with readReplicaPolicy=fastest which is the recommended setting. A known limitation with readReplicaPolicy=fastest is that with networks that add ~3 ms latency (which are common in such installations) there is no substantial difference between local and remote disks (assuming the disk latency might be in the 40/50ms range). Thus, you might still read data from the remote site. Therefore, it is acceptable to use readReplicaPolicy=local to ensure the data is written/read on the local site as long as the local servers are on the same subnet as the clients and the remote servers are not. The readReplicaPolicy=fastest setting will work with either network topology, both sites on the same subnet or each site on the its own subnet, as long as there is a measurable difference in the I/O access time.

2. Set up the CES nodes.

   CES group are needed when the CES networks on each site cannot communicate with each other. By having each site's local nodes in the same CES group, the administrator is able to control where the CES IPs failover to when there is an issue with a specific protocol node. If CES groups are not set up, a CES IP from Site A might attempt to failover to a node on Site B, and because there is no adapter for that IP to alias to on Site B (assuming different subnets), the failover will not succeed. CES groups make it easy to manage what CES nodes can host what CES IPs.

   Set the CES nodes in the cluster to the corresponding groups by issuing the **mmchnode --ces-group** command (CES group names are not case-sensitive). For example:

   ```
   mmchnode --ces-group SiteA -N protocol1A
   mmchnode --ces-group SiteA -N protocol2A
   mmchnode --ces-group SiteB -N protocol1B
   mmchnode --ces-group SiteB -N protocol2B
   ```

   In the example, protocol nodes protocol1A and protocol2A are set to the Site A CES group, protocol nodes protocol1Band protocol2B are set to the Site B CES group.

   For detailed instructions, see *Setting up Cluster Export Services groups in an IBM Spectrum Scale cluster* in *IBM Spectrum Scale: Administration Guide*.

3. Assign CES IPs to the corresponding CES groups. This ensures that IPs that reside on nodes in Site A do not fail over to nodes that reside in Site B and vice versa. Issue the **mmces address change command**. For example:

   ```
   mmces address change --ces-ip 10.18.52.30,10.18.52.31,10.18.52.32,10.18.52.33 --ces-group SiteA
   mmces address change --ces-ip 10.18.60.30,10.18.60.31,10.18.60.32,10.18.60.33 --ces-group SiteB
   ```

4. To verify the CES groups your nodes belong to, issue the **mmces node list** command. The sample output is as follows:

   ```
   Node  Name                         Node Flags      Node Groups
   --------------------------------------------------------------
   10 protocol1B                      none            siteB
   11 protocol2B                      none            siteB
   12 protocol3B                      none            siteB
   13 protocol4B                      none            siteB
   6 protocol1A                       none            siteA
   7 protocol2A                       none            siteA
   8 protocol3A                       none            siteA
   9 protocol4A                       none            siteA
   ```

5. To verify the CES groups your CES IPs belong to, issue the **mmces address list** command. The sample output is as follows:

   ```
   Address         Node               Group     Attribute
   -------------------------------------------------------------------------
   10.18.52.30    protocol1AsiteAobject_singleton_node,object_database_node
   10.18.52.31    protocol2AsiteA      none
   10.18.52.32    protocol3AsiteA      none
   10.18.52.33    protocol4AsiteA      none
   10.18.60.30    protocol1BsiteB      none
   10.18.60.31    protocol2BsiteB      none
   10.18.60.32    protocol3BsiteB      none
   10.18.60.33    protocol4BsiteB      none
   ```

A load balancer is recommended for the protocol stack to cater to a site loss. The load balancer will ensure that you do not encounter issues if using DNS Round Robin when a site goes down and that the host name in the DNS server can resolve all of the IP addresses.

Using NFS, SMB, and Object with a Stretch Cluster

Using NFS and SMB protocols is similar to using a Spectrum Scale cluster that is in one geographical location. All clients can read and write to either site and to any CES IP that they connect with depending on access. If a single protocol node fails at one site, a normal IP failover will still occur within the site, and the client seamlessly fails over with NFS I/O continuing. SMB clients, however, might need to be reconnected. On failures, clients can reconnect to another cluster node because the IP addresses of failing nodes are transferred to another healthy cluster node. Windows SMB clients automatically open a new connection without additional intervention, but the application that is running I/O may need to be restarted. Object has a few more limitations. See the section "Limitations regarding a Stretch Cluster using GPFS synchronous replication" for details. In summary, if your CES networks cannot communicate across sites, you must choose a single site and its CES group to configure with object. During a full site outage, you will need to make the manual fixes described in the Limitations section. A single protocol node failure will still work but you will need to retry after the CES IP moves to a new node within the CES group.

Monitoring and administering a stretch cluster

Monitoring your stretch cluster is the same as monitoring a cluster in a single location, except for the disk setup, and knowing when your disks are down and what site is affected. You can see the disk status using the **mmlsdisk** command. The sample output is as follows:

```
disk         driver   sector     failure holds    holds                    storage
name         type       size       group metadata data  status        availability pool
------------ -------- ------ ----------- -------- ----- ------------- ------------------------
nsd22        nsd         512           2 Yes      Yes   ready         up   system
nsd23        nsd         512           2 Yes      Yes   ready         up   system
nsd24        nsd         512           2 Yes      Yes   ready         up   system
nsd25        nsd         512           2 Yes      Yes   ready         up   system
nsd26        nsd         512           1 Yes      Yes   ready         up   system
nsd27        nsd         512           1 Yes      Yes   ready         up   system
nsd28        nsd         512           1 Yes      Yes   ready         up   system
nsd29        nsd         512           1 Yes      Yes   ready         up   system
nsd30        nsd         512           1 Yes      Yes   ready         up   system
nsd31        nsd         512           3 No       No    ready         up   system
```

A healthy cluster shows all the disks that have a status of up. You can also verify the replication settings using the **mmlsfs -m -M -r -R** command. The sample output is as follows:

```
flag                value                    description
------------------- ------------------------ -----------------------------------
-m                   2                        Default number of metadata replicas
-M                   3                        Maximum number of metadata replicas
-r                   2                        Default number of data replicas
-R                   3                        Maximum number of data replicas
```

If the default number of data and metadata replicas is set to two, this will indicate that you have no disk failures and that your data is being replicated across both failure groups.

Issue the **mmlsdisk ces** command. The sample output is as follows:

```
disk         driver   sector     failure holds    holds                    storage
name         type       size       group metadata data  status        availability pool
------------ -------- ------ ----------- -------- ----- ------------- ------------------------
nsd20        nsd         512           2 Yes      Yes   ready         up   system
nsd21        nsd         512           1 Yes      Yes   ready         up   system
nsd32        nsd         512           3 No        No   ready         up   system
```

If you lose access to one site's storage due to maintenance, network issues, or hardware issues, the disks in the cluster are marked as down and the **mmhealth node show** command results shows them as down. This is acceptable because the stretch cluster can keep operating when an entire site goes down. There can be a negative impact on performance while one site is down, but that is expected.

To see the disk cluster status for the use case, issuing the **mmlsdisk gpfs0** command shows the following information:

```
disk         driver   sector     failure holds    holds                    storage
name         type       size       group metadata data  status        availability pool
------------ -------- ------ ----------- -------- ----- ------------- ------------------------
nsd22        nsd         512           2 Yes      Yes   ready         down   system
nsd23        nsd         512           2 Yes      Yes   ready         down   system
nsd24        nsd         512           2 Yes      Yes   ready         down   system
nsd25        nsd         512           2 Yes      Yes   ready         down   system
nsd26        nsd         512           1 Yes      Yes   ready         up     system
nsd27        nsd         512           1 Yes      Yes   ready         up     system
nsd28        nsd         512           1 Yes      Yes   ready         up     system
nsd29        nsd         512           1 Yes      Yes   ready         up     system
nsd30        nsd         512           1 Yes      Yes   ready         up     system
nsd31        nsd         512           3 No       No    ready         up     system
```

For the use case, the results of the **mmhealth node show -N nsd2B disk** command show three disks:

```
Node name:      nsd2B
Node status:    FAILED
Status Change:  17 min. ago
Component      Status        Status Change     Reasons
---------------------------------------------------------------------------------
GPFS           FAILED        17 min. ago       gpfs_down, quorum_down
NETWORK        HEALTHY       10 days ago       -
FILESYSTEM     DEPEND        17 min. ago       unmounted_fs_check(gpfs1, ces, gpfs0)
DISK           DEPEND        17 min. ago       disk_down(nsd20, nsd22, nsd23)
PERFMON        HEALTHY       10 days ago       -
```

To see all of the failed disks, issue the **mmhealth node show nsd2B** command (without the -N attribute). For the use case, the system displays the following information:

```
Node name:      nsd2B
Component Status        Status Change     Reasons
-----------------------------------------------------------------------
DISK      DEPEND        18 min. ago      disk_down(nsd20, nsd22, nsd23)
nsd1      DEPEND        18 min. ago       -
nsd10     DEPEND        18 min. ago       -
nsd11     DEPEND        18 min. ago       -
nsd12     DEPEND        18 min. ago       -
nsd13     DEPEND        18 min. ago       -
nsd14     DEPEND        18 min. ago       -
nsd15     DEPEND        18 min. ago       -
nsd16     DEPEND        18 min. ago       -
nsd17     DEPEND        18 min. ago       -
nsd18     DEPEND        18 min. ago       -
nsd19     DEPEND        18 min. ago       -
nsd2      DEPEND        18 min. ago       -
nsd20     DEPEND        18 min. ago       disk_down
nsd22     DEPEND        18 min. ago       disk_down
nsd23     DEPEND        18 min. ago       disk_down
nsd24     DEPEND        18 min. ago       disk_down
nsd25     DEPEND        18 min. ago       disk_down
nsd3      DEPEND        18 min. ago       -
nsd4      DEPEND        18 min. ago       -
nsd5      DEPEND        18 min. ago       -
nsd6      DEPEND        18 min. ago       -
nsd7      DEPEND        18 min. ago       -
nsd8      DEPEND        18 min. ago       -
nsd9      DEPEND        18 min. ago       -
```

```
Event     Parameter     Severity   Active Since      Event Message
-------------------------------------------------------------------------------------
disk_down nsd20         WARNING     16 min. ago       Disk nsd20 is reported as not up
disk_down nsd22         WARNING     16 min. ago       Disk nsd22 is reported as not up
disk_down nsd23         WARNING     16 min. ago       Disk nsd23 is reported as not up
disk_down nsd24         WARNING     16 min. ago       Disk nsd24 is reported as not up
disk_down nsd25         WARNING     16 min. ago       Disk nsd25 is reported as not up
```

After the issue is resolved, restart the disks and make sure that the data and metadata replicas are intact. First, ensure that GPFS is active on all nodes. Next, issue the **mmchdisk <filesystem> start-a** command. This informs GPFS to try to access the disks that are marked down and, if possible, to move them back into the up state. This is accomplished by first changing the disk availability from down to recovering. The file system metadata is then scanned and any missing updates (replicated data that was changed while the disk was down) are repaired. If this operation is successful, the availability is then changed to up. If the metadata scan fails, availability is changed to unrecovered. This could occur if too many disks are down. The metadata scan can be re-initiated later by issuing the **mmchdisk** command again. If more than one disk in the file system is down, all of the disks that are down must be started at the same time by issuing **mmchdisk <filesystem> start -a**. If you start them separately and metadata is stored on any disk that remains down, the **mmchdisk start** command fails.

```
mmnsddiscover: Attempting to rediscover the disks. This may take a while ...
mmnsddiscover: Finished.
nsd2A: Rediscovered nsd server access to nsd26.
nsd2A: Rediscovered nsd server access to nsd28.
nsd3C: Rediscovered nsd server access to nsd31.
sdn2B: Rediscovered nsd server access to nsd23.
nsd1B: Rediscovered nsd server access to nsd23.
nsd2B: Rediscovered nsd server access to nsd24.
nsd1B: Rediscovered nsd server access to nsd24.
nsd1A: Rediscoverensd server access to nsd29.
nsd2A: Rediscovered nsd server access to nsd30.
nsd2A: Rediscoved red nsd server access to nsd27.
nsd2B: Rediscovered nsd server access to nsd25.
nsd2B: Rediscovered nsd server access to nsd22.
nsd2A: Rediscovered nsd server access to nsd25.
nsd2A: Rediscovered nsd server access to nsd22.
Scanning file system metadata, phase 1 ...
33 % complete on Fri Feb 3 11:46:41 2017
66 % complete on Fri Feb 3 11:56:57 2017
100 % complete on Fri Feb 3 11:58:24 2017 Scan completed successfully.
Scanning file system metadata, phase 2 ...
Scan completed successfully.
Scanning file system metadata, phase 3 ...
8 % complete on Fri Feb 3 11:58:29 2017
16 % complete on Fri Feb 3 11:58:32 2017
23 % complete on Fri Feb 3 11:58:35 2017
...
91 % complete on Fri Feb 3 11:59:18 2017
95 % complete on Fri Feb 3 11:59:22 2017
98 % complete on Fri Feb 3 11:59:25 2017
100 % complete on Fri Feb 3 11:59:26 2017
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
2.37 % complete on Fri Feb 3 11:59:46 2017 ( 2473984 inodes with total 672770 MB data processed)
3.86 % complete on Fri Feb 3 12:00:07 2017 ( 4734976 inodes with total 1094807 MB data processed)
4.59 % complete on Fri Feb 3 12:00:27 2017 ( 7880704 inodes with total 1301307 MB data processed)
5.30 % complete on Fri Feb 3 12:00:47 2017 ( 11003904 inodes with total 1501577 MB data processed)
6.01 % complete on Fri Feb 3 12:01:07 2017 ( 14077952 inodes with total 1703928 MB data processed)
6.70 % complete on Fri Feb 3 12:01:27 2017 ( 17154048 inodes with total 1896877 MB data processed)
7.36 % complete on Fri Feb 3 12:01:47 2017 ( 20135936 inodes with total 2084748 MB data processed)
```

```
7.97 % complete on Fri Feb 3 12:02:07 2017 ( 22512640 inodes with total 2257626 MB data processed)
8.21 % complete on Fri Feb 3 12:02:27 2017 ( 23322624 inodes with total 2327269 MB data processed)
8.39 % complete on Fri Feb 3 12:02:48 2017 ( 24182784 inodes with total 2377108 MB data processed)
8.52 % complete on Fri Feb 3 12:03:09 2017 ( 25182208 inodes with total 2414040 MB data processed)
8.64 % complete on Fri Feb 3 12:03:29 2017 ( 26166272 inodes with total 2447380 MB data processed)
...
96.58 % complete on Fri Feb 3 12:36:40 2017 ( 198458880 inodes with total 27362407 MB data processed)
96.82 % complete on Fri Feb 3 12:37:00 2017 ( 202438144 inodes with total 27430464 MB data processed)
97.06 % complete on Fri Feb 3 12:37:20 2017 ( 206526720 inodes with total 27498158 MB data processed)
97.30 % complete on Fri Feb 3 12:37:40 2017 ( 210588672 inodes with total 27567944 MB data processed)
97.46 % complete on Fri Feb 3 12:38:00 2017 ( 266730496 inodes with total 27612826 MB data processed)
97.52 % complete on Fri Feb 3 12:38:20 2017 ( 302344960 inodes with total 27629694 MB data processed)
97.59 % complete on Fri Feb 3 12:38:40 2017 ( 330066432 inodes with total 27648547 MB data processed)
100.00 % complete on Fri Feb 3 12:38:52 2017 ( 394185216 inodes with total 27657707 MB data processed)
Scan completed successfully.
```

The recovery time for this command can vary depending on how much data was written while the disks were down. If the disks were down for a long time (greater than 24 hours) and a lot of data was written in that time, it is expected that the **mmchdisk** command could take quite a while to complete. The time needed to bring up the disks depends on the quantity of data changed during the time the disks were down. This command is run while the file data remains accessible to the applications so I/O clients can continue to operate.

IBM Spectrum Scale stretch cluster use case conclusion

Each use case for a stretch cluster will vary. The sample use case represents one tested configuration. For more information see the following topics:

Synchronous mirroring with GPFS replication in *IBM Spectrum Scale: Administration Guide*.

Synchronous mirroring with GPFS replication in *IBM Spectrum Scale: Administration Guide*.

An active-passive cluster in *IBM Spectrum Scale: Administration Guide*

Limitations in *IBM Spectrum Scale: Big Data and Analytics Guide*

Configuration of object for isolated node and network groups in *IBM Spectrum Scale: Administration Guide*

https://docs.openstack.org/mitaka/install-guide-rdo/keystone-services.html

https://docs.openstack.org/mitaka/install-guide-rdo/keystone-install.html

Parameters for performance tuning and optimization in *IBM Spectrum Scale: Administration Guide*

Setting up Cluster Export Services groups in an IBM Spectrum Scale cluster in *IBM Spectrum Scale: Administration Guide*

Performing additional tasks using the installation toolkit

Use this information for using the installation toolkit for tasks such as deploying protocols on existing clusters and adding nodes to an existing installation.

Deploying protocols on an existing cluster

Use this information to deploy protocols on an existing cluster using the installation toolkit.

1. Instantiate your chef zero server using the following command.

   ```
   ./spectrumscale setup -s 192.168.0.1
   ```

2. Designate protocol nodes in your environment to use for the deployment using the following command.

   ```
   ./spectrumscale node add FQDN -p
   ```

3. Designate GUI nodes in your environment to use for the deployment using the following command.

   ```
   ./spectrumscale node add FQDN -g
   ```

4. Configure protocols to point to a file system that will be used as a shared root using the following command.

   ```
   ./spectrumscale config protocols -f FS_Name -m FS_Mountpoint
   ```

5. Configure a pool of export IPs using the following command.

   ```
   ./spectrumscale config protocols -e Comma_Separated_List_of_Exportpool_IPs
   ```

6. Enable NFS on all protocol nodes using the following command.

   ```
   ./spectrumscale enable nfs
   ```

7. Enable SMB on all protocol nodes using the following command.

   ```
   ./spectrumscale enable smb
   ```

8. Enable Object on all protocol nodes using the following command.

   ```
   ./spectrumscale enable object
   ./spectrumscale config object -au admin -ap -dp
   ./spectrumscale config object -e FQDN
   ./spectrumscale config object -f FS_Name -m FS_Mountpoint
   ./spectrumscale config object -o Object_Fileset
   ```

9. Review the configuration prior to deployment using the following command.

   ```
   ./spectrumscale config protocols
   ./spectrumscale config object
   ./spectrumscale node list
   ```

10. Deploy protocols on your defined environment using the following command.

    ```
    ./spectrumscale deploy --precheck
    ./spectrumscale deploy
    ```

After deploying protocols on an existing cluster, deploy protocols authentication.

Deploying protocols authentication on an existing cluster

Use this information to deploy protocols authentication on an existing cluster after deploying protocols using the installation toolkit.

1. Enable file authentication with AD server on all protocol nodes using the following command.

   ```
   ./spectrumscale auth file ad
   ```

   Fill out the template and save the information, and then issue the following commands.

   ```
   ./spectrumscale deploy --precheck
   ./spectrumscale deploy
   ```

2. Enable Object authentication with AD server on all protocol nodes using the following command.

   ```
   ./spectrumscale auth object ad
   ```

   Fill out the template and save the information, and then issue the following commands.

   ```
   ./spectrumscale deploy --precheck
   ./spectrumscale deploy
   ```

Adding nodes, NSDs, or file systems to an existing installation

Use this information to add nodes, NSDs, or file systems to an installation process using the installation toolkit.

- To add nodes to an existing installation, do the following.
  1. Add one or more node types using the following commands.
     - Client nodes:

       ```
       ./spectrumscale node add FQDN
       ```

  - NSD nodes:

    ```
    ./spectrumscale node add FQDN -n
    ```

  - Protocol nodes:

    ```
    ./spectrumscale node add FQDN -p
    ```

  - GUI nodes:

    ```
    ./spectrumscale node add FQDN -g -a
    ```

  For the list of supported node types that can be added, see *spectrumscale command* in *IBM Spectrum Scale: Administration Guide*.

2. Install GPFS on the new nodes using the following commands.

   ```
   ./spectrumscale install --precheck
   ./spectrumscale install
   ```

3. If protocol nodes are being added, deploy protocols using the following commands.

   ```
   ./spectrumscale deploy --precheck
   ./spectrumscale deploy
   ```

- To add NSDs to an existing installation, do the following.
  1. Verify that the NSD server connecting this new disk runs an OS compatible with the installation toolkit and that the NSD server exists within the cluster.
  2. Add NSDs to the installation using the following command.

     ```
     ./spectrumscale nsd add -p FQDN_of_Primary_NSD_Server Path_to_Disk_Device_File
     ```

  3. Run the installation using the following commands.

     ```
     ./spectrumscale install --precheck
     ./spectrumscale install
     ```

- To add file systems to an existing installation, do the following.
  1. Verify that free NSDs exist and that they can be listed by the installation toolkit using the following commands.

     ```
     mmlsnsd
     ./spectrumscale nsd list
     ```

  2. Define the file system using the following command.

     ```
     ./spectrumscale nsd modify NSD -fs File_System_Name
     ```

  3. Deploy the file system using the following commands.

     ```
     ./spectrumscale deploy --precheck
     ./spectrumscale deploy
     ```

Enabling another protocol on an existing cluster that has protocols enabled

Use this information to enable protocols on an existing cluster that has other protocols enabled using the installation toolkit.

1. Use one of the following steps depending on your configuration.
   - Enable NFS on all protocol nodes using the following command.

     ```
     ./spectrumscale enable nfs
     ```

   - Enable SMB on all protocol nodes using the following command.

     ```
     ./spectrumscale enable smb
     ```

   - Enable Object on all protocol nodes using the following commands.

     ```
     ./spectrumscale enable object
     ./spectrumscale config object -au Admin_User -ap Admin_Password -dp Database_Password
     ./spectrumscale config object -e FQDN
     ./spectrumscale config object -f FS_Name -m FS_Mountpoint
     ./spectrumscale config object -o Object_Fileset
     ```

2. Enable the new protocol using the following command.

   ```
   ./spectrumscale deploy --precheck
   ./spectrumscale deploy
   ```

Diagnosing errors during installation, deployment, or upgrade

Use this information to diagnose errors during install, deploy, or upgrade using the installation toolkit.

1. Note the screen output indicating the error. This helps in narrowing down the general failure. When a failure occurs, the screen output also shows the log file containing the failure.
2. Open the log file in an editor such as **`vi`**.
3. Go to the end of the log file and search upwards for the text `FATAL`.
4. Find the topmost occurrence of FATAL (or first FATAL error that occurred) and look above and below this error for further indications of the failure.

   For more information, see *Finding deployment related error messages more easily and using them for failure analysis* in *IBM Spectrum Scale: Problem Determination Guide*.

Preparing a cluster that contains ESS for adding protocols

Use this information to prepare a cluster that contains ESS for adding functions such as NFS, SMB, Object, GUI, and Performance Monitoring.

Before adding protocols to a cluster that contains ESS, ensure the following:

- The cluster containing ESS is active and online.
- RHEL 7.x is installed on nodes that are going to serve as protocol nodes.
- RHEL 7.x base repository is set up on nodes that are going to serve as protocol nodes.

1. Use the ESS GUI or CLI to create a CES shared root file system.

   Note: The CES shared root file system must be at least 4 GB in size and it must not be encrypted.
2. On all nodes that are going to serve as protocol nodes, download and extract the IBM Spectrum Scale protocols standard or advanced packages. For information on extracting packages, see "Extracting the IBM Spectrum Scale software on Linux nodes" on page 231.
3. On all nodes that are going to serve as protocol nodes, install core GPFS RPMs from the `/usr/lpp/mmfs/5.x.x.x/gpfs_rpms`.

   Attention: Ensure that you do not install packages such as `perfmon`, `gui`, `callhome`, `java`, and `protocols` at this time. These components will be installed and configured with the protocols deployment.

   The core GPFS RPMs that need to be installed include:
   - `gpfs.base`
   - `gpfs.ext`
   - `gpfs.gpl`
   - `gpfs.license.xx`
   - `gpfs.gskit`
   - `gpfs.docs`
   - `gpfs.msg.en_US`
   - `gpfs.compression`
   - `gpfs.adv` (optional)
   - `gpfs.crypto` (optional)
4. Add the nodes that are going to serve as protocol nodes to the cluster using the **`mmaddnode`** command.
5. Enable the Cluster Configuration Repository (CCR) on the cluster, if it is not enabled already.

Attention: If GPFS levels between protocol nodes and ESS differ significantly, ensure that the nodes with the newer code level of GPFS are designated as both quorum and manager nodes. For example, old ESS systems with GPFS 4.1.0-8 are incompatible with CES. These ESS systems can be a part of a cluster with protocols only if they are not designated as quorum and manager nodes.

This aspect requires careful planning and administration because:

- By default, ESS nodes are designated as quorum and manager nodes.
- In some cases, there might not be extra nodes outside of ESS to make sure that only nodes with the newer code level are designated as quorum and manager nodes.

6. Verify that passwordless SSH is working between all nodes in the cluster.
7. Verify that firewall ports are set correctly.
8. Verify that the CES shared root file system is mounted and that it is set to auto mount.
9. Set NFSv4 ACLs for file systems that are going to be used for protocol data, if they are not set already.
10. On one of the nodes that are going to serve as protocol nodes, run the installation toolkit to start deploying protocols.

    Note:

    - Designate only the nodes that you have planned to use as protocol nodes. Do not designate existing ESS nodes such as EMS or I/O nodes as protocol nodes.
    - Point to existing file systems only. Do not attempt to create new file systems or NSDs for this purpose.

    For information about deploying protocols in a cluster, see **spectrumscale command** in *IBM Spectrum Scale: Command and Programming Reference* and "Deploying protocols on an existing cluster" on page 307.

Related concepts:

"ESS awareness with the installation toolkit" on page 294
You can use the installation toolkit to install IBM Spectrum Scale and deploy protocols in a cluster containing Elastic Storage Server (ESS).

Adding an IBM Spectrum Archive Enterprise Edition (EE) node in an IBM Spectrum Scale cluster

Use this information to add an IBM Spectrum Archive Enterprise Edition (EE) node in an IBM Spectrum Scale cluster.

1. Use the installation toolkit to install IBM Spectrum Scale and create a cluster.

   For more information, see "Using the installation toolkit to perform installation tasks: Explanations and examples" on page 272.
2. If the node on which IBM Spectrum Archive Enterprise Edition (EE) is going to be installed is running on RHEL 6.x, install IBM Spectrum Scale packages on that node manually using the RPMs from a protocol node.
3. Add the IBM Spectrum Archive Enterprise Edition (EE) node to your cluster at any time after the cluster is created.

   For more information, see *Adding nodes to a GPFS cluster* in *IBM Spectrum Scale: Administration Guide*.
4. Before installing IBM Spectrum Archive Enterprise Edition (EE), enable Data Management API (DMAPI) on the file system to be used with IBM Spectrum Archive Enterprise Edition (EE).
   a. Unmount the file system to be used with IBM Spectrum Archive Enterprise Edition (EE).

      Note: If this file system is also used for protocols, you cannot unmount the file system unless you issue the **mmshutdown** command on all protocol nodes.
   b. Enable DMAPI using the **mmchfs** *Device* **–z yes** command.

      In this command example, *Device* is the device name of the file system.
   c. Mount the file system again and issue the **mmstartup** on all protocol nodes if you used the **mmshutdown** command in a preceding step.
5. Install and configure IBM Spectrum Archive Enterprise Edition (EE).

For detailed information on using IBM Spectrum Archive and IBM Spectrum Scale together, see Active Archive Implementation Guide with Spectrum Scale Object and IBM Spectrum Archive Redpaper. Chapters 2 and 3 of this document contain specific details about cluster configuration and how to add the IBM Spectrum Archive node to the IBM Spectrum Scale cluster.

Installing IBM Spectrum Scale by using the graphical user interface (GUI)

You can use the GUI to install and configure IBM Spectrum Scale system. The installation GUI is used only for installing the system and a separate management GUI needs to be used for configuring and managing the system.

The scope of the installation GUI is to install the IBM Spectrum Scale software on cluster nodes, create an IBM Spectrum Scale cluster, and configure NTP. Other initial tasks like creating file systems, creating NSDs, setting up NFS and SMB protocols, object storage, or authentication of protocol users are out of scope of the installation GUI. You need to perform those tasks either through CLI or management GUI.

Prerequisites

The following are the prerequisites for installing IBM Spectrum Scale through installation GUI:

- On a cluster where the GUI is installed, other nodes in the cluster need to be able to send notifications to the GUI through HTTP requests. For this purpose, curl, a command line tool and library for transferring data with URLs is used. Therefore, it is necessary to make sure that all nodes in the cluster have curl installed and configured correctly so that it can send HTTP requests to the GUI node. Verify whether the curl is working properly by issuing `curl -k https://[GUI NODE NAME OR IP]:443` from the other cluster nodes, which should return the HTML code of the GUI.

  If curl is not configured correctly or cannot send HTTP requests to the GUI node, the following error message appears in the `mmfs.log` of that node:

  ```
  [E] An attempt to send notification to the GUI subsystem failed. response=curl: (28) connect() timed out! rc=28
  ```
- The Network Time Protocol (NTP) must be installed on all nodes on which the IBM Spectrum Scale is being installed through installation GUI.
- The prerequisites that are applicable for installing the IBM Spectrum Scale system through CLI is applicable for installation through the GUI as well. For more information on the prerequisites for installation, see “Installation prerequisites” on page 224.

The installation GUI requires port 9080 to be designated for http and port 9443 for https. That is, the installer GUI URL of the http and https connections are `http://<install server IP or host name>:9080` and `https://<install server IP or host name>:9443` respectively.

Note: To start the graphical installation, the IBM Spectrum Scale package must be extracted. By default, the installation images are extracted to the target directory `/usr/lpp/mmfs/5.0.x.x` depending on the release version.

By selecting different parameters of the **`spectrumscale installgui`** command, you can start, stop, and view the status of the installation GUI.

```
spectrumscale installgui [-h] {start,stop,status} ...
```

Example:

To start the installation GUI, issue the following command:

```
./spectrumscale installgui start
```

To view the status of the processes that are running on the installation GUI, issue the following command:

```
./spectrumscale installgui status
```

The installation process through the GUI automatically stops when you exit the installation GUI. To stop installation process through the CLI, issue the following command:

```
./spectrumscale installgui stop
```

Note: The password to access the IBM Spectrum Scale installer GUI is `Passw0rd`.

In case the installation GUI fails to launch, you can use the logs located at the following location for debugging: `/usr/lpp/mmfs/5.0.0.0/installer/gui/wlp/usr/servers/installgui/logs`

Limitations of the installation GUI

The following are the limitations of the installation GUI:

- The scope of the installation GUI is to install the IBM Spectrum Scale software on cluster nodes, create an IBM Spectrum Scale cluster, and configure NTP. Other initial tasks like creating file systems, creating NSDs, setting up NFS and SMB protocols, object storage, or authentication of protocol users are out of scope of the installation GUI. You need to perform those tasks either through CLI or management GUI.
- It supports only a fresh installation of an IBM Spectrum Scale cluster. That is, it does not support to update the system from a previous release to the latest release.
- It does not support adding nodes to and removing nodes from the existing IBM Spectrum Scale cluster
- You cannot uninstall the existing IBM Spectrum Scale cluster through the GUI.

Protocol node IP further configuration

The **mmces** commands for adding or moving IPs are as follows.

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

Checking that protocol IPs are currently configured

The **mmces address list** command shows all the currently configured protocol IPs.

Note: The term CES IP refers to the Cluster Export Services IP, or a protocol IP used specifically for NFS, SMB, and Object protocol access. All CES IPs must have an associated hostname and reverse DNS lookup must be configured for each. For more information, see *Adding export IPs* in “Deploying protocols” on page 279.

```
mmces address list
```

The system displays output similar to the following:

```
Address Node Group Attribute
-------------------------------------------------------------------------
10.0.0.101 prt001st001 none none
10.0.0.102 prt002st001 none none
10.0.0.103 prt003st001 none none
10.0.0.104 prt004st001 none none
10.0.0.105 prt005st001 none none
10.0.0.106 prt006st001 none none
```

Additional IPs can be added to a node. In this example, IPs `10.0.0.108` and `10.0.0.109` are added to protocol node 5 (`prt005st001`).

```
mmces address add --ces-node prt005st001 --ces-ip 10.0.0.108,10.0.0.109
mmces address list
```

The system displays output similar to the following:

```
Address Node Group Attribute
-------------------------------------------------------------------------
10.0.0.101 prt001st001 none none
10.0.0.102 prt002st001 none none
10.0.0.103 prt003st001 none none
10.0.0.104 prt004st001 none none
10.0.0.105 prt005st001 none none
10.0.0.106 prt006st001 none none
10.0.0.108 prt005st001 none none
10.0.0.109 prt005st001 none none
```

IPs can also be moved among nodes manually. This is helpful if IP allocation becomes unbalanced among the nodes. Continuing from the prior example, `prt005st001` now has three protocol IPs whereas all other nodes have a single protocol IP. To balance this out a bit better, `10.0.0.109` will be manually moved to node `prt004st001`.

```
mmces address move –ces-ip 10.0.0.109 --ces-node prt004st001
mmces address list
```

The system displays output similar to the following:

```
Address Node Group Attribute
-------------------------------------------------------------------------
10.0.0.101 prt001st001 none none
10.0.0.102 prt002st001 none none
10.0.0.103 prt003st001 none none
10.0.0.104 prt004st001 none none
10.0.0.105 prt005st001 none none
10.0.0.106 prt006st001 none none
10.0.0.108 prt005st001 none none
10.0.0.109 prt004st001 none none
```

List all protocol services enabled

Run the following command from any protocol node to list the running services on all protocol nodes:

```
mmces service list -a
```

The system displays output similar to the following:

```
Enabled services: SMB NFS OBJ
prt001st001: SMB is not running, NFS is not running, OBJ is running
prt002st001: SMB is running, NFS is running, OBJ is not running
prt003st001: SMB is running, NFS is running, OBJ is running
```

Object protocol further configuration

If the Object protocol was enabled during installation, use the following information to verify the Object protocol configuration and to enable features such as unified file and object access and multi-region object deployment.

Object is currently set up, enabled, and configured by the installation toolkit, but a few additional steps are necessary to ready the Object protocol for use. Verify the Object protocol by running a few tests.

Verifying the Object protocol portion of the installation

Perform the following simple example OpenStack client and unified file and object access commands to verify your installation. You should be able to list all users and projects (called accounts in unified file and object access), list the unified file and object access containers, upload a sample object to a container, and list that container and see the object. These commands should complete with no errors.

```
# source $HOME/openrc
# openstack user list
# openstack project list
# swift stat
# date > object1.txt
# swift upload test_container object1.txt
object1.txt
# swift list test_container
object1.txt
```

When the SELinux mode is `Enforcing` and a user on the protocol node needs to run Swift helper commands, the **runcon** command must be used. Otherwise, the system might generate SELinux failure logs. For example, the *.recon files in the `/var/cache/swift` directory are updated by Swift helper services such as replicator and updater.

1. To run the helper command, type: `runcon -t swift_t -r system_r` *`swift helper command`*
2. To check if the SELinux context is correct, run the **matchpathcon** command: `matchpathcon -V /var/cache/swift/*`

   The system displays the following output:

   ```
   /var/cache/swift/account.recon verified.
   /var/cache/swift/container.recon has context
   unconfined_u:object_r:var_t:s0, should be system_u:object_r:swift_var_cache_t:s0
   /var/cache/swift/object.recon verified.
   ```
3. To fix the SELinux context in the file, run the **restorecon** command: `restorecon /var/cache/swift/container.recon`

For more information about SELinux consideration, see "SELinux considerations" on page 204.

Enabling multi-region object deployment initially

For multi-region object deployment, each region is a separate cluster and on each cluster IBM Spectrum Scale for object storage is installed independently using the installer.

In a single cluster, the installer completely installs the object protocol on all the nodes within that cluster. However, in a multi-region object deployment environment, each cluster is installed independently and the object protocol on the independent clusters is linked together.

The object portion of the IBM Spectrum Scale installer contains an option to indicate if multi-region support is to be enabled. If it needs to be enabled, the installer determines if the cluster being installed is the first cluster or if the cluster is a subsequent one and it will be added to an existing environment. This information is passed to the **mmobj swift base** command that the installer runs to install the object protocol.

To set up an initial multi-region environment, issue the following command on the 1st cluster after it has been installed:

```
mmobj multiregion enable
```

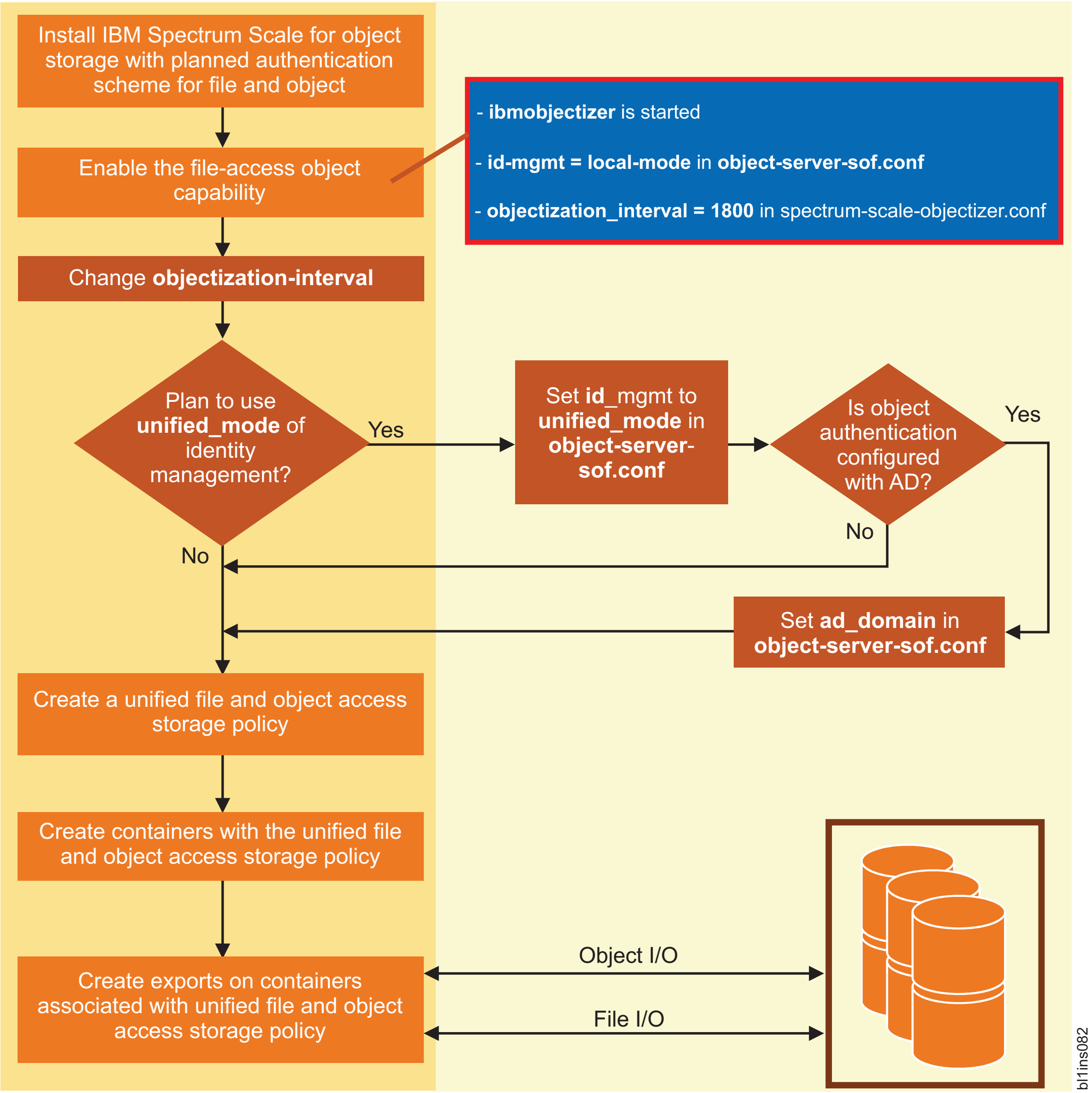
For information on adding a new region to a multi-region object deployment environment, see *Adding a region in a multi-region object deployment* in *IBM Spectrum Scale: Administration Guide*.

Installing and using unified file and object access

Unified file and object access gets installed when you install IBM Spectrum Scale for object storage in an IBM Spectrum Scale cluster. No additional steps are required for installing unified file and object access.

For information about managing and administering unified file and object access, see *Unified file and object access in IBM Spectrum Scale* in *IBM Spectrum Scale: Administration Guide*.

The high-level flow of administering unified file and object access is shown in the following diagram. The flow of changing the identity management mode for unified file and object access with the **id_mgmt** parameter is also shown.



Install IBM Spectrum Scale for object storage with planned authentication scheme for file and object
Enable the file-access object capability
- ibmobjectizer is started
- id-mgmt = local-mode in object-server-sof.conf
- objectization_interval = 1800 in spectrum-scale-objectizer.conf
Change objectization-interval
Plan to use unified_mode of identity management?
Yes
No
Set id_mgmt to unified_mode in object-server-sof.conf
Is object authentication configured with AD?
Yes
No
Set ad_domain in object-server-sof.conf
Create a unified file and object access storage policy
Create containers with the unified file and object access storage policy
Create exports on containers associated with unified file and object access storage policy
Object I/O
File I/O
bl1ins082

For detailed steps, see *Administering unified file and object access* in *IBM Spectrum Scale: Administration Guide*.

Enabling unified file and object access after upgrading from IBM Spectrum Scale 4.2 or later to 4.2.2 or later

Use these steps to enable unified file and object access after upgrading from IBM Spectrum Scale Version 4.2 or later to 4.2.2 or later.

1. Enable the file-access object capability.

   For detailed steps, see *Enabling the file-access object capability* in *IBM Spectrum Scale: Administration Guide*.

2. By default, the ibmobjectizer service interval is set to 30 minutes. Set the ibmobjectizer service interval to another value according to your requirement.

   For detailed steps, see *Setting up the objectizer service interval* in *IBM Spectrum Scale: Administration Guide*.

3. By default, the identify management mode for unified file and object access is set to `local_mode`. Change the identity management mode according to your requirement.

   For detailed steps, see *Configuring authentication and setting identity management modes for unified file and object access* in *IBM Spectrum Scale: Administration Guide*.

For the other unified file and object access tasks that you can perform, see *Administering unified file and object access* in *IBM Spectrum Scale: Administration Guide*.

Chapter 5. Installing IBM Spectrum Scale on AIX nodes

There are three steps for installing GPFS on AIX nodes.

Before you begin installation, read "Planning for GPFS" on page 133 and the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

Do not attempt to install GPFS if you do not have the prerequisites listed in "Hardware requirements" on page 133 and "Software requirements" on page 134.

Ensure that the **PATH** environment variable on each node includes **/usr/lpp/mmfs/bin**.

The installation process includes:

1. "Creating a file to ease the AIX installation process"
2. "Verifying the level of prerequisite software"
3. "Procedure for installing GPFS on AIX nodes" on page 320

Creating a file to ease the AIX installation process

Creation of a file that contains all of the nodes in your GPFS cluster prior to the installation of GPFS, will be useful during the installation process. Using either host names or IP addresses when constructing the file will allow you to use this information when creating your cluster through the **mmcrcluster** command.

For example, create the file **/tmp/gpfs.allnodes**, listing the nodes one per line:

```
k145n01.dpd.ibm.com
k145n02.dpd.ibm.com
k145n03.dpd.ibm.com
k145n04.dpd.ibm.com
k145n05.dpd.ibm.com
k145n06.dpd.ibm.com
k145n07.dpd.ibm.com
k145n08.dpd.ibm.com
```

Verifying the level of prerequisite software

Before you can install GPFS, verify that your system has the correct software levels installed.

If your system *does not* have the prerequisite AIX level, refer to the appropriate installation manual before proceeding with your GPFS installation. See the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) for the latest software levels.

To verify the software version, run the command:

```
WCOLL=/tmp/gpfs.allnodes dsh "oslevel"
```

The system should display output similar to:

```
7.1.0.0
```

Procedure for installing GPFS on AIX nodes

These installation procedures are generalized for all levels of GPFS. Ensure you substitute the correct numeric value for the modification (*m*) and fix (*f*) levels, where applicable. The modification and fix level are dependent upon the current level of program support.

Follow these steps to install the GPFS software using the **installp** command:

1. "Accepting the electronic license agreement"
2. "Creating the GPFS directory"
3. "Creating the GPFS installation table of contents file" on page 321
4. "Installing the GPFS man pages" on page 321
5. "Installing GPFS over a network" on page 321
6. "Verifying the GPFS installation" on page 321

Accepting the electronic license agreement

The GPFS software license agreement is shipped and viewable electronically. The electronic license agreement must be accepted before software installation can continue.

GPFS has three different editions based on functional levels:

- IBM Spectrum Scale Standard Edition
- IBM Spectrum Scale Advanced Edition
- IBM Spectrum Scale Data Management Edition

For IBM Spectrum Scale Standard Edition and IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition installations, the installation cannot occur unless the appropriate license agreements are accepted.

When using the **installp** command, use the **-Y** flag to accept licenses and the **-E** flag to view license agreement files on the media.

GPFS license agreements are retained on the AIX system after installation completes. These license agreements can be viewed after installation in the following directories:

`/usr/swlag/GPFS_standard` (IBM Spectrum Scale Standard Edition)

`/usr/swlag/GPFS_advanced` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition)

Creating the GPFS directory

To create the GPFS directory:

1. On any node create a temporary subdirectory where GPFS installation images will be extracted. For example:
   ```
   mkdir /tmp/gpfslpp
   ```
2. Copy the installation images from the CD-ROM to the new directory, by issuing:
   ```
   bffcreate -qvX -t /tmp/gpfslpp -d /dev/cd0/AIX all
   ```
   This command places these GPFS installation files in the images directory:
   ```
   gpfs.base
   gpfs.docs.data
   gpfs.msg.en_US
   gpfs.license.xx
   gpfs.compression
   gpfs.gskit
   ```

`gpfs.ext`

`gpfs.crypto` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only)

`gpfs.adv` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only)

Creating the GPFS installation table of contents file

To create the GPFS installation table of contents file:

1. Make the new image directory the current directory:
   ```
   cd /tmp/gpfslpp
   ```
2. Use the **inutoc .** command to create a **.toc** file. The **.toc** file is used by the **installp** command.
   ```
   inutoc .
   ```

Installing the GPFS man pages

In order to use the GPFS man pages you must install the **gpfs.docs.data** image.

The GPFS manual pages will be located at **/usr/share/man/**.

Installation consideration: The **gpfs.docs.data** image need not be installed on all nodes if man pages are not desired or local file system space on the node is minimal.

Installing GPFS over a network

Install GPFS according to these directions, where *localNode* is the name of the node on which you are running:

1. If you are installing on a shared file system network, ensure the directory where the GPFS images can be found is NFS exported to all of the nodes planned for your GPFS cluster (**/tmp/gpfs.allnodes**).
2. Ensure an acceptable directory or mountpoint is available on each target node, such as **/tmp/gpfslpp**. If there is not, create one:
   ```
   WCOLL=/tmp/gpfs.allnodes mmdsh "mkdir /tmp/gpfslpp"
   ```
3. If you are installing on a shared file system network, to place the GPFS images on each node in your network, issue:
   ```
   WCOLL=/tmp/gpfs.allnodes mmdsh "mount localNode:/tmp/gpfslpp /tmp/gpfslpp"
   ```

   Otherwise, issue:
   ```
   WCOLL=/tmp/gpfs.allnodes mmdsh "rcp localNode:/tmp/gpfslpp/gpfs* /tmp/gpfslpp"
   WCOLL=/tmp/gpfs.allnodes mmdsh "rcp localNode:/tmp/gpfslpp/.toc /tmp/gpfslpp"
   ```
4. Install GPFS on each node:
   ```
   WCOLL=/tmp/gpfs.allnodes mmdsh "installp -agXYd /tmp/gpfslpp gpfs"
   ```

For information about using `rcp` with IBM Spectrum Scale, see “Remote file copy command” on page 145.

Verifying the GPFS installation

Verify that the installation procedure placed the required GPFS files on each node by running the **lslpp** command on *each* node:

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

The system returns output similar to the following:

```
Fileset                          Level      State Description
----------------------------------------------------------------------------
Path: /usr/lib/objrepos
```

```
gpfs.adv                          5.0.0.0     COMMITTED GPFS Advanced Features
gpfs.base                         5.0.0.0     COMMITTED GPFS File Manager
gpfs.crypto                       5.0.0.0     COMMITTED GPFS Cryptographic Subsystem
gpfs.ext                          5.0.0.0     COMMITTED GPFS Extended Features
gpfs.gskit                        8.0.50-79   COMMITTED GPFS GSKit Cryptography Runtime
gpfs.compression                  5.0.0.0     COMMITTED GPFS Compresson Libraries
gpfs.msg.en_US                    5.0.0.0     COMMITTED GPFS Server Messages - U.S. English
gpfs.license.std                  5.0.0.0     COMMITTED IBM Spectrum Scale Standard Edition License

Path: /etc/objrepos
gpfs.base                         5.0.0.0     COMMITTED GPFS File Manager

Path: /usr/share/lib/objrepos
gpfs.docs.data                    5.0.0.0     COMMITTED GPFS Server Manpages and Documentation
```

The output that is returned on your system can vary depending on whether you have IBM Spectrum Scale Standard Edition or IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition installed.

Note: The path returned by `lslpp -l` shows the location of the package control data used by **installp**. The listed path does not show GPFS file locations. To view GPFS file locations, use the `-f` flag.

Chapter 6. Installing IBM Spectrum Scale on Windows nodes

There are several steps for installing GPFS on Windows nodes. The information in this topic points you to the detailed steps.

Before you begin installation, read the following:
- "Planning for GPFS" on page 133
- The IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html)
- "GPFS for Windows overview" and all of its subtopics

Do not install GPFS unless you have the prerequisites listed in "Hardware requirements" on page 133 and "Software requirements" on page 134.

The installation process includes:
1. "Installing GPFS prerequisites" on page 327
2. "Procedure for installing GPFS on Windows nodes" on page 330
3. "Configuring a mixed Windows and UNIX cluster" on page 332

To install GPFS for Windows, first configure your Windows systems as described in "Installing GPFS prerequisites" on page 327. This includes steps such as joining an Active Directory domain and installing Cygwin from the Cygwin website (www.cygwin.com), which provides a UNIX-like environment on Windows. GPFS installation is simple once the prerequisites are completed. Finally, if your system will be part of a cluster that includes UNIX nodes, follow the steps described in "Configuring a mixed Windows and UNIX cluster" on page 332. This includes creating the GPFS Administration service, installing OpenSSH, and other requirements. Complete this process before performing configuration steps common to all GPFS supported platforms.

Note: Throughout this information, UNIX file name conventions are used. For example, the GPFS cluster configuration data is stored in the **/var/mmfs/gen/mmsdrfs** file. On Windows, the UNIX namespace starts under the Cygwin installation directory, which by default is **%SystemDrive%\cygwin64**, so the GPFS cluster configuration data is stored in the **C:\cygwin64\var\mmfs\gen\mmsdrfs** file.

GPFS for Windows overview

GPFS for Windows participates in a new or existing GPFS cluster in conjunction with AIX and Linux systems.

Support includes the following:
- Core GPFS parallel data services
- Windows file system semantics
- A broad complement of advanced GPFS features
- User identity mapping between Windows and UNIX
- Access to GPFS 3.4 and/or later file systems

Identity mapping between Windows and UNIX user accounts is a key feature. System administrators can explicitly match users and groups defined on UNIX with those defined on Windows. Users can maintain file ownership and access rights from either platform. System administrators are not required to define an identity map. GPFS automatically creates a mapping when one is not defined.

GPFS supports the unique semantic requirements posed by Windows. These requirements include case-insensitive names, NTFS-like file attributes, and Windows file locking. GPFS provides a bridge between a Windows and POSIX view of files, while not adversely affecting the functions provided on AIX and Linux.

GPFS for Windows provides the same core services to parallel and serial applications as are available on AIX and Linux. GPFS gives parallel applications simultaneous access to files from any node that has GPFS mounted, while managing a high level of control over all file system operations. System administrators and users have a consistent command interface on AIX, Linux, and Windows. With few exceptions, the commands supported on Windows are identical to those on other GPFS platforms. See "GPFS limitations on Windows" for a list of commands that Windows clients do not support.

GPFS support for Windows

The IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) lists the levels of the Windows operating system that are supported by GPFS.

Limited GPFS support for the Windows platform first appeared in GPFS 3.2.1. Subsequent GPFS releases have expanded the set of available functions and features so that now most GPFS capabilities supported on the UNIX platforms are also supported on Windows.

GPFS limitations on Windows

GPFS for Windows supports most of the GPFS features that are available on AIX and Linux, but some limitations apply.

The following limitations apply to configuring and operating a file system on a Windows node:

- File systems must have been created with GPFS 3.1.2.5 or later.
- You cannot upgrade an existing file system that was created with GPFS 3.1 or earlier.

The remaining GPFS for Windows limitations apply only to the Windows nodes in a cluster:

- The following GPFS commands are not supported on Windows:
  - **mmafm*** commands
  - **mmapplypolicy**
  - **mmaudit*** commands
  - **mmbackup**
  - **mmbackupconfig**, **mmrestoreconfig**
  - **mmcall*** commands
  - **mmcdp*** commands
  - **mces*** commands
  - **mmcheckquota**, **mmdefedquota**, **mmedquota**, **mmlsquota**, **mmrepquota**, **mmsetquota**
  - **mmclone**
  - **mmcloud*** commands
  - **mmdelacl**, **mmeditacl**
  - **mmhadoop*** commands
  - **mmhealth**
  - **mmimgbackup**, **mmimgrestore**
  - **mmkeyserv**
  - **mmnetverify**
  - **mmnfs*** commands

  - **mmperfmon**, **mmpmon**
  - **mmqosnice**
  - **mmsed**
  - **mmsmb*** commands
  - **mmsnmp*** commands
  - **mmsysmon*** commands
  - **mmuserauth**
  - All **GNR** commands
- The GPFS Application Programming Interfaces (APIs) are not supported on Windows.
- The native Windows backup utility is not supported.
- Symbolic links that are created on UNIX-based nodes are specially handled by GPFS Windows nodes; they appear as regular files with a size of 0 and their contents cannot be accessed or modified.
- GPFS on Windows nodes attempts to preserve data integrity between memory-mapped I/O and other forms of I/O on the same computation node. However, if the same file is memory mapped on more than one Windows node, data coherency is not guaranteed between the memory-mapped sections on these multiple nodes. In other words, GPFS on Windows does not provide distributed shared memory semantics. Therefore, applications that require data coherency between memory-mapped files on more than one node might not function as expected.
- GPFS on Windows is not supported on the AFM cluster.

File system name considerations

GPFS file system names should be no longer than 31 characters if the file system will be mounted on a Windows node. GPFS file system names are used as Windows file system labels, and Windows limits the length of these labels to 31 characters. Any attempt to mount a file system with a long name will fail.

You can rename a file system using a command like the following:

```
mmchfs gpfs_file_system_name_that_is_too_long -W gpfs_name_1
```

File name considerations

File names created on UNIX-based GPFS nodes that are not valid on Windows are transformed into valid short names. A name may not be valid either because it is a reserved name like NUL or COM2, or because it has a disallowed character like colon (:) or question mark (?).

See the MSDN Library description of Naming Files, Paths, and Namespaces (msdn.microsoft.com/en-us/library/aa365247(VS.85).aspx) for complete details.

Windows applications can use short names to access GPFS files with Windows file names that are not valid. GPFS generates unique short names using an internal algorithm. You can view these short names using **dir /x** in a DOS command prompt.

Table 43 shows an example:

Table 43. Generating short names for Windows

UNIX	Windows
foo+bar.foobar	FO~23Q_Z.foo
foo|bar.-bar	FO~TD}C5._ba
f|bar.-bary	F_AMJ5!._ba

Case sensitivity

Native GPFS is case-sensitive; however, Windows applications can choose to use case-sensitive or case-insensitive names.

This means that case-sensitive applications, such as those using Windows support for POSIX interfaces, behave as expected. Native Win32 applications (such as Windows Explorer) have only case-aware semantics.

The case specified when a file is created is preserved, but in general, file names are case insensitive. For example, Windows Explorer allows you to create a file named **Hello.c**, but an attempt to create **hello.c** in the same folder will fail because the file already exists. If a Windows node accesses a folder that contains two files that are created on a UNIX node with names that differ only in case, Windows inability to distinguish between the two files might lead to unpredictable results.

Antivirus software

If more than one GPFS Windows node is running antivirus software that scans directories and files, shared files only need to be scanned by one GPFS node. It is not necessary to scan shared files more than once.

When you run antivirus scans from more than one node, schedule the scans to run at different times to allow better performance of each scan, as well as to avoid any conflicts that might arise because of concurrent exclusive access attempts by the antivirus software from multiple nodes. Note that enabling real-time antivirus protection for GPFS volumes could significantly degrade GPFS performance and cause excessive resource consumption.

Tip: Consider using a single, designated Windows node to perform all virus scans.

Differences between GPFS and NTFS

GPFS differs from the Microsoft Windows NT File System (NTFS) in its degree of integration into the Windows administrative environment, Windows Explorer, and the desktop.

The differences are as follows:

- Manual refreshes are required to see any updates to the GPFS namespace.
- You cannot use the recycle bin.
- You cannot use distributed link tracking. This is a technique through which shell shortcuts and OLE links continue to work after the target file is renamed or moved. Distributed link tracking can help you locate the link sources in case the link source is renamed or moved to another folder on the same or different volume on the same computer, or moved to a folder on any computer in the same domain.
- You cannot use NTFS change journaling. This also means that you cannot use the Microsoft Indexing Service or Windows Search Service to index and search files and folders on GPFS file systems.

GPFS does not support the following NTFS features:

- File compression (on individual files or on all files within a folder)
- Encrypted directories
- Encrypted files (GPFS file encryption is administered through GPFS-specific commands. For more information, see *Encryption* in *IBM Spectrum Scale: Administration Guide*.)
- Quota management (GPFS quotas are administered through GPFS-specific commands)
- Reparse points
- Defragmentation and error-checking tools
- Alternate data streams
- Directory Change Notification

- The assignment of an access control list (ACL) for the entire drive
- A change journal for file activity
- The scanning of all files or directories that a particular SID owns (**FSCTL_FIND_FILES_BY_SID**)
- Generation of AUDIT and ALARM events specified in a System Access Control List (SACL). GPFS is capable of storing SACL content, but will not interpret it.
- Windows sparse files API
- Transactional NTFS (also known as TxF)

Access control on GPFS file systems

GPFS provides support for the Windows access control model for file system objects.

Each GPFS file or directory has a Security Descriptor (SD) object associated with it and you can use the standard Windows interfaces for viewing and changing access permissions and object ownership (for example, Windows Explorer Security dialog panel). Internally, a Windows SD is converted to an NFS V4 access control list (ACL) object, which ensures that access control is performed consistently on other supported operating systems. GPFS supports all discretionary access control list (DACL) operations, including inheritance. GPFS is capable of storing system access control list (SACL) objects, but generation of AUDIT and ALARM events specified in SACL contents is not supported.

An important distinction between GPFS and Microsoft Windows NT File Systems (NTFS) is the default set of permissions for the root (top-level) directory on the file system. On a typical NTFS volume, the DACL for the top-level folder has several inheritable entries that grant full access to certain special accounts, as well as some level of access to nonprivileged users. For example, on a typical NTFS volume, the members of the local group **Users** would be able to create folders and files in the top-level folder. This approach differs substantially from the traditional UNIX convention where the root directory on any file system is only writable by the local **root** superuser by default. GPFS adheres to the latter convention; the root directory on a new file system is only writable by the UNIX user **root**, and does not have an extended ACL when the file system is created. This is to avoid impacting performance in UNIX-only environments, where the use of extended ACLs is not common.

When a new GPFS file system is accessed from a Windows client for the first time, a security descriptor object is created for the root directory automatically, and it will contain a noninheritable DACL that grants full access to the local **Administrators** group and read-only access to the local **Everyone** group. This allows only privileged Windows users to create new files and folders. Because the root directory DACL has no inheritable entries, new objects will be created with a default DACL that grants local **Administrators** and **SYSTEM** accounts full access. Optionally, the local system administrator could create a subdirectory structure for Windows users, and explicitly set DACLs on new directories as appropriate (for example, giving the necessary level of access to nonprivileged users).

Note: Some applications expect to find NTFS-style permissions on all file systems and they might not function properly when that is not the case. Running such an application in a GPFS folder where permissions have been set similar to NTFS defaults might correct this.

Installing GPFS prerequisites

This topic provides details on configuring Windows systems prior to installing GPFS.

Perform the following steps:

1. "Configuring Windows" on page 328
   a. "Assigning a static IP address" on page 328
   b. "Joining an Active Directory domain" on page 328
   c. "Disabling User Account Control" on page 328
   d. "Disabling the Windows firewall" on page 329

e. "Installing the Tracefmt and Tracelog programs (optional)" on page 329

2. From here on, all GPFS installation steps must be executed as a special user with Administrator privileges. Further, all GPFS administrative operations (such as creating a cluster and file systems) must also be issued as the same user. This special user will depend on your cluster type. For Windows homogeneous clusters, run GPFS commands as a member of the **Domain Admins** group or as a domain account that is a member of the local **Administrators** group. For clusters with both Windows and UNIX nodes, run GPFS commands as **root**, a special domain user account described in "Creating the GPFS administrative account" on page 333.
3. "Installing Cygwin" on page 329

See the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) for the latest:

- Software recommendations
- Configuration information

For additional requirements when setting up clusters that contain both Windows nodes and AIX or Linux nodes, see "Configuring a mixed Windows and UNIX cluster" on page 332.

Configuring Windows

This topic provides some details on installing and configuring Windows on systems that will be added to a GPFS cluster.

The IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) lists the versions of the Windows operating system that are supported by GPFS.

Assigning a static IP address

GPFS communication requires invariant static IP addresses for each GPFS node.

Joining an Active Directory domain

All Windows systems in the same GPFS cluster should be members of the same Active Directory domain. Join these systems to the Windows domain before adding them to a GPFS cluster.

GPFS expects that all Windows nodes in a cluster are members of the same domain. This gives domain users a consistent identity and consistent file access rights independent of the system they are using. The domain controllers, which run the Active Directory Domain Services, are not required to be members of the GPFS cluster.

Refer to your Windows Server documentation for information on how to install and administer Active Directory Domain Services.

Disabling User Account Control

On Windows Server 2008 nodes, you must disable User Account Control (UAC) for GPFS to operate correctly. UAC needs to be disabled for the entire system, not just turned off for GPFS administrative users.

However, if UAC is enabled, some GPFS functions may not work properly. Therefore, it is recommended that UAC be disabled, regardless of the Windows OS version. On the systems with UAC enabled, GPFS administrative users run in **Admin Mode** such as when a program is started with the **Run as Administrator** option.

To disable UAC on Windows systems, follow these steps:

1. Open the **System Configuration** application under **Administrative Tools**.

2. Select the **Tools** tab and scroll down to select **Disable UAC**.
3. Click **Launch**.

Note: This change requires a reboot.

Limitations: Windows is responsible for providing file ownership information to GPFS. If UAC is disabled, the default ownership of files may be affected. Files from users who are members of an administrative group will be owned by the administrative group, but files from users without administrative membership will be mapped to their user ID. This restriction is documented on the Microsoft Support website (support.microsoft.com/kb/947721).

Disabling the Windows firewall

GPFS requires that you modify the default Windows Firewall settings.

The simplest change that will allow GPFS to operate properly is to disable the firewall. Open **Windows Firewall** in the Control Panel and click **Turn Windows firewall on or off**, and select **Off** under the General tab. For related information, see *GPFS port usage* in *IBM Spectrum Scale: Administration Guide*.

Installing the Tracefmt and Tracelog programs (optional)

GPFS diagnostic tracing (**mmtracectl**) on Windows uses the Microsoft programs called **tracefmt.exe** and **tracelog.exe**. These programs are not included with Windows but can be downloaded from Microsoft. The **tracefmt.exe** and **tracelog.exe** programs are only for tracing support and are not required for normal GPFS operations.

The **tracefmt.exe** and **tracelog.exe** programs are included with the Windows Driver Kit (WDK) as well as the Windows SDK. You can download either of these packages from the Microsoft Download Center (www.microsoft.com/download).

To allow GPFS diagnostic tracing on Windows using the WDK, follow these steps:
1. Download the latest version of the Windows Driver Kit (WDK) from Microsoft.
2. Install the Tools feature of the WDK on some system to obtain a copy of **tracefmt.exe** and **tracelog.exe**.
3. Locate and copy **tracefmt.exe** and **tracelog.exe** to both the **/usr/lpp/mmfs/bin** and **/usr/lpp/mmfs/win** directories to ensure **mmtracectl** properly locates these programs.

For additional information about GPFS diagnostic tracing, see *The GPFS trace facility* in *IBM Spectrum Scale: Problem Determination Guide*.

Installing Cygwin

Cygwin is a POSIX environment available for Windows and can be downloaded from the Cygwin website (www.cygwin.com). GPFS uses this component to support many of its administrative scripts. System administrators have the option of using either the **GPFS Admin Command Prompt** or **GPFS Admin Korn Shell** to run GPFS commands.

Cygwin must be installed before installing GPFS. It is a software package that provides a Unix-like environment on Windows and provides runtime support for POSIX applications and includes programs such as **grep**, **ksh**, **ls**, and **ps**.

When running Cygwin setup, only the standard packages are installed by default. GPFS requires installation of additional packages, which are listed in "Installing the 64-bit version of Cygwin" on page 330.

Note: Starting with GPFS 4.1.1, the 32-bit version of Cygwin is no longer supported for Windows nodes running GPFS. Users that are running GPFS 4.1 with the 32-bit version of Cygwin installed must upgrade to the 64-bit version of Cygwin before installing GPFS 4.1.1. For more information, see "Upgrading from the 32-bit version of Cygwin to the 64-bit version of Cygwin." For users on GPFS releases prior to 4.1 (SUA based), see "Upgrading from GPFS 3.5" on page 384.

Upgrading from the 32-bit version of Cygwin to the 64-bit version of Cygwin

Follow these instructions to upgrade from the 32-bit version of Cygwin to the 64-bit version of Cygwin:

1. Uninstall GPFS 4.1 and reboot.
2. Uninstall IBM GSKit for GPFS and reboot.
3. Uninstall the GPFS 4.1 license.
4. Stop and delete any Cygwin 32-bit services, such as OpenSSH, that might have been configured.
5. Do *not* uninstall the 32-bit version of Cygwin yet, or you may lose GPFS configuration information.
6. Install the 64-bit version of Cygwin using the instructions in "Installing the 64-bit version of Cygwin."
7. Install the GPFS 4.1.1 license for the appropriate edition; for example, `gpfs.ext-4.1.1-Windows-license.msi`.
8. Install the appropriate GPFS 4.1.1 edition; for example, `gpfs.ext-4.1.1.`*x*`-Windows.msi`.
9. Install IBM GSKit for GPFS.
10. Uninstall the 32-bit version of Cygwin completely.
11. Follow the procedures in "Installing and configuring OpenSSH on Windows nodes" on page 334.

Installing the 64-bit version of Cygwin

To install the 64-bit version of Cygwin for Windows, follow these steps:

1. Logon to the Windows node as the account you will use for GPFS administrative operations.
2. Go to the Cygwin website (www.cygwin.com), and click on the **Install Cygwin** link in the upper-left pane.
3. Download and start the installation of the **setup-x86_64.exe** file.
4. Follow the prompts to continue with the install. The default settings are recommended until the Select Packages dialog is displayed. Then, select the following packages (use the Search field to quickly find these packages):
   - diffutils: A GNU collection of diff utilities
   - flip: Convert text file line endings between Unix and DOS formats
   - m4: GNU implementation of the traditional Unix macro processor
   - mksh: MirBSD Korn Shell
   - perl: Perl programming language interpreter
   - procps: System and process monitoring utilities
   - openssh: The OpenSSH server and client programs (only required if you plan on mixing Linux, AIX and Windows nodes in the same cluster)
5. Click the **Next** button, and continue to follow the prompts to complete the installation.
6. If you are using a mixed cluster environment (Windows and Linux/AIX), follow the steps in "Installing and configuring OpenSSH on Windows nodes" on page 334.

Procedure for installing GPFS on Windows nodes

IBM provides GPFS as Windows Installer packages (MSI), which allow both interactive and unattended installations. Perform the GPFS installation steps as the **Administrator** or some other member of the **Administrators** group.

Before installing GPFS on Windows nodes, verify that all the installation prerequisites have been met. For more information, see "Installing GPFS prerequisites" on page 327.

To install GPFS on a Windows node, follow these steps:

1. Run the following license installation package from the product media and accept the license:

   `gpfs.ext-5.0.x.x-Windows-license.msi` (IBM Spectrum Scale Standard Edition)

   Rebooting is normally not required.
2. Download and install the latest service level of GPFS from the IBM Support Portal: Downloads for General Parallel File System(www.ibm.com/support/entry/portal/Downloads/Software/Cluster_software/General_Parallel_File_System). The GPFS package name includes the GPFS version (for example, `gpfs.ext-5.0.x.x-Windows.msi`). For the latest information on the supported Windows operating system versions, see the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

   Under some circumstances, the installation process will prompt you to reboot the systems when it is required. You do not need to install the GPFS 5.0.x.x package included on the media before installing the latest update.
3. Download and install the latest level of IBM GSKit for GPFS. GSKit (**gpfs.gskit-*x.x.x.x*.msi**) comes as a single package that works with the IBM Spectrum Scale Standard Edition.

   Note: For this release, the IBM GSKit version must be at least 8.0.50-79 or higher.

To upgrade GPFS, do the following:

1. Uninstall your current GPFS packages (for example, `gpfs.base-4.x.x.0-Windows.msi`, `gpfs.ext-4.x.x.0-Windows.msi`, and `gpfs.gskit-8.0.50.x.msi`).
2. Reboot.
3. Install the latest PTF packages (for example, `gpfs.ext-5.0.x.x-Windows.msi`).

When upgrading, you do not need to uninstall and reinstall the license package unless you are explicitly instructed to do so by IBM. In addition, **gpfs.gskit** only needs to be upgraded if the update (zip file) contains a newer version.

For more information, refer to Chapter 3, “Steps for establishing and starting your IBM Spectrum Scale cluster,” on page 221.

The GPFS installation package provides some options that you can select by installing GPFS from a Command Prompt. Property values control the options. The command line syntax is:

```
msiexec.exe /package <Package> [Optional Parameters] [Property=Value]
```

The following property values are available:

AgreeToLicense=yes

This property applies to the license package `gpfs.ext-5.0.x-Windows-license.msi`. It allows GPFS to support an installation procedure that does not require user input. IBM recommends that you perform at least one interactive installation on a typical system before attempting an unattended installation. This will help identify any system configuration issues that could cause the installation to fail.

The following command installs GPFS without prompting you for input or generating any error dialog boxes:

```
msiexec /package gpfs.ext-5.0.x-Windows-license.msi /passive AgreeToLicense=yes
```

The **msiexec.exe** executable file supports display options other than **/passive**. See **msiexec** documentation for details.

RemoteShell=no

This property applies to the product package (for example, `gpfs.ext-5.0.x-Windows-license.msi`). It is equivalent to running **mmwinservctl set –remote-shell no**, but performs this configuration change before **mmwinserv** initially starts. This option is available to satisfy security policies that restrict network communication protocols.

You can verify that GPFS is installed correctly by opening the **Programs and Features** control panel. **IBM Spectrum Scale** should be included in the list of installed programs. The program's version should match the version of the update package.

The GPFS software license agreement is shipped and is viewable electronically. The license agreement will remain available in the `%SystemDrive%\Program Files\IBM\GPFS\5.0.0.0\license` directory for future access.

Running GPFS commands

Once GPFS is installed, the account you use for GPFS administrative operations (such as creating a cluster and file systems) will depend on your cluster type.

For Windows clusters, run GPFS commands as a member of the **Domain Admins** group or as a domain account that is part of the local **Administrators** group. For clusters with both Windows and UNIX nodes, run GPFS commands as **root**, a special domain user account described in “Creating the GPFS administrative account” on page 333.

You must run all GPFS commands from either a **GPFS Admin Command Prompt** or a **GPFS Admin Korn Shell**. The GPFS administrative commands may not work properly if issued from the Cygwin Terminal. Open a new **GPFS Admin Command Prompt** or **GPFS Admin Korn Shell** after installing GPFS so that it uses the updated **PATH** environment variable required to execute GPFS commands.

GPFS Admin Command Prompt and **GPFS Admin Korn Shell** can be accessed using their desktop shortcuts or from under the **IBM GPFS** Start menu group. Each of these program links starts its respective shell using the **Run as Administrator** option.

Configuring a mixed Windows and UNIX cluster

For GPFS clusters that include both Windows and UNIX nodes, this topic describes the additional configuration steps needed beyond those described in “Installing GPFS prerequisites” on page 327.

For mixed clusters, perform the following steps:

1. Optionally, install and configure identity mapping on your Active Directory domain controller (see “Identity Management for UNIX (IMU)”).
2. Create the root administrative account (see “Creating the GPFS administrative account” on page 333).
3. Edit the Domain Group Policy to give root the right to log on as a service (see “Allowing the GPFS administrative account to run as a service” on page 333.
4. Configure the GPFS Administration service (**mmwinserv**) to run as root (see “Configuring the GPFS Administration service” on page 334).
5. Install and configure OpenSSH (see “Installing and configuring OpenSSH on Windows nodes” on page 334).

Complete this process before performing configuration steps common to all GPFS supported platforms.

Identity Management for UNIX (IMU)

GPFS can exploit a Windows Server feature called Identity Management for UNIX (IMU) to provide consistent identities among all nodes in a cluster.

GPFS expects that all Windows nodes in a cluster are members of the same Active Directory domain. This gives domain users a consistent identity and consistent file access rights independent of the system they are using.

In addition, GPFS can exploit the Identity Management for UNIX (IMU) service for mapping users and groups between Windows and UNIX. IMU is an optional component of Microsoft Windows Server (starting with Server 2003 R2) that can be installed on domain controllers. GPFS does not require IMU.

For IMU installation and configuration information, see *Identity management on Windows* in *IBM Spectrum Scale: Administration Guide*.

Creating the GPFS administrative account

GPFS uses an administrative account in the Active Directory domain named **root** in order to interoperate with UNIX nodes in the cluster. Create this administrative account as follows:

1. Create a domain user with the logon name **root**.
2. Add user **root** to the **Domain Admins** group or to the local **Administrators** group on each Windows node.
3. In **root Properties/Profile/Home/LocalPath**, define a **HOME** directory such as **C:\Users\root\home** that does not include spaces in the path name and is not the same as the profile path.
4. Give **root** the right to log on as a service as described in "Allowing the GPFS administrative account to run as a service."

Step 3 is required for the Cygwin environment (described in "Installing Cygwin" on page 329) to operate correctly. Avoid using a path that contains a space character in any of the path names. Also avoid using **root**'s profile path (for example, **C:\User\root**). OpenSSH requires specific permissions on this directory, which can interfere with some Windows applications.

You may need to create the **HOME** directory on each node in your GPFS cluster. Make sure that **root** owns this directory.

Allowing the GPFS administrative account to run as a service

Clusters that depend on a **root** account to interoperate with UNIX nodes in a cluster will need to configure the GPFS Administrative Service (**mmwinserv**) to run as the **root** account. For this, **root** needs to be assigned the right to log on as a service. See "Configuring the GPFS Administration service" on page 334 for details.

The right to log on as a service is controlled by the Local Security Policy of each Windows node. You can use the Domain Group Policy to set the Local Security Policy on all Windows nodes in a GPFS cluster.

The following procedure assigns the *log on as a service* right to an account when the domain controller is running on Windows Server 2008:

1. Open **Group Policy Management** (available under **Administrative Tools**).
2. In the console tree, expand ***Forest name*/Domains/*Domain name*/Group Policy Objects**.
3. Right click **Default Domain Policy** and select **Edit**.
4. In the console tree of the Group Policy Management Editor, expand down to **Computer Configuration/Policies/Windows Settings/Security Settings/Local Policies/User Rights Assignment**.
5. Double click the **Log on as a service policy**.
6. Check **Define these policy settings if necessary**.
7. Use **Add User or Group...** to include the ***DomainName*\root** account in the policy, then click **OK**.

Refer to your *Windows Server* documentation for a full explanation of Local Security Policy and Group Policy Management.

Configuring the GPFS Administration service

GPFS for Windows includes a service called **mmwinserv**. In the Windows Services management console, this service has the name GPFS Administration. **mmwinserv** supports GPFS operations such as **autoload** and remote command execution in Windows GPFS clusters. The Linux and AIX versions of GPFS do not have a similar component. The **mmwinserv** service is used on all Windows nodes starting with GPFS 3.3.

The GPFS installation package configures **mmwinserv** to run using the default **LocalSystem** account. This account supports Windows GPFS clusters. For clusters that include both Windows and UNIX nodes, you must configure **mmwinserv** to run as **root**, the GPFS administrative account. Unlike **LocalSystem**, **root** can access the Identity Management for UNIX (IMU) service and can access other GPFS nodes as required by some cluster configurations.

For IMU installation and configuration information, see *Identity management on Windows* in *IBM Spectrum Scale: Administration Guide*. For information on supporting administrative access to GPFS nodes, see the *Requirements for administering a GPFS file system* topic in the *IBM Spectrum Scale: Administration Guide*.

Before configuring **mmwinserv** to run as **root**, you must first grant **root** the right to run as a service. For details, see “Allowing the GPFS administrative account to run as a service” on page 333.

Use the GPFS command **mmwinservctl** to set and maintain the GPFS Administration service's log on account. **mmwinservctl** must be run on a Windows node. You can run **mmwinservctl** to set the service account before adding Windows nodes to a cluster. You can also use this command to change or update the account on nodes that are already in a cluster. GPFS can be running or stopped when executing **mmwinservctl**, however, refrain from running other GPFS administrative commands at the same time.

In this example, **mmwinservctl** configures three nodes before they are added to a GPFS cluster containing both Windows and UNIX:

```
mmwinservctl set -N node1,node2,node3 --account mydomain/root --password mypwd --remote-shell no
```

Whenever root's password changes, the **mmwinserv** logon information needs to be updated to use the new password. The following command updates on all Windows nodes in a cluster with a new password:

```
mmwinservctl set -N all --password mynewpwd
```

As long as **mmwinserv** is running, the service will not be affected by an expired or changed password and GPFS will continue to function normally. However, GPFS will not start after a system reboot when **mmwinserv** is configured with an invalid password. If for any reason the Windows domain or root password changes, then **mmwinservctl** should be used to update the domain and password. The domain and password can also be updated on a per node basis by choosing **Administrative Tools** > **Computer Management** > **Services and Applications** > **Services**, and selecting **GPFS Administration**. Choose **File** > **Properties** > **Logon** and update the *<domain>\username* and the password.

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

Installing and configuring OpenSSH on Windows nodes

If using a mixed cluster, OpenSSH must be configured on the Windows nodes. Refer to the Cygwin FAQ (www.cygwin.com/faq.html) and documentation on how to setup **sshd**. Replace the usage of the account **cyg_server** in the Cygwin documentation with **root** when setting up a privileged account for **sshd**.

The following are some guidelines in addition to the Cygwin instructions on setting up **sshd**:

1. Verify that all nodes can be pinged among themselves by host name, Fully Qualified Domain Name (FQDN) and IP address.

2. If not using IPv6, disable it. For more information, see How to disable IPv6 or its components in Windows (support.microsoft.com/kb/929852).
3. Check that passwd contains the privileged user that you plan to use for GPFS operations, as well as its correct home path:

```
$ cat /etc/passwd | grep "root"

root:unused:11103:10513:U-WINGPFS\root,S-1-5-21-3330551852-1995197583-3793546845-1103:/cygdrive/c/home/root:/bin/bash
```

   If the user is not listed, rebuild your passwd:

   ```
   mkpasswd -l -d wingpfs > /etc/passwd
   ```

4. From the Cygwin shell, run **/usr/bin/ssh-host-config** and respond **yes** to the prompts. When prompted to enter the value of CYGWIN for the daemon, enter **ntsec**. Specify **root** in response to the query for the new user name. You may receive the following warning:

   ```
   ***Warning:  The specified account 'root' does not have the
   ***Warning:  required permissions or group memberships.  This may
   ***Warning:  cause problems if not corrected; continuing...
   ```

   As long as the account (in this case, **root**) is in the local Administrators group, you can ignore this warning.
5. When the installation is complete, enter the following:

   ```
   $ net start sshd

   The CYGWIN sshd service is starting.
   The CYGWIN sshd service was started successfully.
   ```

Note: The OpenSSH READMEs are available at /usr/share/doc/openssh. Also see the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

Once OpenSSH is installed, the GPFS administrative account **root** needs to be configured so that it can issue **ssh** and **scp** commands without requiring a password and without producing any extraneous messages. This kind of passwordless access is required from any node used for GPFS administration to all other nodes in the cluster.

For additional information, see *Requirements for administering a GPFS file system* in *IBM Spectrum Scale: Administration Guide* and *Troubleshooting Windows problems* in *IBM Spectrum Scale: Problem Determination Guide*.

Configuring the Windows HPC server

In order for Windows HPC Server to function with GPFS, disable dynamic hosts file and re-enable dynamic DNS updates by doing the following

1. On all nodes in the Windows HPC cluster, open the hosts file **%systemroot%\system32\drivers\etc\hosts** and change **ManageFile = true** to **ManageFile = false**.
2. On the HPC head node, execute the following from HPC PowerShell, assuming all nodes are part of the head node's Active Directory domain.

   Go to **Start > All Programs > Microsoft HPC Pack > HPC PowerShell** and execute:

   ```
   Set-HpcNetwork -EnterpriseDnsRegistrationType WithConnectionDnsSuffix
   ```

Chapter 7. Installing Cloud services on IBM Spectrum Scale nodes

The Cloud services installation is not included in the IBM Spectrum Scale installation toolkit workflow. It needs to be installed separately on top of the IBM Spectrum Scale cluster. However, the Cloud services RPMs are available with the IBM Spectrum Scale package.

Before you begin, ensure that you have created a node class. For more information, see the *Creating a node class* topic in the *IBM Spectrum Scale: Administration Guide*. Also, ensure that your server meets the required prerequisites. For more information, see "Software requirements for Cloud services" on page 209

Creating a user-defined node class for Transparent cloud tiering or Cloud data sharing

This topic provides step-by-step instructions for creating and configuring the required node class to specify which IBM Spectrum Scale nodes are used for Cloud services.

Note: You need to run this command only once on any node on the IBM Spectrum Scale cluster.

You can install Cloud services on a maximum of four nodes per node group (any combination of NSD or CES nodes) on the IBM Spectrum Scale cluster (which can have a maximum of four node groups). Before you install Cloud services, you need to create a node class that specifies the IP addresses (or fully qualified host names) of the nodes where the server packages are going to be installed. You can use any host name that the **-N** option accepts. For details, see *mmcrnodeclass command* in *IBM Spectrum Scale: Command and Programming Reference*.

You can enable and manage independent groups of Cloud services nodes in different node classes for use with different network configurations per node class. Nodes are mutually exclusive to each node class and cannot be shared with another node class that has Cloud services nodes. For example, the node class `TCTNodeClass1` has node1 and node2 and the node class `TCTNodeClass2` has node3 and node 4. All nodes are Cloud services enabled nodes. In this case, the node class `TCTNodeClass2` cannot enable node1 as a Cloud services node under the `TCTNodeClass2` node class because it is already marked for Cloud services usage under the `TCTNodeClass1` node class. Additionally, each group of Cloud services enabled nodes under a node class cannot share file system with Cloud services enabled nodes from a different node class. And each Cloud services enabled node under a node class is limited to use only one common file system with each node in that node class. Therefore, you must manage each group of Cloud services nodes within a node class as a pool of nodes that require a single common file system.

Note: It is recommended that you set up at least two nodes so that you have good availability for the service if a node were to go down. Ensure to use the GPFS cluster IP address that gets displayed when you run the **`mmlscluster`** command.

To create a node class, issue a command according to this syntax:

```
mmcrnodeclass ClassName -N {Node[,Node...] | NodeFile | NodeClass}
```

For example, to create a node class called *TCTNodeClass* by using three nodes 10.10.10.10, 11.11.11.11, and 12.12.12.12, issue this command:

```
mmcrnodeclass TCTNodeClass -N 10.10.10.10,11.11.11.11,12.12.12.12
```

To verify that the node class is created, issue this command:

```
mmnlsnodeclass
```

The system displays output similar to this output:

```
Node Class Name Members
--------------------- -----------------------------------------------------------
TCTNodeClass node10,node11,node12
```

Next step: “Installation steps”

Installation steps

This topic describes the procedure for installing Cloud services on IBM Spectrum Scale nodes.

Cloud services creates the following folders after installation:

- `/var/MCStore`
- `<filesystem>/.mcstore`
- `/opt/ibm/MCStore`

Do not make any changes to these folders in order to avoid any undesired results.

The Cloud services RPMs are available with the IBM Spectrum Scale Advanced Edition or Data Management Edition. You can verify this by issuing this command: **`rpm -qa | grep gpfs.adv`**. Depending on the target Linux distribution, the RPMs are available in the following paths where *_rpms or *_debs is the component specific directory:

- Red Hat Enterprise Linux 7.x: `/usr/lpp/mmfs/5.0.x.x/*_rpms/rhel7`
- Ubuntu 16.04 LTS: `/usr/lpp/mmfs/5.0.x.x/*_debs/ubuntu16`
- SLES 12: `/usr/lpp/mmfs/5.0.x.x/*_rpms/sles12`

With each build of IBM Spectrum Scale, two RPMs are available for Cloud services:

- gpfs.tct.server-x.x.x.x86_64.rpm. This is the server package that is installed on the Cloud services nodes. These nodes must have outbound WAN connectivity to be able to support data migration between IBM Spectrum Scale cluster and the cloud object storage.
- gpfs.tct.client-x.x.x.x86_64.rpm. This is the client package that contains the Cloud services binary files (specifically the data manager commands). This package must be installed on all remaining cluster nodes to activate ILM-based file migration or file recall that is initiated from that node (every node in the cluster is safe). The client RPM is not required on the nodes where the server package is installed.

  Note: It is important to install the client package on every node so that files that are deleted from the IBM Spectrum Scale cluster are deleted efficiently from the cloud storage tier. If this package is not installed, files are deleted less aggressively and only as a result of running the reconcile service.

Note: For policy-based migrations, it is recommended that the file system manager node (rather all nodes with manager roles) is installed at least with the client RPM. This ensures that if any migration request goes to the file system manager node that is not a Transparent cloud tiering node, the request is forwarded to one of the server nodes in the node class, and the migration seamlessly takes place. If the RPM is not there on the file system manager, then migration fails. Transparent cloud tiering is needed on all clients (not just for migration requests) or transparent recall does not work properly and delete processing is not handled efficiently for files transferred to the cloud.

1. Copy the server RPMs to the desired nodes.
2. Install Cloud services package on each node by issuing the following command depending on your OS:
   - For RHEL: **`rpm -ivh gpfs.tct.server-x.x.x.x86_64.rpm`**
   - For Power Linux: **`rpm -ivh gpfs.tct.server-x.x.x.ppc64le.rpm`**

   Note: By default, the Cloud services is installed under the `/opt/ibm/MCStore` directory.

3. Install the client package on the desired nodes by issuing the following command depending on your OS:
   - For RHEL: **rpm -ivh gpfs.tct.client-x.x.x.x86_64.rpm**
   - For SLES: **zypper install gpfs.tct.client-x.x.x.x86_64.rpm**
   - For Debian: **sudo dpkg -i gpfs.tct.client*x86_64.deb**
   - For Power Linux: **rpm -Uvh gpfs.tct.client-x.x.x.ppc64le.rpm**

Chapter 8. Setting up a Cloud services cluster

This topic provides step-by-step instructions for quickly setting up a Cloud services node group that can support Transparent cloud tiering or Cloud data sharing on the IBM Spectrum Scale cluster.

In the multi-node setup, a maximum of 4 of any combination of CES or NSD nodes are supported as a Cloud services node group -- a minimum of two are recommended for high availability. The node group is bound to a single file system. Data migration or recall workload is shared across all the nodes. During a transparent recall or migration, the ILM policy gathers a list of files to be migrated, and the list is divided equally among all the nodes.

Note: Once a multi-node setup is up and running, the Transparent cloud tiering GUI components are available on the IBM Spectrum Scale GUI.

Perform the following steps for setting up a Cloud services cluster:

1. Install the Cloud services RPMs on the required IBM Spectrum Scale nodes. For more information, see Chapter 7, "Installing Cloud services on IBM Spectrum Scale nodes," on page 337.
2. Designate the nodes where you installed the RPMs as Cloud services nodes. For more information, see the *Designating the transparent cloud tiering nodes* topic in the *IBM Spectrum Scale: Administration Guide*.
3. Start Cloud services on all nodes. For more information, see the *Starting the Cloud services* topic in the *IBM Spectrum Scale: Administration Guide*.
4. Create a cloud account. For simpler configurations, use the built-in key manager that comes with Cloud services. Use KLM when KLM is already running on Spectrum Scale (for other reasons) or if specific security requirements that drive its usage. For more information, see the *Creating a cloud storage account* topic in the *IBM Spectrum Scale: Administration Guide*.
5. Define cloud storage access points (CSAP). For more information, see the *Defining cloud storage access points (CSAP)* topic in the *IBM Spectrum Scale: Administration Guide*.
6. Create a Cloud services. For more information, see the *Creating a Cloud services* topic in the *IBM Spectrum Scale: Administration Guide*.
7. Configure a key manager. For more information, see the *Creating a key manager* topic in the *IBM Spectrum Scale: Administration Guide*.
8. Create a container pair set and test your configuration. For more information, see the *Creating a container pair set* topic in the *IBM Spectrum Scale: Administration Guide*.
9. Make sure you back up your configuration and your key manager. For more information, see the *Backing up the Cloud services configuration* topic in the *IBM Spectrum Scale: Administration Guide*.

Now that a Cloud services node group setup is ready, you need to decide whether you are going to use the node group for tiering or sharing.

If you are doing Transparent cloud tiering, do the following steps to verify that the cluster is ready for transparent recall/migration based on an ILM policy:

- Create a sample policy, *migrate.policy*, and apply this policy by using the Cloud services nodes by with the command: **`mmapplypolicy gpfs -P migrate.policy -N FirstCloud`**

  Note: If you administer the `defaultHelperNodes` settings, you do not need to specify the `-N` parameter to distribute workload here.

You can confirm that data migration is happening across multiple nodes by looking at the following events:

- Each Transparent cloud tiering node has policy files under /tmp.
- Each Transparent cloud tiering node starts reflecting the migration metrics for the amount of data that is passed through it.

Do not forget to set up a cron job to do your regular maintenance tasks such as reconciling files between the file systems and the cloud objects storage and deleting cloud objects. For more information, see the *Administering files for Transparent cloud tiering* topic in the *IBM Spectrum Scale: Administration Guide*.

Note: If you change the IP address or network settings of any of the Cloud services nodes once it is operational, do the following steps:

1. Stop the Cloud services (by using the **mmcloudgateway service stop** command), the way it is mandated to stop IBM Spectrum Scale services while you change the node IP address.
2. To change the IP address, it is recommended to follow the steps that are provided in the *Changing IP addresses and host names* topic in the *IBM Spectrum Scale: Administration Guide*.

For more information, see *Configuring and tuning your system for Cloud services* topic in the *IBM Spectrum Scale: Administration Guide*.

Chapter 9. Installing and configuring IBM Spectrum Scale management API

The IBM Spectrum Scale management API implementation is based on the IBM Spectrum Scale GUI stack. Installation and configuration of GUI sets up the API infrastructure. That is, you do not need to perform any API-specific deployment procedures.

Ensure that the following tasks are completed to start using the IBM Spectrum Scale management API to monitor and manage the IBM Spectrum Scale system:

1. Install the GUI server. Use any of the following options to install the GUI:
   a. *Manually installing IBM Spectrum Scale management GUI* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
   b. *Installing IBM Spectrum Scale management GUI by using the spectrumscale installation toolkit* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
2. Initialize the GUI either by accessing the GUI through the browser or by using the **/usr/lpp/mmfs/gui/cli/initgui** command.
3. The default user name and password to access the IBM Spectrum Scale GUI is `admin` and `admin001` respectively. Due to security reasons, it is recommended to change the default password after the first login. The password can be changed through the GUI or by using **/usr/lpp/mmfs/gui/cli/chuser admin -p newPassword** command.
4. Create new users and provide the required permissions. You can either use the **Access** > **GUI Users** GUI page or the **usr/lpp/mmfs/gui/cli/mkuser** command to create new users and assign roles. For more information on how to create GUI users and assigning user roles, see *Managing GUI users* section in the *IBM Spectrum Scale: Administration Guide*.

   Every API call to the IBM Spectrum Scale needs a valid GUI user and password. For example, curl -k **-u admin:admin001** -XGET -H content-type:application/json "https://<IP address>:443/scalemgmt/v2/info". The user roles assigned to the user determine actions that can be performed.

Upgrading IBM Spectrum Scale management API from 4.2.2 to 4.2.3.x

The API Version 1 was introduced in the IBM Spectrum Scale 4.2.2 release and the API Version 2 is introduced in the 4.2.3 release. You cannot upgrade the IBM Spectrum Scale management API from 4.2.2 to 4.2.3 or later as the architecture and syntax are entirely different in these versions. If you are already using API in 4.2.2, uninstall the scalemgmt rpm and then install the 4.2.3 GUI rpms to start using API Version 2.

Chapter 10. Installing Active File Management

The following topics describe how to install Active File Management.

Installation and upgrade of Active File Management (AFM)

Consider the following while installing and upgrading Active File Management (AFM).

The home and cache are two separate operational clusters separated from each other through LAN or WAN. There are no special considerations on the cluster setup for using AFM functions. AFM functions are available in Standard Edition of IBM Spectrum Scale or GPFS.

The home cluster can be a legacy NAS or a cluster running GPFS version 3.5 or earlier, or IBM Spectrum Scale 4.1 or later. The cache cluster can run GPFS 3.5, or IBM Spectrum Scale 4.1 or later. User-extended attributes, ACLs and sparse files are not supported on a home cluster that is running GPFS version 3.5 or earlier, if home is a legacy NFS export.

The NFS v3 or GPFS protocol can be used for communication. The GPFS protocol can be used only if the home is a cluster running GPFS or IBM Spectrum Scale.

Nodes must be identified at the cache cluster that can act as gateway nodes. Gateway nodes should preferably be configured in the cache cluster before creating filesets and starting applications.

Nodes must be identified at the home cluster that can act as NFS servers. If the GPFS protocol is planned to be used, the home filesystem must be remote mounted on all the gateway nodes in the cache cluster. Gateway nodes and NFS servers must preferably be configured in the cache and home clusters before creating filesets and starting applications.

At any point in time, the versions on the home or the cache cluster can be upgraded independent of the other. For example, a home cluster can be running on GPFS version 3.5 and the cache cluster on IBM Spectrum Scale version 4.1. If you are using the GPFS protocol for AFM filesets, the 3.5 filesystem at home must be remote mounted on the 4.1 cache cluster.

Note: While upgrading to IBM Spectrum Scale 4.2.2 or later, the cache cluster must be upgraded before considering upgrade of the home cluster.

Requirements for UID and GID on the cache and home clusters

This topic describes the UID and GID requirements on the cache and home clusters.

User IDs and group IDs must be managed the same way across the cache and home clusters. However, for AFM relationships, which are using the native GPFS protocol, where user IDs are different on the home and the cache, the IDs can be remapped using GPFS UID remapping. For a detailed procedure, see *Configuring ID mappings in IMU* in *IBM Spectrum Scale: Administration Guide*.

Recommended worker1threads on cache cluster

This topic specifies the number of worker1threads on cache cluster.

It is recommended to increase the **worker1threads** to twice the number of AFM filesets using GPFS backend, that exist on all filesystems in the cache cluster. For NFS protocol, the number of **worker1threads** can be equal to the number of AFM filesets.

For example, if a cluster has 50 caches using NFS protocol and 50 caches using GPFS protocol, the **worker1threads** must be at least 150 (50 + 2*50).

Inode limits to set at cache and home

To control the number of inodes used by a cache or home fileset, you can specify a limit on the number of inodes for that fileset during fileset creation time (see *mmcrfileset command* in *IBM Spectrum Scale: Command and Programming Reference* and the `--inode-limit` option).

If home is a dependent fileset in the filesystem, no inode limit is enforced. The limits applicable to the filesystem would hold good for the dependent fileset.

Note: AFM does not perform cache eviction for cache fileset, even if the inode limit is reached. Cache eviction is applicable for block quotas.

Creating an AFM relationship by using the NFS protocol

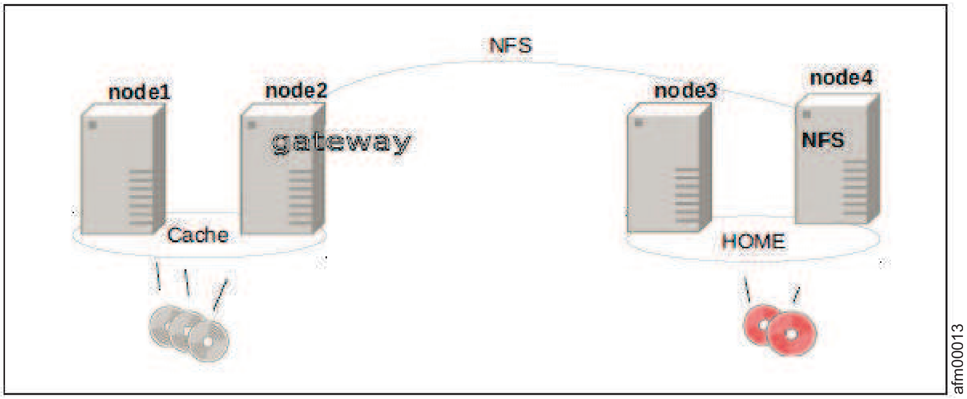


Figure 36. A demonstration setup of an AFM relationship

Setting up the home cluster

This topic lists steps to set up the home cluster.

1. Set up a home cluster. For more information, see *Creating your GPFS cluster* in *IBM Spectrum Scale: Administration Guide*.
2. Configure a cache relationship for AFM filesets using the home cluster you just created.

   The relationship between the home cluster and the cache cluster is set up by using NFS exports defined at home cluster. The home cluster exports NFS mount points that AFM cache cluster uses to synchronize data.
3. Create a file system and mount this file system on the home cluster. For more information, see *Managing file systems* in *IBM Spectrum Scale: Administration Guide*. The home side of the relationship is defined at the NFS share level. The home contains all the data available from the NFS mount point.
4. At the home cluster, create and link one or more filesets. For more information, see *Filesets* in *IBM Spectrum Scale: Administration Guide*. These filesets are used to set up NFS exports at the home cluster. These export paths are fileset junction paths where filesets are linked.
5. Export the fileset, one fileset at a time. Do one of the following:

   Note: In this example, the IP address of the Gateway node of the cache cluster is 192.168.1.2. As an Administrator, ensure that NFS exports are accessible only to nodes at the cache.
   - If you are using KNFS, complete the following steps for KNFS export of the home filesystem:

a. Add fileset junction path to the `/etc/export` file of home cluster export servers. Exported path must have the necessary permissions. **`no_root_squash`** and **`rw`** are mandatory and **`fsid`** is optional. The following is an example of an NFS export entry with `fsid` at home side:

```
/gpfs/fs1/sw2 192.168.1.2(rw,nohide,insecure,no_subtree_check,sync,no_root_squash,fsid=1069)
```

b. Re-start the NFS services on home cluster export servers. This starts all the necessary NFS services and exports the given path to be used by the AFM cache.

c. If the file system goes down, you must export the file system again. The gateway nodes at the cache site must have access to the exported directory and can be mounted using NFS.

- If you want to configure a file system on the home cluster with protocols nodes, complete the following steps:

  a. Use the mmnfs export add command to create export on junction path or the gpfs path of user choice.

  ```
  mmnfs export add /ibm/gpfs0/nfsexport --client \
      "192.168.1.2(Access_Type=RW,Squash=no_root_squash,SecType=sys)"
  ```

  Use the mmnfs export list command to list NFS exports:

  ```
      mmnfs export list
  ```

  The system displays output similar to this:

  ```
  Path Delegations Clients
  ----------------------------------------
  ```

  `/ibm/gpfs0/nfsexport none *` Do not edit `/etc/exports` or any other NFS configuration files manually, and do not restart NFS services after the export is created

6. After you export filesets at the home cluster, run **`mmafmconfig enable /ibm/gpfs0/nfsexport`**. For more information about the command, see *mmafmconfig* in *IBM Spectrum Scale: Command and Programming Reference*.

   Note:

   a. Ensure that you add the IP addresses of all gateway nodes of the cache cluster. Multiple IP addresses can be indicated by a comma-separated list. Update the list of IP addresses whenever you add or remove a gateway node.

   b. Ensure that the KNFS or NFS server at home is restarted, and home exports are available. CES NFS does not require a restart.

   c. If both NFS and GPFS start automatically at the start-up time, ensure that GPFS starts before NFS as NFS can export GPFS only if it is loaded. If NFS starts before GPFS, run **`exportfs -r`**.

   d. If an Ubuntu server is used as an AFM DR secondary server, configure the AFM secondary server so that the NFS server service `rpc.mountd` does not start with *--manage-gids* . `rpc.mountd` *`--manage-gids`* is not applicable for CES NFS or Native GPFS protocol.

Setting up the cache cluster

This topic lists steps to set up the cache cluster.

1. To identify the nodes of the GPFS cluster that function as the application nodes and the nodes that function as gateway nodes, run the following command:

   ```
   mmlscluster
   ```

2. After you identify the nodes, run the following command to assign the role of a gateway node to the identified nodes:

   ```
   mmchnode --gateway -N Node1,Node2,...
   ```

3. To ensure that GPFS has started, run the following command:

   ```
   mmgetstate -a
   ```

4. To mount the file system, run the following command:

   ```
   mmmount filesystemname
   ```

5. To create an AFM fileset and link the fileset, run the following command:

   ```
   mmcrfileset filesystemname Fileset -p afmTarget=Home:Home-Exported-Path
   --inode-space=new -p afmMode=single-writer | read-only | local-updates | independent-writer
   ```

   ```
   mmlinkfileset filesystemname fileset -J /filesystem-path/fileset
   ```

6. Access the AFM fileset. **mmafmctl filesystemname getstate** at cache displays the fileset state and other details.

Example of creating an AFM relationship by using the NFS protocol

The following is an example of creating an AFM relationship between the home cluster and the cache cluster by using the NFS protocol.

Home can be configured by using CES NFS or default NFS available with operating system:

If you are using CES NFS -

1. Create a directory path (For example - /gpfs/fshome/fset001) and export this path.
2. Run the command **mmnfs export add /ibm/gpfs0/export1 -c "<client Nodes IP/ range>(Access_Type=RO,Squash=root_squash"**. For more information about the command, see *mmnfs* in *IBM Spectrum Scale: Command and Programming Reference*.

If you are using default NFS, enter the following in /etc/export:

```
/gpfs/fshome/fset001 node2(rw,no_root_squash,no_subtree_check,fsid=101)
node4:/gpfs # exportfs -a
node4:/gpfs # exportfs
/gpfs/fshome/fset001
node2.site
```

Cache:

#designate a node as gateway

```
node1:~ # mmchnode --gateway -N node2
```

```
Wed Oct 8 22:35:42 CEST 2014: mmchnode: Processing node node2.site
mmchnode: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

# create an afm fileset

```
node1:/gpfs/cache # mmcrfileset fs1 fileset_SW -p afmtarget=node4:/gpfs/fshome/fset001 -p
afmmode=single-writer --inode-space new
```

```
Fileset fileset_SW created with id 1 root inode 131075.
```

# link the fileset

Note: Test the NFS mounts from the gateway nodes to the home cluster by using the standard operating system mount command before you create and link the fileset.

```
node1:/gpfs/cache # mmlinkfileset fs1 fileset_SW -J /gpfs/cache/fileset_SW
```

```
Fileset fileset_SW linked at /gpfs/cache/fileset_SW
```

# to verify the state after creating/linking the fileset

```
node1:/gpfs/cache # mmafmctl fs1 getstate
```

```
Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
------------ -------------- ------------- ------------ ------------ -------------
fileset_SW nfs://node4/gpfs/fshome/fset001 Inactive
```

# stat the cache directory, to activate AFM

```
node1:/gpfs/cache # ls -l
          total 1
          drwx------ 4 root root 4096 Oct 8 20:38 fileset_SW
          dr-xr-xr-x 2 root root 32768 Jan 1 1970 .snapshots
```

# the fileset is active and numExec is 1 (for the ls)

```
node1:/gpfs/cache # mmafmctl fs1 getstate
          Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
           ------------ -------------- ------------- ------------ ------------------------
          fileset_SW nfs://node4/gpfs/fshome/fset001 Active node2 0 1
```

#Queue requests on the gateway

```
node1:/gpfs/cache # ls -l fileset_SW/
          total 96
          drwx------ 65535 root root 32768 Oct 9 20:14 .afm
          drwx------ 65535 root root 32768 Oct 9 20:14 .ptrash
          dr-xr-xr-x 2 root root 32768 Jan 1 1970 .snapshots
```

#fileset shows revalidation operations executed

```
node1:/gpfs/cache # mmafmctl fs1 getstate
           Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
           ------------ -------------- ------------- ------------ ------------------------
           fileset_SW nfs://node4/gpfs/fshome/fset001 Active node2 0 5
```

# creating four new files in the CACHE fileset (cache side)

```
node1:/gpfs/cache/fileset_SW # for i in 1 2 3 4 ; do date >> file$i; done
```

# verify, that the files were written (on cache)

```
node1:/gpfs/cache/fileset_SW # ls -l
          total 96
          drwx------ 65535 root root 32768 Oct 9 20:14 .afm
          -rw-r--r-- 1 root root 30 Oct 9 20:25 file1
          -rw-r--r-- 1 root root 30 Oct 9 20:25 file2
          -rw-r--r-- 1 root root 30 Oct 9 20:25 file3
          -rw-r--r-- 1 root root 30 Oct 9 20:25 file4
          drwx------ 65535 root root 32768 Oct 9 20:14 .ptrash
          dr-xr-xr-x 2 root root 32768 Jan 1 1970 .snapshots
```

# verify cache state

```
node1:/gpfs/cache/fileset_SW # mmafmctl fs1 getstate
         Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
         ------------ -------------- ------------- ------------ ------------------------
         fileset_SW nfs://node4/gpfs/fshome/fset001 Dirty node2 8 5
```

# wait for sometime, up to async delay and check cache state to see it is flushed to home

```
node1:/gpfs/cache/fileset_SW # mmafmctl fs1 getstate
         Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
         ------------ -------------- ------------- ------------ ------------------------
         fileset_SW nfs://node4/gpfs/fshome/fset001 Active node2 0 13
```

Creating an AFM relationship by using GPFS protocol

The following topics describe how to set up the home and cache cluster.

Setting up the home cluster

This topic lists the steps to set up the home cluster.

1. Create a home cluster. For more information, see *Creating your GPFS cluster* in *IBM Spectrum Scale: Administration Guide*.
2. Create a file system at the home cluster you just created. For more information, see *Managing file systems* in *IBM Spectrum Scale: Administration Guide*.
3. Create filesets at the home cluster. For more information, see *Filesets* in *IBM Spectrum Scale: Administration Guide*.
4. Enable remote access to the file system you just created at the home cluster. For more information, see *Accessing a remote GPFS file system* in *IBM Spectrum Scale: Administration Guide*.
5. To configure the exported path at the home cluster for AFM, run **mmafmconfig enable /ibm/gpfs0/nfsexport** at the home cluster.

The file system with its filesets, is now accessible to the AFM cache cluster.

Setting up the cache cluster

This topic lists the steps to set up the cache cluster.

1. To determine the nodes of the GPFS cluster that functions as application nodes and the nodes that function as the gateway nodes, run the **mmchnode --gateway -N Node1,Node2,...** command.
2. To start GPFS, run the **mmstartup -a** command.
3. To mount the file system, run the **mmmount Device -a** command.
4. Mount the home filesystem on the cache cluster remotely.
5. To create an AFM fileset and link it, run the following command:

   ```
   mmcrfileset Device Fileset -p afmTarget=Home-Path --inode-space=new
   -p afmMode=single-writer | read-only | local-updates | independent-writer
   mmlinkfileset device fileset -J /fsmount-path/fileset
   ```

6. Access the AFM fileset.

   mmafmctl Device getstate run on the cache cluster displays the fileset state and other details.

Example of creating an AFM relationship by using the GPFS protocol

How to create an AFM relationship between the home cluster and cache cluster by using the GPFS protocol.

#designate a node as gateway

node1:~ # mmchnode --gateway -N node2

```
Wed Oct 8 22:35:42 CEST 2014: mmchnode: Processing node node2.site
mmchnode: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

# Remote mount the home filesystem on the cache

node1:/var/mmfs/ssl # mmremotefs show all

```
Local Name Remote Name Cluster name Mount Point Mount Options Automount Drive Priority
remoteHOME fshome node3.home /remote/fshome rw no – 0
```

node1:/var/mmfs/ssl # mmmount remoteHOME -a

```
Thu Oct 23 23:28:32 CEST 2014: mmmount: Mounting file systems ...
```

#Create AFM with GPFS protocol as the transport layer

```
node1:/var/mmfs/ssl # mmcrfileset fs1 fileset_IW -p afmtarget=gpfs:///remote/fshome -p
afmmode=iw --inode-space=new
 Fileset fileset_IW created with id 1 root inode 131075.

node1:/var/mmfs/ssl # cd /gpfs/cache

node1:/gpfs/cache # mmlinkfileset fs1 fileset_IW
         Fileset fileset_IW linked at /gpfs/cache/fileset_IW

node1:/gpfs/cache # mmafmctl fs1 getstate
         Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
         ------------ -------------- ------------- ------------ ------------ -------------
         fileset_IW gpfs:///remote/fshome Inactive

node1:/gpfs/cache # cd fileset_IW
node1:/gpfs/cache/fileset_IW # ll
        total 97
        drwx------ 65535 root root 32768 Oct 23 23:47 .afm
        drwx------ 65535 root root 32768 Oct 23 23:47 .pconflicts
        drwx------ 65535 root root 32768 Oct 23 23:47 .ptrash
        dr-xr-xr-x 2 root root 32768 Jan 1 1970 .snapshots
node1:/gpfs/cache/fileset_IW #mmmount remoteHOME -a
        # access the fileset / wait 60 seconds
        node1:/gpfs/cache/fileset_IW # mmafmctl fs1 getstate
        Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
        ------------ -------------- ------------- ------------ ------------ -------------
        fileset_IW gpfs:///remote/fshome Active node1 0 13
```

Configuration changes in an existing AFM relationship

See the following examples:

Gateway nodes in the cache cluster

You can add a new gateway or remove the existing gateway nodes by using the **mmchnode** command.

AFM automatically adjusts the existing filesets to use the newly-configured gateways.

Ensure that queues are empty, or existing gateway nodes are shut down before running the **mmchnode** command..

If the gateway node changes are made with all the gateway nodes in the Active state, the cluster might appear hung, or cluster wide waiters might occur after the **mmchnode** command is run. The cluster wide waiters disappear when you recycle the active gateway nodes.

If the existing gateway nodes are a part of a mapping, the gateway nodes cannot be removed. You can remove the gateway nodes from the mapping using **mmafmconfig** command.

Note: Ensure that you add the IP addresses of all gateway nodes of the cache cluster. Update the list of IP addresses whenever you add or remove a gateway node.

The NFS server at the home cluster

The NFS server, the mount path, or the IP address at the home cluster can be changed.

The existing AFM filesets at cache must be updated to point to the new target. The home cluster and filesystem do not change. Therefore, any change can be reflected in the cache cluster by using the **mmafmctl failover** command with the –target-only option.

However, in the case of NFS server change, the new NFS server must be in the same home cluster and must have the same architecture as the existing NFS server in the target path. In other cases, the failover must be performed without the –target-only option. If the target protocol changes from NSD to NFS or vice-versa, the **mmafmctl failover** command must be used without the –target-only option.

For more details, see "Changing home of AFM cache" on page 65.

Chapter 11. Installing and upgrading AFM-based Disaster Recovery

The primary and the secondary are two separate operational clusters separated from each other through LAN or WAN. AFM DR functions are available in Advanced Edition or Data Management Edition of IBM Spectrum Scale. All nodes in both the primary and secondary clusters must be running the Advanced Edition or Data Management Edition.

The NFS v3 or GPFS protocol can be used for communication. Similar to AFM filesets, gateway nodes and NFS servers must be planned on the primary and secondary clusters. Gateway nodes and NFS servers must preferably be configured in the primary and secondary clusters before creating filesets and starting applications. If you are planning to use GPFS protocol, the primary filesystem must be remote mounted on all the gateway nodes in the secondary cluster.

Note: While upgrading to IBM Spectrum Scale 4.2.2 or later, the primary cluster must be upgraded before considering upgrade of the secondary cluster.

Requirements for UID/GID on primary and secondary clusters

User IDs and group IDs must be managed the same way across the primary and secondary clusters.

However, for AFM DR relationships, which use native GPFS protocol, where user IDs are different on the primary and secondary clusters, the IDs may be mapped again using GPFS UID remapping.

Recommended worker1threads on primary cluster

IBM recommends that you increase the **worker1threads** to twice the number of AFM DR filesets using GPFS backend, that exist on all filesystems in the cache cluster.

For example, if a cluster has 50 primary filesets using NFS protocol and 50 primary using GPFS protocol, the **worker1threads** must be at least 150 (50 + 2*50). For NFS protocol, the number of **worker1threads** can be equal to the number of primary filesets.

NFS setup on the secondary cluster

NFS setup on the secondary cluster follows the same rules as NFS setup on the AFM home cluster.

Creating an AFM-based DR relationship

This topic lists the steps to create an AFM-based Async DR relationship.

Complete the following steps to create and use an AFM-based Asynchronous DR relationship:

1. Create a new primary fileset. Run on a primary cluster.

   Create the primary fileset using **mmcrfileset** command. The primary can be connected to the secondary using NFSv3 protocol or the NSD protocol. All AFM parameters for writable filesets (single writer / independent writer) are applicable to a primary fileset. A primary fileset is not revalidated and does not check the secondary for changes because it is expected that changes always originate from the primary.

   A primary fileset is a writable fileset so all file operations performed on this fileset are replayed at the secondary fileset using the same mechanism as single writer and independent writer modes. Unlike other AFM modes the secondary, or target fileset is an AFM fileset that has a relationship with a primary. The secondary fileset is enforced as read-only. AFM parameters such as **Async Delay**, number of flush threads and parallel write can be used on primary filesets.

When a primary fileset is created a unique primary ID is generated. When creating the primary fileset you need to specify the path to the secondary fileset though it may not exist at the time of primary creation. In the following example, the secondary is not created but the path is provided in **mmcrfileset** command.

```
# mmcrfileset fs1 primary2 -p afmMode=primary --inode-space=new -p
afmTarget=nfs://c2m3n06/ibm/fs1/secondary2 -p afmRPO=720
Fileset primary2 created with id 19 root inode 7340035.
Primary Id (afmPrimaryId) 15997179941099568310-C0A8747F557F0086-19
```

Note: If you are using CES NFS at home, replace `c2m3n06` with *ces_ip_of_secondary_node*.

2. Create the secondary fileset. Run on a secondary cluster.

   Get the primary id of the GPFS fileset on the primary side (**afmPrimaryId**) before the actual conversion. Use **mmafmctl getPrimaryId** command on the GPFS fileset on the primary side.

   ```
   # mmafmctl fs1 getPrimaryId -j primary2
   Primary Id (afmPrimaryId) 15997179941099568310-C0A8747F557F0086-19
   ```

   Create the secondary fileset using the **mmcrfileset** command.

   ```
   mmcrfileset fs1 secondary2 -p afmMode=secondary –p
   afmPrimaryId=15997179941099568310-C0A8747F557F0086-19
   --inode-space new
   ```

3. Link the secondary fileset on the secondary cluster using the **mmlinkfileset** command.

   Run **mmlinkfileset fs1 secondary2 -J /ibm/fs1/secondary2**.

   The primary does not check the secondary for changes. If you are using quotas, ensure that the same value is set for quotas on primary and secondary. On a primary fileset, eviction is disabled by default and filesets do not expire. If you are using NFS, ensure that the NFS export on the secondary site is accessible from the gateway nodes in the primary cluster. If you are using the NSD protocol, the secondary file system needs to be mounted on the gateway nodes at the primary cluster.

4. Restart NFS on secondary.

   Note: If you are using CES NFS, restart is not required.

5. Link the primary fileset on the primary cluster. Link the primary fileset using **mmlinkfileset** command. Linking the fileset creates the first RPO snapshot on the primary called **psnap0**.

   ```
   mmlinkfileset fs1 primary2 -J /ibm/fs1/primary2
   ```

   After the primary and secondary are linked, the RPO snapshot (**psnap0**) gets queued from the primary fileset which gets replayed on the secondary fileset. The Async DR filesets are now ready for use.

- Do not run **mmafmconfig** command on the secondary site. Run **mmafmctl gpfs0 getstate** on the primary to know the primary gateway node.
- Check **fsid** and **primary id** on the secondary, and ensure that any two secondary filesets do not share the same **fsid** or **primary id**.
- **psnap0** must be created at both sites for the filesets to synchronize. In cases like node shutdown, process failure; **psnap0** might not be created. Hence, filesets do not synchronize with the secondary. Unlinking and re-linking the filesets re-creates **psnap0**. The filesets then synchronize with the secondary.

Converting GPFS filesets to AFM DR

This topic lists the steps to convert GPFS independent filesets to primary or secondary filesets.

Complete the following steps to convert GPFS-independent filesets to primary or secondary:

1. Ensure that primary and secondary sites have the same data using trucking method.

An existing GPFS independent fileset can be converted to primary or secondary. If the fileset on the primary site has data, the secondary site must be synchronized with the same data. This process is termed as trucking. Trucking must be inband.

Inband trucking: Copying the data from the primary to the secondary while setting up the relationship. Inband trucking is limited by the network bandwidth between the primary and the secondary.

Conversion of a regular independent fileset to AFM primary with **mmafmctl** command must be performed by specifying the `--check-metadata` option that verifies that the fileset does not contain objects with attributes that are disallowed in a primary fileset. These include the following:

- Files with **Immutable** and **AppendOnly** attributes
- Special files (such as devices)
- Dependent fileset
- Clones where the source belongs to a snapshot

2. Get the primary id of the GPFS fileset before the actual conversion by using the **mmafmctl getprimaryid** command on the GPFS fileset.
3. Convert the fileset on the secondary site to a secondary and set the primary id using the **mmchfileset** or **mmafmctl** command with the `convertToSecondary` option.

   Ensure that the NFS export on the secondary site is accessible on the primary, if NFS is used for defining AFM target on primary site. If GPFS protocol is used for the target, the secondary file system should be remote mounted on the primary site.

   Note: If you are establishing the secondary using the out-of-band option, you must first complete the data copy and ensure that the primary and the secondary have the same data before you configure the secondary with the primary's unique ID.
4. Restart NFS on the secondary.
5. Convert the fileset on the primary site to a primary by using **mmafmctl** command. Run on the primary cluster. Gateway nodes must be defined in the primary site and the file system must be mounted on all gateway nodes before doing this conversion. Run **mmafmctl** with **convertToPrimary** option.

   ```
   mmafmctlDevice convertToPrimary-j FilesetName
   [ --afmtarget Target { --inband | --secondary-snapname SnapshotName }]
     [ --check-metadata | --nocheck-metadata ] [--rpo RPO] [-s LocalWorkDirectory
   ```

   --afmtarget and **--inband** are mandatory options for the first conversion command on the fileset. Conversion can get interrupted in the middle due to unforeseen reasons or in case of rare errors when psnap0 creation is not successful. In such cases, fileset is converted to a primary but left in `PrimInitFail` state.

   Based on the reason behind the failure, the administrator must re-run the conversion command without any argument. Alternatively, the fileset can be converted back to normal GPFS filesets and converted again using the conversion command with arguments.

   The **--afmtarget** option mentions the fileset path on the secondary site.

   The **--inband** option is used for inband trucking. Primary id gets generated and the first RPO snapshot psnap0 gets created. The entire data on the snapshot is queued to the secondary. Once the data has replayed on the secondary after Step 3 (following), that is, the primary and secondary are connected, it creates a psnap0 on the secondary ensuring that the **psnap0** on the primary and the secondary are the same. At this point, one can consider a relationship has been established.

   The **--check-metadata** option checks if the disallowed types (like immutable/append-only files, clones where the source belongs to a snapshot etc) are present in the GPFS fileset on the primary site before the conversion. Conversion fails with this option if the disallowed types still exist on the primary side. --check-metadata is not mandatory and performs a scan of the entire fileset to verify its contents, and while it can be excluded if the fileset is known to be permissible for conversion, it should be used when in doubt. This is the default option.

   The **--no check-metadata** option is used to proceed with conversion without checking for the disallowed types.

The **--rpo** option specifies the RPO interval in minutes for this primary fileset. By default, RPO is disabled. You can use the **mmchfileset** command to modify the *afmRPO* value of the AFM DR fileset. The --secondary-snapname is not applicable for converting AFM or GPFS filesets. This option is used while establishing a new primary, as discussed in subsequent sections.

Gateway node assignments must be finalized and done preferably before conversion of GPFS or AFM filesets to primary. If no gateway is present on the primary cluster during conversion, then primary fileset might remain in the **PrimInitFail** state.

After the primary and secondary are connected with `psnap0` from any one side, the primary is in `Active` state. The two filesets are ready for use.

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

Note: Parallel data transfers are not applicable to trucking even if the AFM target is mapping. Resync does not split data transfers even if parallel data transfer is configured, and the target is a mapping.

Converting AFM relationship to AFM DR

A working AFM single writer (SW) or independent writer (IW) relationship can be converted to a primary/secondary relationship.

Complete the following steps:

Note: In case of multiple IW caches to the same home, you can convert only one to primary.

1. Ensure all contents are cached. AFM fileset must be in active state by flushing queues and for filesets that have contents at home, the complete namespace should be constructed at the cache using stat on all entries, to avoid orphans. In SW/IW filesets, some files might not be cached or some files might be evicted. All such files should be cached using prefetch. Complete the following steps to ensure that all contents are present and are up to-date in the SW/IW caches:
   a. Ensure that the storage capacity on the cache fileset is the same as on the home and the set quotas match.
   b. Run **mmchfileset filesystem sw/iw cache -p afmEnableAutoEviction=no** to disable automatic eviction.
   c. Ensure that **afmPrefetchThreshold** is set to 0 on the SW/IW cache.
   d. Run a policy scan on the home to get the list of files and use the list in **mmafmctl prefetch** on the cache to ensure all files are cached, or run a policy scan on the cache to test the cached flag of each file and report on any that are not fully cached.
2. Convert the fileset on the primary site to a primary using **mmafmctl**. The primary id is generated and a `psnap0` is created on the primary site. AFM gateway nodes must be defined in the primary site, and the file system is mounted on all gateway nodes before conversion. By default, RPO is disabled. You can use the **mmafmctl convertToPrimary** command to enable RPO. You can use the **mmchfileset** command later to enable RPOs.
3. Convert the home to a secondary and set the primary id by using **mmchfileset** or **mmafmctl** with the **convertToSecondary** option. Run on the primary cluster.

After the primary and secondary are converted and connected through primary ID, the `psnap0` queued from the primary fileset is played on the secondary fileset. The two filesets are ready for use. For more information, see *mmafmctl* command in *IBM Spectrum Scale: Command and Programming Reference*.

Note:

- IW/SW fileset must communicate at-least once to home before conversion. Newly created and inactive filesets might not convert successfully. When you convert a fileset in inactive state, it will convert to primary but will not create psnap0. Next access of the primary fileset will trigger recovery and create the psnap0 and move the psnap0 and the pending changes to home.

- If applications are in progress on the cache fileset during conversion, some inodes might be orphans and the `–check-metadata` option might show failures. It might be useful to use the `–nocheck-metadata` option in such cases.
- If cached files had been evicted from SW/IW cache, conversion with the `–check-metadata` option might show failures. It might be useful to use the `–nocheck-metadata` option in such cases.
- If home of an IW fileset is running applications during conversion, IW should revalidate with home to pull in all the latest data before conversion. During conversion, if any file/directory is not present in cache, it might result in a conflict error and fileset might go into NeedsResync state, AFM automatically fixes the conflicts during next recovery.
- You cannot convert a SW fileset that is in an unmounted state or NeedsResync state.
- Resync does not split data transfers even if parallel data transfer is configured, and the target is a mapping.

Changing configuration in an existing AFM DR relationship

See the following examples of changing gateway nodes and NFS server:

Changing NFS server at secondary

The NFS server, or mount path or IP address on secondary can change.

Existing AFM primary filesets need to be updated to point to the new target. As the secondary cluster and filesystem do not change, any of these changes can be reflected in the cache using **`mmafmctl`** with the **`changeSecondary –target-only`** option. If the NFS server changes, the new NFS server must be in the same secondary cluster and the architecture must be the same as the existing NFS server in the target path. If the NFS server is not in the same secondary cluster or the architecture is not the same, the **`changeSecondary`** must be performed without the **`–target-only`** option. If the target protocol changes from NSD to NFS or vice-versa, **`mmafmctl changeSecondary`** must be used without **`–target-only`**.

Changing gateway nodes in primary

You can add new gateway or remove existing gateway nodes using the **`mmchnode`** command. AFM automatically adjusts the existing filesets to use the latest configured gateways.

You must shutdown all the existing gateway nodes and then add or remove gateway using **`mmchnode`** command on a cluster that is currently running applications on AFM DR filesets. If the gateway nodes are changed while all gateway nodes are active, the gateway nodes might not be responding, or cluster-wide waiters might be observed after running **`mmchnode`**. Recycle the active gateway nodes.

It is not possible to remove existing gateway nodes if they are part of a mapping. You can remove the gateway nodes from the mapping using **`mmafmconfig`** command.

Note: Whenever you add or remove a gateway node, ensure that you update the list of IP addresses in the export map at home.

Chapter 12. Installing call home

The call home feature collects files, logs, traces, and details of certain system health events from different nodes and services. These details are shared with the IBM support center for monitoring and problem determination.

Prerequisites for installing call home

The following criteria must be met to use the call home functionality on your GPFS cluster:

- Red Hat Enterprise Linux 7.x, Ubuntu 16.04 LTS, and SLES 12 nodes are supported.
- Intel x86_64 and PowerPC LE and BE are supported.
- On IBM Z platform only Red Hat Enterprise Linux 7.x, Ubuntu 16.04 LTS, and SLES 12 are supported.
- At least one call home group must be defined.
- GPFS CCR (Clustered Configuration Repository) must be enabled.
- The call home node must be able to access the following IP addresses and ports. In the configurations with a proxy, these IP addresses and ports should be accessible over the proxy connection.
  - Host name: esupport.ibm.com
  - IP address: 129.42.56.189, 129.42.60.189, 129.42.52.189
  - Port number: 443

  The recommendation is to open 129.42.0.0/18.
- The call home node needs to have at least 1 GB free space under the data package directory in the dataStructureDump directory. It is preferable to have 2 - 3 GB of free space. The current dataStructureDump directory can be determined using the following command:

  ```
  mmdiag --config | grep "dataStructureDump "
  ```

  and is `tmp/mmfs` by default. You can change the default value of the `dataStructureDump` directory using the **`mmchconfig`** command.

  The call home node needs to have 1 GB free space under the data package directory (2 - 3 GB preferred).
- There must not be multiple nodes with the same short host name within the GPFS cluster.
- Only GPFS clusters where scp is configured as the file transfer method are supported. The file transfer method that is currently configured can be detected by running the `mmlscluster | grep "Remote file copy"` command:

  ```
  [root@g4240-51 ~]# mmlscluster | grep "Remote file copy"
  Remote file copy command: /usr/bin/scp
  ```

  Note: Call home is not supported in a sudo wrapper environment.

Steps for installing call home

The installation toolkit can be used for installing and upgrading call home RPMs, and for enabling and configuring the call home feature. For more information, see “Enabling and configuring call home using the installation toolkit” on page 289.

For information on configuring call home manually, see the *Monitoring the IBM Spectrum Scale system by using call home* section in the *IBM Spectrum Scale: Problem Determination Guide*.

Note: You must install the packages on all the nodes from which you want to gather data, and all the nodes where you want to upload data. The GPFS cluster nodes that are unsupported, unconfigured, or do not have call home installed, do not influence the call home capabilities.

1. Install the call home RPMs in the following order:

   Important: If you are using Ubuntu, use the corresponding *.deb files.

   a. gpfs.java-5.0.0-0.x86_64.rpm
   b. gpfs.callhome-ecc-client-5.0.0-0.000.noarch.rpm
   c. gpfs.callhome-5.0.0-0
      1) For RHEL: gpfs.callhome-5.0.0-0.000.el7.noarch.rpm
      2) For SLES: gpfs.callhome-5.0.0-0.000.sles12.noarch.rpm
2. Ensure that the following dependencies are resolved when you install call home:
   - Dependencies for gpfs.callhome-4.2 and later:
     - perl-List-MoreUtils
     - perl-Module-Runtime
     - perl-Try-Tiny
     - perl-Module-Implementation
     - perl-Params-Validate
     - perl-DateTime-Locale
     - perl-Sub-Install
     - perl-Params-Util
     - perl-Data-OptList
     - perl-Package-DeprecationManager
     - perl-Package-Stash-XS
     - perl-Package-Stash
     - perl-Class-Load
     - perl-Class-Singleton
     - perl-TimeDate
     - perl-DateTime
     - perl-DateTime-TimeZone
     - perl-Data-Dumper package

Chapter 13. Installing file audit logging

The following topics describe the requirements, limitations, and steps to install file audit logging.

Requirements and limitations for file audit logging

Use this information to understand the requirements and limitations for installing file audit logging.

RPM and package requirements

Every node that is capable of hosting any combination of brokers, zookeepers, producers, and consumers must have the following packages installed:

- GPFS Java (`gpfs.java rpm/package`)
- `openssl` and `libssl-dev` Linux libraries
- `librdkafka` (`gpfs.librdkafka rpm/package`)
- Kafka (`gpfs.kafka rpm/package`)

OS and hardware requirements

- RHEL 7.x on x86, RHEL 7.x on Power8 Little Endian, or Ubuntu 16.04/16.04.01 on x86.
- Linux Kernel on all platforms must be greater than or equal to 31000123.
- Minimum of three Linux quorum nodes running on approved OS and hardware (zookeepers).
- Minimum of three nodes to act as message queue servers (brokers) running on approved OS and hardware.

  Note: The nodes acting as zookeepers and brokers can be the same nodes (for example, a node acting as a zookeeper can also take on the broker role and vice versa).
- Message queue and broker nodes require 5 GB of local disk space (for the `/opt/kafka directory`) for every file system that is monitored by file audit logging. If this amount of space is not available, file audit logging enablement will fail. 10 GB of local disk space (for the `/opt/kafka directory`) is recommended and values lower than that will result in a warning when you enable a file system for file audit logging.

Security requirements and limitations

- SELinux in enforcing mode is not supported. For more information about support for SELinux in IBM Spectrum Scale, 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)).
- Root authority is required to run **`mmmsgqueue`** and **`mmaudit`**.
- The following TCP ports must be open on all nodes in the cluster:
  - 2181, 9092, and 9093 along with the range 2888:3888
- Sudo wrappers are not supported.

Restrictions imposed by mixed environments and protocols

- Events generated on non-Linux nodes will not be audited.
- Events generated on SLES Linux nodes will not be audited.
- IBM Spectrum Scale file audit logging has full support for the following protocols (support for all other protocols should be considered limited):
  - NFS ganesha
  - SMB
  - Native UNIX file access

- Multi-cluster environments are not supported. Remote mounts work, but access is not fully audited. Some error messages might be seen depending on which cluster file audit logging is enabled on.
- Events are not generated at or below the `cesSharedRoot` path.

GPFS file system requirements and limitations

- File audit logging can be enabled only for file systems that have been created or upgraded to IBM Spectrum Scale 5.0.0 or later.
- Space provisioning must be considered to store the generated events in the `.audit_log` fileset.
- The `.audit_log` fileset is protected from tampering. It cannot be easily deleted to free up space in the file system. This is done by creating the fileset in the IAM noncompliant mode, which allows expiration dates to be set on the files containing the audit records within the fileset.
- Events are not generated for file system activity within the file audit logging fileset itself.

GPFS and `spectrumscale` functional limitations

- The **`mmrestorefs`** command is not supported when restoring to a file system that contains a file audit Logging fileset.
- Conversion of a file audit logging fileset to AFM DR is not supported.

Miscellaneous requirements

- File audit logging is available in the IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only.

Manually installing file audit logging

Use this information to manually install file audit logging on your system.

If you do not have three protocol nodes, you do not want to use protocol nodes as your message queue servers, or you want the flexibility of using the file audit logging CLI commands, the following steps can be performed to provide access to the most customizable setup:

Important: Before you can install file audit logging, you must have the required packages for IBM Spectrum Scale installed on the node.

1. Ensure that the following two software packages (based on your OS) are installed on every node in the cluster that acts as a broker, Zookeeper, producer, or any combination of the three.
   - For Red Hat Enterprise Linux:
     - `gpfs.kafka-5.0*.rpm`
     - `gpfs.librdkafka-5.0*.rpm`

     These packages are available in `/usr/lpp/mmfs/5.0.0.x/gpfs_rpms/rhel7`.
   - For Ubuntu Linux:
     - `gpfs.kafka-5.0*.deb`
     - `gpfs.librdkafka-5.0*.deb`

     These packages are available in `/usr/lpp/mmfs/5.0.0.x/gpfs_debs/ubuntu16`.

   Note: For more information, see "Requirements and limitations for file audit logging" on page 361.
2. Use the **`mmmsgqueue enable -N <Nodes>`** command where **`<Nodes>`** is one of the following options:
   - A comma-separated list of at least three qualifying nodes.
   - The full path to a file that contains at least three qualifying nodes with each node on a separate line.
   - An existing node class that contains at least three qualifying nodes.

Note: This step will make the message queue use eligible Linux quorum nodes as Zookeepers and the nodes specified by the **-N** option as message queue servers/brokers.

3. Use the **mmaudit** command for each file system that you want to enable for file audit logging.

   Note: By using the CLI command versus the installation toolkit, you have more options including the ability to choose different devices to host the audit fileset.

Once you have enabled file audit logging for the desired file systems, the audit information is written to the defined audit filesets as described in the "Introduction to file audit logging" on page 121 topic.

If you would prefer to use the installation toolkit instead, see "Enabling and configuring file audit logging using the installation toolkit" on page 287.

Reinstalling file audit logging

Use this information to ensure a clean reinstall of file audit logging.

These steps assume that you have a working IBM Spectrum Scale cluster running 5.0.0, that file audit logging is enabled, and that you have access to the installation RPMs (for example, gpfs.kafka-5.0.0-0.ppc64le.rpm and gpfs.librdkafka-5.0.0-0.ppc64le.rpm) because you will not be able to run file audit logging with the directories removed unless you do a complete reinstall of the RPMs.

Note:

- The listed RPMs are examples for the PPC LE platform. The package names are slightly different for other operating systems and platforms.
- Performing the following steps will remove the configuration and logs entirely.

1. Run the following command for all file systems that are enabled for file audit logging. Substitute the file system name in the command:
   ```
   mmaudit <file system> disable
   ```
2. Run the following command to disable the message queue and remove the configuration:
   ```
   mmmsgqueue config --remove
   ```
3. Run the following command:
   ```
   mmdsh -N all rm -rf /opt/kafka
   ```
4. Remove the gpfs.librdkafka and gpfs.kafka packages from all nodes in the cluster.
5. You are now ready to install file audit logging. For more information, see *Manually installing file audit logging* or *Enabling and configuring file audit logging using the installation toolkit* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Chapter 14. Steps to permanently uninstall GPFS and/or Protocols

GPFS maintains a number of files that contain configuration and file system related data. Since these files are critical for the proper functioning of GPFS and must be preserved across releases, they are not automatically removed when you uninstall GPFS.

Follow these steps if you do not intend to use GPFS on any of the nodes in your cluster and you want to remove all traces of GPFS:

Attention: After following these steps and manually removing the configuration and file system related information, you will permanently lose access to all of your current GPFS data.

1. Unmount all GPFS file systems on all nodes by issuing the **mmumount all -a** command.
2. Issue the **mmdelfs** command for each file system in the cluster to remove GPFS file systems.
3. Issue the **mmdelnsd** command for each NSD in the cluster to remove the NSD volume ID from the device.

   If the NSD volume ID is not removed and the disk is again used with GPFS at a later time, you receive an error message when you specify the **mmcrnsd** command. For more information, see *NSD creation fails with a message referring to an existing NSD* in *IBM Spectrum Scale: Problem Determination Guide*.
4. Issue the **mmshutdown -a** command to shutdown GPFS on all nodes.
5. Uninstall GPFS from each node:
   - For your Linux nodes, run the de-installation program to remove GPFS for the correct version of the packages for your hardware platform and Linux distribution. For example, on SLES and Red Hat Enterprise Linux nodes:

     `rpm -e gpfs.crypto` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only)

     `rpm -e gpfs.adv` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only)

     ```
     rpm -e gpfs.ext
     rpm -e gpfs.gpl
     rpm -e gpfs.license.xx
     rpm -e gpfs.msg.en_US
     rpm -e gpfs.compression
     rpm -e gpfs.base
     rpm -e gpfs.docs
     rpm -e gpfs.gskit
     ```

     On Ubuntu Linux nodes:

     `dpkg -P gpfs.crypto` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only)

     `dpkg -P gpfs.adv` (IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition only)

     ```
     dpkg -P gpfs.ext
     dpkg -P gpfs.gpl
     dpkg -P gpfs.license.xx
     dpkg -P gpfs.msg.en-us
     ```

```
dpkg -P gpfs.compression
dpkg -P gpfs.base
dpkg -P gpfs.docs
dpkg -P gpfs.gskit
```

- For your AIX nodes:

  ```
  installp -u gpfs
  ```

- For your Windows nodes, follow these steps:
  a. Open **Programs and Features** in the Control Panel.
  b. Uninstall **IBM General Parallel File System**.
  c. Reboot the system.
  d. From a Command Prompt, run the following command:
     sc.exe delete mmwinserv

6. Remove the **/var/mmfs** and **/usr/lpp/mmfs** directories.
7. Remove all files that start with **mm** from the **/var/adm/ras** directory.
8. Remove **/tmp/mmfs** directory and its content, if present.

Cleanup procedures required if reinstalling with the installation toolkit

Before you can reinstall with the installation toolkit, you must perform some cleanup procedures first.

You can use the following procedures to clean up various stages of the previous GPFS installation and protocols deployment.

Starting the cleanup process

1. Clean up the installer directory by issuing the following command:

   ```
   mmdsh -N NodeList rm -rf /usr/lpp/mmfs/5.0.0.0
   ```

   Note: The installer directory in this command depends upon the release version.
2. Check /etc/resolv.conf for any changes that pointed the authentication directory to DNS, and remove them.
3. If the object protocol was enabled and storage policies were created, use the **mmobj policy list -v** command and save the output list before you execute the **mmces service disable OBJ** command.

Cleaning up the authentication configurations

Use these commands to completely remove the current authentication configuration for all protocols, NFS/SMB only, or Object only. Be aware that doing so may cause loss of access to previously written data.

1. Remove the configurations. To do so, issue the following commands:

   ```
   mmuserauth service remove --data-access-method file
   mmuserauth service remove --data-access-method object
   ```

2. Delete the ID mapping information when authentication was configured with AD. To do so, use the following command.

   Note: Once this ID mapping information is deleted, you might not be able to access existing files (or you might experience other unforeseen access issues), so do this only if you do not need access to existing data.

   ```
   mmuserauth service remove --data-access-method file --idmapdelete
   ```

Disabling CES

Disable CES on every node *nodeNameX*. To do so, issue the following command:

```
mmchnode -N nodeNameX --ces-disable
```

Protocol cleanup steps

1. Use the following steps to clean up some of the object state on the system:

   ```
   mmdsh -N cesNodes systemctl stop postgresql-obj
   rm -rf path_to_cesSharedRoot/object/keystone
   ```

   Delete the objectization temp directory. In the following example, the temp directory was created in filesystem `fs1` at `/ibm/fs1/ibmobjectizer/tmp`:

   ```
   rm -rf  /ibm/fs1/ibmobjectizer/tmp
   ```

2. Remove the fileset created for object.

   In the following example, the configured fileset `object_fileset` in file system `fs1` was linked at `/gpfs/fs1`.

   ```
   mmlsfileset fs1
   mmunlinkfileset fs1 object_fileset
   mmdelfileset fs1 object_fileset -f
   ```

3. Remove any fileset created for an object storage policy.

   Run the **mmobj policy list -v** command. If for some reason **mmobj policy list -v** cannot be executed (For example, if the Object Protocol or CES was already disabled) or you cannot detect which filesets are used for object storage policies, contact the IBM Support Center.

   For example:

   If the **mmobj policy list** returned the following and the filesets got created in file system fs1:

```
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"
```

   Use the following commands for each row except the first:

   ```
   mmunlinkfileset fs1 sof-policy1
   mmdelfileset fs1 sof-policy1 -f
   ```

4. Clean up installer remnants. To do so, issue the following command:

   ```
   mmdsh -N NodeList rm -rf /root/.chef
   ```

5. Clear the **cesSharedRoot** configuration by issuing the following command:

   ```
   mmchconfig cesSharedRoot=DEFAULT
   ```

6. Check your recent package installs to determine which packages need to be removed by issuing the following commands:

   RHEL 7.x and SLES 12

   ```
   rpm -qa --last|more
   ```

   Ubuntu 16

   ```
   dpkg -l
   ```

   Once you have determined which packages need to be removed, do so by issuing the following commands, as needed:

   RHEL 7.x and SLES 12

   ```
   mmdsh -N NodeList yum erase gpfs.smb nfs-ganesha-mount \
   nfs-ganesha-vfs PyQt4 nfs-ganesha-utils phonon-backend-gstreamer phonon sip qt-x11 kde-filesystem \
   qt qt-settings libmng nfs-ganesha-proxy nfs-ganesha-nullfs nfs-ganesha-gpfs nfs-ganesha chef -y

   mmdsh -N NodeList yum erase python-webob MySQL-python mariadb-server perl-DBD-MySQL mariadb \
   python-futures python-warlock python-jsonpatch python-jsonschema python-jsonpointer python-cmd2 \
   python-cliff pyparsing memcached python-oslo-utils python-oslo-serialization python-oslo-i18n \
   python-babel babel python-keyring python-stevedore python-prettytable python-pbr python-oslo-config \
   ```

```
python-netaddr python-iso8601 python-dnspython python-urllib3 python-six python-requests python-paste-deploy \
python-tempita python-paste python-netifaces python-simplejson python-greenlet python-eventlet -y

mmdsh -N NodeList yum erase gpfs.smb gpfs.smb-debuginfo sssd-ipa sssd python-sssdconfig sssd-proxy \
sssd-ldap sssd-krb5 libipa_hbac sssd-krb5-common sssd-common-pac sssd-ad sssd-common cyrus-sasl-gssapi c-ares \
libsss_idmap libdhash yp-tools ypbind -y

mmdsh -N NodeList yum erase python-routes python-repoze-lru python-oauthlib python-crypto \
python-qpid-common python-qpid python-dogpile-cache python-fixtures python-dogpile-core python-testtools \
python-extras python-oslo-context python-kombu python-anyjson python-amqp python-mimeparse \
python-markupsafe PyYAML python-passlib libyaml python-ibm-db-sa python-ibm-db python-sqlparse \
python-memcached python-webob python-posix_ipc python-sqlalchemy python-pbr python-greenlet -y

mmdsh -N NodeList yum erase nfs-ganesha gpfs.gss.pmsensors gpfs.gss.pmcollector gpfs.pm-ganesha boost-regex -y

mmdsh -N NodeList yum erase gpfs.smb python-oslo-i18n openstack-utils qt-settings nfs-ganesha-gpfs nfs-ganesha -y

mmdsh -N NodeList yum erase python-ordereddict python-routes python-passlib python-amqp python-crypto -y

mmdsh -N NodeList yum erase mariadb-libs -y

mmdsh -N NodeList yum erase pmswift nfs-ganesha-gpfs nfs-ganesha gpfs.smb -y

mmdsh -N NodeList yum erase python-cryptography python-enum34 swift3 python-pyeclib liberasurecode \
openstack-utils crudini python-msgpack python-wsgiref python-extras python-pycparser python-ply python-cffi \
python-ibm-db openstack-selinux xmlsec1 python-pyasn1 python-mimeparse python-retrying \
postgresql-server postgresql python-psycopg2 python-repoze-who jerasure gf-complete python-zope-interface \
postgresql-libs pytz python-oslo-context python-webob python-six nfs-ganesha-gpfs nfs-ganesha -y


mmdsh -N NodeList yum erase spectrum-scale-object spectrum-scale-object-selinux \
openstack-swift openstack-swift-account openstack-swift-container openstack-swift-object openstack-swift-proxy \
python-swiftclient swift3 swiftonfile keystonemiddleware openstack-keystone python-keystone python-keystoneauth1 \
python-cinderclient python-glanceclient python-keystoneclient python-neutronclient python-novaclient \
python-openstackclient python-swiftclient -y
```

Ubuntu 16

```
mmdsh -N NodeList dpkg -P gpfs.smb nfs-ganesha-mount nfs-ganesha-vfs \
PyQt4 nfs-ganesha-utils phonon-backend-gstreamer phonon sip qt-x11 kde-filesystem qt qt-settings libmng \
nfs-ganesha-proxy nfs-ganesha-nullfs nfs-ganesha-gpfs nfs-ganesha chef -y

mmdsh -N NodeList dpkg -P python-webob MySQL-python mariadb-server perl-DBD-MySQL mariadb \
python-futures python-warlock python-jsonpatch python-jsonschema python-jsonpointer python-cmd2 \
python-cliff pyparsing memcached python-oslo-utils python-oslo-serialization python-oslo-i18n python-babel \
babel python-keyring python-stevedore python-prettytable python-pbr python-oslo-config python-netaddr \
python-iso8601 python-dnspython python-urllib3 python-six python-requests python-paste-deploy \
python-tempita python-paste python-netifaces python-simplejson python-greenlet python-eventlet -y

mmdsh -N NodeList dpkg -P gpfs.smb gpfs.smb-debuginfo sssd-ipa sssd python-sssdconfig \
sssd-proxy sssd-ldap sssd-krb5 libipa_hbac sssd-krb5-common sssd-common-pac sssd-ad sssd-common \
cyrus-sasl-gssapi c-ares libsss_idmap libdhash yp-tools ypbind -y

mmdsh -N NodeList dpkg -P python-routes python-repoze-lru python-oauthlib python-crypto \
python-qpid-common python-qpid python-dogpile-cache python-fixtures python-dogpile-core python-testtools \
python-extras python-oslo-context python-kombu python-anyjson python-amqp python-mimeparse \
python-markupsafe PyYAML python-passlib libyaml python-ibm-db-sa python-ibm-db python-sqlparse \
python-memcached python-webob python-posix_ipc python-sqlalchemy python-pbr python-greenlet -y

mmdsh -N NodeList dpkg -P nfs-ganesha gpfs.gss.pmsensors gpfs.gss.pmcollector gpfs.pm-ganesha boost-regex -y

mmdsh -N NodeList dpkg -P gpfs.smb python-oslo-i18n openstack-utils qt-settings nfs-ganesha-gpfs nfs-ganesha -y

mmdsh -N NodeList dpkg -P python-ordereddict python-routes python-passlib python-amqp python-crypto -y

mmdsh -N NodeList dpkg -P mariadb-libs -y

mmdsh -N NodeList dpkg -P pmswift nfs-ganesha-gpfs nfs-ganesha gpfs.smb -y

mmdsh -N NodeList dpkg -P python-cryptography python-enum34 swift3 python-pyeclib liberasurecode \
openstack-utils crudini python-msgpack python-wsgiref python-extras python-pycparser python-ply python-cffi \
python-ibm-db openstack-selinux xmlsec1 python-pyasn1 python-mimeparse python-retrying postgresql-server \
postgresql python-psycopg2 python-repoze-who jerasure gf-complete python-zope-interface postgresql-libs \
pytz python-oslo-context python-webob python-six nfs-ganesha-gpfs nfs-ganesha -y
```

```
mmdsh -N NodeList dpkg -P spectrum-scale-object spectrum-scale-object-selinux \
openstack-swift openstack-swift-account openstack-swift-container openstack-swift-object openstack-swift-proxy \
python-swiftclient swift3 swiftonfile keystonemiddleware openstack-keystone python-keystone python-keystoneauth1 \
python-cinderclient python-glanceclient python-keystoneclient python-neutronclient python-novaclient \
python-openstackclient python-swiftclient -y
```

7. Issue **rpm -qa --last|more** or **dpkg -l** on all nodes again, as the preceding list does not contain everything in the new builds.

   If the pmswift package still exists, remove it as follows:

   RHEL 7.x and SLES 12

   ```
   mmdsh -N NodeList rpm -e pmswift-5.0.0-0.noarch –noscripts
   ```

   Ubuntu 16

   ```
   mmdsh -N NodeList dpkg -P pmswift-5.0.0-0.noarch –noscripts
   ```

   You might also need to remove gpfs.smb.

8. Clean up additional chef files. To do so, issue the following commands:

   ```
   mmdsh -N NodeList rm -rf /var/chef
   mmdsh -N NodeList rm -rf /etc/chef
   mmdsh -N NodeList rm -rf /opt/chef
   mmdsh -N NodeList rm -rf /root/.chef
   mmdsh -N NodeList rm -rf /root/.berkshelf
   ```

9. Clean up performance monitoring tool files. To do so, issue the following commands:

   ```
   mmdsh -N NodeList rm -rf /opt/IBM/zimon*
   mmdsh -N NodeList rm -rf /usr/IBM/zimon*
   mmdsh -N NodeList rm -rf /var/log/cnlog/zimon*
   mmdsh -N NodeList rm -rf /var/lib/yum/repose/x86_64/7Server/*zimon*
   mmdsh -N NodeList rm -rf /var/lib/yum/repose/ppc64/7Server/*zimon*
   ```

10. Clean up CTDB. To do so, issue the following commands:

    ```
    mmdsh -N NodeList rm -rf /var/lib/ctdb
    ```

11. Clean up Swift files. To do so, issue the following commands:

    ```
    mmdsh -N NodeList rm -rf /etc/swift
    mmdsh -N NodeList rm -rf /var/lib/mysql
    mmdsh -N NodeList rm -rf /etc/keystone
    mmdsh -N NodeList rm -rf /var/lib/keystone
    mmdsh -N NodeList rm -rf /root/openrc
    mmdsh -N NodeList rm -rf /var/cache/yum/x86_64/7Server/*
    mmdsh -N NodeList rm -rf /var/cache/yum/ppc64/7Server/*
    mmdsh -N NodeList rm -rf /var/log/maria*
    mmdsh -N NodeList rm -rf /usr/share/pixmaps/comps/maria*
    mmdsh -N NodeList rm -rf /var/log/keystone
    mmdsh -N NodeList rm -rf /var/spool/cron/keystone
    mmdsh -N NodeList rm -rf /usr/lib/python2.7/site-packages/sos/plugins/openstack_keystone*
    mmdsh -N NodeList rm -rf /tmp/keystone-signing-swift
    mmdsh -N NodeList rm -rf /usr/lib/python2.7/site-packages/sos/plugins/*
    mmdsh -N NodeList rm -rf /var/lib/yum/repos/x86_64/7Server/icm_openstack
    mmdsh -N NodeList rm -f /usr/lib/python2.7/site-packages/swiftonfile-2.5.0_0-py2.7.egg*
    mmdsh -N NodeList rm -f /usr/bin/*objectizer*.pyc
    mmdsh -N NodeList rm -f /usr/bin/generate_dbmap.pyc
    mmdsh -N NodeList systemctl disable openstack-keystone.service
    mmdsh -N NodeList systemctl disable openstack-swift-container-updater.service
    mmdsh -N NodeList systemctl disable openstack-swift-container-update
    mmdsh -N NodeList systemctl disable openstack-swift-object-updater.service
    mmdsh -N NodeList systemctl disable openstack-swift-container-auditor.service
    mmdsh -N NodeList systemctl disable openstack-swift-container-replicator.service
    mmdsh -N NodeList systemctl disable openstack-swift-container.service
    mmdsh -N NodeList systemctl disable openstack-swift-object-replicator.service
    mmdsh -N NodeList systemctl disable openstack-swift-object.service
    mmdsh -N NodeList systemctl disable openstack-keystone.service
    mmdsh -N NodeList systemctl disable openstack-swift-account-reaper.service
    mmdsh -N NodeList systemctl disable openstack-swift-account-auditor.service
    mmdsh -N NodeList systemctl disable openstack-swift-account-replicator.service
    mmdsh -N NodeList systemctl disable openstack-swift-account.service
    mmdsh -N NodeList systemctl disable openstack-swift-proxy.service
    mmdsh -N NodeList systemctl disable openstack-swift-object-auditor.service
    mmdsh -N NodeList systemctl disable openstack-swift-object-expirer.service
    ```

```
mmdsh -N NodeList systemctl disable openstack-swift-container-reconciler.service
mmdsh -N NodeList rm -rf
/var/lib/yum/yumdb/o/0b01eb65826df92befd8c161798cb842fa3c941e-openstack-utils-2015.1-201502031913.ibm.el7.7-noarch
```

12. Reboot the nodes. To do so, issue the following command:

```
mmdsh -N NodeList shutdown -r now
```

If some nodes do not fully come up, do the following:

a. Power off that node.

b. Wait one minute, then power the node back on.

Note: Some nodes might need to be powered off and on more than once.

13. Clean up Yum on all nodes. To do so, issue the following commands:

```
mmdsh -N NodeList yum clean all
mmdsh -N NodeList rm -rf /etc/yum.repos.d/gpfs.repo /etc/yum.repos.d/icm* /etc/yum.repos.d/ces.repo
mmdsh -N NodeList rm -rf /etc/yum.repos.d/epel* /etc/yum.repos.d/rdo*
mmdsh -N NodeList rm -rf /var/lib/yum/repos/x86_64/7Server/*
mmdsh -N NodeList rm -rf /var/lib/yum/repos/ppc64/7Server/*
```

14. Remove GPFS.

Notes:

- This step is not required if you know that GPFS has not changed between old and new builds.
- This step is not required if you prefer to perform an upgrade of GPFS.
- This step is not required if you would like to keep the base GPFS installation intact and merely rerun the protocols deployment.
- (To permanently remove GPFS, see Chapter 14, “Steps to permanently uninstall GPFS and/or Protocols,” on page 365.)

a. Check which GPFS packages are on each node by issuing the following command:

RHEL 7.x and SLES 12

```
mmdsh -N NodeList rpm -qa|grep gpfs
```

Ubuntu 16

```
mmdsh -N NodeList dpkg -l|grep gpfs
```

The system displays output similar to the following:

```
gpfs.ext-5.0.0-0.x86_64
gpfs.msg.en_US-5.0.0-0.noarch
gpfs.gskit-8.0.50-79.x86_64
gpfs.license.xx-5.0.0-0.x86_64
gpfs.crypto-5.0.0-0.x86_64
gpfs.adv-5.0.0-0.x86_64
gpfs.docs-5.0.0-0.noarch
gpfs.base-5.0.0-0.x86_64
gpfs.gpl-5.0.0-0.noarch
```

b. Before removing anything, make sure that GPFS is shut down on all nodes. To do so, issue the following command:

```
mmshutdown -a
```

c. Remove the packages by issuing the following commands *in the order shown*.

Note: When you remove gpfs.base, you will lose **mmdsh** access.
Therefore, be sure to remove gpfs.base last, as shown here.

RHEL 7.x and SLES 12

```
mmdsh -N NodeList rpm -e gpfs.base-debuginfo-5.0.0-0.x86_64
mmdsh -N NodeList rpm -e gpfs.crypto-5.0.0-0.x86_64
mmdsh -N NodeList rpm -e gpfs.adv-5.0.0-0.x86_64
mmdsh -N NodeList rpm -e gpfs.ext-5.0.0-0.x86_64
mmdsh -N NodeList rpm -e gpfs.msg.en_US-5.0.0-0.noarch
mmdsh -N NodeList rpm -e gpfs.gskit-8.0.50-79.x86_64
mmdsh -N NodeList rpm -e gpfs.compression-5.0.0-0.x86_64
mmdsh -N NodeList rpm -e gpfs.license.xx-5.0.0-0.x86_64
mmdsh -N NodeList rpm -e gpfs.docs-5.0.0-0.noarch
mmdsh -N NodeList rpm -e gpfs.gpl-5.0.0-0.noarch
mmdsh -N NodeList rpm -e gpfs.callhome-5.0.0-0.000.el7.noarch
mmdsh -N NodeList rpm -e gpfs.base-5.0.0-0.x86_64
```

Ubuntu 16

```
mmdsh -N NodeList dpkg -P gpfs.base-debuginfo_5.0.0-0.x86_64
mmdsh -N NodeList dpkg -P gpfs.crypto_5.0.0-0.x86_64
mmdsh -N NodeList dpkg -P gpfs.adv_5.0.0-0.x86_64
mmdsh -N NodeList dpkg -P gpfs.ext_5.0.0-0.x86_64
mmdsh -N NodeList dpkg -P gpfs.msg.en-us_5.0.0-0.all
mmdsh -N NodeList dpkg -P gpfs.gskit_8.0.50-79.x86_64
mmdsh -N NodeList dpkg -P gpfs.compression_5.0.0-0.x86_64
mmdsh -N NodeList dpkg -P gpfs.license.xx_5.0.0-0.x86_64
mmdsh -N NodeList dpkg -P gpfs.docs_5.0.0-0.noarch
mmdsh -N NodeList dpkg -P gpfs.gpl_5.0.0-0.noarch
mmdsh -N NodeList dpkg -P gpfs.base_5.0.0-0.x86_64
```

Note: In the preceding commands, the package name `gpfs.license.xx` needs to be changed depending on the IBM Spectrum Scale product edition.

15. Reinstall GPFS.
16. Proceed to "Installation prerequisites" on page 224 and "Using the installation toolkit to perform installation tasks: Explanations and examples" on page 272.

Note: If you wish to remove all cluster configurations, you can also apply the **mmdelnode -f** command to each node; however, if you choose to do so, you will also have to remake `cluster/nsds/filesystems`.

Uninstalling the Performance Monitoring tool

You can uninstall the Performance Monitoring tool by running the following commands on all nodes that has monitoring enabled on it.

To uninstall the sensor on the node, use the **`rpm -evh gpfs.gss.pmsensors`** command.

To uninstall the collector on the node, use the **`rpm -evh gpfs.gss.pmcollector`** command.

To uninstall the Object proxy on the node, use the **`rpm -evh pmswift`** command.

To uninstall the NFS proxy on the node, use the **`rpm -evh gpfs.pm-ganesha`** command.

For reinstalling sensors and the collector, see "Manually installing the Performance Monitoring tool" on page 250.

Uninstalling the IBM Spectrum Scale management GUI

Do the following to uninstall management GUI and remove the performance monitoring components that are installed for the GUI:

1. Issue the **`systemctl stop`** command as shown in the following example:

   ```
   systemctl stop gpfsgui
   ```

2. If the GUI runs on an IBM Spectrum Scale cluster where sudo wrapper is enabled, export the name of the user that was configured as the file system administrator as an environment variable by issuing the following command:

   ```
   export SUDO_USER=gpfsadmin
   ```

   In this example, the name of the sudo user is *gpfsadmin*. Exporting the environment variable is necessary so that the uninstall process get to know which user name must be used to run the administrative file system commands that are necessary while uninstalling the GUI. For more information on how to configure the IBM Spectrum Scale GUI to use sudo wrapper, see *Configuring IBM Spectrum Scale(tm) GUI to use sudo wrapper* in *IBM Spectrum Scale: Administration Guide*.

3. Issue the following command to clean up the GUI database:
   ```
   psql postgres postgres -c "drop schema fscc cascade"
   ```
4. Issue `rpm -e gpfs.gui-5.0.0-0.noarch` to remove the GUI rpm.
5. Issue one of the following commands depending on the platform to remove the GUI package.
   - Red Hat Enterprise Linux and SLES
     ```
     rpm -e gpfs.gui-5.0.0-0.noarch
     ```
   - Ubuntu
     ```
     dpkg -r gpfs.gui_5.0.0-0_amd64.deb
     ```

Removing nodes from management GUI-related node class

If you are reinstalling IBM Spectrum Scale, in some scenarios you might need to remove some nodes from the node classes that are used by the management GUI.

For information about these node classes, see "Node classes used for the management GUI" on page 257. If you want to remove the GUI designation of a node, you must remove it from the `GUI_MGT_SERVERS` node class. After you remove a node from this node class, it is no longer designated as a GUI node and GUI services are not started on this node after reinstalling IBM Spectrum Scale. To remove a node from the `GUI_MGT_SERVERS` node class, use the **mmchnodeclass** command only when the GUI services are not running.

On the node that you want to remove from the `GUI_MGT_SERVERS` node class, run the following commands:

```
systemctl stop gpfsgui
mmchnodeclass GUI_MGMT_SERVERS delete -N guinode
```

These commands stop the GUI services and remove the GUI node from the `GUI_MGT_SERVERS` node class.

Note: If you use **mmchnodeclass** to change the `GUI_MGT_SERVERS` node class while the GUI services are running, the management GUI adds the removed node to the `GUI_MGT_SERVERS` node class again.

Permanently uninstall Cloud services and clean up the environment

Before setting up a fresh multi-node, you must clean up the environment.

Perform the following steps to do a cleanup of the environment and be able to create a fresh multi-node setup.

1. Recall any data that is migrated to the cloud storage by Transparent cloud tiering. For more information, see the *Recalling files from the cloud storage tier* topic in the *IBM Spectrum Scale: Administration Guide*.
2. Delete the container pair set by issuing a command similar to this:
   ```
   mmcloudgateway containerpairset delete --cloud-nodeclass cloud1 --container-pair-set-name vm206
   ```

   For more information, see *Creating a container pair set* topic in the *IBM Spectrum Scale: Administration Guide*.
3. Delete the Cloud services by issuing a command similar to this:
   ```
   mmcloudgateway cloudservice delete --cloud-nodeclass cloud --cloud-service-name newServ
   ```

   For more information, see *Creating a Cloud services* topic in the *IBM Spectrum Scale: Administration Guide*.

4. Delete the CSAP by issuing a command similar to this:

   ```
   mmcloudgateway cloudStorageAccessPoint delete --cloud-nodeclass cloud --cloud-storage-access-point-name  csap1
   ```

   For more information, see *Defining cloud storage access points* topic in the *IBM Spectrum Scale: Administration Guide*.
5. Delete the cloud storage account by issuing a command similar to this:

   ```
   mmcloudgateway account delete --cloud-nodeclass cloud --account-name new
   ```

   For more information, see the *Managing a cloud storage account* topic in the *IBM Spectrum Scale: Administration Guide*.
6. Stop the Cloud services on all nodes by issuing a command similar to this:

   ```
   mmcloudgateway service stop -N cloud1
   ```

   For more information, see the Stopping the *Transparent cloud tiering* topic in the *IBM Spectrum Scale: Administration Guide*.
7. Disable Cloud services nodes on the node group by issuing a command similar to this:

   ```
   mmchnode --cloud-gateway-disable -N vm1,vm2 --cloud-gateway-nodeclass cloud1
   ```

   For more information, see the *Designating the Transparent cloud tiering nodes* topic in the *IBM Spectrum Scale: Administration Guide*.
8. Delete the node class if required. For more information, see the **mmdelnodeclass** command in the *IBM Spectrum Scale: Command and Programming Reference*
9. Uninstall the Cloud services RPMs (both server and client) from all the nodes by issuing the following command:
   - `rpm -e gpfs.tct.server-1-1.4.x86_64`
   - `rpm -e gpfs.tct.client-x.x.x.x86_64.rpm`

Chapter 15. Upgrading

To upgrade to a newer version of IBM Spectrum Scale, first consider the version that you are upgrading from and then consider coexistence and compatibility issues.

IBM Spectrum Scale supports a limited form of backward compatibility between two adjacent releases. Limited backward compatibility allows you to temporarily operate with a mixture of IBM Spectrum Scale nodes running on the newer version and nodes running an earlier version:

- Within a cluster this enables you to perform an online upgrade to the new IBM Spectrum Scale version of the code provided upgrade from your current version to the newer version is supported.
- In a multicluster environment this allows the individual clusters to be upgraded on their own schedules. Access to the file system data can be preserved even though some of the clusters might still be running at an earlier version.

Note: Nodes with Cluster Export Services (CES) do not allow backwards compatibility. All CES nodes must be at the same version prior to returning client access.

Online upgrades allow you to install new IBM Spectrum Scale code one node at a time while maintaining access to the file systems without shutting down IBM Spectrum Scale on other nodes. However, you must upgrade all nodes within a short time. The time dependency exists because some IBM Spectrum Scale features in the newer version become available on each node as soon as the node is upgraded, while other features will not become available until you upgrade all participating nodes.

You can also perform **offline upgrades**, if you can shut down the entire cluster. An offline upgrade is similar to the online upgrade procedure. However, due to the entire cluster being offline, it is possible to jump straight to the latest code level instead of hopping to an intermediate level, as may be required during an online upgrade.

Once all nodes have been upgraded to the new code, you must finalize the upgrade by running the commands **mmchconfig release=LATEST** and **mmchfs -V full** (or **mmchfs -V compat**). Also, certain new features may require you to run the **mmmigratefs** command to enable them.

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

IBM Spectrum Scale upgrade consists of these topics:

- “Upgrading to IBM Cloud Object Storage software level 3.7.2 and above” on page 407
- “Upgrading IBM Spectrum Scale components with the installation toolkit” on page 407
- “Migrating from Express Edition to Standard Edition” on page 417
- “Completing the upgrade to a new level of IBM Spectrum Scale” on page 418
- “Reverting to the previous level of IBM Spectrum Scale” on page 422
- “Coexistence considerations” on page 423
- “Compatibility considerations” on page 423
- “Considerations for IBM Spectrum Protect for Space Management” on page 424
- “GUI user role considerations” on page 424
- “Applying maintenance to your GPFS system” on page 424

IBM Spectrum Scale supported upgrade paths

Use this information to understand the supported upgrade paths for IBM Spectrum Scale.

The following table lists the supported online upgrade paths for IBM Spectrum Scale.

Table 44. IBM Spectrum Scale supported online upgrade paths

Upgrading from:	**To: 3.5**	**To: 4.1.0.x**	**To: 4.1.1.x**	**To: 4.2.0.x**	**To: 4.2.1.x**	**To: 4.2.2.x**	**To: 4.2.3.x**	**To: 5.0.0.x**
3.5	✓	✓	✓	X	X	X	X	X
4.1.0.x	--	✓	✓	✓	✓	✓	✓	X
4.1.1.x	--	--	✓	✓	✓	✓	✓	X
4.2.0.x	--	--	--	✓	✓	✓	✓	✓
4.2.1.x	--	--	--	--	✓	✓	✓	✓
4.2.2.x	--	--	--	--	--	✓	✓	✓
4.2.3.x	--	--	--	--	--	--	✓	✓
5.0.0.x	--	--	--	--	--	--	--	✓

- ✓: Supported
- X: Not supported
- --: Not applicable

Remember:

- Online upgrade support for protocol nodes depends on which services have been enabled and deployed. Different protocol services have different levels of online upgrade support. For more information, see “Online upgrade support for protocols and performance monitoring” on page 388.
- Offline upgrade can be performed from all supported versions to all supported versions.
- IBM Spectrum Scale installation toolkit can be used for upgrades from version 4.1.1.x onwards.

Upgrade scenario 1: Direct online upgrades from IBM Spectrum Scale version 4.1.1.x to 5.0.x.x cannot be done. Perform an online upgrade to version 4.2.x (4.2.0.x, 4.2.1.x, 4.2.2.x, or 4.2.3.x) and then do another online upgrade to version 5.0.x.x. However, offline upgrade from version 4.1.1.x to 5.0.x.x can be done.

Upgrade scenario 2: Online upgrade from IBM Spectrum Scale version 4.2.x to 5.0.x.x can be done.

Related information:

IBM Spectrum Scale Software Version Recommendation Preventive Service Planning

ESS 5.3 supported upgrade paths

Upgrading to IBM Spectrum Scale 5.0.x from IBM Spectrum Scale 4.2.y

IBM Spectrum Scale 5.0.x release supports node-at-a-time upgrade if the previous nodes in the cluster are running IBM Spectrum Scale 4.2.y release. IBM Spectrum Scale nodes with 4.2.y release can coexist and interoperate with nodes that are running IBM Spectrum Scale 5.0.x, except for performance monitoring, SMB, and object. However, new functions that depend on format changes will not be available until all nodes have been upgraded.

To upgrade a cluster to IBM Spectrum Scale 5.0.x release from IBM Spectrum Scale 4.2.y release, perform the following steps:

1. Stop all user activity in the file systems on the designated node.

   For information about how to stop and unmount NFS running over IBM Spectrum Scale file systems, see *Unmounting a file system after NFS export* in *IBM Spectrum Scale: Administration Guide*.
2. Follow any local administrative backup procedures to ensure protection of your file system data in the event of a failure.
3. Cleanly unmount the mounted IBM Spectrum Scale file systems on the designated node. Do not use force to unmount the designated node.
4. Stop IBM Spectrum Scale on the node to be upgraded in the cluster using the **mmshutdown** command.

   For example: **mmshutdown -N k164n04**, where k164n04 is the designated node.

   Note:

   It is not required to terminate **mmccrmonitor** and other mm processes because these are handled by GPFS upgrade post-install scripts.
5. Upgrade to IBM Spectrum Scale 5.0.x release depending on the operating system.

   Attention: Starting with release 4.2.3, IBM Spectrum Scale Express Edition is no longer available. For information on upgrading from IBM Spectrum Scale Express Edition 4.2.2.x or earlier to IBM Spectrum Scale Standard Edition 4.2.3.x or later, see "Migrating from Express Edition to Standard Edition" on page 417.

   For Linux nodes:

   - Extract the IBM Spectrum Scale software as described in the "Extracting the IBM Spectrum Scale software on Linux nodes" on page 231 topic. For information about the location of extracted installation images, see "Location of extracted packages" on page 235.
   - On **Red Hat Enterprise Linux (RHEL) and SLES nodes**, issue the following commands to upgrade GPFS packages:
     - For IBM Spectrum Scale Standard Edition, issue the following commands:

       ```
       rpm -Fvh gpfs.base*.rpm gpfs.doc*.rpm gpfs.gpl*rpm gpfs.gskit*rpm gpfs.msg*rpm
       gpfs.ext*rpm gpfs.license.std*.rpm
       rpm -ivh gpfs.compression*.rpm
       ```
     - For IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition, issue the following commands:

       [For Advanced Edition]

       ```
       rpm -Fvh gpfs.base*.rpm gpfs.doc*.rpm gpfs.gpl*rpm gpfs.gskit*rpm gpfs.msg*rpm
       gpfs.ext*rpm gpfs.adv*rpm gpfs.crypto*rpm gpfs.license.adv*.rpm
       rpm -ivh gpfs.compression*.rpm
       ```

       [For Data Management Edition]

       ```
       rpm -Fvh gpfs.base*.rpm gpfs.doc*.rpm gpfs.gpl*rpm gpfs.gskit*rpm gpfs.msg*rpm
       gpfs.ext*rpm gpfs.adv*rpm gpfs.crypto*rpm gpfs.license.dm*.rpm
       rpm -ivh gpfs.compression*.rpm
       ```

       Note: These packages must be installed to use the AFM-based Async DR, Encryption, and Transparent cloud tiering features.

    - gpfs.adv-5.0.0-0*.rpm
    - gpfs.crypto-5.0.0-0*.rpm

- On **Ubuntu Linux nodes**, do the following:
  - For IBM Spectrum Scale Standard Edition, issue the following command:

    ```
    dpkg -i gpfs.base*deb gpfs.gpl*deb gpfs.license.std*deb gpfs.gskit*deb
    gpfs.msg*deb gpfs.doc*deb gpfs.ext*deb gpfs.compression*deb
    ```

  - For IBM Spectrum Scale Advanced Edition, issue the following command:

    ```
    dpkg -i gpfs.base*deb gpfs.gpl*deb gpfs.license.adv*deb gpfs.gskit*deb gpfs.msg*deb
    gpfs.doc*deb gpfs.ext*deb  gpfs.adv*deb gpfs.crypto*deb gpfs.compression*deb
    ```

  - For IBM Spectrum Scale Data Management Edition, issue the following command:

    ```
    dpkg -i gpfs.base*deb gpfs.gpl*deb gpfs.license.dm*deb gpfs.gskit*deb gpfs.msg*deb
    gpfs.doc*deb gpfs.ext*deb  gpfs.adv*deb gpfs.crypto*deb gpfs.compression*deb
    ```

  - To optionally install the performance monitoring tool on Ubuntu 16, issue the following command:

    ```
    dpkg -i gpfs.gss.pmsensors_*.U16.*.deb
    gpfs.gss.pmcollector_*.U16.*.deb
    ```

- After successful upgrade, rebuild the GPFS portability layer (GPL). For more information, see "Building the GPFS portability layer on Linux nodes" on page 237.

For Hadoop users: To upgrade the IBM Spectrum Scale Hadoop Connector, see *Upgrading IBM Spectrum Scale connector* in *IBM Spectrum Scale: Administration Guide*.

Note: After upgrading from release 4.2.y to 5.0.x, you might see the `pmcollector service critical error` on GUI nodes. In this case, restart the `pmcollector` service by running the **`systemctl restart pmcollector`** command on all GUI nodes.

For AIX nodes:

Note: Specific to AIX, when upgrading from 4.2.y to 5.0.x, you must upgrade to 5.0.0 first and then you can upgrade to 5.0.x.

- Copy the installation images and install the IBM Spectrum Scale licensed programs as described in the Chapter 5, "Installing IBM Spectrum Scale on AIX nodes," on page 319 section.

For Windows nodes

a. Open the **Programs and Features** control panel and remove **IBM Spectrum Scale Standard Edition 4.2.y**.
   1) Uninstall IBM Spectrum Scale 4.2.y and reboot.
   2) Uninstall IBM GPFS GSKit 8.0.x.x and reboot.
   3) Uninstall the IBM Spectrum Scale 4.2.y license.

b. Copy the installation images and install the IBM Spectrum Scale licensed program as described in Installing IBM Spectrum Scale on Windows nodes.

   Note: For Windows upgrades, it is required that all IBM Spectrum Scale administration commands are executed from the 5.0.x node.

6. Start IBM Spectrum Scale on the designated node in the cluster using the **`mmstartup`** command. For example: **`mmstartup -N k164n04`**, where `k164n04` is the designated node.
7. Mount the file systems if this is not done automatically when the IBM Spectrum Scale daemon starts.

Repeat the preceding steps on all nodes to upgrade GPFS on all nodes in the cluster.

Attention: If you are using components such as protocols, performance monitoring, management GUI, call home, and Transparent cloud tiering, these components must also be upgraded to the newer code level after base GPFS is upgraded. For information on upgrading these components, see step 8 on page 379.

8. On Linux distributions, if you are using protocols, performance monitoring, management GUI, call home, and Transparent cloud tiering, upgrade these components as follows.
   a. Upgrade Object packages on nodes running on supported Linux distributions. For information on upgrading Object packages, see "Upgrading object packages" on page 389.
   b. Upgrade NFS packages on nodes running on supported Linux distributions. For information on upgrading NFS packages, see "Upgrading NFS packages" on page 392.
   c. Upgrade SMB packages on nodes running on supported Linux distributions. For information on upgrading SMB packages, see "Upgrading SMB packages" on page 393.
   d. Upgrade performance monitoring packages. For information on upgrading performance monitoring packages, see "Manually upgrading the performance monitoring tool" on page 397 and "Manually upgrading pmswift" on page 395.
   e. Upgrade management GUI packages. For information on upgrading management GUI packages, see "Manually upgrading the IBM Spectrum Scale management GUI" on page 397.
   f. Upgrade call home packages.

      Important: Ensure that all scheduled tasks are disabled, and that no call home tasks are running. Then install the call home RPMs in the following order:
      1) gpfs.java-5.0.0-0.x86_64.rpm
      2) gpfs.callhome-ecc-client-5.0.0-0.000.noarch.rpm

      For more information, see "Upgrading call home configuration" on page 393.
   g. Upgrade Cloud services packages. For information on upgrading Cloud services packages, see "Upgrading Cloud services" on page 399.

   Note: When upgrading to IBM Spectrum Scale 5.0.x from IBM Spectrum Scale 4.2.y or later, you can also use the installation toolkit on supported Linux nodes. For more information, see "Upgrading IBM Spectrum Scale components with the installation toolkit" on page 407.

   If you were using only GPFS, after upgrading to the newer version you can install other available components depending on your requirements.
   - To optionally install protocols, see "Installing Cluster Export Services as part of installing IBM Spectrum Scale on Linux systems" on page 243.
   - To optionally install the performance monitoring tool, see "Manually installing the Performance Monitoring tool" on page 250.
   - To optionally install the IBM Spectrum Scale GUI, see "Manually installing IBM Spectrum Scale management GUI" on page 252.
   - To optionally install the call home feature, see *Monitoring the IBM Spectrum Scale system by using call home* in *IBM Spectrum Scale: Problem Determination Guide*.
   - To optionally install file audit logging, see Chapter 13, "Installing file audit logging," on page 361.
   - To optionally install the Transparent cloud tiering feature, see Chapter 7, "Installing Cloud services on IBM Spectrum Scale nodes," on page 337.
   - To optionally install the Scale Management API, see Chapter 9, "Installing and configuring IBM Spectrum Scale management API," on page 343.
   - To optionally install the CES BLOCK service, see *Configuring and enabling the BLOCK service* in *IBM Spectrum Scale: Administration Guide*.

When all nodes in the cluster have been successfully upgraded to the new IBM Spectrum Scale level, proceed to completing the upgrade. For more information, see "Completing the upgrade to a new level of IBM Spectrum Scale" on page 418.

Upgrading to IBM Spectrum Scale 4.2.y from IBM Spectrum Scale 4.1.x

IBM Spectrum Scale 4.2 release supports node-at-a-time upgrade if the previous nodes in the cluster are running IBM Spectrum Scale 4.1.x release. IBM Spectrum Scale 4.1.x nodes can coexist and interoperate with nodes that are running IBM Spectrum Scale 4.2. However, new functions that depend on format changes will not be available until all nodes have been upgraded.

Note: If you have protocols in use, to upgrade to the IBM Spectrum Scale 4.2.y release, you must have the IBM Spectrum Scale 4.2 release installed and running on the system. Before proceeding with the upgrade, see the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

To upgrade a cluster to IBM Spectrum Scale 4.2.y release from IBM Spectrum Scale 4.1.x release, perform the following steps:

1. Stop all user activity in the file systems on the designated node.

   For information about how to stop and unmount NFS running over IBM Spectrum Scale file systems, see *Unmounting a file system after NFS export* in *IBM Spectrum Scale: Administration Guide*.
2. Cleanly unmount the mounted IBM Spectrum Scale file system. Do not use force to unmount the designated node.
3. Follow any local administrative backup procedures to ensure protection of your file system data in the event of a failure.
4. Stop IBM Spectrum Scale on the node to be upgraded in the cluster using the **mmshutdown** command.

   For example: **mmshutdown -N k164n04**, where k164n04 is the designated node.
5. Upgrade to IBM Spectrum Scale 4.2 release depending on the operating system:

   For Linux nodes:

   - Extract the IBM Spectrum Scale software as described in the "Extracting the IBM Spectrum Scale software on Linux nodes" on page 231 topic.

     The installable images will be extracted to the following location: /usr/lpp/mmfs/4.2.*

     For SLES and Red Hat Enterprise Linux nodes:

     – For IBM All Spectrum Scale Editions: **rpm -Fvh /usr/lpp/mmfs/4.2.*/gpfs*.rpm**

     – For IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition:

       rpm -ivh gpfs.adv-4.2.*.rpm - This must be installed to use the Async disaster recovery feature.

       rpm -ivh gpfs.crypto-4.2.*.rpm

     – For IBM Spectrum Scale Standard Edition and IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition:

       To optionally install the call home feature, see *Monitoring the IBM Spectrum Scale system by using call home* in *IBM Spectrum Scale: Problem Determination Guide*.

       To optionally install the Performance Monitoring tool, see the "Manually installing the Performance Monitoring tool" on page 250 topic.

       To optionally install the IBM Spectrum Scale GUI, see the "Manually installing IBM Spectrum Scale management GUI" on page 252 topic.

       Note: When upgrading to the IBM Spectrum Scale 4.2 from IBM Spectrum Scale 4.1.1, RHEL7 users can refer to the "Upgrading IBM Spectrum Scale components with the installation toolkit" on page 407 topic for more information on deploying protocols using the installation toolkit.
   - For Debian and Ubuntu Linux users:

     – For IBM Spectrum Scale Express Edition, run the following command:

     ```
     dpkg -i gpfs.base*deb gpfs.gpl*deb gpfs.gskit*deb
     gpfs.msg*deb gpfs.doc*deb
     ```

     – For IBM Spectrum Scale Standard Edition, run the following command:

```
dpkg -i gpfs.base*deb gpfs.gpl*deb gpfs.gskit*deb
gpfs.msg*deb gpfs.doc*deb gpfs.ext*deb
```

  - For IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition, run the following command:

```
dpkg -i gpfs.base*deb gpfs.gpl*deb gpfs.gskit*deb gpfs.msg*deb
gpfs.doc*deb gpfs.ext*deb  gpfs.adv*deb gpfs.crypto*deb
```

  - To optionally install the Performance Monitoring tool on Debian 7, run the following command:

```
dpkg -i gpfs.gss.pmcollector_4.2.*.D7*deb
gpfs.gss.pmsensors_4.2.*D7*.deb
```

  - To optionally install the Performance Monitoring tool on Ubuntu 14, run the following command:

```
dpkg -i gpfs.gss.pmsensors_4.2.*.U14.*.deb
gpfs.gss.pmcollector_4.2.*.U14.*.deb
```

- After successful upgrade, you must rebuild the GPL layer. For more information, see the “Building the GPFS portability layer on Linux nodes” on page 237 topic.

For Hadoop users: To upgrade the IBM Spectrum Scale Hadoop Connector, see *Upgrading IBM Spectrum Scale connector* in *IBM Spectrum Scale: Administration Guide*.

For AIX nodes:

Note: Specific to AIX, when upgrading from 4.1.x to 4.2.y, you must upgrade to 4.2.0.0 first and then you can upgrade to 4.2.y.

- Copy the installation images and install the IBM Spectrum Scale licensed programs as described in the Chapter 5, “Installing IBM Spectrum Scale on AIX nodes,” on page 319 section.

  Note: If you use SMIT to upgrade IBM Spectrum Scale on AIX, IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition users must install `gpfs.adv`.

For Windows nodes

a. Open the **Programs and Features** control panel and remove **IBM Spectrum Scale Express Edition 4.1.x** or **IBM Spectrum Scale Standard Edition 4.1.x**.
   1) Uninstall IBM Spectrum Scale 4.1.1 and reboot.
   2) Uninstall IBM GPFS GSKit 8.0.x.x and reboot.
   3) Uninstall the IBM Spectrum Scale 4.1.1 license.

b. Copy the installation images and install the IBM Spectrum Scale licensed program as described in Installing IBM Spectrum Scale on Windows nodes.

   Note: For Windows upgrades, it is required that all IBM Spectrum Scale administration commands are executed from the 4.2 node.

6. Start IBM Spectrum Scale on the designated node in the cluster using the **`mmstartup`** command. For example: **`mmstartup -N k164n04`**, where `k164n04` is the designated node.
7. Mount the file systems if this is not done automatically when the IBM Spectrum Scale daemon starts.

When all nodes in the cluster have been successfully upgraded to the new IBM Spectrum Scale level, proceed to the “Completing the upgrade to a new level of IBM Spectrum Scale” on page 418 topic.

Upgrading to IBM Spectrum Scale 4.1.1.x from GPFS V4.1.0.x

Use the following information to upgrade from V4.1.0.x to V4.1.1.x.

For Linux:

1. Prior to upgrading GPFS on a node, all applications that depend on GPFS (For example, DB2) must be stopped. Any GPFS file systems that are NFS exported must be unexported prior to unmounting GPFS file systems.

2. Stop GPFS on the node. Verify that the GPFS daemon has terminated and that the kernel extensions have been unloaded (**mmfsenv -u**). If the command **mmfsenv -u** reports that it cannot unload the kernel extensions because they are "busy", then the install can proceed, but the node must be rebooted after the install. By "busy" this means that some process has a "current directory" in some GPFS file system directory or has an open file descriptor. The freeware program lsof can identify the process and the process can then be killed. Retry **mmfsenv -u** and if that succeeds then a reboot of the node can be avoided.
3. Upgrade GPFS using the RPM command as follows (make sure you are in the same directory as the files) :

   For SLES or RHEL systems

   ```
   rpm -Fvh gpfs*.rpm
   ```

   For Debian systems

   ```
   dpkg -i gpfs*.deb
   ```

4. You can verify the installation of the GPFS SLES or RHEL Linux RPMs on each node. To check that the software has been successfully installed, use the **rpm** command:

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

   The system should return output similar to the following:

   ```
   gpfs.gpl-4.1.1-1
   gpfs.docs-4.1.1-1
   gpfs.msg.en_US-4.1.1-1
   gpfs.base-4.1.1-1
   gpfs.gskit-8.0.50-40
   ```

   If you have the GPFS Standard Edition or the GPFS Advanced Edition installed, you should also see the following line in the output:

   ```
   gpfs.ext-4.1.1-1
   ```

   If you have the GPFS Advanced Edition installed, you should also see the following line in the output:

   ```
   gpfs.crypto-4.1.1-1
   ```

5. You can verify the installation of the GPFS Debian Linux packages on each node. To check that the software has been successfully installed, use the **dpkg** command:

   ```
   dpkg -l | grep gpfs
   ```

   The system should return output similar to the following:

   ```
   ii gpfs.base 4.1.1-1     GPFS File Manager
   ii gpfs.docs 4.1.1-1      GPFS Server Manpages and Documentation
   ii gpfs.gpl 4.1.1-1         GPFS Open Source Modules
   ii gpfs.gskit 8.0.50-40   GPFS GSKit Cryptography Runtime
   ii gpfs.msg.en-us 4.1.1-1 GPFS Server Messages - U.S. English
   ```

   If you have the GPFS Standard Edition or the GPFS Advanced Edition installed, you should also see the following line in the output:

   ```
   ii gpfs.ext 4.1.1-1      GPFS Extended Features
   ```

   If you have the GPFS Advanced Edition installed, you should also see the following line in the output:

   ```
   ii gpfs.crypto 4.1.1-1    GPFS Cryptographic Subsystem
   ```

6. Recompile any GPFS portability layer modules you may have previously compiled using the **mmbuildgpl** command.

For AIX:

1. Prior to upgrading GPFS on a node, all applications that depend on GPFS (For example, DB2) must be stopped. Any GPFS file systems that are NFS exported must be unexported prior to unmounting GPFS file systems.
2. To upgrade directly from 4.1.0.x to 4.1.1.x, you must put all Unnnnnn.gpfs*.bff images from 4.1.1.0 and 4.1.1.x in the same directory prior to running the **installp** command or SMIT.
3. Stop GPFS on the node. Verify that the GPFS daemon has terminated and that the kernel extensions have been unloaded (**mmfsenv -u**). If the command **mmfsenv -u** reports that it cannot unload the kernel extensions because they are "busy", then the install can proceed, but the node must be rebooted after the install. By "busy" this means that some process has a "current directory" in some GPFS filesystem directory or has an open file descriptor. The freeware program **lsof** can identify the process and the process can then be killed. Retry **mmfsenv -u** and if that succeeds then a reboot of the node can be avoided.
4. Upgrade GPFS using the **installp** command or via SMIT on the node. If you are in the same directory as the install packages, an example command might be:

   ```
   installp -agXYd . gpfs
   ```
5. You can verify that the installation procedure placed the required GPFS files on each node by running the **lslpp** command

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

   The system should return output similar to the following:

   ```
   Fileset                 Level      State       Description
   ----------------------------------------------------------------------------
   Path: /usr/lib/objrepos
    gpfs.base             4.1.1-1 COMMITTED GPFS File Manager
    gpfs.crypto           4.1.1-1 COMMITTED GPFS Cryptographic Subsystem
    gpfs.ext              4.1.1-1 COMMITTED GPFS Extended Features
    gpfs.gskit           8.0.50.40 COMMITTED GPFS GSKit Cryptography Runtime
    gpfs.msg.en_US        4.1.1-1 COMMITTED GPFS Server Messages - U.S. English
   Path: /etc/objrepos
    gpfs.base             4.1.1-1 COMMITTED GPFS File Manager
   Path: /usr/share/lib/objrepos
    gpfs.docs.data        4.1.1-1 COMMITTED GPFS Server Manpages and
   Documentation
   ```

   The output that is returned on your system can vary depending on if you have the GPFS Express Edition, GPFS Standard Edition, or GPFS Advanced Edition installed.

For Windows:

1. Downloaded the GPFS 4.1.1-x update package into any directory on your system.

   Note: This update requires a prior level of GPFS version 4.1.x on your system. In such a case, you need to simply uninstall the previous 4.1.x level and proceed with installing this upgrade. Upgrading from a prior 4.1 level does not require installing the GPFS license package again. However if you are upgrading directly from version 3.5 or prior (any level), or installing version 4.1 for the first time on your system, you must first install the GPFS license package (gpfs.base-4.1-Windows-license.msi) before installing this update.
2. Extract the contents of the ZIP archive so that the .msi file it includes is directly accessible to your system.
3. Uninstall the system's current version of GPFS using the Programs and Features control panel. If prompted to reboot the system, do this before installing the update package.

Upgrading from GPFS 3.5

If you are upgrading from GPFS 3.5, you can choose between the following upgrade options.

- "Online upgrade, most functionality remains available"
- "Offline upgrade, complete cluster shutdown" on page 387

Note:

- Given the age of GPFS 3.5, you might need to upgrade your operating system to a version supported by the latest release of IBM Spectrum Scale (GPFS). Operating system upgrades must be done before upgrading IBM Spectrum Scale (GPFS) on each node. For more information, see "Operating system upgrades while upgrading from GPFS 3.5."

  For the list of supported operating system levels for respective IBM Spectrum Scale versions, see IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).
- For Linux systems that are on IBM Spectrum Scale (GPFS) version 4.1 or later, when the operating system is updated and the new GPFS RPMs are installed, you need to use the **mmbuildgpl** command to rebuild the GPFS kernel modules. If you need to rebuild the GPFS portability after upgrading operating system on GPFS 3.5 systems, see Building the GPFS portability nodes.

Operating system upgrades while upgrading from GPFS 3.5

If you want to upgrade the operating system while upgrading from GPFS 3.5, use the following steps.

1. Disable the GPFS autoload setting that causes GPFS to be started whenever the operating system reboots by issuing the following command.

   ```
   mmchconfig autoload=no -N NodeName
   ```

   NodeName is the name of the node on which the operating system upgrade will be done.
2. Unmount GPFS file systems on the node by issuing the following command.

   ```
   mmumount all -N NodeName
   ```
3. Stop GPFS on the node by issuing the following command.

   ```
   mmshutdown -N NodeName
   ```
4. Upgrade the operating system according to the instructions supplied by the operating system vendor.
5. After the operating system is upgraded and any required reboots are complete, follow the steps for upgrading GPFS.
6. Re-enable the GPFS autoload setting by issuing the following command.

   ```
   mmchconfig autoload=yes -N NodeName
   ```

Online upgrade, most functionality remains available

Upgrading to the 5.0.z version of IBM Spectrum Scale (GPFS) from version 3.5 requires three separate online upgrade procedures.

1. The first procedure is to upgrade to IBM Spectrum Scale (GPFS) 4.1.1.x.
2. The second procedure is to upgrade to IBM Spectrum Scale (GPFS) 4.2.y from 4.1.1.x.
3. The third procedure is to upgrade to IBM Spectrum Scale (GPFS) 5.0.z from 4.2.y.

The upgrade can be done directly to IBM Spectrum Scale (GPFS) 5.0.z only if the entire GPFS cluster is shut down and all nodes are fully upgraded before restarting the cluster.

For each node in the cluster, do the following steps.

1. Perform any necessary operating system upgrades.
2. If you did not need to do operating system upgrade, unmount GPFS file systems and shut down GPFS.

3. Extract the IBM Spectrum Scale installation image so that all installation files (rpm, deb, bff, msi, and so on) are available.
4. Run the necessary platform-specific commands to upgrade GPFS.
   - SLES and RHEL:

     ```
     rpm -Fvh /usr/lpp/mmfs/4.1.1/gpfs_rpms/{gpfs.base.*.rpm,gpfs.crypto.*.rpm,gpfs.ext.*.rpm,\
     gpfs.docs.*.rpm,gpfs.gpl.*.rpm,gpfs.gskit.*.rpm,gpfs.msg.*.rpm}

     rpm -Fvh /usr/lpp/mmfs/4.1.1/zimon_rpms/{gpfs.gss.pmsensors*.rpm,gpfs.gss.pmcollector*.rpm}
     ```

     Note: The `gpfs-crypto` package is available only with IBM Spectrum Scale Advanced Edition.
   - Debian Linux or Ubuntu:

     ```
     dpkg -i /usr/lpp/mmfs/4.1.1/{gpfs.base.*.deb,gpfs.crypto.*.deb,gpfs.ext.*.deb,gpfs.docs.*.deb,\
     gpfs.gpl.*.deb,gpfs.gskit.*.deb,gpfs.msg.*.deb}
     ```
   - AIX:

     ```
     installp -agXYd . gpfs
     ```
   - Windows:
     a. Uninstall GPFS 3.5 and reboot.
     b. Uninstall the GPFS 3.5 license.
     c. Uninstall Open Secure Shell for GPFS 3.5 on Windows, if applicable.
     d. Disable any SUA daemon, such as OpenSSH, that might have been configured. Do *not* uninstall SUA yet, or you may lose GPFS configuration information.
     e. Install the 64-bit version of Cygwin. For more information, see "Installing Cygwin" on page 329.
     f. Install `gpfs.ext-4.1.1.x-Windows-license.msi`.
     g. Install `gpfs.ext-4.1.1.x-Windows.msi`.
     h. Install IBM® GSKit for GPFS.
     i. Uninstall SUA completely.
     j. Configure passwordless SSH in Cygwin for mixed (Linux, AIX, Windows) clusters or use **`mmwinservctl`** on a Windows only cluster.
5. On Linux systems, rebuild the GPFS portability layer (GPL) with the **`mmbuildgpl`** command. For more information, see "Building the GPFS portability layer on Linux nodes" on page 237.
6. Restart GPFS by issuing the **`mmstartup`** command and remount file systems by issuing the **`mmmount`** command, if necessary.

   Repeat the preceding steps on each node in the cluster one by one after restarting GPFS.

   After all the nodes in the cluster are upgraded to 4.1.1.x, repeat this sequence of steps (1 to 6) on all nodes in the cluster one by one for upgrading from **4.1.1.x to 4.2.y**. For this upgrade path, use the following platform-specific commands instead of those listed in Step 4:
   - SLES and RHEL:

     ```
     rpm -Fvh /usr/lpp/mmfs/4.2.y.z/gpfs_rpms/<OS>/{gpfs.adv.*.rpm,gpfs.base.*.rpm,gpfs.crypto.*.rpm,\
     gpfs.ext.*.rpm,gpfs.docs.*.rpm,gpfs.gpl.*.rpm,gpfs.gskit.*.rpm,gpfs.msg.*.rpm,\
     gpfs.callhome*.rpm,gpfs.license.XX*.rpm}

     rpm -Fvh /usr/lpp/mmfs/4.2.y.z/zimon_rpms/<OS>{gpfs.gss.pmsensors*.rpm,gpfs.gss.pmcollector*.rpm}
     ```

     Where *<OS>* is the operating system specific directory such as `rhel7` or `sles12`.

     Note:
     - The `gpfs-adv` package is available only with IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition.
     - Depending on the IBM Spectrum Scale edition that you are installing, install the appropriate `gpfs.license` package.
   - Debian Linux or Ubuntu:

```
dpkg -i /usr/lpp/mmfs/4.2.y.z/gpfs_debs/<OS>/{gpfs.adv.*.deb,gpfs.base.*.deb,gpfs.crypto.*.deb,\
gpfs.ext.*.deb,gpfs.docs.*.deb,gpfs.gpl.*.deb,gpfs.gskit.*.deb,gpfs.msg.*.deb,\
gpfs.callhome*.deb,gpfs.license.XX*.deb}

dpkg -i /usr/lpp/mmfs/4.2.y.z/zimon_debs/<OS>{gpfs.gss.pmsensors*.deb,gpfs.gss.pmcollector*.deb}
```

Where <*OS*> is the operating system specific directory such as `ubuntu16`.

Note: Depending on the IBM Spectrum Scale edition that you are installing, install the appropriate `gpfs.license` package.

- AIX:

  ```
  installp -agXYd . gpfs
  ```

- Windows:
  a. Uninstall GPFS 4.1 and reboot.
  b. Uninstall the GPFS 4.1 license.
  c. Uninstall IBM GSKit for GPFS.
  d. Install the 64-bit version of Cygwin. For more information, see "Installing Cygwin" on page 329.
  e. Install `gpfs.ext-4.2.x.x-Windows-license.msi`.
  f. Install `gpfs.ext-4.2.x.x-Windows.msi`.
  g. Install IBM® GSKit for GPFS.
  h. Configure passwordless SSH in Cygwin for mixed (Linux, AIX, Windows) clusters or use **mmwinservctl** on a Windows only cluster.

Repeat the preceding steps on each node in the cluster one by one after restarting GPFS.

After all the nodes in the cluster are upgraded to 4.2.y, repeat this sequence of steps (1 to 6) on all nodes in the cluster one by one for upgrading from **4.2.y to 5.0.z**. For this upgrade path, use the following platform-specific commands instead of those listed in Step 4:

- SLES and RHEL:

  ```
  rpm -Fvh /usr/lpp/mmfs/5.0.y.z/gpfs_rpms/<OS>{gpfs.adv.*.rpm,gpfs.base.*.rpm,gpfs.crypto.*.rpm,\
  gpfs.ext.*.rpm,gpfs.docs.*.rpm,gpfs.gpl.*.rpm,gpfs.gskit.*.rpm,gpfs.msg.*.rpm,gpfs.callhome*.rpm,\
  gpfs.license.XX*.rpm,gpfs.compression*.rpm}

  rpm -Fvh /usr/lpp/mmfs/5.0.y.z/zimon_rpms/<OS>{gpfs.gss.pmsensors*.rpm,gpfs.gss.pmcollector*.rpm}
  ```

  Where <*OS*> is the operating system specific directory such as `rhel7` or `sles12`.

  Note:
  – The `gpfs-adv` package is available only with IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition.
  – Depending on the IBM Spectrum Scale edition that you are installing, install the appropriate `gpfs.license` package.

- Ubuntu:

  ```
  dpkg -i /usr/lpp/mmfs/5.0.y.z/gpfs_debs/<OS>{gpfs.adv.*.deb,gpfs.base.*.deb,gpfs.crypto.*.deb,\
  gpfs.ext.*.deb,gpfs.docs.*.deb,gpfs.gpl.*.deb,gpfs.gskit.*.deb,gpfs.msg.*.deb,gpfs.callhome*.deb,\
  gpfs.license.XX*.deb,gpfs.compression*.deb}

  dpkg -i /usr/lpp/mmfs/5.0.y.z/zimon_debs/<OS>{gpfs.gss.pmsensors*.deb,gpfs.gss.pmcollector*.deb}
  ```

  Where <*OS*> is the operating system specific directory such as `ubuntu16`.

  Note: Depending on the IBM Spectrum Scale edition that you are installing, install the appropriate `gpfs.license` package.

- AIX:

  ```
  installp -agXYd . gpfs
  ```

- Windows:
    a. Uninstall GPFS 4.2.y and reboot.
    b. Uninstall the GPFS 4.2.y license.
    c. Uninstall IBM GSKit for GPFS.
    d. Install the 64-bit version of Cygwin. For more information, see “Installing Cygwin” on page 329.
    e. Install `gpfs.ext-5.0.x.x-Windows-license.msi`.
    f. Install `gpfs.ext-5.0.x.x-Windows.msi`.
    g. Install IBM® GSKit for GPFS.
    h. Configure passwordless SSH in Cygwin for mixed (Linux, AIX, Windows) clusters or use **`mmwinservctl`** on a Windows only cluster.

   Note: At this point, the code is at the upgraded level, but the cluster is not upgraded and new functions cannot be configured until the upgrade is completed.

7. Complete the upgrade as documented in “Completing the upgrade to a new level of IBM Spectrum Scale” on page 418.

The 5.0.z code level is now applied on the whole cluster and all file systems. For GPFS-only configurations, the process is now complete.

Offline upgrade, complete cluster shutdown

This scenario is applicable if you can shut down the entire cluster and perform the upgrade. The steps in this procedure are similar to the steps in the online upgrade procedure. However, due to the entire cluster being offline, it is possible to jump straight to the latest code level instead of hopping to an intermediate level, as required during an online upgrade.

1. Perform any necessary operating system upgrades. Confirm that the upgraded operating system level is one that is supported by the version of IBM Spectrum Scale to which the system is being upgraded.
2. If you did not need to do operating system upgrade, unmount GPFS file systems and shut down GPFS.
3. Extract the IBM Spectrum Scale installation image so that all installation files (rpm, deb, bff, msi, and so on) are available.
4. Run the necessary platform-specific commands to upgrade GPFS.
    - SLES and RHEL Linux:

      ```
      rpm -Fvh /usr/lpp/mmfs/5.0.y.z/gpfs_rpms/<OS>{gpfs.adv.*.rpm,gpfs.base.*.rpm,gpfs.crypto.*.rpm,\
      gpfs.ext.*.rpm,gpfs.docs.*.rpm,gpfs.gpl.*.rpm,gpfs.gskit.*.rpm,gpfs.msg.*.rpm,gpfs.callhome*.rpm,\
      gpfs.license.XX*.rpm,gpfs.compression*.rpm}
      rpm -Fvh /usr/lpp/mmfs/5.0.y.z/zimon_rpms/<OS>{gpfs.gss.pmsensors*.rpm,gpfs.gss.pmcollector*.rpm}
      ```

      Where *<OS>* is the operating system specific directory such as `rhel7` or `sles12`.

      Note:
      - The `gpfs-crypto` and the `gpfs-adv` packages are available only with IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition.
      - Depending on the IBM Spectrum Scale edition that you are installing, install the appropriate `gpfs.license` package.
    - Ubuntu:

      ```
      dpkg -i /usr/lpp/mmfs/5.0.y.z/gpfs_debs/<OS>{gpfs.adv.*.deb,gpfs.base.*.deb,gpfs.crypto.*.deb,\
      gpfs.ext.*.deb,gpfs.docs.*.deb,gpfs.gpl.*.deb,gpfs.gskit.*.deb,gpfs.msg.*.deb,gpfs.callhome*.deb,\
      gpfs.license.XX*.deb,gpfs.compression*.deb}

      dpkg -i /usr/lpp/mmfs/5.0.y.z/zimon_debs/<OS>{gpfs.gss.pmsensors*.deb,gpfs.gss.pmcollector*.deb}
      ```

      Where *<OS>* is the operating system specific directory such as `ubuntu16`.

Note: Depending on the IBM Spectrum Scale edition that you are installing, install the appropriate `gpfs.license` package.

- AIX:

  ```
  installp -agXYd . gpfs
  ```

- Windows:
  a. Uninstall GPFS 3.5 and reboot.
  b. Uninstall the GPFS 3.5 license.
  c. Uninstall Open Secure Shell for GPFS 3.5 on Windows, if applicable.
  d. Disable any SUA daemon, such as OpenSSH, that might have been configured. Do *not* uninstall SUA yet, or you may lose GPFS configuration information.
  e. Install the 64-bit version of Cygwin. For more information, see "Installing Cygwin" on page 329.
  f. Install `gpfs.ext-5.0.x.x-Windows-license.msi`.
  g. Install `gpfs.ext-5.0.x.x-Windows.msi`.
  h. Install IBM® GSKit for GPFS.
  i. Uninstall SUA completely.
  j. Configure passwordless SSH in Cygwin for mixed (Linux, AIX, Windows) clusters or use **`mmwinservctl`** on a Windows only cluster.

5. On Linux systems, rebuild the GPFS portability layer (GPL) with the **`mmbuildgpl`** command. For more information, see "Building the GPFS portability layer on Linux nodes" on page 237.
6. Restart GPFS by issuing the **`mmstartup`** command and remount file systems by issuing the **`mmmount`** command, as necessary.

   At this point all code is at the upgraded level, but the cluster is not upgraded and new functions cannot be configured until the upgrade is completed.

   Note: At this point, all code is at the upgraded level, but the cluster is not upgraded and new functions cannot be configured until the upgrade is completed.
7. Complete the upgrade as documented in "Completing the upgrade to a new level of IBM Spectrum Scale" on page 418.

The 5.0.z code level is now applied on the whole cluster and all file systems. For GPFS-only configurations, the process is now complete.

At this point, you can use the installation toolkit for installation or deployment of new functions and for future upgrades. Alternatively, you can perform function additions and upgrades manually.

Attention: Regardless of which method you are using, if you want to install and configure CES protocols after upgrading to the latest level, you must create a CES shared root file system of size 4 GB or greater. This file system is used for CES protocol state data and it must remain mounted and up on all protocol nodes for the protocols to function. You can also create this file system from multiple NSDs, each of which are replicated to help with redundancy. For more information, see *Setting up Cluster Export Services shared root file system* in *IBM Spectrum Scale: Administration Guide*.

- If you want to add new functions manually, see "Manually installing the IBM Spectrum Scale software packages on Linux nodes" on page 233.
- If you want to add new functions using the installation toolkit, see "Installing IBM Spectrum Scale on Linux nodes with the installation toolkit" on page 258.

Online upgrade support for protocols and performance monitoring

Use this information to understand the online upgrade support for protocols and performance monitoring in IBM Spectrum Scale.

Online upgrade support for protocol nodes depends on which services have been enabled and deployed. Different protocol services have different levels of online upgrade support.

- NFS: Having different versions of NFS on protocol nodes in the cluster during the course of a node-at-a-time online upgrade can be tolerated.
- SMB: All protocol nodes running the SMB service must have the same version installed at any time. This requires a brief outage of the SMB service to upgrade SMB to the newer version across all protocol nodes. SMB update in an IBM Spectrum Scale cluster is done in two phases. The protocol nodes in the cluster are divided in two halves and SMB is updated on each half one by one.
- Object: Because object operations interact with all nodes, all object nodes must be at the same level. Therefore, object services need to be stopped across the cluster for the duration of the upgrade. Other protocol services can continue to be run.
- Performance monitoring: Having two different versions on collector nodes will result in a loss of communication and could potentially result in missing performance metrics until collector versions match.

Upgrading object packages

Use these steps to upgrade IBM Spectrum Scale for object storage packages.

Upgrading object packages to 5.0.x from 4.2.x

Use these steps to upgrade IBM Spectrum Scale for object storage packages from 4.2.x to 5.0.x.

Attention:

- All protocols must also be upgraded to the newer code level along with base GPFS.
- The object storage packages on all protocol nodes must be upgraded to the same level at the same time.
- Online upgrade of the object protocol is not supported due to changes that make it incompatible with servers running on the previous level. The upgrade must be scheduled at a time when downtime is appropriate.

1. On all protocol nodes, upgrade `gpfs.ext` and `spectrum-scale-object` to the 5.0.x version.
2. Select one of the protocol nodes to use for upgrade processing. This node must have GPFS services started and the **mmgetstate** command results must show that the node is active.
3. From the selected node, issue the following command to stop Object protocol related services on all protocol nodes.
   ```
   mmces service stop obj -a
   ```
4. From the selected protocol node, issue the following command to migrate the existing Object protocol configuration to be consistent with the level of installed packages.
   ```
   mmobj config manage --version-sync
   ```
5. Issue the following command to start Object protocol related services on all protocol nodes.
   ```
   mmces service start obj -a
   ```

Upgrading object packages to version 4.2.3.x from 4.2.2.x

Use these steps to upgrade IBM Spectrum Scale for object storage packages from version 4.2.2.x to 4.2.3.x.

Attention:

- All protocols must also be upgraded to the newer code level along with base GPFS.
- Rolling upgrade of the Object protocol is not supported due to changes in the Keystone database schema that make it incompatible with servers running on the previous level. The upgrade must be scheduled at a time when downtime is appropriate.

1. Issue the following command to stop Object protocol related services on all protocol nodes.

```
mmces service stop obj -a
```

2. On all protocol nodes, issue the following command to update the IBM Spectrum Scale for Object Storage packages.

   ```
   yum upgrade 'spectrum-scale-object*'
   ```

   Note: You must set up the Yum repository before using the **yum upgrade** command to upgrade IBM Spectrum Scale packages. For more information, see *Repository setup* in "Installation prerequisites" on page 224.

3. On the node on which the object database runs, issue the following command to start the `postgresql-obj` service.

   ```
   systemctl start postgresql-obj
   ```

   Note: The node on which the object database runs has the attribute `object_database_node`.

4. On the node on which the object database runs, issue the following command to upgrade the database schema.

   ```
   keystone-manage db_sync
   ```

5. On the node on which the object database runs, issue the following command to stop the `postgresql-obj` service.

   ```
   systemctl stop postgresql-obj
   ```

6. On the node on which the object database runs, make changes in the listed sections of the `/etc/keystone/keystone-paste.ini` file as follows.

   Update the [pipeline:admin_api] section

   - Remove the following middleware:

     ```
     crud_extension
     ```

   - Add the following middleware at the beginning of the pipeline:

     ```
     cors
     ```

   - Rearrange the middleware sequence:

     Old: `build_auth_context token_auth admin_token_auth`

     New: `admin_token_auth build_auth_context token_auth`

   The updated pipeline is as follows:

   ```
   pipeline = cors sizelimit url_normalize request_id admin_token_auth
   build_auth_context token_auth json_body ec2_extension s3_extension admin_service
   ```

   Note: The pipeline cannot be replaced completely because there might be other middleware added.

   Update the [pipeline:api_v3] section

   - Remove the following middleware:

     ```
     simple_cert_extension
     revoke_extension
     federation_extension
     oauth1_extension
     endpoint_filter_extension
     ```

   - Add the following middleware at the beginning of the pipeline:

     ```
     cors
     ```

   - Rearrange the middleware sequence:

     Old: `build_auth_context token_auth admin_token_auth`

     New: `admin_token_auth build_auth_context token_auth`

   The updated pipeline is as follows:

   ```
   pipeline = cors sizelimit url_normalize request_id admin_token_auth
   build_auth_context token_auth json_body ec2_extension_v3 s3_extension service_v3
   ```

Note: The pipeline cannot be replaced completely because there might be other middleware added.

Update the `[pipeline:public_api]` section

- Remove the following middleware:

  ```
  user_crud_extension
  ```

- Rearrange the middleware sequence:

  Old: `build_auth_context token_auth admin_token_auth`

  New: `admin_token_auth build_auth_context token_auth`

- Add the following middleware at the beginning of the pipeline:

  ```
  cors
  ```

The updated pipeline is as follows:

```
cors sizelimit url_normalize request_id admin_token_auth build_auth_context
token_auth json_body ec2_extension public_service
```

Note: The pipeline cannot be replaced completely because there might be other middleware added.

Add the `[filter:cors]` section with the following contents

```
use = egg:oslo.middleware#cors
oslo_config_project = keystone
```

Ensure that the `[filter:request_id]` section is as follows

```
[filter:request_id]
use = egg:oslo.middleware#request_id
```

Ensure that the `[filter:s3_extension]` section is as follows

```
[filter:s3_extension]
use = egg:keystone#s3_extension
```

7. On the node on which the object database runs, issue the following command to save the modified `/etc/keystone/keystone-paste.ini` file to CCR.

   ```
   mmccr fput keystone-paste.ini /etc/keystone/keystone-paste.ini
   ```

8. Issue the following command to start the Object protocol related services on all protocol nodes.

   ```
   mmces service start obj -a
   ```

Upgrading object packages to version 4.2.2.x from 4.2.1.x

Use these steps to upgrade IBM Spectrum Scale for object storage packages from version 4.2.1.x to 4.2.2.x.

Attention: All protocols must also be upgraded to the newer code level along with base GPFS.

1. Do these steps on all protocol nodes one by one.
   a. Issue the following command to suspend the node.

      `mmces node suspend -N` *`NodeName`*

   b. Issue the following command to stop Object protocol related services.

      ```
      mmces service stop obj
      ```

   c. Issue the following command to update the IBM Spectrum Scale for object storage RPM.

      ```
      yum upgrade spectrum-scale-object*.rpm
      ```

      Note: You must set up the Yum repository before using the **`yum upgrade`** command to upgrade IBM Spectrum Scale packages. For more information, see *Repository setup* in “Installation prerequisites” on page 224.

2. On the node on which the object database is running, issue the following command to start the `postgresql-obj` service.

   ```
   systemctl start postgresql-obj
   ```

3. On the node on which the object database is running, issue the following command to upgrade the database schema.

   ```
   keystone-manage db_sync
   ```

4. On the node on which the object database is running, issue the following command to stop the `postgresql-obj` service.

   ```
   systemctl stop postgresql-obj
   ```

5. On the node on which the object database is running, issue the following command to update CCR with object configuration files.

   ```
   mmccr fput account-server.conf /etc/swift/account-server.conf
   mmccr fput container-reconciler.conf /etc/swift/container-reconciler.conf
   mmccr fput container-server.conf /etc/swift/container-server.conf
   mmccr fput object-expirer.conf /etc/swift/object-expirer.conf
   mmccr fput object-server.conf /etc/swift/object-server.conf
   mmccr fput proxy-server.conf /etc/swift/proxy-server.conf
   mmccr fput swift.conf /etc/swift/swift.conf
   mmccr fput keystone.conf /etc/keystone/keystone.conf
   mmccr fput keystone-paste.ini /etc/keystone/keystone-paste.ini
   mmccr fput logging.conf /etc/keystone/logging.conf
   ```

6. Issue the following command to resume the nodes.

   ```
   mmces node resume -N ProtocolNodeList
   ```

   ProtocolNodeList is a comma-separated list of protocol nodes.

7. Issue the following command to start the object services on all protocol nodes.

   ```
   mmces service start obj –N ProtocolNodeList
   ```

   ProtocolNodeList is a comma-separated list of protocol nodes.

Upgrading NFS packages

Use these steps on each protocol node one by one to upgrade IBM Spectrum Scale NFS packages.

Attention:

- All protocols must also be upgraded to the newer code level along with base GPFS.
- After you upgrade from an earlier version to the current version, it might be necessary to unmount and mount the file system on the NFS clients to ensure continued access.

1. Issue the following command to suspend the node.

   ```
   mmces node suspend -N Node
   ```

2. Issue the following command to stop NFS services on the node.

   ```
   mmces service stop nfs -N Node
   ```

3. Issue the following command to upgrade NFS packages on the node.

   ```
   rpm -Uvh nfs-ganesha*.rpm
   ```

4. Issue the following command to start NFS services on the node.

   ```
   mmces service start nfs -N Node
   ```

5. Issue the following command to resume the node.

   ```
   mmces node resume -N Node
   ```

For information on migrating from CNFS to CES NFS, see *Migration of CNFS clusters to CES clusters* in *IBM Spectrum Scale: Administration Guide*.

Upgrading SMB packages

Use these steps to upgrade IBM Spectrum Scale SMB packages.

Attention: All protocols must also be upgraded to the newer code level along with base GPFS.

SMB package update in an IBM Spectrum Scale cluster is done in two phases. The protocol nodes in the cluster are divided in two halves and SMB is updated on each half one by one. In the following steps, *NodeList1* is the comma-separated list of nodes in the first half of the cluster and *NodeList2* is the comma-separated list of nodes in the second half of the cluster.

Note: All protocol nodes running the SMB service must have the same version of `gpfs.smb` installed at any time. This requires a brief outage of the SMB service to upgrade `gpfs.smb` to the newer version across all protocol nodes. The procedure outlined here is intended to reduce the outage to a minimum.

1. On the first half of the protocol nodes, do the following steps.
   a. Issue the following command to suspend the nodes.
      ```
      mmces node suspend -N NodeList1
      ```
   b. Issue the following command to stop CTDB and SMB daemons on the nodes.
      ```
      mmces service stop smb -N NodeList1
      ```
   c. Issue the following command to upgrade the SMB package on each node in the first half of protocol nodes.
      ```
      rpm -Uvh gpfs.smb*.rpm
      ```
2. On the second half of the protocol nodes, do the following steps.
   a. Issue the following command to suspend the nodes.
      ```
      mmces node suspend -N NodeList2
      ```
   b. Issue the following command to stop CTDB and SMB daemons on the nodes.
      ```
      mmces service stop smb -N NodeList2
      ```
3. On the first half of the protocol nodes, do the following steps.
   a. Issue the following command to start CTDB and SMB daemons on the nodes.
      ```
      mmces service start smb -N NodeList1
      ```
   b. Issue the following command to resume the nodes.
      ```
      mmces node resume -N NodeList1
      ```
4. On the second half of the protocol nodes, do the following steps.
   a. Issue the following command to upgrade the SMB package on each node in the second half of protocol nodes.
      ```
      rpm -Uvh gpfs.smb*.rpm
      ```
   b. Issue the following command to start CTDB and SMB daemons on the nodes.
      ```
      mmces service start smb -N NodeList2
      ```
   c. Issue the following command to resume the nodes.
      ```
      mmces node resume -N NodeList2
      ```

Upgrading call home configuration

Use these steps to upgrade call home configuration for IBM Spectrum Scale .

Call home configuration changes to be made while upgrading to IBM Spectrum Scale 5.0.x from IBM Spectrum Scale 4.2.1

When you upgrade from IBM Spectrum Scale 4.2.1. to IBM Spectrum Scale 5.0.0, the existing group configuration will be incompatible with the new format.

Follow one of the given methods to deploy the existing group configuration after an upgrade:

- If the existing call home groups do not need to be saved, you can use the `mmcallhome group auto --force` command to automatically delete all the current groups, and create new groups with the necessary configuration.
- If you want to save the existing call home group configuration, follow the given steps:
  1. Find the current call home group configuration using the **`mmcallhome group list`** command.
  2. Delete all listed call home groups using the **`mmcallhome group delete <groupName>`** command.
  3. Add all the call home groups that were listed in step 1, using the following command:

     ```
     mmcallhome group add GroupName server [--node {all | ChildNode[,ChildNode...]}]
     ```
  4. Set the new groups to track the global settings.
     1. Run the **`mmcallhome capability list`**, **`mmcallhome proxy list`**, or **`mmcallhome schedule list`** commands to view the current global settings and default settings for the new groups.

        The system displays output similar to this:

        ```
        [root@g5001-21 ~]# mmcallhome capability list
        group    callHomeNode           status
        -------- ---------------------- ----------
        global    ---                   enabled
        g1       g5001-21.localnet.com  disabled
        g2       g5001-22.localnet.com  disabled
        ```
     2. For each group, execute the **`mmcallhome capability`**, **`mmcallhome proxy`**, the **`mmcallhome schedule`** commands to configure the system with the existing settings available in the global field.

        After each change for each group, the corresponding group entry will disappear from the list command output. Therefore, the new group uses the global value for their setting. The system displays output similar to this:

        ```
        [root@g5001-21 ~]# mmcallhome capability list
        group    callHomeNode           status
        -------- ---------------------- ----------
        global    ---                   enabled
        g1       g5001-21.localnet.com  disabled
        g2       g5001-22.localnet.com  disabled
        ```

        Change the capability to `enabled` for the group g1. The g1 group is set to use the global value, and is not listed in the output of the **`mmcallhome capability list`** command anymore, as shown in the sample output:

        ```
        [root@g5001-21 ~]# mmcallhome capability enable accept
        Call home enabled has been set to true

        Additional messages:
        License acceptance specified on command line. Callhome enabled.
        [root@g5001-21 ~]# mmcallhome capability list
        group    callHomeNode           status
        -------- ---------------------- ----------
        global    ---                   enabled
        g2       g5001-22.localnet.com  disabled
        ```

        After this procedure is successfully completed, the **`mmcallhome capability list`**, **`mmcallhome proxy list`**, and **`mmcallhome schedule list`** commands must list only one global set of values.

Call home configuration changes to be made while upgrading to IBM Spectrum Scale 4.2.1 from IBM Spectrum Scale 4.2.0

The existing call home configurations will be lost while upgrading from IBM Spectrum Scale 4.2.0 to IBM Spectrum Scale 4.2.1 or later versions.

If you want to deploy the existing configuration after an upgrade, follow the given steps:

1. Query and save your existing configuration prior to the upgrade, using the following commands:

```
# mmcallhome group list
# mmcallhome capability list
# mmcallhome info list
# mmcallhome proxy list
# mmcallhome schedule list
# cat /usr/lpp/mmfs/data/callhome/etc/ENmain.properties | grep 'CustomerId\|Email\|CountryCode'
# cat /usr/lpp/mmfs/data/callhome/etc/ENlocation.properties | grep CustomerName
```

2. Execute the upgrade.
3. Reconfigure call home with the configuration that has been saved in step 1, using the **mmcallhome** command. For more information on how to configure call home, see the *Configuring call home to enable manual and automated data upload* section in the *IBM Spectrum Scale: Problem Determination Guide*.

Manually upgrading pmswift

To upgrade pmswift you can either uninstall and reinstall the pmswift rpm or use the native rpm upgrade command: `rpm -Uvh pmswift-version-release.noarch.rpm`. After upgrading, restart the service using the following command: `systemctl restart pmswiftd.service`.

Uninstall pmswift-version-release.noarch.rpm

1. Stop the pmsensors.service using the following command:

   ```
   systemctl stop pmsensors.service
   ```

2. If uninstalling pmswift-4.1.1-4 or later, stop the pmswiftd.service using the following command:

   ```
   systemctl stop  pmswiftd.service
   ```

   If uninstalling pmswift-4.1.1-3, stop the pmprovidor.service using the following command:

   ```
   systemctl stop pmprovider.service
   ```

3. Uninstall the pmswift rpm using the following command:

   ```
   rpm -evh --nodeps pmswift
   ```

   If you are uninstalling pmswift-4.1.1-3, it should edit the Object configuration files for all Object servers and remove the entries created at the time of installation. The Object configuration files in `/etc/swift/ directory` are:

   - account - *.conf
   - container - *.conf
   - object - *.conf
   - proxy - *.conf

   This should also edit the sensors configuration file, `/opt/IBM/zimon/ZIMonSensors.cfg`, to remove the Object related sensors entries created at the time of installation. If you are uninstalling pmswift-4.1.1-4 or later these files will be left alone.

4. Ensure that following directories/files are removed. If they are not removed, you can, remove them manually.
   a. `/usr/local/swiftmon` directory or `/usr/local/pmswift` directory
   b. `/var/log/swiftmon` directory or `/var/log/pmswift` directory
   c. `/var/run/swiftmon` directory or `/var/runpmswift.pid` file
   d. For pmswift-4.1.1-4 and later remove `/etc/rc.d/init.d/pmswift` file and for pmswift-4.1.1-3 remove `/etc/rc.d/init.d/pmprovider` file
   e. For pmswift-4.1.1-3 `SwiftAccount.cfg`, `SwiftContainer.cfg`, `SwiftObject.cfg` and `SwiftProxy.cfg` files from within the Performance Monitoring tool's installation directory, `/opt/IBM/zimon/`.

5. Ensure that for pmswift-4.1.1-3 the pmprovider.service and for pmswift-4.1.1-4 and later the pmswftd.service is not available anymore by running the following command:

   ```
   systemctl daemon-reload
   ```

Install pmswift-version-release.noarch.rpm

1. Install the pmswift rpm using the following command:

   ```
   rpm -ivh pmswift-version-release.noarch.rpm
   ```

2. Ensure that following directories/files have been created:
   a. /usr/local/pmswift directory
   b. /var/log/pmswift directory
   c. /etc/logrotate.d/pmswift file
   d. /etc/rc.d/init.d/pmswiftd file
   e. SwiftAccount.cfg, SwiftContainer.cfg, SwiftObject.cfg and SwiftProxy.cfg files in the Performance Monitoring tool's installation directory, /opt/IBM/zimon/.
3. Edit the Object configuration files for all Object servers that reside in CCR, using the /usr/local/pmswift/bin/pmswift-config-swift set command. CCR will then propagate modified configuration files to /etc/swift/ directory on all the protocol nodes within the cluster. The modified configuration files are:
   - account - *.conf
   - container - *.conf
   - object - *.conf
   - proxy - *.conf
4. Edit the sensors configuration file information stored in the CCR using the /usr/local/pmswift/bin/pmswift-config-zimon set command to add the following Object related sensors entries:

   ```
   {
     # SwiftAccount operational metrics
     name = "SwiftAccount"
     period = 1
     type = "generic"
     restrict= "cesNodes"
   },
   {
     # SwiftContainer operational metrics
     name = "SwiftContainer"
     period = 1
     type = "generic"
     restrict= "cesNodes"
   },
   {
     # SwiftObject operational metrics
     name = "SwiftObject"
     period = 1
     type = "generic"
     restrict= "cesNodes"
   },
   {
     # SwiftProxy operational metrics
     name = "SwiftProxy"
     period = 1
     type = "generic"
     restrict= "cesNodes"
   },
   ```

   These entries are then automatically propagated to the ZIMonSensors.cfg file in /opt/IBM/zimon on all the nodes in the cluster.
5. Start the pmswiftd.service using the following command:

   ```
   systemctl start pmswiftd.service
   ```

6. Start the pmsensors.service using the following command:

   ```
   systemctl start pmsensors.service
   ```

Manually upgrading the performance monitoring tool

You can upgrade performance monitoring by uninstalling and then reinstalling the new version of the tool.

Prerequisites:

1. Before uninstalling, ensure that you stop sensors and the collector. For more information see, *Starting and stopping the performance monitoring tool* in *IBM Spectrum Scale: Problem Determination Guide*.
2. For information on uninstalling, see "Uninstalling the Performance Monitoring tool" on page 371.
3. For information on manual installation, see "Manually installing the Performance Monitoring tool" on page 250.
4. If federation is used for performance monitoring, the collectors participating in a federation must be of the same version number.

To upgrade the tool the `gpfs.gss.pmsensors` and `gpfs.gss.pmcollector` packages need to be upgraded:

1. Uninstall the tool and then reinstall the new version of the tool, or use the operating system's native package upgrade mechanism. For example, `rpm -Uvh gpfs.gss.pmsensors-version-release.el7.x86_64.rpm`.

   Note: Ensure that the pmsensors and pmcollector are updated using the appropriate rpm packages for the platform.
2. Restart the sensors on the protocol nodes.
3. Restart the collector on the node that previously ran the collector.

   Note: If multiple collectors are used, ensure that they all have the same version.
4. Verify that the collector and the sensor are running on the node by issuing the **`systemctl status pmcollector`** command or the **`systemctl status pmsensors`** command.

   Note: If additional sensor packages such as `pmswift` or `gpfs.pm-ganesha` are used, these packages should be upgraded as well.

For more information about the Performance Monitoring tool, see *Configuring the Performance Monitoring tool* in *IBM Spectrum Scale: Problem Determination Guide*.

Manually upgrading the IBM Spectrum Scale management GUI

You can upgrade the IBM Spectrum Scale management GUI to the latest version to get the latest features. You can upgrade one GUI node at a time without shutting down IBM Spectrum Scale on other nodes to ensure high availability.

Prerequisite

Ensure that you are aware of the following details before you start the upgrade process:

- Upgrading the GUI from previous release to the latest release overwrites the customizations that are made in the property file that is located at: `/usr/lpp/mmfs/gui/conf/gpfsgui.properties`. It is recommended to take backup of the property file to save all such customizations before you upgrade the GUI. If sudo wrappers are in use, the *GPFS_ADMIN* attribute in the `gpfsgui.properties` file must be set again after you upgrade the GUI to 4.2.2 or 4.2.3. No need to update the *GPFS_ADMIN* attribute starting the 5.0.0 release because the sudo wrapper user information is not stored in the `gpfsgui.properties` file anymore.
- If an external authentication server, such as AD or LDAP, is used to authenticate the GUI users, connections to AD and LDAP systems are stored in the `/opt/ibm/wlp/usr/servers/gpfsgui/server.xml`

file. This file is overwritten during the upgrade. Ensure that you save the edited `server.xml` file to make the similar changes on the replaced file after the upgrade.

- Local users, user groups, user roles, and snapshot rules are not affected with an upgrade, because they exist in the cluster configure repository (CCR).
- Data in the postgres database is retained.
- All IBM Spectrum Scale packages must be of the same release on the GUI node. For example, do not mix the 4.2.3 GUI rpm with a 4.2.2 base rpm. However, GUI PTFs and fixes can usually be applied without having to install the corresponding PTF or fix of the base package. This method is helpful if you just want to resolve a GUI issue without changing anything on the base layer
- Release levels can be different in GUI nodes and other nodes of the cluster. However, the minimum release level of the cluster must be 4.2.0.0 or later for the GUI to function.
- The user ID "scalemgmt" must not be already used as the GUI requires this user ID to run IBM Spectrum Scale GUI Websphere Java process.
- The ports 80, 443, 474080, and 47443 must not be already used by other processes.
- You can directly upgrade the IBM Spectrum Scale GUI from 4.2.0.0 or later to the latest version.

Perform the following steps to upgrade the management GUI from 4.2.1.x or later to 4.2.2.x or later:

1. Stop the GUI services on the node by issuing the **systemctl stop gpfsgui** command.
2. Ensure that the latest packages are available at the required location. For more information on the latest packages that are required for different platforms, see "Manually installing IBM Spectrum Scale management GUI" on page 252.

   For information about the location of extracted installation images, see "Location of extracted packages" on page 235.
3. Upgrade the GUI package. For upgrading the previously installed package, use **rpm -Fvh** or **rpm -Uvh** options. The **rpm -Fvh** is used for upgrading the existing installed package and **rpm -Uvh** is used for installing the package and upgrading the package as well.

   For example: `rpm -Fvh gpfs.gui-5.0.0-0.noarch.rpm`
4. If there is a Java version upgrade, upgrade the Java package as shown in the following example:

   ```
   rpm -Fvh gpfs.java-5.0.0-0.x86_64.rpm
   ```

   Note: If you are upgrading the GUI (`gpfs.gui`) package, the GPFS Java (`gpfs.java`) package must also be upgraded to the same version.
   Java packages are platform-dependent. For more information on the latest Java package that is required for each platform, see "Manually installing IBM Spectrum Scale management GUI" on page 252.
5. After installing the packages, enable the performance monitoring tools in the GUI, if it is required. For more information, see "Enabling performance tools in management GUI" on page 255.
6. If the minimum release level set for IBM Spectrum Scale is not same as the GUI version, change the release level by issuing the **mmchconfig release=LATEST** command. As changing the minimum release level affects the cluster behavior, refer the **mmchconfig** command man page and other related topics before you make this configuration change. For more information, see "Completing the upgrade to a new level of IBM Spectrum Scale" on page 418.
7. Start the GUI by issuing the **systemctl start gpfsgui** command.
8. To make sure that the GUI and performance tool are started on the boot process, issue the following commands:

   ```
   systemctl enable gpfsgui.service
   systemctl enable pmsensors.service
   systemctl enable pmcollector.service
   ```
9. Issue the **systemctl status gpfsgui** command to verify the GUI service status.
10. Issue the **systemctl status pmcollector** and **systemctl status pmsensors** commands to verify the status of the performance tool.

Upgrading Cloud services

To upgrade to a newer version of Cloud services, first consider the version that you are migrating from and then consider coexistence and compatibility issues.

You can migrate from one Cloud services version to another. Cloud services metrics too are upgraded. For more information, see *List of metrics for Cloud services* topic in the *IBM Spectrum Scale: Administration Guide*.

Note: Before performing any migration, ensure that you back up the configuration files. For more information, see the *Backing up the Cloud services configuration* topic in the *IBM Spectrum Scale: Administration Guide*.

Cloud services migration consists of these topics:

- "Upgrading to Transparent cloud tiering 1.1.2 from Transparent cloud tiering 1.1.0 or 1.1.1"
- "Upgrading to Cloud services 1.1.2.1 from 1.1.2" on page 400
- "Upgrading to Cloud services 1.1.3 from 1.1.2" on page 400
- "Upgrading to Cloud services 1.1.3 from 1.1.0 and 1.1.1" on page 401
- "Upgrading to Cloud services 1.1.4 from 1.1.2.x" on page 401
- "Upgrading to Cloud services 1.1.4 from 1.1.3" on page 402

Upgrading to Transparent cloud tiering 1.1.2 from Transparent cloud tiering 1.1.0 or 1.1.1

This topic describes the procedure for upgrading to Transparent cloud tiering 1.1.2 from Transparent cloud tiering 1.1.0 or 1.1.1.

Ensure the following before you start the upgrade the process:

- Transparent cloud tiering service is stopped on all the server nodes by using the **mmcloudgateway service stop** command.
- IBM Spectrum Scale is at release 4.2.2. Migration might fail if your IBM Spectrum Scale release is different.
- No data migration or recall is in progress.

Perform the following steps:

1. Copy the `gpfs.tct.server-1.1.2.x86_64.rpm` file to each of the nodes that is specified as Transparent cloud tiering server nodes.
2. Run this command:
   ```
   rpm -Uvh gpfs.tct.server-1.1.2.x86_64.rpm
   ```
   Note: Transparent cloud tiering statistics on the IBM Spectrum Scale GUI is enabled only when the `ENABLE_MCSTORE` parameter in the `/usr/lpp/mmfs/gui/conf/gpfsgui.properties` is set to "true". During upgrade from IBM Spectrum Scale 4.2.1.1 to 4.2.2, this parameter is reset to "false", even though its original value is "true", and the Transparent cloud tiering statistics does not show up on the IBM Spectrum Scale GUI. Therefore, it is recommended to manually set the value of the property to "true" after an upgrade. To activate this change, you must restart the `gpfsgui.service`. All other configurations will be intact after the upgrade, and you can continue to perform data migration seamlessly as before.
3. To upgrade the client packages, issue the following command:
   ```
   rpm -Uvh gpfs.tct.client-1.1.2.1.rpm
   ```
4. To upgrade the Debian package, issue the following command:

```
sudo dpkg -i gpfs.tct.client*.deb
```

5. To verify the version after upgrade, issue the following command:

   ```
   mmcloudgateway service version
   ```

Upgrading to Cloud services 1.1.2.1 from 1.1.2

This topic describes the procedure for upgrading to Cloud services 1.1.2.1 from 1.1.2.

Ensure the following before you start the upgrade the process:

- Transparent cloud tiering service is stopped on all the server nodes by using the **mmcloudgateway service stop** command.
- IBM Spectrum Scale is at release 4.2.2.1. Migration might fail if your IBM Spectrum Scale release is different.
- No data migration or recall is in progress.

Perform the following steps:

1. Copy the gpfs.tct.server-1.1.2.1.x86_64.rpm file to each of the nodes that is specified as Transparent cloud tiering server nodes.
2. Run this command:

   ```
   rpm -Uvh gpfs.tct.server-1.1.2.1.x86_64.rpm
   ```

   Note: Transparent cloud tiering statistics on the IBM Spectrum Scale GUI is enabled only when the ENABLE_MCSTORE parameter in the /usr/lpp/mmfs/gui/conf/gpfsgui.properties is set to "true". During upgrade from IBM Spectrum Scale 4.2.2 to 4.2.2.1, this parameter is reset to "false", even though its original value is "true", and the Transparent cloud tiering statistics does not show up on the IBM Spectrum Scale GUI. Therefore, it is recommended to manually set the value of the property to "true" after an upgrade. To activate this change, you must restart the gpfsgui.service. All other configurations will be intact after the upgrade, and you can continue to perform data migration seamlessly as before.
3. For upgrading the client systems, issue this command:

   ```
   rpm -Uvh gpfs.tct.client-1.1.2.1.rpm
   ```
4. For upgrading the Debian package, issue this command:

   ```
   sudo dpkg -i gpfs.tct.client-1.1.3*.deb
   ```

Upgrading to Cloud services 1.1.3 from 1.1.2

This topic describes the procedure for upgrading to Cloud services 1.1.3 from 1.1.2.

Ensure the following before you start the upgrade the process:

- Transparent cloud tiering service is stopped on all the server nodes by using the **mmcloudgateway service stop** command.
- IBM Spectrum Scale is at release 4.2.3. Migration might fail if your IBM Spectrum Scale release is different.
- No data migration or recall is in progress.

Perform the following steps:

1. Copy the gpfs.tct.server-1.1.3.x86_64.rpm file to each of the nodes that is specified as Transparent cloud tiering server nodes.
2. Run this command:

   ```
   rpm -Uvh gpfs.tct.server-1.1.3.x86_64.rpm
   ```
3. To upgrade the client nodes, issue this command:

   ```
   rpm -Uvh gpfs.tct.client-1.1.3*.rpm
   ```

4. To upgrade the Debian package, issue this command:

   ```
   sudo dpkg -i gpfs.tct.client-1.1.3*.deb
   ```

5. Verify the version by using the **mmcloudgateway service version** command. The system displays output similar to this:

   ```
   Cluster minReleaseLevel:   4.2.2.0

    Node  Daemon node name             TCT Type          TCT Version     Equivalent Product Version
   ------------------------------------------------------------------------------------------------
       1  c350f1u1b7.pok.stglabs.ibm.com  Server            1.1.3           4.2.3
   ```

   Note: This upgrade takes longer than others because of the database schema change. The larger the database of migrated files, the longer this upgrade will take.

Upgrading to Cloud services 1.1.3 from 1.1.0 and 1.1.1

This topic describes the procedure for upgrading to Cloud services 1.1.3 from 1.1.0 or 1.1.1.

Ensure the following before you start the upgrade the process:

- Transparent cloud tiering service is stopped on all the server nodes by using the **mmcloudgateway service stop** command.
- IBM Spectrum Scale is at release 4.2.3. Migration might fail if your IBM Spectrum Scale release is different.
- No data migration or recall is in progress.

Perform the following steps:

1. Copy the `gpfs.tct.server-1.1.3.x86_64.rpm` file to each of the nodes that is specified as Transparent cloud tiering server nodes.
2. Run this command:

   ```
   rpm -Uvh gpfs.tct.server-1.1.3.x86_64.rpm
   ```

3. To upgrade the client nodes, issue the following command:

   ```
   rpm -Uvh gpfs.tct.client-1.1.3*.rpm
   ```

4. To upgrade the Debian package, issue the following command:

   ```
   sudo dpkg -i gpfs.tct.client-1.1.3*.deb
   ```

5. Verify the version by using the **mmcloudgateway service version** command. The system displays output similar to this:

   ```
   Cluster minReleaseLevel:   4.2.2.0

    Node  Daemon node name             TCT Type          TCT Version     Equivalent Product Version
   ------------------------------------------------------------------------------------------------
       1  c350f1u1b7.pok.stglabs.ibm.com  Server            1.1.3           4.2.3
   ```

Upgrading to Cloud services 1.1.4 from 1.1.2.x

This topic describes the procedure for upgrading to Cloud services 1.1.4 from 1.1.2.x

Ensure the following before you start the upgrade the process:

- Uninstalling the old rpm (**rpm -e** command) and then installing the new rpm (**rpm -i**) will not work. The upgrade must be done with the **rpm -Uvh** command only.
- Transparent cloud tiering service is stopped on all the server nodes by using the **mmcloudgateway service stop** command.
- IBM Spectrum Scale is at release 5.0.0. Upgrade might fail if your IBM Spectrum Scale release is different.
- No data migration or recall is in progress.

Perform the following steps:

1. Copy the `gpfs.tct.server-1.1.4.x86_64.rpm` file to each of the nodes that is specified as Transparent cloud tiering server nodes.
2. Run this command:
   ```
   rpm -Uvh gpfs.tct.server-1.1.4.x86_64.rpm
   ```
3. To upgrade the client nodes, issue this command:
   ```
   rpm -Uvh gpfs.tct.client-1.1.4*.rpm
   ```
4. To upgrade the Debian package, issue this command:
   ```
   sudo dpkg -i gpfs.tct.client-1.1.4*.deb
   ```
5. Verify the version by using the **`mmcloudgateway service version`** command. The system displays output similar to this:
   ```
   Cluster minReleaseLevel:   5.0.0.0

    Node  Daemon node name            TCT Type         TCT Version     Equivalent Product Version
   -----------------------------------------------------------------------------------------------
       1  c350f1u1b7.pok.stglabs.ibm.com  Server           1.1.4          5.0.0
   ```

   Note: This upgrade takes longer than others because of the database schema change. The larger the database of migrated files, the longer this upgrade will take.

Note:

- After the upgrade, the previous sensors are no more valid and Cloud services performance monitoring will fail on the GUI. Therefore, you must remove the old sensors and populate new ones. For more information, see "Upgrading the Cloud services sensors" on page 403.
- After the upgrade, do not use any transparent recall policy that was created by using IBM Spectrum Scale 4.2.x and earlier releases. You will experience hangs during transparent recalls if the old policies are reused. Therefore, you must remove the older policy (which was being applied before upgrade) and enable the transparent recall after upgrade.

Upgrading to Cloud services 1.1.4 from 1.1.3

This topic describes the procedure for upgrading to Cloud services 1.1.4 from 1.1.3.

Ensure the following before you start the upgrade the process:

- Uninstalling the old rpm (**`rpm -e`** command) and then installing the new rpm (**`rpm -i`**) will not work. The upgrade must be done with the **`rpm -Uvh`** command only.
- Transparent cloud tiering service is stopped on all the server nodes by using the **`mmcloudgateway service stop`** command.
- IBM Spectrum Scale is at release 5.0.0. Migration might fail if your IBM Spectrum Scale release is different.
- No data migration or recall is in progress.

Perform the following steps:

1. Copy the `gpfs.tct.server-1.1.4.x86_64.rpm` file to each of the nodes that is specified as Transparent cloud tiering server nodes.
2. Run this command:
   ```
   rpm -Uvh gpfs.tct.server-1.1.4.x86_64.rpm
   ```
3. To upgrade the client nodes, issue this command:
   ```
   rpm -Uvh gpfs.tct.client-1.1.4*.rpm
   ```
4. To upgrade the Debian package, issue this command:
   ```
   sudo dpkg -i gpfs.tct.client-1.1.4*.deb
   ```
5. To upgrade the SLES package, issue this command:

```
rpm -Uvh gpfs.tct.server-1.1.4*.rpm
```

6. Verify the version by using the **mmcloudgateway service version** command. The system displays output similar to this:

   ```
   Cluster minReleaseLevel:   5.0.0.0

    Node  Daemon node name             TCT Type         TCT Version     Equivalent Product Version
   ---------------------------------------------------------------------------------------------
       1  c350f1u1b7.pok.stglabs.ibm.com  Server           1.1.4          5.0.0
   ```

   Note: This upgrade takes longer than others because of the database schema change. The larger the database of migrated files, the longer this upgrade will take.

After the upgrade, the previous sensors are no more valid and Cloud services performance monitoring will fail on the GUI. Therefore, you must remove the old sensors and populate new ones. For more information, see "Upgrading the Cloud services sensors."

Upgrading the Cloud services sensors

The sensors in earlier versions of the Cloud services are no longer valid in the 1.1.4 version. Therefore you must change the sensors after upgrading to Cloud services version 1.1.4 from either 1.1.3.x or 1.1.2.x version as described below.

You must know which sensors are configured and active.

Perform the following steps to upgrade the sensors:

1. Remove configuration of the old sensors by issuing the following commands:
   - `mmperfmon config delete --sensors MCStoreGPFSStats`
   - `mmperfmon config delete --sensors MCStoreIcstoreStats`
   - `mmperfmon config delete --sensors MCStoreLWEStats`
2. Remove the old sensors files from the `/opt/IBM/zimon` folder of every Cloud services node:
   - `rm MCStoreGPFSStats.cfg`
   - `rm MCStoreIcstoreStats.cfg`
   - `rm MCStoreLWEStats.cfg`
3. Copy the following sensor files from the `/opt/ibm/MCStore/config/` folder to the `/opt/IBM/zimon` folder:
   - TCTDebugDbStats.cfg
   - TCTDebugLweDestroyStats.cfg
   - TCTFsetGpfsConnectorStats.cfg
   - TCTFsetIcstoreStats.cfg
   - TCTFsGpfsConnectorStats.cfg
   - TCTFsIcstoreStats.cfg
4. On each Cloud services server node, create a new `MCStore-sensor-definition.cfg` file under `/opt/IBM/zimon/` and copy the following sensors text into it:

   ```
   sensors=
     {
        #Transparent cloud tiering statistics
        name = "TCTDebugDbStats"
        period = 10
        type = "Generic"
      },

     {
       #Transparent cloud tiering statistics
       name = "TCTDebugLweDestroyStats"
       period = 10
       type = "Generic"
   ```

```
  },
  {
     #Transparent cloud tiering statistics
     name = "TCTFsetGpfsConnectorStats"
     period = 10
     type = "Generic"
   },

  {
    #Transparent cloud tiering statistics
    name = "TCTFsetIcstoreStats"
    period = 10
    type = "Generic"
  },
  {
     #Transparent cloud tiering statistics
     name = "TCTFsGpfsConnectorStats"
     period = 10
     type = "Generic"
   },

  {
    #Transparent cloud tiering statistics
    name = "TCTFsIcstoreStats"
    period = 10
    type = "Generic"
  }
```

5. Specify the following command to register the sensors:

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

6. Specify the following command to verify that all the sensors are added:

   ```
   mmperfmon config show
   ```

   The system displays output similar to this:

   ```
   # This file has been generated automatically and SHOULD NOT
   # be edited manually.  It may be overwritten at any point
   # in time.

   cephMon = "/opt/IBM/zimon/CephMonProxy"
   cephRados = "/opt/IBM/zimon/CephRadosProxy"
   colCandidates = "jupiter-vm856.pok.stglabs.ibm.com", "jupiter-vm934.pok.stglabs.ibm.com"
   colRedundancy = 2
   collectors = {
    host = ""
    port = "4739"
   }
   config = "/opt/IBM/zimon/ZIMonSensors.cfg"
   ctdbstat = ""
   daemonize = T
   hostname = ""
   ipfixinterface = "0.0.0.0"
   logfile = "/var/log/zimon/ZIMonSensors.log"
   loglevel = "info"
   mmcmd = "/opt/IBM/zimon/MMCmdProxy"
   mmdfcmd = "/opt/IBM/zimon/MMDFProxy"
   mmpmon = "/opt/IBM/zimon/MmpmonSockProxy"
   piddir = "/var/run"
   release = "5.0.0-0"
   sensors = {
    name = "CPU"
    period = 1
   },
   {
    name = "Load"
    period = 1
   },
   ```

```
{
 name = "Memory"
 period = 1
},
{
 name = "Network"
 period = 1
},
{
 name = "Netstat"
 period = 10
},
{
 name = "Diskstat"
 period = 0
},
{
 name = "DiskFree"
 period = 600
},
{
 name = "Infiniband"
 period = 0
},
{
 name = "GPFSDisk"
 period = 0
},
{
 name = "GPFSFilesystem"
 period = 10
},
{
 name = "GPFSNSDDisk"
 period = 10
 restrict = "nsdNodes"
},
{
 name = "GPFSPoolIO"
 period = 0
},
{
 name = "GPFSVFS"
 period = 10
},
{
 name = "GPFSIOC"
 period = 0
},
{
 name = "GPFSVIO64"
 period = 0
},
{
 name = "GPFSPDDisk"
 period = 10
 restrict = "nsdNodes"
},
{
 name = "GPFSvFLUSH"
 period = 0
},
{
 name = "GPFSNode"
 period = 10
},
{
 name = "GPFSNodeAPI"
 period = 10
},
{
 name = "GPFSFilesystemAPI"
 period = 10
},
{
 name = "GPFSLROC"
 period = 0
},
{
 name = "GPFSCHMS"
 period = 0
},
{
 name = "GPFSAFM"
 period = 0
},
{
 name = "GPFSAFMFS"
 period = 0
},
{
 name = "GPFSAFMFSET"
 period = 0
},
{
 name = "GPFSRPCS"
 period = 10
},
{
 name = "GPFSWaiters"
 period = 10
},
{
 name = "GPFSFilesetQuota"
 period = 3600
 restrict = "jupiter-vm856.pok.stglabs.ibm.com"
},
{
 name = "GPFSFileset"
 period = 300
 restrict = "jupiter-vm934.pok.stglabs.ibm.com"
},
{
 name = "GPFSPool"
 period = 300
 restrict = "jupiter-vm934.pok.stglabs.ibm.com"
},
{
 name = "GPFSDiskCap"
 period = 86400
 restrict = "jupiter-vm856.pok.stglabs.ibm.com"
},
{
 name = "TCTDebugDbStats"
 period = 10
 type = "Generic"
},
{
 name = "TCTDebugLweDestroyStats"
 period = 10
 type = "Generic"
},
{
 name = "TCTFsetGpfsConnectorStats"
```

```
 period = 10
 type = "Generic"
},
{
 name = "TCTFsetIcstoreStats"
 period = 10
 type = "Generic"
},
{
 name = "TCTFsGpfsConnectorStats"
 period = 10
 type = "Generic"
},
{
 name = "TCTFsIcstoreStats"
 period = 10
 type = "Generic"
}
smbstat = ""
```

7. Specify the following commands to restart your `pmsensors` and `pmcollector` on the nodes:
   - `service pmsensors restart`
   - `service pmcollector restart`
8. Verify the status of your `pmsensors` and `pmcollector` on the nodes:
   - `mservice pmsensors status`
   - `service pmcollector status`

Upgrading to IBM Cloud Object Storage software level 3.7.2 and above

If you have used IBM Cloud Object Storage software level system (3.7.x and below) as the cloud storage tier for some time and then upgraded to IBM Cloud Object Storage software level 3.7.2 or above, you must perform certain configurations. This is required because of a change in computation of the entity tag (ETag) in IBM Cloud Object Storage software level 3.7.2.

Do the following steps to upgrade to IBM Cloud Object Storage software level 3.7.2 and above:

1. To display the configured cloud account, issue this command: **`mmcloudgateway account list`**
2. Edit `/var/MCStore/.mcstore_settings` and change the configuration property "<cloudname>.provider=cleversafe" to "<cloudname>.provider=cleversafe-new".
3. Restart the Transparent cloud tiering service by issuing this command: **`mmcloudgateway service restart`**

Upgrading IBM Spectrum Scale components with the installation toolkit

You can use the installation toolkit to upgrade core GPFS, NFS, SMB, object, management GUI, and performance monitoring from the current version of IBM Spectrum Scale to the next higher version. For example, you can use the installation toolkit to upgrade from IBM Spectrum Scale release 4.2.x to release 5.0.x.

For information on the upgrade flow, see "IBM Spectrum Scale installation toolkit - upgrade process flow" on page 413.

Important:

- Protocol upgrades are non-concurrent and they cause brief outages to user access. Schedule an outage window for the duration of the upgrade, which is approximately 20 minutes per protocol node.
  - For NFS, it is recommended to quiesce all I/O for the duration of the upgrade. NFS experiences I/O pauses, and depending upon the client, mounts and disconnects during the upgrade.

- SMB requires quiescing all I/O for the duration of the upgrade. Due to the SMB clustering functionality, differing SMB levels cannot co-exist within a cluster at the same time. This requires a full outage of SMB during the upgrade.
- For object, it is recommended to quiesce all I/O for the duration of the upgrade. Object service will be down or interrupted at multiple times during the upgrade process. Clients might experience errors or they might be unable to connect during this time. They must retry as appropriate.

- The IBM Spectrum Scale cluster that you are upgrading must be healthy. A healthy cluster in the context of this upgrade has the following characteristics:
  - GPFS must be up and running on all nodes defined in the cluster definition file.
  - All protocols defined on each protocol node must be up and running, if the protocol has been initially enabled.
  - The whole cluster is operational.
- Before proceeding with the IBM Spectrum Scale upgrade, ensure that Transparent cloud tiering is not active on the cluster. If it is active, shut down Transparent cloud tiering services on all nodes by issuing the **mmcloudgateway service stop** command. If Transparent cloud tiering is active, the IBM Spectrum Scale upgrade might fail.
- The installation toolkit does not support upgrading a cluster containing manually installed protocols, if the entire protocols stack was not originally installed in the cluster. To upgrade these clusters, use the manual procedures. For more information, see Chapter 15, "Upgrading," on page 375.
- Starting with the IBM Spectrum Scale 4.2.3 release, the installation toolkit supports the installation, deployment, and upgrade of Advanced (gpfs.adv) and Crypto (gpfs.crypto) packages. The upgrade of these packages is handled by the installation toolkit according to your IBM Spectrum Scale product edition.
  - For IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition, if the gpfs.adv and gpfs.crypto packages are installed on the system that you are upgrading, these packages are also upgraded.
  - For IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition, if the gpfs.adv and gpfs.crypto packages are not installed on the system that you are upgrading, these packages are installed.
  - These packages are not available with IBM Spectrum Scale Standard Edition.

  If you have IBM Spectrum Scale Standard Edition version 4.2.2.x installed, you can use the installation toolkit to upgrade to IBM Spectrum Scale Advanced Edition or IBM Spectrum Scale Data Management Edition version 4.2.3 or later. In this scenario, you must apply the license for the applicable edition.

Before proceeding with upgrade using the installation toolkit, ensure the following:

- The prerequisite steps for using the installation toolkit are done. This is particularly important if you are using the installation toolkit to upgrade a cluster that was created manually. For more information, see "Preparing to use the installation toolkit" on page 270.
- Review limitations of the installation toolkit related to upgrade. For more information, see "Limitations of the installation toolkit" on page 263.

For performing an upgrade by using the installation toolkit, a key requirement is the cluster definition file that you can obtain by using one of the following ways:

1. Use the **./spectrumscale config populate** command to retrieve the cluster configuration and populate it in the cluster definition file.
2. Use an existing cluster definition file if it exists in the current version installer configuration directory.

For more information, see Step 4.

Do the following steps to upgrade IBM Spectrum Scale using the installation toolkit:

1. Download the new IBM Spectrum Scale self-extracting package using the sub-steps and then place it on the installer node.

a. Go to the IBM Spectrum Scale page on Fix Central, select the new **spectrumscale** package and then click **Continue**.

b. Choose the download option **Download using Download Director** to download the new **spectrumscale** package and place it in the wanted location on the install node.

   Note: Although it is not recommended but if you must use the download option **Download using your browser (HTTPS)**, instead of clicking the down arrow to the left of the package name, you must right-click on the package name and select the **Save Link As..** option. If you click the download arrow, the browser might hang.

2. Extract the new IBM Spectrum Scale self-extracting package by using the package name (for example, `/tmp/Spectrum_Scale_Protocols_Standard-5.0.0.0_x86_64-Linux_install`).

   This creates a new directory structure (`/usr/lpp/mmfs/5.0.0.0/`).

3. Accept the license agreement. After the self-extracting package extraction is done several next steps and instructions are displayed on the screen including:
   - cluster installation and protocol deployment
   - upgrading an existing cluster using the installation toolkit
   - adding nodes to an existing cluster using the installation toolkit
   - adding NSDs or file systems to an existing cluster using the installation toolkit

4. Depending on your setup, do one of the following:
   - If you are upgrading an IBM Spectrum Scale cluster that was created by using the installation toolkit, do these steps to upgrade by using the installation toolkit.

     Note: If you have used the installation toolkit for the initial installation and deployment, it is assumed that the prerequisites are already in place.

     a. If you want to change the installer node before the upgrade, issue the `spectrumscale setup -s` *`InstallNodeIP`* `command` to set up the new installer node.

     b. Create a backup of the cluster definition file present in the installer configuration directory of the target version. For example: `/usr/lpp/mmfs/5.0.0.0/installer/configuration`

        This backup can be used in case the config populate command fails.

     c. From the installer directory of the target version that you are upgrading to, populate the cluster definition file with the current cluster state by issuing the **`./spectrumscale config populate`** command. For more information, see "Populating cluster definition file with current cluster state using the installation toolkit" on page 292 and "Limitations of config populate option of the installation toolkit" on page 292.

        If the config populate command fails, before proceeding with the upgrade, do the following.

        1) Copy the cluster definition file from the `/usr/lpp/mmfs/4.2.x.x/installer/configuration` directory to the `/usr/lpp/mmfs/5.0.0.0/installer/configuration` directory.

        2) Update the cluster definition file with the cluster state change operations that were done after the initial installation and deployment by using the **`./spectrumscale`** commands. These cluster state change operations include:
           – Node-related changes such as adding nodes
           – NSD-related changes such as adding or modifying NSDs
           – File system changes such as creating or modifying file systems
           – Protocols-related changes such as enabling or disabling protocols

   - If you are upgrading a manually created IBM Spectrum Scale cluster, do these steps to upgrade by using the installation toolkit.

     a. Ensure that the prerequisite steps for using the installation toolkit are completed. For more information, see "Preparing to use the installation toolkit" on page 270.

     b. Configure your cluster definition file as explained in the following topics:
        – "Setting up the installer node" on page 272

   - “Defining the cluster topology for the installation toolkit” on page 273

   c. From the installer directory of the target version that you are upgrading to, populate the cluster definition file with the current cluster state by issuing the **./spectrumscale config populate** command. For more information, see “Populating cluster definition file with current cluster state using the installation toolkit” on page 292 and “Limitations of config populate option of the installation toolkit” on page 292.

      If the config populate command fails, before proceeding with the upgrade, update the cluster definition file with the cluster state change operations that were done after the initial installation and deployment by using the **./spectrumscale** commands. These cluster state change operations include:

      - Node-related changes such as adding nodes
      - NSD-related changes such as adding or modifying NSDs
      - File system changes such as creating or modifying file systems
      - Protocols-related changes such as enabling or disabling protocols

5. Ensure that the cluster definition file is accurate and reflects the current state of your IBM Spectrum Scale cluster.

   Note: If you are upgrading from IBM Spectrum Scale release 4.2.1 or earlier, ensure that the operating system and architecture fields in the output of the **./spectrumscale node list** command are updated. These fields are automatically updated when you perform the upgrade precheck with **./spectrumscale upgrade --precheck** or you can update these fields by issuing the **./spectrumscale config update** command.

   If you want to upgrade IBM Spectrum Scale on IBM Spectrum Archive Enterprise Edition (EE) nodes using the installation toolkit, you must perform some additional steps to manage IBM Spectrum Archive components. For more information, see “Upgrading IBM Spectrum Scale on IBM Spectrum Archive Enterprise Edition (EE) nodes using the installation toolkit” on page 411.

6. Run the upgrade precheck using the **./spectrumscale upgrade --precheck** command.

   For any cluster with existing performance monitoring collectors, the upgrade prechecks might find that a reconfiguration is required. To enable the installation toolkit to perform the reconfiguration, issue the following command:

   ```
   ./spectrumscale config perfmon -r on
   ```

   To keep your existing algorithms, issue the following command:

   ```
   ./spectrumscale config perfmon -r off
   ```

   If you are upgrading to IBM Spectrum Scale version 4.2.2 or later, the operating system and architecture fields in the cluster definition file must be updated. The upgrade precheck updates the operating system and CPU architecture fields in the cluster definition file. Alternatively, you can issue the **spectrumscale config update** command to update the operating system and CPU architecture fields in the cluster definition file.

7. Run the upgrade process using the **./spectrumscale upgrade** command.

   Note:

   - Accept the warnings for system impact and interruption during the upgrade.
   - If you are upgrading to IBM Spectrum Scale release 4.2.2.1 or later on SLES 12 nodes, ignore the following warning generated by the local zypper repository created by the installation toolkit:

     ```
     WARN: Chef::Config[:zypper_check_gpg] was not set.
     All packages will be installed without gpg signature checks. This is a security hazard.
     ```

   After the upgrade is completed, the installation toolkit automatically runs the upgrade post checks. You can manually run the upgrade post checks by issuing the **./spectrumscale upgrade --postchecks** command.

8. After the upgrade process is done, complete the upgrade to the new code level to take advantage of the new functionality. For more information, see "Completing the upgrade to a new level of IBM Spectrum Scale" on page 418.

Installing new functions after upgrade

You can use the installation toolkit to install new functions that are available in the new version that you have upgraded to and that are supported by the installation toolkit. Use the following steps to do so:

1. Enable the functions in the cluster definition file using the installation toolkit.
2. Issue the **`./spectrumscale install`** or the **`./spectrumsacle deploy`** command.

Failure recovery

If a failure occurs during an upgrade, examine the log at the location provided in the output from the installation toolkit to determine the cause. More information on each error is provided in the output from the installation toolkit.

Certain failures will cause the upgrade process to stop. In this case, the cause of the failure must be addressed on the node on which it occurred. Once the problem has been addressed, the upgrade command can be run again. This does not affect any nodes that were upgraded successfully and the upgrade process continues from the point of failure.

An example of a failure during upgrade is:

A protocol is enabled in the configuration file, but is not running on a node.

Using the **mmces service list** command on the node that failed highlights which process is not running (the output from the installation toolkit also reports which component failed). Make sure the component is started on all nodes with **mmces service start nfs | smb | obj**, or alternatively disable the protocol in the configuration using **spectrumscale disable nfs|smb|object** if the component has been intentionally stopped.

CAUTION:

When a protocol is disabled, the protocol is stopped on all protocol nodes in the cluster and all protocol-specific configuration data is removed.

For information on troubleshooting installation, deployment, and upgrade related issues, see *Resolving most frequent problems related to installation, deployment, and upgrade* in *IBM Spectrum Scale: Problem Determination Guide*.

Upgrading IBM Spectrum Scale on IBM Spectrum Archive Enterprise Edition (EE) nodes using the installation toolkit

On an IBM Spectrum Archive Enterprise Edition (EE) node, if you want to upgrade IBM Spectrum Scale, some steps need to be taken before and after the upgrade to manage the IBM Spectrum Archive related components.

Use this information to upgrade IBM Spectrum Scale on IBM Spectrum Archive Enterprise Edition (EE) nodes by using the installation toolkit.

Note: For latest information about IBM Spectrum Archive Enterprise Edition (EE) commands, refer to *IBM Spectrum Archive Enterprise Edition (EE) documentation on IBM Knowledge Center*.

| 1. Perform the following steps that must be carried out on IBM Spectrum Archive Enterprise Edition
| (EE) components prior to an IBM Spectrum Scale upgrade.

| a. Stop IBM Spectrum Archive (LTFS) by issuing the following command on all IBM Spectrum
| Archive Enterprise Edition (EE) nodes.

```
ltfsee stop
```

   b. Unmount the media by issuing the following command on all IBM Spectrum Archive Enterprise Edition (EE) nodes.

```
umount /ltfs
```

   c. Deactivate failover operations by issuing the following command on all IBM Spectrum Archive Enterprise Edition (EE) nodes.

```
dsmmigfs disablefailover
```

   d. Stop the HSM daemons by issuing the following command on all IBM Spectrum Archive Enterprise Edition (EE) nodes.

```
dsmmigfs stop
```

   e. Stop the HSM service by issuing the following command on all IBM Spectrum Archive Enterprise Edition (EE) nodes.

```
systemctl stop hsm.service
```

After these steps, proceed with downloading, extracting, and doing other steps prior to an installation toolkit upgrade precheck. For more information, see "Upgrading IBM Spectrum Scale components with the installation toolkit" on page 407.

2. Run the installation toolkit upgrade precheck by issuing the following commands.

```
cd /usr/lp/mmfs/5.0.x.x/installer
./spectrumscale upgrade --precheck
```

   Where 5.0.x.x is the release specific directory.

   This precheck fails and lists all RPMs that are failing the dependency check. Ensure that the RPMs listed are what is expected when IBM Spectrum Archive Enterprise Edition (EE) is installed.

3. Set the environment variable to bypass the installation toolkit RPM dependency check.

```
export SSFEATURE_OVERRIDE_EXT_PKG_DEPS=true
```

4. Rerun the installation toolkit upgrade precheck by issuing the following command.

```
./spectrumscale upgrade --precheck
```

   This precheck should pass if no other problems are detected.

5. Run the installation toolkit upgrade by issuing the following command.

```
./spectrumscale upgrade
```

6. Upgrade IBM Spectrum Archive Enterprise Edition (EE), if needed.
7. Perform the following steps that must be carried out on IBM Spectrum Archive Enterprise Edition (EE) components after a successful IBM Spectrum Scale upgrade.

   a. Start the HSM service by issuing the following command on all IBM Spectrum Archive Enterprise Edition (EE) nodes.

```
systemctl start hsm.service
```

   b. Start the HSM daemons by issuing the following command on all IBM Spectrum Archive Enterprise Edition (EE) nodes.

```
dsmmigfs start
```

   c. Activate failover operations by issuing the following command on all IBM Spectrum Archive Enterprise Edition (EE) nodes.

```
dsmmigfs enablefailover
```

   d. Mount the media by issuing the following command on all IBM Spectrum Archive Enterprise Edition (EE) nodes.

```
ltfs -o devname=DEVICE /ltfs
```

   e. Start IBM Spectrum Archive (LTFS) by issuing the following command on all IBM Spectrum Archive Enterprise Edition (EE) nodes.

```
ltfsee start
```

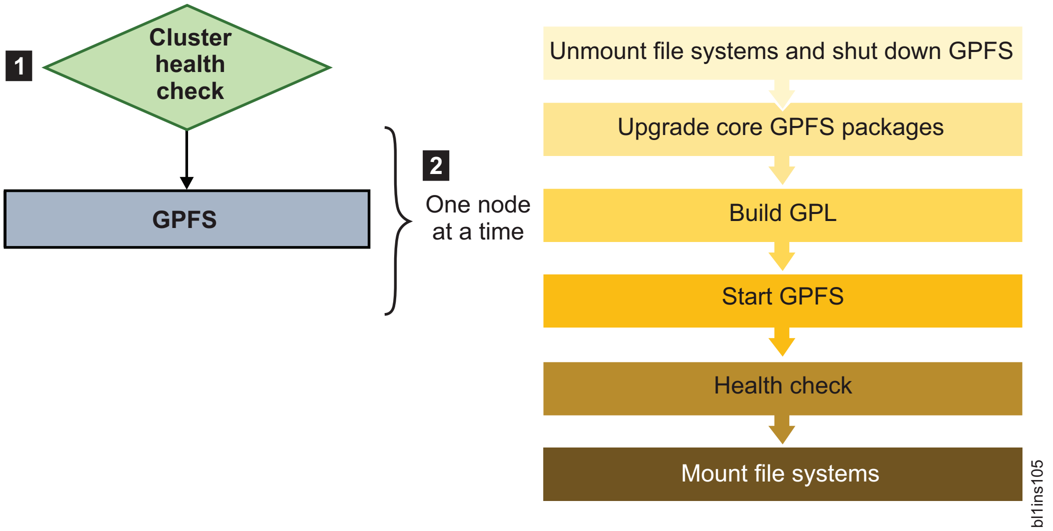
After these steps, you must complete the upgrade to new code level to take advantage of the new functionality. For more information, see “Completing the upgrade to a new level of IBM Spectrum Scale” on page 418.

IBM Spectrum Scale installation toolkit - upgrade process flow

The upgrade procedure using the installation toolkit comprises two phases.

In the first phase, all the non-protocol nodes are upgraded. In the second phase, all the protocol nodes are upgraded. The second phase is executed in two stages. NFS, object, performance monitoring, and GUI functions are upgraded in the first stage of phase 2. SMB and CES are upgraded in the second stage of phase 2.

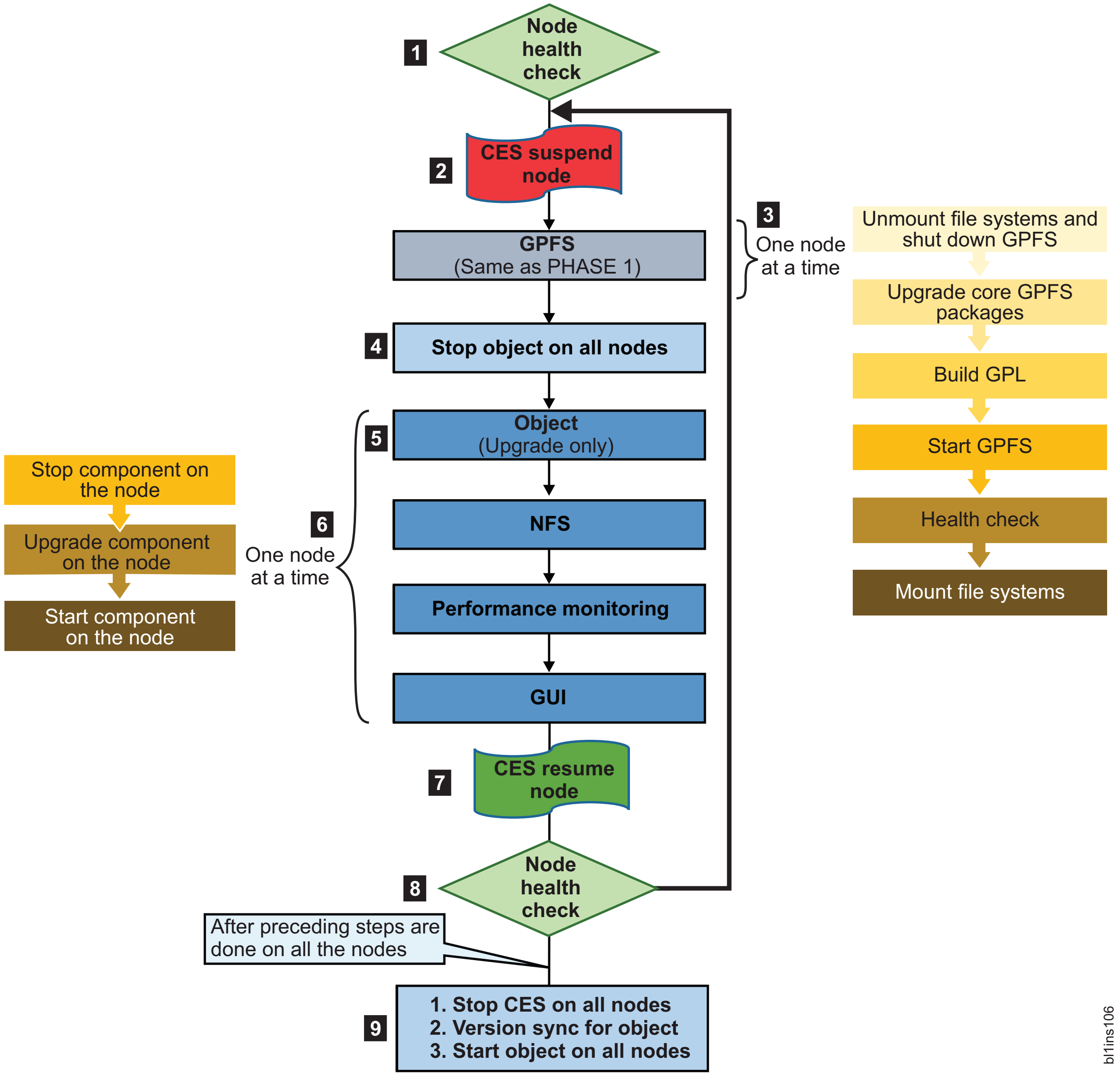
Phase 1 - upgrade non-protocol nodes



In this phase, all non-protocol nodes are upgraded one node at a time. The first step is to check the cluster health. After this, the upgrade process starts on the first non-protocol node.

1	Check the cluster health.
2	Perform following steps on each non-protocol node, one node at a time. 1. Unmount file systems and shut down GPFS. **Attention:** At this point, I/O from this node stops. 2. Upgrade core GPFS packages. 3. Build GPFS portability layer (GPL). 4. Start up GPFS. 5. Perform node health check. 6. Mount the file systems. **Attention:** At this point, I/O from this node resumes. These steps are repeated on the remaining non-protocol nodes, one node at a time. After all of the non-protocol nodes are upgraded, phase 1 is complete.

Phase 2 - upgrade protocol nodes (stage 1)

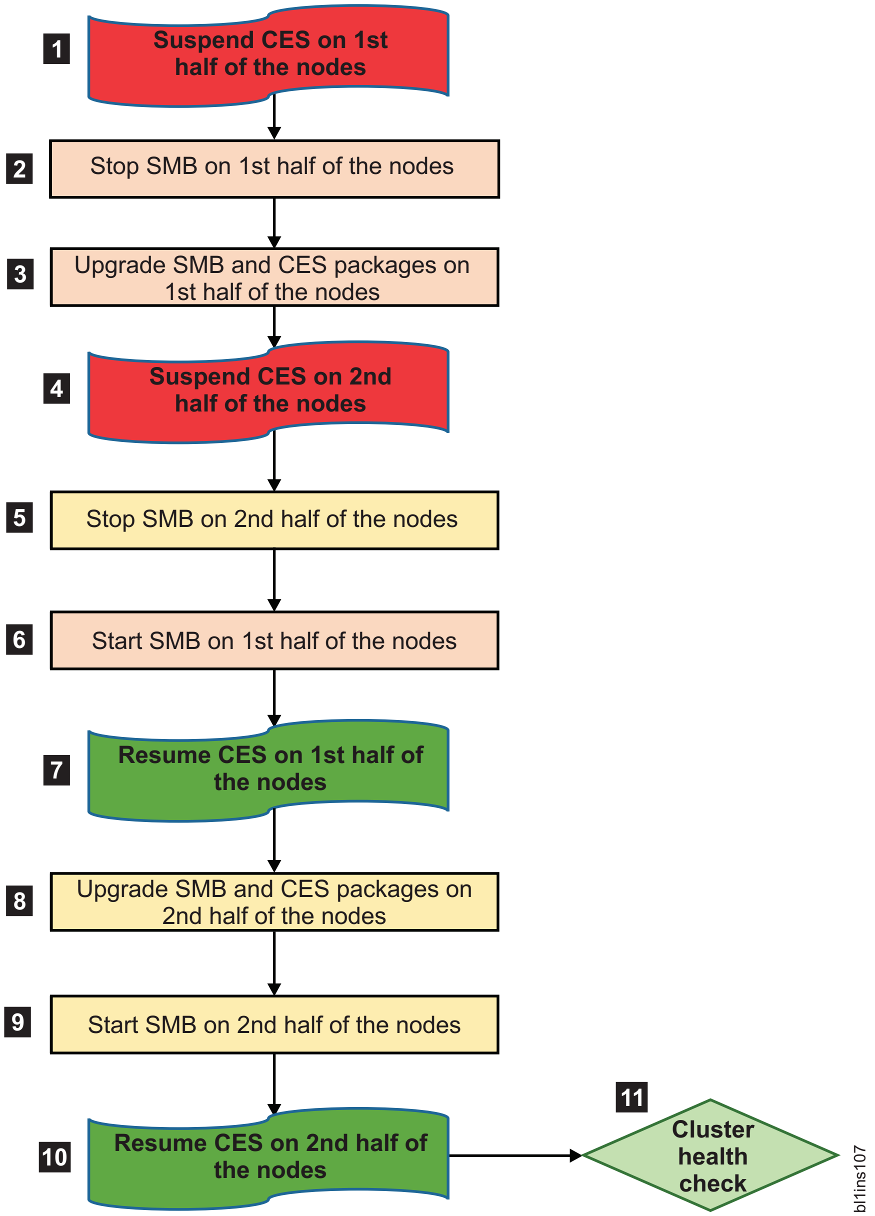


In this stage of phase 2, all protocol nodes are upgraded one node at a time.

1	Check the node health.
2	Suspend CES on the node. **Attention:** At this point, I/O from this node stops.
3	Upgrade core GPFS on the node. This procedure is same as phase 1.
4	Stop object services on all protocol nodes in the cluster. **Attention:** At this point, object I/O stops in the whole cluster.
5	Upgrade object package.

6	Upgrade NFS, performance monitoring, and GUI as follows: 1. Stop the component. 2. Upgrade the component. 3. Start the component.
7	Resume CES on the node. **Attention:** At this point, I/O from this node resumes.
8	Check the node health. Steps 2 to 8 are repeated on all protocol nodes, one node at a time.
9	After the preceding steps are repeated on all protocol nodes, do the following steps. 1. Stop CES services. 2. Perform object version sync. 3. Start object services on all protocol nodes. **Attention:** At this point, object I/O resumes in the whole cluster. This concludes the first stage of phase 2.

Phase 2 - upgrade protocol nodes (stage 2)


1
Suspend CES on 1st half of the nodes
2
Stop SMB on 1st half of the nodes
3
Upgrade SMB and CES packages on 1st half of the nodes
4
Suspend CES on 2nd half of the nodes
5
Stop SMB on 2nd half of the nodes
6
Start SMB on 1st half of the nodes
7
Resume CES on 1st half of the nodes
8
Upgrade SMB and CES packages on 2nd half of the nodes
9
Start SMB on 2nd half of the nodes
10
Resume CES on 2nd half of the nodes
11
Cluster health check
bl1ins107

In this stage of phase 2, all the protocol nodes are divided into two halves and each half is upgraded one by one as follows.

1	Suspend CES on the first half of nodes.
2	Stop SMB on the 1st half of nodes. **Attention:** At this point, SMB I/O on this half of nodes stops and CES IPs on these nodes are migrated to the nodes in the other half.
3	Upgrade SMB and CES packages on the 1st half of nodes.
4	Suspend CES on the 2nd half of nodes. **Attention:** At this point, SMB I/O stops on the whole cluster for a brief period.

5	Stop SMB on the 2nd half of nodes.
6	Start SMB on the 1st half of nodes.
7	Resume CES on the 1st half of nodes. **Attention:** At this point, SMB I/O resumes on this half of nodes.
8	Upgrade SMB and CES packages on the 2nd half of nodes.
9	Start SMB on the 2nd half of nodes.
10	Resume CES on the 2nd half of nodes. **Attention:** At this point, SMB I/O resumes in the whole cluster.
11	Check the cluster health.

After these steps are done, the installation toolkit automatically runs upgrade post checks. You can manually run the upgrade post checks by issuing the **`./spectrumscale upgrade --postchecks`** command.

You must complete the upgrade to the new code level to take advantage of the new functionality. For more information, see "Completing the upgrade to a new level of IBM Spectrum Scale" on page 418.

Migrating from Express Edition to Standard Edition

Starting with release 4.2.3, IBM Spectrum Scale Express Edition is no longer available. Use this information for migrating from IBM Spectrum Scale Express Edition 4.2.2.x or earlier to IBM Spectrum Scale Standard Edition 4.2.3.x or later.

Before you begin, extract the IBM Spectrum Scale software as described in the "Extracting the IBM Spectrum Scale software on Linux nodes" on page 231 topic. For information about the location of extracted installation images, see "Location of extracted packages" on page 235.

Use one of the following sets of steps depending on the operating system and the IBM Spectrum Scale version.

Red Hat Enterprise Linux and SLES

- To migrate from 4.1.1.x or 4.2.0.x or 4.2.1.x Express Edition to 4.2.3 Standard Edition or later on Red Hat Enterprise Linux and SLES, update the existing RPMs and install new RPMs by issuing this command.

  ```
  rpm -Uvh gpfs*rpm
  ```

- To migrate from 4.2.2.x Express Edition to 4.2.3 Standard Edition or later on Red Hat Enterprise Linux and SLES, use these steps.
  1. Uninstall the Express Edition license RPM by issuing the following command.

     ```
     rpm -e gpfs.license.exp*.rpm
     ```

  2. Update the existing RPMs and install new RPMs by issuing the following command.

     ```
     rpm -Uvh gpfs*rpm
     rpm -ivh gpfs.compression*.rpm
     rpm -ivh gpfs.license.std*.rpm
     ```

Debian and Ubuntu

- To migrate from 4.1.1.x or 4.2.0.x or 4.2.1.x Express Edition to 4.2.3 Standard Edition or later on Debian and Ubuntu, update the existing packages and install new packages by issuing the following command.

  ```
  dpkg -i gpfs*deb
  ```

- To migrate from 4.2.2.x Express Edition to 4.2.3 Standard Edition or later on Debian and Ubuntu, use these steps.

1. Uninstall the Express Edition license package by issuing the following command.

   ```
   dpkg -r gpfs.license.exp*deb
   ```

2. Update the existing packages and install new packages by issuing the following command.

   ```
   dpkg -i gpfs*deb
   ```

AIX

- Depending on whether you want to be able to revert to the previous version without reinstalling or uninstalling or not, do one of the following sets of steps.
  - If you want to be able to revert to previous version without reinstalling or uninstalling, do one of following sets of steps:
    - To migrate from 4.2.0.x or 4.2.1.x Express Edition to 4.2.3 Standard Edition or later on AIX, use these steps.
      1. Update existing packages to 4.2.3 level or a later version by using the PTF images on FixCentral.
      2. Install additional packages from CD images.
    - To migrate from 4.2.2.x Express Edition to 4.2.3 Standard Edition on AIX, use these steps.
      1. Update existing packages to 4.2.3 level or a later version by using the PTF images on FixCentral.
      2. Install additional packages from CD images.
      3. After a successful update, uninstall the `gpfs.license.exp` package.

    To revert to the previous version, do the following steps:

    1. Uninstall all additional packages installed from CD images.
    2. Reject the PTF images.
    3. If you are reverting to 4.2.2.x and you had uninstalled the `gpfs.license.exp` package for migrating to Standard Edition, install the `gpfs.license.exp` package.
  - If you can reinstall or uninstall to revert to the previous version, do one of the following sets of steps:
    - To migrate from 4.2.0.x or 4.2.1.x Express Edition to 4.2.3 Standard Edition or later on AIX, install 4.2.3 packages from CD images.
    - To migrate from 4.2.2.x Express Edition to 4.2.3 Standard Edition or later on AIX, use these steps.
      1. Install 4.2.3 or a later version packages from CD images.
      2. After a successful update, uninstall the `gpfs.license.exp` package.

Windows

- To migrate from 4.2.2.x or earlier Express Edition to Standard Edition 4.2.3 or later on Windows, use these steps.
  1. Uninstall the current IBM Spectrum Scale packages (For example, `gpfs.base-4.x.x.0-Windows.msi`, `gpfs.gskit-8.0.x.x.msi`, and `gpfs.x.license.msi`).
  2. Reboot the system.
  3. Install `gpfs.ext-4.x.x.-Windows-license.msi`, `gpfs.ext-4.x.x.0-Windows.msi`, and `gpfs.gskit-8.0.x.x.msi` packages.

Completing the upgrade to a new level of IBM Spectrum Scale

It is a good idea to use the cluster for a while with the new level of IBM Spectrum Scale installed, until you are sure that you are ready to permanently upgrade the cluster to the new level.

When you are ready to permanently upgrade the cluster, follow the steps in this topic to complete the upgrade. If you decide not to complete the upgrade, you can revert to the previous level of IBM Spectrum Scale. For more information, see “Reverting to the previous level of IBM Spectrum Scale” on page 422.

Before you begin this task, verify that you have upgraded all the nodes in the cluster to the latest licensed version of IBM Spectrum Scale.

When you run `mmchconfig release=LATEST` in Step 2 of these directions, you can add other parameters and their values to the command line:

```
mmchconfig release=LATEST <parameterN=value> <parameterN+1=value>...
```

The following table describes the parameters that are referred to in this topic.

Table 45. Other parameters with **mmchconfig release=LATEST**

Parameter	Purpose	Comment
--accept-empty-cipherlist-security	You must specify this parameter if you want to continue running the cluster with the lowest level of security for communications between nodes or with other clusters. That is, you want the **cipherList** attribute to remain set to **EMPTY**or **undefined**. For more information, see the topic *Security mode* in the *IBM Spectrum Scale: Administration Guide*	It is a good idea to have some level of security for cluster communications: • Set **cipherList** to **AUTHONLY** or to a supported cipher: `mmchconfig cipherList=AUTHONLY` • Do not include the parameter **--accept-empty-cipherlist-security** when you run `mmchconfig release=LATEST.` For more information, see the topic *Security mode* in the *IBM Spectrum Scale: Administration Guide*
mmfsLogTimeStampISO8601={**yes** | **no**}	Setting this parameter to **no** allows the cluster to continue running with the earlier log time stamp format. For more information, see the topic *Security mode* in the *IBM Spectrum Scale: Administration Guide*	• Set **mmfsLogTimeStampISO8061** to **no** if you save log information and you are not yet ready to switch to the new log time stamp format. After you complete the upgrade, you can change the log time stamp format at any time with the **mmchconfig** command. • Omit this parameter if you are ready to switch to the new format. The default value is **yes**

1. Verify that the SHA message digest and the **cipherList** configuration variable are set to valid values.

   Note:

   - The SHA message digest is a hash result that is generated by a cryptographic hash function.
   - The **cipherList** variable specifies the security mode for communications among nodes in the cluster and with nodes in other clusters.

   Follow these steps:

   a. Display the current values by entering the following command. The listing shows both the command and example output:

   ```
   # mmauth show .
   Cluster name:        zounds.cluster (this cluster)
   Cipher list:         (none specified)
   SHA digest:          (undefined)
   File system access:  (all rw)
   ```

   b. If the value for the `SHA digest` is `(undefined)`, follow these steps:

      1) Enter the following command to generate a public/private key pair and an SHA message digest:

      ```
      mmauth genkey new
      ```

2) Enter `mmauth show .` again and verify that the value for `SHA digest` is no longer `(undefined)`.

c. If the value for **cipherList** is `(none specified)` or `EMPTY`, do one of the following actions:

- If you want a level of security in communications between nodes and with other clusters, follow these steps:
  1) Set **cipherList** to AUTHONLY or to a supported cipher:
     ```
     mmauth update . -l AUTHONLY
     ```
  2) Enter `mmauth show .` again and verify that the value for **cipherList** is no longer `(none specified)` or `EMPTY`.
- If you do not want a level of security cluster communications, let **cipherList** remain set to `(none specified)` or `EMPTY`.

2. Enter the following command to upgrade the cluster configuration data, upgrade the file systems, and enable new functionality. You can add the parameters that are described in Table 45 on page 419 to the command line:
   ```
   mmchconfig release=LATEST
   ```

   Note: Until you run the **`mmchconfig release=LATEST`** command, the management GUI might not be fully operational at the new code level.

   Important: If the **mmchconfig** command detects any nodes that are not available or cannot be reached, it lists the names of those nodes. If any such nodes are listed, correct the problem and run the command again until it verifies all the nodes and completes successfully.

   Note: If the **mmchconfig** command fails with an error message that indicates that **cipherlist** is set to **EMPTY**, do one of the following actions:

   - If you want the cluster to run with a higher security mode than **EMPTY**, set **cipherList** to `AUTHONLY` or to a supported cipher:
     ```
     mmauth update . -l AUTHONLY
     ```

     Return to the first part of Step 2 and run the **mmchconfig** command as before.
   - If you want the cluster to continue with the security mode set to **EMPTY**, return to the first part of Step 2 and run the **mmchconfig** command with the additional parameter **--accept-empty-cipherlist-security**.

3. If you have not already done so, assign an appropriate GPFS license to each of the nodes in the cluster. See "IBM Spectrum Scale license designation" on page 129 for a detailed description of the GPFS license types. To see what the minimum required GPFS license is for each of the nodes in the cluster, enter the following command:
   ```
   mmlslicense -L
   ```

   To assign a GPFS server license to the nodes that require it, enter the following command:
   ```
   mmchlicense server -N NodeList
   ```

   To assign a GPFS client license to the nodes that require it, enter:
   ```
   mmchlicense client -N NodeList
   ```

4. Enable backward-compatible format changes or upgrade all file systems to the latest metadata format changes.

Attention: Before you continue with this step, it is important to understand the differences between **mmchfs -V compat** and **mmchfs -V full**:

- If you enter **mmchfs -V compat**, only changes that are backward compatible with GPFS 3.5 are enabled. Nodes in remote clusters that are running GPFS 3.5 will still be able to mount the file system. Nodes running GPFS 3.4 or earlier will no longer be able to mount the file system. Running in the compatibility mode might prevent enablement of new functionality that relies on disk structures available only with the latest version.
- If you enter **mmchfs -V full**, all new functions that require different on-disk data structures are enabled. Nodes in remote clusters that run an older GPFS version will no longer be able to mount the file system. If any nodes that run an older GPFS version have mounted the file system at the time this command is entered, the **mmchfs** command fails. This consideration might also apply to minor releases within the same major release. For example, release 4.1.0.x and release 4.1.1.x might have different metadata formats.

To enable backward-compatible format changes, enter the following command:

```
mmchfs FileSystem -V compat
```

To upgrade all file systems to the latest metadata format changes, enter the following command:

```
mmchfs FileSystem -V full
```

Certain new file system features might require more processing that cannot be handled by the **mmchfs -V** command alone. To fully activate such features, in addition to **mmchfs -V**, you must also run the **mmmigratefs** command. An example of such a feature is enabling fast extended attributes for file systems older than GPFS 3.4.

Note: The first mount of a file system after you run **mmchfs -V** might fail with a no-disk-space error. This situation might occur if the file system is relatively low on metadata disk space (10% or less free). If so, enter the command again. Typically the file system is mounted without a problem after the initial failed mount.

After completing the upgrade, you might need to enable cluster configuration repository (CCR) and fast extended attributes (fastea).

5. Enable cluster configuration repository (CCR) and fast extended attributes (fastea) as follows, if required.
   a. Enable the CCR in the cluster as follows.

      ```
      mmchcluster --ccr-enable
      ```

   b. Enable fastea in the cluster as follows.
      1) Check if fastea is enabled.

         ```
         mmlsfs FileSystemName --fastea
         ```

         A sample output is as follows.

         ```
         flag                value                    description
         ------------------- ------------------------ -----------------------------------
          --fastea           Yes                      Fast external attributes enabled?
         ```

      2) If it is not enabled, issue the following command to enable fastea.

         ```
         mmmigratefs FileSystemName --fastea
         ```

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

   Note: The following features are a few among several IBM Spectrum Scale features that require fast extended attributes (fastea) to be enabled to work.
   - Clones
   - Independent filesets
   - Encryption
   - Active file management (AFM)

6. If you use the **mmbackup** command to back up your file system and you have not done a full backup since GPFS 3.3 or later, a full backup might be necessary. For more information, see *File systems backed up using GPFS 3.2 or earlier versions of mmbackup* in *IBM Spectrum Scale: Administration Guide*.

Reverting to the previous level of IBM Spectrum Scale

If you decide not to continue the upgrade to the latest level of GPFS, and you have not yet issued the **mmchfs -V** command, you can reinstall the earlier level of GPFS.

Important: Once a file system has been upgraded explicitly by issuing the **mmchfs -V full** command, the disk images can no longer be read by a prior version of GPFS. You will be required to re-create the file system from the backup media and restore the content if you choose to go back after this command has been issued. The same rules apply for file systems that are newly created with GPFS 5.0.

You can revert back to GPFS 4.2.x.

If you have performed backups with the **mmbackup** command using the 5.0 version and decide to revert to an earlier version, you must rebuild the **mmbackup** shadow database using the **mmbackup** command with either the **-q** or **--rebuild** option.

The procedure differs depending on whether you have issued the **mmchconfig release=LATEST** command or not.

Reverting to a previous level of GPFS when you have *not* issued mmchconfig release=LATEST

If you have **not** issued the **mmchconfig release=LATEST** command, perform these steps.

1. Stop all user activity in the file systems.
2. Cleanly unmount all mounted GPFS file systems. Do not use force unmount.
3. Stop GPFS on all nodes in the cluster:
   ```
   mmshutdown -a
   ```
4. Run the appropriate uninstallation program to remove GPFS from each node in the cluster. For example:
   - For Linux nodes (this example is only applicable on the IBM Spectrum Scale Standard Edition):
     ```
     rpm -e gpfs.gpl gpfs.license.std gpfs.base gpfs.docs gpfs.gskit gpfs.msg.en_US gpfs.ext gpfs.compression
     ```
   - For AIX nodes:
     ```
     installp -u gpfs
     ```
   - For Windows nodes, open the **Programs and Features** control panel and remove **IBM General Parallel File System**.

   For the remaining steps, see IBM Spectrum Scale in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/ibmspectrumscale_welcome.html), and search for the appropriate *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* for your release.
5. Copy the installation images of the previous GPFS licensed program on all affected nodes.
6. Install the original install images and all required PTFs.
7. For Linux nodes running GPFS, you must rebuild the GPFS portability layer.
8. Reboot all nodes.

Reverting to a previous level of GPFS when you *have* issued mmchconfig release=LATEST

If you *have* issued the **mmchconfig release=LATEST** command, you must rebuild the cluster. Perform these steps.

1. Stop all user activity in the file systems.
2. Cleanly unmount all mounted GPFS file systems. Do not use force unmount.
3. Stop GPFS on all nodes in the cluster:

   ```
   mmshutdown -a
   ```
4. Export the GPFS file systems by issuing the **mmexportfs** command:

   ```
   mmexportfs all -o exportDataFile
   ```
5. Delete the cluster:

   ```
   mmdelnode -a
   ```
6. Run the appropriate uninstallation command to remove GPFS from each node in the cluster. For example:
   - For Linux nodes:

     ```
     rpm -e gpfs.gpl gpfs.license.xx gpfs.base gpfs.docs gpfs.msg.en_US gpfs.ext gpfs.compression
     ```
   - For AIX nodes:

     ```
     installp -u gpfs
     ```
   - For Windows nodes, open the **Programs and Features** control panel and remove **IBM General Parallel File System**.

   For the remaining steps, see IBM Spectrum Scale in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/ibmspectrumscale_welcome.html), and search for the appropriate *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* for your release.
7. Copy the installation images of the previous GPFS licensed program on all affected nodes.
8. Install the original installation images and all required PTFs.
9. For Linux nodes running GPFS, you must rebuild the GPFS portability layer.
10. Reboot all nodes.
11. Recreate your original cluster using the **mmcrcluster** command. Run the **mmchlicense** command to set the appropriate licenses after the cluster is created.
12. Use the **mmchconfig** command to restore any previously set configuration settings that are compatible with GPFS 4.2 or below.
13. Import the file system information using the **mmimportfs** command. Specify the file created by the **mmexportfs** command from Step 4:

    ```
    mmimportfs all -i exportDataFile
    ```
14. Start GPFS on all nodes in the cluster, issue:

    ```
    mmstartup -a
    ```
15. Mount the file systems if this is not done automatically when the GPFS daemon starts.

Coexistence considerations

Each GPFS cluster can have multiple GPFS file systems that coexist on the cluster, but function independently of each other. In addition, each file system might have different data management programs.

Note: The GPFS Data Management API (DMAPI) and GPFS file system snapshots can coexist; however, access to the files in a snapshot using DMAPI is restricted. For more information, see *IBM Spectrum Scale: Command and Programming Reference*.

Compatibility considerations

All applications that ran on the previous release of GPFS will run on the new level of GPFS. File systems that were created under the previous release of GPFS can be used under the new level of GPFS.

Considerations for IBM Spectrum Protect for Space Management

Upgrading to GPFS 3.3 or beyond requires consideration for IBM Spectrum Protect for Space Management. IBM Spectrum Protect for Space Management requires that all nodes in a cluster are configured with a DMAPI file handle size of 32 bytes.

During upgrade, it is possible to have older versions of GPFS that have a DMAPI file handle size of 16 bytes. Until all nodes in the cluster have been updated to the latest release and the DMAPI file handle size has been changed to 32 bytes, IBM Spectrum Protect for Space Management must be disabled. Run **mmlsconfig dmapiFileHandleSize** to see what value is set.

After all nodes in the cluster are upgraded to the latest release and you change the DMAPI file handle size to 32 bytes, you can enable IBM Spectrum Protect for Space Management.

The DMAPI file handle size is configured with the **dmapiFileHandleSize** option. For more information about this option, see *GPFS configuration attributes for DMAPI* in *IBM Spectrum Scale: Command and Programming Reference*.

GUI user role considerations

When the IBM Spectrum Scale is updated from 4.2.0.x to 4.2.1, no default group is created for the user role *User Administrator*.

If you need to assign *User Administrator* role to the admin user, you need to create a user group with *User Administrator* role. Perform the following steps to create a user group with *User Administrator* role:

1. Select **Access > GUI Users** in the IBM Spectrum Scale management GUI.
2. Click **Create Group** under **Groups** tab.
3. In the Create User Group window, enter the user group as *UserAdmin*.
4. Select user group role as **User Administrator**.

Now, you are ready to assign the User Administrator role to the admin users.

Note: This process is not required for fresh installation of IBM Spectrum Scale system. In case of fresh installation, the *User Administrator* role is granted to the group *UserAdmin* by default.

Applying maintenance to your GPFS system

Before applying maintenance to your GPFS system, there are several things you should consider.

Remember that:

1. There is limited interoperability between GPFS 5.0.x nodes and nodes running GPFS 4.2.y. This function is intended for short-term use. You will not get the full functions of GPFS 5.0.x until all nodes are using GPFS 5.0.x.
2. Interoperability between maintenance levels will be specified on a per-maintenance-level basis. See the IBM Spectrum Scale FAQ in IBM Knowledge Center(www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).
3. When applying maintenance, the GPFS file systems will need to be unmounted and GPFS shut down on the node being upgraded. Any applications using a GPFS file system should be shut down before applying maintenance to avoid application errors or possible system halts.

   Note: If the AIX Alternate Disk Install process is used, the new image is placed on a different disk. The running install image is not changed, and GPFS is not shut down. The new image is not used until the node is rebooted.

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

To download fixes, go to the IBM Support Portal: Downloads for General Parallel File System(www.ibm.com/support/entry/portal/Downloads/Software/Cluster_software/General_Parallel_File_System) and obtain the fixes for your hardware and operating system.

To install the latest fixes:

- For Linux nodes, see Chapter 4, “Installing IBM Spectrum Scale on Linux nodes and deploying protocols,” on page 223.
- For AIX nodes, see Chapter 5, “Installing IBM Spectrum Scale on AIX nodes,” on page 319.
- For Windows nodes, see Chapter 6, “Installing IBM Spectrum Scale on Windows nodes,” on page 323.

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-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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

Product Number: 5725-Q01
5641-GPF
5725-S28
5765-ESS

Printed in USA

GC27-9219-03