IBM Spectrum Scale RAID
Version 5.0.1

# *Administration*

SC27-9271-00

IBM Spectrum Scale RAID
Version 5.0.1

# *Administration*

SC27-9271-00

**Note**
Before using this information and the product it supports, read the information in "Notices" on page 297.

This edition applies to version 5 release 0 modification 1 of the following product and to all subsequent releases and modifications until otherwise indicated in new editions:
- IBM Spectrum Scale RAID (product number 5641-GRS)

IBM welcomes your comments; see the topic "How to submit your comments" on page xi. 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

# Figures

# Tables

# About this information

This information is intended as a guide for administering IBM Spectrum Scale™ RAID on Power Systems™ servers.

## Who should read this information

This information is intended for administrators of systems on Power Systems servers that include IBM Spectrum Scale RAID.

## Related information

### ESS information

The ESS 5.3.1 library consists of these information units:

- *Elastic Storage Server: Quick Deployment Guide*, SC27-9205
- *Elastic Storage Server: Problem Determination Guide*, SC27-9208
- *Elastic Storage Server: Command Reference*, SC27-9246
- *IBM Spectrum Scale RAID: Administration*, SC27-9206
- *IBM ESS Expansion: Quick Installation Guide (Model 084)*, SC27-4627
- *IBM ESS Expansion: Installation and User Guide (Model 084)*, SC27-4628
- *IBM ESS Expansion: Hot Swap Side Card - Quick Installation Guide (Model 084)*, GC27-9210
- *Installing the Model 024, ESLL, or ESLS storage enclosure*, GI11-9921
- *Removing and replacing parts in the 5147-024, ESLL, and ESLS storage enclosure*
- *Disk drives or solid-state drives for the 5147-024, ESLL, or ESLS storage enclosure*
- For information about the DCS3700 storage enclosure, see:
  - *System Storage® DCS3700 Quick Start Guide*, GA32-0960-04:
  - *IBM® System Storage DCS3700 Storage Subsystem and DCS3700 Storage Subsystem with Performance Module Controllers: Installation, User's, and Maintenance Guide*, GA32-0959-07:

    **http://www.ibm.com/support/docview.wss?uid=ssg1S7004920**
- For information about the IBM Power Systems EXP24S I/O Drawer (FC 5887), see IBM Knowledge Center :

  **http://www.ibm.com/support/knowledgecenter/8247-22L/p8ham/p8ham_5887_kickoff.htm**

For more information, see IBM Knowledge Center:

http://www-01.ibm.com/support/knowledgecenter/SSYSP8_5.3.1/sts531_welcome.html

For the latest support information about IBM Spectrum Scale RAID, see the IBM Spectrum Scale RAID FAQ in IBM Knowledge Center:

**http://www.ibm.com/support/knowledgecenter/SSYSP8/gnrfaq.html**

### Switch information

ESS release updates are independent of switch updates. Therefore, it is recommended that Ethernet and Infiniband switches used with the ESS cluster be at their latest switch firmware levels. Customers are responsible for upgrading their switches to the latest switch firmware. If switches were purchased through IBM, review the minimum switch firmware used in validation of this ESS release available in

*Customer networking considerations* section of *Elastic Storage Server: Quick Deployment Guide*.

### Other related information

For information about:

- IBM Spectrum Scale, see IBM Knowledge Center:
  **http://www.ibm.com/support/knowledgecenter/STXKQY/ibmspectrumscale_welcome.html**
- IBM Spectrum Scale call home, see Understanding call home.
- IBM POWER8® servers, see IBM Knowledge Center:
  **http://www.ibm.com/support/knowledgecenter/POWER8/p8hdx/POWER8welcome.htm**
- Extreme Cluster/Cloud Administration Toolkit (xCAT), go to the xCAT website :
  **http://xcat.org/**
- Mellanox OFED Release Notes:
  - 4.3: https://www.mellanox.com/related-docs/prod_software/Mellanox_OFED_Linux_Release_Notes_4_3-1_0_1_0.pdf
  - 4.1: https://www.mellanox.com/related-docs/prod_software/Mellanox_OFED_Linux_Release_Notes_4_1-1_0_2_0.pdf

## Conventions used in this information

Table 1 describes the typographic conventions used in this information. UNIX file name conventions are used throughout this information.

*Table 1. Conventions*

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

*Table 1. Conventions (continued)*

| Convention | Usage |
|---|---|
| *item...* | Ellipses indicate that you can repeat the preceding item one or more times. |
| \| | In *synopsis* statements, vertical lines separate a list of choices. In other words, a vertical line means *Or*.<br><br>In the left margin of the document, vertical lines indicate technical changes to the information. |

## How to submit your comments

Your feedback is important in helping us to produce accurate, high-quality information. You can add comments about this information in IBM Knowledge Center:

**http://www.ibm.com/support/knowledgecenter/SSYSP8/sts_welcome.html**

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

**scale@us.ibm.com**

# Summary of changes

This topic summarizes changes to the IBM Spectrum Scale RAID information. A vertical line (|) to the left of text and illustrations indicates technical changes or additions made to the previous edition of the information.

**Summary of changes for IBM Spectrum Scale RAID version 5.0.1.1 as updated, June 2018**

Changes to this release include the following:

**Managing IBM Spectrum Scale RAID with the `mmvdisk` command**

A new section has been added that introduces the **`mmvdisk`** command for managing IBM Spectrum Scale RAID.

The **`mmvdisk`** command simplifies IBM Spectrum Scale RAID administration. It also enforces and encourages consistent best practices for IBM Spectrum Scale RAID servers, recovery groups, vdisk NSDs, and file systems.

**Commands**

The following lists the modifications to the commands:

**New commands**

The following commands are new:

- **`mmvdisk`**
- **`mmvdisk nodeclass`**
- **`mmvdisk server`**
- **`mmvdisk recoverygroup`**
- **`mmvdisk vdiskset`**
- **`mmvdisk filesystem`**
- **`mmvdisk pdisk`**
- **`mmvdisk vdisk`**

**Changed commands**

The following commands, command descriptions, or both were changed:

- **`mmchpdisk`**

**GUI changes**

The following are the main changes in this release:

- Prior to the ESS 5.3.1 release, the Create File System wizard in the **Files** > **File Systems** page supported creating file systems with a system pool that can store only metadata. You can now configure the system pool for storing both data and metadata while creating file systems.
- On mmvdisk-enabled ESS clusters, the GUI disk replacement procedure uses the **`mmvdisk`** command instead of the **`mmchcarrier`** command.

**Messages**

No changes.

**Summary of changes for IBM Spectrum Scale RAID version 5.0.0 as updated, March 2018**

Changes to this release include the following:

**IBM Spectrum Scale RAID management API**

Added the following API endpoints:

- GET /gnr/recoverygroups
- GET /gnr/recoverygroups/{recoveryGroupName}
- GET /gnr/recoverygroups/{recoveryGroupName}/pdisks
- GET /gnr/recoverygroups/{recoveryGroupName}/pdisks/{pdiskName}
- GET /gnr/recoverygroups/{recoveryGroupName}/vdisks
- GET /gnr/recoverygroups/{recoveryGroupName}/vdisks/{vdiskName}

For more information on the newly added endpoints, see Chapter 9, "IBM Spectrum Scale RAID management API," on page 101.

**Commands**

The following lists the modifications to the commands:

**Changed commands**

The following commands, command descriptions, or both were changed:

- **mmchfirmware**
- **mmchpdisk**

**Messages**

The following new messages are added in this release:

**New messages**

6027-3845, 6027-3846, 6027-3847, 6027-3848, and 6027-3849

**Summary of changes for IBM Spectrum Scale RAID version 4.2.3 as updated, May 2017**

Changes to this release include the following:

**Commands**

The following lists the modifications to the commands:

**Changed commands**

The following commands, command descriptions, or both were changed:

- **mmchfirmware**
- **mmchpdisk**

**Summary of changes for IBM Spectrum Scale RAID version 4.2.2 as updated, January 2017**

Changes to this release include the following:

**Commands**

The following lists the modifications to the commands:

**Changed commands**

The following commands, command descriptions, or both were changed:

- **mmchfirmware**
- **mmlsfirmware**

**Scripts**

The following lists the modifications to the scripts:

**Changed scripts**

The following scripts, script descriptions, or both were changed:

- **mkrginput**
- **topselect**
- **topsummary**

**Messages**

The following lists the new messages that are added in this release:

**New messages**

6027-3812, 6027-3813, 6027-3814, 6027-3815, 6027-3816, 6027-3817, 6027-3818, 6027-3819, 6027-3820, 6027-3821, 6027-3822, 6027-3823, 6027-3824, 6027-3825, 6027-3826, 6027-3827, 6027-3828, 6027-3829, 6027-3830, 6027-3831, 6027-3832, 6027-3833, 6027-3836, 6027-3837, 6027-3838, 6027-3839, 6027-3840, 6027-3841, 6027-3843, 6027-3844

**Changed messages**

6027-1860, 6027-1861, 6027-1864, 6027-1876, 6027-3016, 6027-3041, 6027-3091, 6027-3092, 6027-3802, 6027-3804, 6027-3805, 6027-3806, 6027-3807, 6027-3808, 6027-3809, 6027-3811

**Deleted messages**

6027-1859, 6027-3803

**Documentation changes**

The following main documentation updates are done in this release:

- Modified the monitoring information to add detailed documentation on system health monitoring, performance monitoring, and capacity monitoring.
- Restructured the Troubleshooting section to add more information about the troubleshooting procedures and also to align the troubleshooting information with the standards set for other IBM storage products.

## Summary of changes for IBM Spectrum Scale RAID version 4 release 5.0 as updated, August 2016

Changes to this release of IBM Spectrum Scale RAID include the following:

**ESS core**

- IBM Spectrum Scale RAID version 4.2.1-0 efix2
- Updated GUI
- Changes from ESS 4.0.x

**Support of Red HatEnterprise Linux 7.1**

- No changes from ESS 3.0, 3.5, and 4.0 (see those sections in this document for more information).
- Change in ESS (same as ESS 4.0.5) kernel release 3.10.0-229.34.1.el7.ppc64

**Support of MLNX_OFED_LINUX-3.3-1.0.4.1**

- Updated from MLNX_OFED_LINUX-3.2-2.0.0.1 (ESS 4.0.3, 4.0.5)
- Updated from MLNX_OFED_LINUX-3.1-1.0.6.1 (ESS 4.0, 4.0.1, 4.0.2)
- Updated from MLNX_OFED_LINUX-3.1-1.0.0.2 (ESS 3.5.x)
- Updated from MLNX_OFED_LINUX-2.4-1.0.2 (ESS 3.0.x)

- Support for PCIe3 LP 2-port 100 Gb EDR InfiniBand adapter x16 (FC EC3E)
  - Requires System FW level FW840.20 (SV840_104)
  - No changes from ESS 4.0.3

**Install Toolkit**
- Updated Install Toolkit
- Updates from ESS 4.0.x

**Updated firmware rpm**
- Updated firmware for IBM PCIe x8 Cache SAS RAID Internal Adapter
- Support for updated drive FW
- Changes from ESS 4.0.5

**Summary of changes for IBM Spectrum Scale RAID version 4 release 2.0.3 as updated, May 2016**

Changes to this release of IBM Spectrum Scale RAID include the following:

**Commands**

The following lists the modifications to the commands:

**Changed commands**

The following commands, command descriptions, or both were changed:
- **mmdelcomploc**
- **mmlsfirmware**

**Scripts**

The following lists the modifications to the scripts:

**Changed scripts**

The following scripts, script descriptions, or both were changed:
- **gnrhealthcheck**

**Summary of changes for IBM Spectrum Scale RAID version 4 release 2.0 as updated, January 2016**

Changes to this release of IBM Spectrum Scale RAID include the following:

**Commands**

The following lists the modifications to the commands:

**Changed commands**

The following commands, command descriptions, or both were changed:
- **mmchcarrier**
- **mmchrecoverygroup**
- **mmcrrecoverygroup**
- **mmlsfirmware**
- **mmlspdisk**

**Messages**

The following lists the new messages that are added in this release:

**New messages**

6027-3801, 6027-3802, 6027-3803, 6027-3804, 6027-3805, 6027-3806, 6027-3807, 6027-3808, 6027-3809, 6027-3810

**Summary of changes for IBM Spectrum Scale RAID version 4 release 1.1 as updated, October 2015**

Changes to this release of IBM Spectrum Scale RAID, the IBM Spectrum Scale RAID information, or both include the following:

**The ability to create recovery group stanza files with a single data (non-log) declustered array**

**Commands**

The following lists the modifications to the commands:

**Changed commands**

The following commands, command descriptions, or both were changed:

- **mmchenclosure**
- **mmlsenclosure**
- **mmlsfirmware**

**Scripts**

The following lists the modifications to the scripts:

**Changed scripts**

The following scripts, script descriptions, or both were changed:

- **mkrginput**

**Messages**

The following lists the changed messages:

**Changed messages**

The following messages, message descriptions, or both were changed:

6027-3045, 6027-3046, MS0103, MS0104, MS0105, MS0241, MS0242, MS0243

**Summary of changes for GPFS™ Native RAID version 4 release 1.0.8 as updated, May 2015**

Changes to this release of GNR, the GNR information, or both include the following:

**Documented commands**

The following lists the modifications to the documented commands:

**New commands**

The following commands are new:

- **mmaddcompspec**
- **mmdelcompspec**

**Changed commands**

The following commands, command descriptions, or both were changed:

- **mmaddcomp**
- **mmchcomp**
- **mmchcomploc**

- **mmchfirmware**
- **mmchrecoverygroup**
- **mmcrrecoverygroup**
- **mmdelcomp**
- **mmdelcomploc**
- **mmdiscovercomp**
- **mmlscomp**
- **mmlscomploc**
- **mmlscompspec**
- **mmlsfirmware**
- **mmlsrecoverygroup**
- **mmsyncdisplayid**

**Messages**

The following lists the new and changed messages:

**New messages**

6027-3095, 6027-3096, 6027-3097, 6027-3098, 6027-3099, 6027-3800

**Changed messages**

The following messages, message descriptions, or both were changed:

6027-3045, 6027-3046

# Chapter 1. Introducing IBM Spectrum Scale RAID

This topic describes the basic concepts, features, and functions of IBM Spectrum Scale RAID: redundancy codes, end-to-end checksums, data declustering, and administrator configuration, including recovery groups, declustered arrays, virtual disks, and virtual disk NSDs.

IBM Spectrum Scale RAID is a software implementation of storage RAID technologies within IBM Spectrum Scale. Using conventional dual-ported disks in a JBOD configuration, IBM Spectrum Scale RAID implements sophisticated data placement and error-correction algorithms to deliver high levels of storage reliability, availability, and performance. Standard GPFS file systems are created from the NSDs defined through IBM Spectrum Scale RAID.

IBM Spectrum Scale RAID is available with the IBM Elastic Storage Server (ESS) 5.1 for Power®. ESS is a high-capacity, high-performance storage solution that combines IBM Power Systems servers, storage enclosures, drives, software (including IBM Spectrum Scale RAID), and networking components. ESS uses a building-block approach to create highly-scalable storage for use in a broad range of application environments.

## Overview

IBM Spectrum Scale RAID integrates the functionality of an advanced storage controller into the GPFS NSD server. Unlike an external storage controller, where configuration, LUN definition, and maintenance are beyond the control of IBM Spectrum Scale, IBM Spectrum Scale RAID itself takes on the role of controlling, managing, and maintaining physical disks - hard disk drives (HDDs) and solid-state drives (SSDs).

Sophisticated data placement and error correction algorithms deliver high levels of storage reliability, availability, serviceability, and performance. IBM Spectrum Scale RAID provides a variation of the GPFS network shared disk (NSD) called a *virtual disk,* or *vdisk*. Standard NSD clients transparently access the vdisk NSDs of a file system using the conventional NSD protocol.

The features of IBM Spectrum Scale RAID include:

- **Software RAID**

  IBM Spectrum Scale RAID, which runs on standard Serial Attached SCSI (SAS) disks in a dual-ported JBOD array, does not require external RAID storage controllers or other custom hardware RAID acceleration.
- **Declustering**

  IBM Spectrum Scale RAID distributes client data, redundancy information, and spare space uniformly across all disks of a JBOD. This approach reduces the rebuild (disk failure recovery process) overhead and improves application performance compared to conventional RAID.
- **Pdisk-group fault tolerance**

  In addition to declustering data across disks, IBM Spectrum Scale RAID can place data and parity information to protect against groups of disks that, based on characteristics of a disk enclosure and system, could possibly fail together due to a common fault. The data placement algorithm ensures that even if all members of a disk group fail, the error correction codes will still be capable of recovering erased data.
- **Checksum**

  An end-to-end data integrity check, using checksums and version numbers, is maintained between the disk surface and NSD clients. The checksum algorithm uses version numbers to detect silent data corruption and lost disk writes.

- **Data redundancy**

  IBM Spectrum Scale RAID supports highly reliable 2-fault-tolerant and 3-fault-tolerant Reed-Solomon-based parity codes and 3-way and 4-way replication.
- **Large cache**

  A large cache improves read and write performance, particularly for small I/O operations.
- **Arbitrarily-sized disk arrays**

  The number of disks is not restricted to a multiple of the RAID redundancy code width, which allows flexibility in the number of disks in the RAID array.
- **Multiple redundancy schemes**

  One disk array can support vdisks with different redundancy schemes, for example Reed-Solomon and replication codes.
- **Disk hospital**

  A disk hospital asynchronously diagnoses faulty disks and paths, and requests replacement of disks by using past health records.
- **Automatic recovery**

  Seamlessly and automatically recovers from primary server failure.
- **Disk scrubbing**

  A disk scrubber automatically detects and repairs latent sector errors in the background.
- **Familiar interface**

  Standard IBM Spectrum Scale command syntax is used for all configuration commands, including maintaining and replacing failed disks.
- **Flexible hardware configuration**

  Support of JBOD enclosures with multiple disks physically mounted together on removable carriers.
- **Journaling**

  For improved performance and recovery after a node failure, internal configuration and small-write data are journaled to solid-state disks (SSDs) in the JBOD or to non-volatile random-access memory (NVRAM) that is internal to the IBM Spectrum Scale RAID servers.

## IBM Spectrum Scale RAID features

This section introduces three key features of IBM Spectrum Scale RAID and how they work: data redundancy using RAID codes, end-to-end checksums, and declustering.

## RAID codes

IBM Spectrum Scale RAID corrects for disk failures and other storage faults automatically by reconstructing the unreadable data using the available data redundancy of a Reed-Solomon code or *N*-way replication. IBM Spectrum Scale RAID uses the reconstructed data to fulfill client operations, and in the case of disk failure, to rebuild the data onto spare space. IBM Spectrum Scale RAID supports 2- and 3-fault-tolerant Reed-Solomon codes and 3-way and 4-way replication, which respectively detect and correct up to two or three concurrent faults[1]. The redundancy code layouts that IBM Spectrum Scale RAID supports, called *tracks*, are illustrated in Figure 1 on page 3.

1. An $f$-fault-tolerant Reed-Solomon code or a $(1 + f)$-way replication can survive the *concurrent* failure of $f$ disks or read faults. Also, if there are $s$ equivalent spare disks in the array, an $f$-fault-tolerant array can survive the *sequential* failure of $f + s$ disks where disk failures occur between successful rebuild operations.

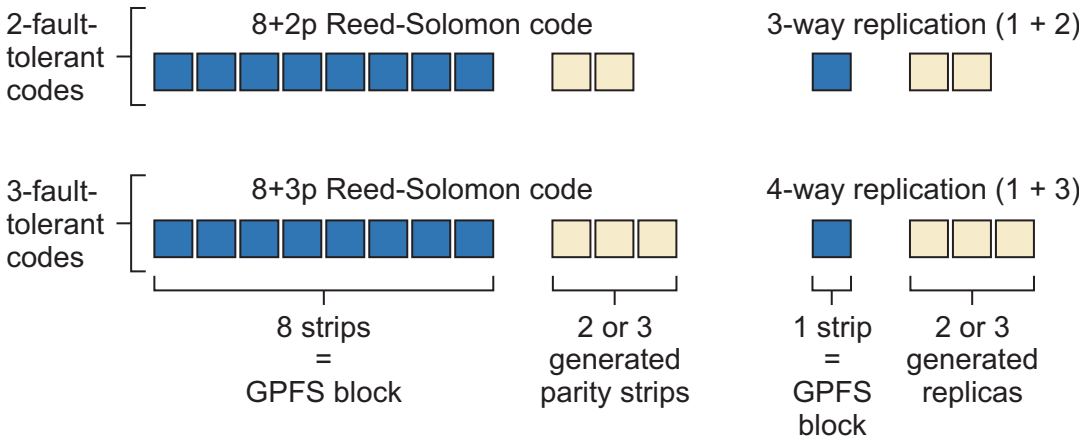



*Figure 1. Redundancy codes supported by IBM Spectrum Scale RAID.* IBM Spectrum Scale RAID supports 2- and 3-fault-tolerant Reed-Solomon codes, which partition a GPFS block into eight data strips and two or three parity strips. The *N*-way replication codes duplicate the GPFS block on *N* - 1 replica strips.

Depending on the configured RAID code, IBM Spectrum Scale RAID creates redundancy information automatically. Using a Reed-Solomon code, IBM Spectrum Scale RAID divides a GPFS block of user data equally into eight *data strips* and generates two or three redundant *parity strips*. This results in a stripe or track width of 10 or 11 strips and storage efficiency of 80% or 73%, respectively (excluding user-configurable spare space for rebuild operations).

Using *N*-way replication, a GPFS data block is replicated simply *N* − 1 times, in effect implementing 1 + 2 and 1 + 3 redundancy codes, with the strip size equal to the GPFS block size. Thus, for every block/strip that is written to the disks, *N* replicas of that block/strip are also written. This results in a track width of three or four strips and storage efficiency of 33% or 25%, respectively.

## End-to-end checksum

Most implementations of RAID codes implicitly assume that disks reliably detect and report faults, hard-read errors, and other integrity problems. However, studies have shown that disks do not report some read faults and occasionally fail to write data, while actually claiming to have written the data. These errors are often referred to as silent errors, phantom-writes, dropped-writes, and off-track writes. To cover for these shortcomings, IBM Spectrum Scale RAID implements an end-to-end checksum that can detect silent data corruption caused by either disks or other system components that transport or manipulate the data.

When an NSD client is writing data, a checksum of 8 bytes is calculated and appended to the data before it is transported over the network to the IBM Spectrum Scale RAID server. On reception, IBM Spectrum Scale RAID calculates and verifies the checksum. Then, IBM Spectrum Scale RAID stores the data, a checksum, and version number to disk and logs the version number in its metadata for future verification during read.

When IBM Spectrum Scale RAID reads disks to satisfy a client read operation, it compares the disk checksum against the disk data and the disk checksum version number against what is stored in its metadata. If the checksums and version numbers match, IBM Spectrum Scale RAID sends the data along with a checksum to the NSD client. If the checksum or version numbers are invalid, IBM Spectrum Scale RAID reconstructs the data using parity or replication and returns the reconstructed data and a newly generated checksum to the client. Thus, both silent disk read errors and lost or missing disk writes are detected and corrected.

## Declustered RAID

Compared to conventional RAID, IBM Spectrum Scale RAID implements a sophisticated data and spare space disk layout scheme that allows for arbitrarily sized disk arrays while also reducing the overhead to clients when recovering from disk failures. To accomplish this, IBM Spectrum Scale RAID uniformly spreads or *declusters* user data, redundancy information, and spare space across all the disks of a declustered array. Figure 2 compares a conventional RAID layout versus an equivalent declustered array.

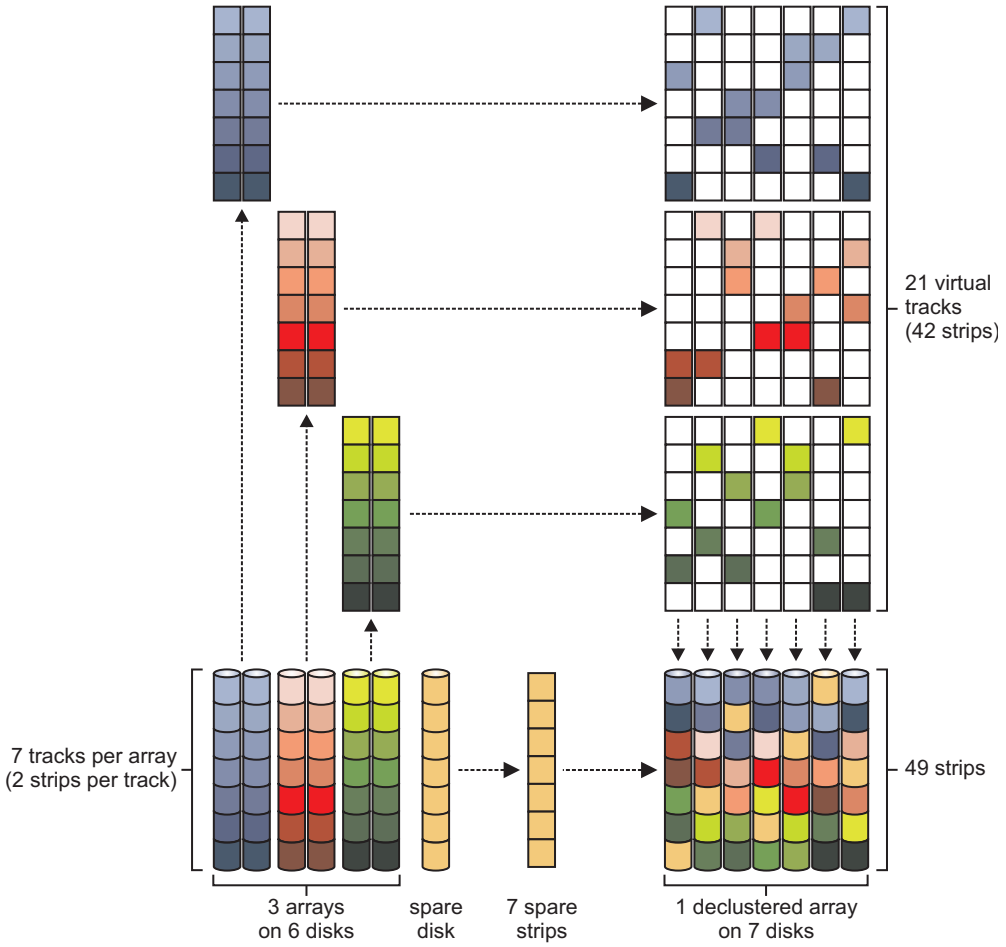



*Figure 2. Conventional RAID versus declustered RAID layouts.* This figure is an example of how IBM Spectrum Scale RAID improves client performance during rebuild operations by using the throughput of all disks in the declustered array. This is illustrated here by comparing a conventional RAID of three arrays versus a declustered array, both using seven disks. A conventional 1-fault-tolerant 1 + 1 replicated RAID array in the lower left is shown with three arrays of two disks each (data and replica strips) and a spare disk for rebuilding. To decluster this array, the disks are divided into seven tracks, two strips per array, as shown in the upper left. The strips from each group are then combinatorially spread across all seven disk positions, for a total of 21 virtual tracks, per the upper right. The strips of each disk position for every track are then arbitrarily allocated onto the disks of the declustered array of the lower right (in this case, by vertically sliding down and compacting the strips from above). The spare strips are uniformly inserted, one per disk.

As illustrated in Figure 3 on page 5, a declustered array can significantly shorten the time that is required to recover from a disk failure, which lowers the rebuild overhead for client applications. When a disk fails, erased data is rebuilt using all the operational disks in the declustered array, the bandwidth of which is greater than that of the fewer disks of a conventional RAID group. Furthermore, if an additional disk fault occurs during a rebuild, the number of impacted tracks requiring repair is markedly less than the previous failure and less than the constant rebuild overhead of a conventional array.

The decrease in declustered rebuild impact and client overhead can be a factor of three to four times less than a conventional RAID. Because IBM Spectrum Scale stripes client data across all the storage nodes of a cluster, file system performance becomes less dependent upon the speed of any single rebuilding storage array.

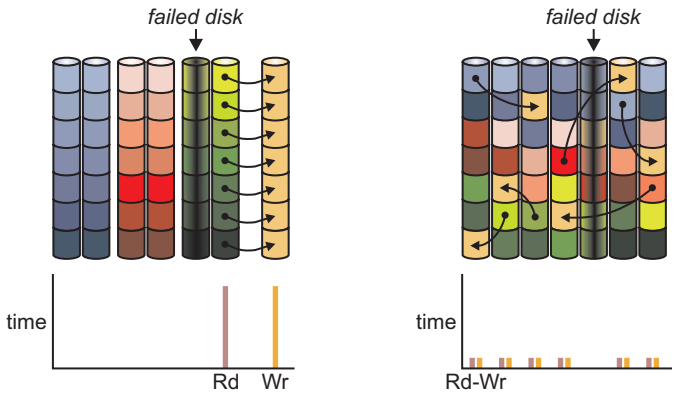



*Figure 3. Lower rebuild overhead in declustered RAID versus conventional RAID.* When a single disk fails in the 1-fault-tolerant 1 + 1 conventional array on the left, the redundant disk is read and copied onto the spare disk, which requires a throughput of 7 strip I/O operations. When a disk fails in the declustered array, all replica strips of the six impacted tracks are read from the surviving six disks and then written to six spare strips, for a throughput of two strip I/O operations. The bar chart illustrates disk read and write I/O throughput during the rebuild operations.

## Disk configurations

This section describes recovery group and declustered array configurations.

## Recovery groups

IBM Spectrum Scale RAID divides disks into *recovery groups* where each is physically connected to two servers: primary and backup. All accesses to any of the disks of a recovery group are made through the active server of the recovery group, either the primary or backup.

Building on the inherent NSD failover capabilities of IBM Spectrum Scale, when an IBM Spectrum Scale RAID server stops operating because of a hardware fault, software fault, or normal shutdown, the backup IBM Spectrum Scale RAID server seamlessly takes over control of the associated disks of its recovery groups.

Typically, a JBOD array is divided into two recovery groups controlled by different primary IBM Spectrum Scale RAID servers. If the primary server of a recovery group fails, control automatically switches over to its backup server. Within a typical JBOD, the primary server for a recovery group is the backup server for the other recovery group.


*Figure 4. Minimal configuration of two IBM Spectrum Scale RAID servers and one storage JBOD.* IBM Spectrum Scale RAID server 1 is the primary controller for the first recovery group and backup for the second recovery group. IBM Spectrum Scale RAID server 2 is the primary controller for the second recovery group and backup for the first recovery group. As shown, when server 1 fails, control of the first recovery group is taken over by its backup server 2. During the failure of server 1, the load on backup server 2 increases by 100% from one to two recovery groups.

## Declustered arrays

A declustered array is a subset of the physical disks (pdisks) in a recovery group across which data, redundancy information, and spare space are declustered. The number of disks in a declustered array is determined by the RAID code-width of the vdisks that will be housed in the declustered array. For more information, see "Virtual disks." There can be one or more declustered arrays per recovery group. Figure 5 illustrates a storage JBOD with two recovery groups, each with four declustered arrays.

A declustered array can hold one or more vdisks. Since redundancy codes are associated with vdisks, a declustered array can simultaneously contain both Reed-Solomon and replicated vdisks.

If the storage JBOD supports multiple disks physically mounted together on removable carriers, removal of a carrier temporarily disables access to all the disks in the carrier. Thus, pdisks on the same carrier should not be in the same declustered array, as vdisk redundancy protection would be weakened upon carrier removal.

Declustered arrays are normally created at recovery group creation time but new ones can be created or existing ones grown by adding pdisks at a later time.

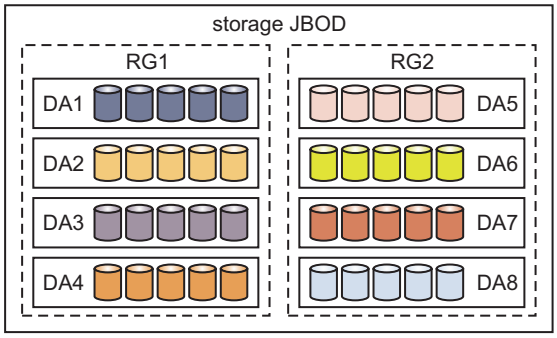



*Figure 5. Example of declustered arrays and recovery groups in storage JBOD.* This figure shows a storage JBOD with two recovery groups, each recovery group with four declustered arrays, and each declustered array with five disks.

## Virtual and physical disks

A virtual disk (vdisk) is a type of NSD, implemented by IBM Spectrum Scale RAID across all the physical disks (pdisks) of a declustered array. Multiple vdisks can be defined within a declustered array, typically Reed-Solomon vdisks for GPFS user data and replicated vdisks for GPFS metadata.

## Virtual disks

Whether a vdisk of a particular capacity can be created in a declustered array depends on its redundancy code, the number of pdisks and equivalent spare capacity in the array, and other small IBM Spectrum Scale RAID overhead factors. The **mmcrvdisk** command can automatically configure a vdisk of the largest possible size given a redundancy code and configured spare space of the declustered array.

In general, the number of pdisks in a declustered array cannot be less than the widest redundancy code of a vdisk plus the equivalent spare disk capacity of a declustered array. For example, a vdisk using the 11-strip-wide 8 + 3p Reed-Solomon code requires at least 13 pdisks in a declustered array with the equivalent spare space capacity of two disks. A vdisk using the 3-way replication code requires at least five pdisks in a declustered array with the equivalent spare capacity of two disks.

Vdisks are partitioned into virtual tracks, which are the functional equivalent of a GPFS block. All vdisk attributes are fixed at creation and cannot be subsequently altered.

### Physical disks

IBM Spectrum Scale RAID uses pdisks to store user data and IBM Spectrum Scale RAID internal configuration data.

A pdisk is a conventional, rotating magnetic-media hard disk drive (HDD) or a solid-state disk (SSD). All pdisks in a declustered array must have the same capacity.

It is assumed that pdisks are dual-ported. In this type of configuration, one or more paths are connected to the primary IBM Spectrum Scale RAID server and one or more paths are connected to the backup server. Typically, there are two redundant paths between an IBM Spectrum Scale RAID server and connected JBOD pdisks.

### Solid-state disks

Early versions of ESS used several solid-state disks (SSDs) in each recovery group in order to redundantly log changes to its internal configuration and fast-write data in non-volatile memory, accessible from the primary or backup IBM Spectrum Scale RAID servers after server failure.

Later versions of ESS typically use NVRAM, with a single SSD per recovery group that is only used as a backup for the NVRAM in case of failure.

## IBM Spectrum Scale RAID with pdisk-group fault tolerance

IBM Spectrum Scale RAID has a revised placement algorithm for distributing strips of the redundancy code. The revision can allow survival of larger units of concurrent disk failures than what was possible in previous versions of IBM Spectrum Scale RAID. The placement algorithm is aware of the hardware groupings of disks that are present in the system and attempts to segregate individual strips of a redundancy code stripe across as many groups as possible.

For example, if the hardware configuration includes four disk enclosures and a vdisk has been created with four-way replication, each strip of the vdisk's four-way stripe can be placed on a separate enclosure. Furthermore, if a complete enclosure (potentially many tens or hundreds of disks) were to fail, the surviving redundancy code strips on other enclosures would ensure no data loss. This revised placement is significantly different from the placement exhibited in previous versions of IBM Spectrum Scale RAID, which made no attempt to segregate strips across hardware groupings of disks and thus could have placed all four redundancy code strips within one enclosure. The loss of that one enclosure would cause the data to be unavailable.

IBM Spectrum Scale RAID uses redundancy codes that are user selected for user data and system selected for configuration data. The selected redundancy code, available disk space, and current disk hardware configuration all play a role with regard to which types of failures can be survived. IBM Spectrum Scale RAID selects a minimum of five-way replication for its internal configuration data and requires a certain amount of physical disk space to be available for describing the system. IBM Spectrum Scale RAID also discovers the disk hardware groups automatically and uses this discovery in a periodic rebalance of the redundancy code strips. If the disk hardware configuration changes (if a new disk enclosure is added to the recovery group, for example), IBM Spectrum Scale RAID recognizes the change automatically and performs a rebalancing operation automatically in the background. Additionally, a rebuild operation in the event of hardware failure is also cognizant of the hardware groupings, so failed redundancy code strips are rebuilt in a manner that is aware of the current disk hardware grouping.

## Pdisk-group fault tolerance: an example

Every data stripe (including user data and system configuration data) within the IBM Spectrum Scale RAID system is protected through a distinct form of redundancy. Each of these data stripes has a set of disks within which they constrain their strip placement. Each stripe of the data (for which there are many stripes in each whole) has individual strips that serve in the redundancy code protection of the object's data. The placement of these strips has been distributed across a set of pdisks residing within a set of drawers. These drawers reside within a set of enclosures.

Figure 6 shows a sample stripe placement for a vdisk that was using a RAID redundancy code of 4WayReplication (that is, four duplicate copies of each data strip). The pdisk-group fault-tolerant placement has chosen to place the four strips of the stripe across four drawers in the two enclosures available to this recovery group.
By segregating each individual strip across as wide a set of disk groups as possible, IBM Spectrum Scale

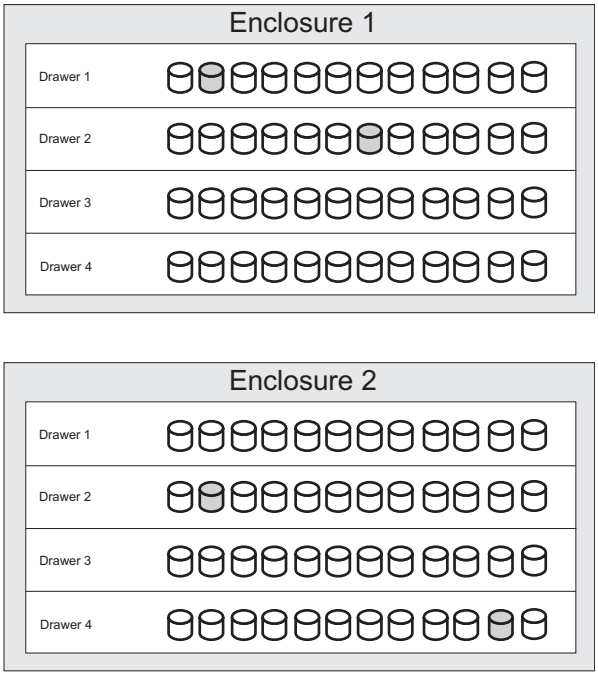



*Figure 6. Strips across JBOD enclosures*

RAID ensures that the loss of any set of disk groups up to fault tolerance of the RAID redundancy code is survivable.

Figure 7 on page 9 shows an example of the same configuration after the loss of a full enclosure and one drawer from the second enclosure.

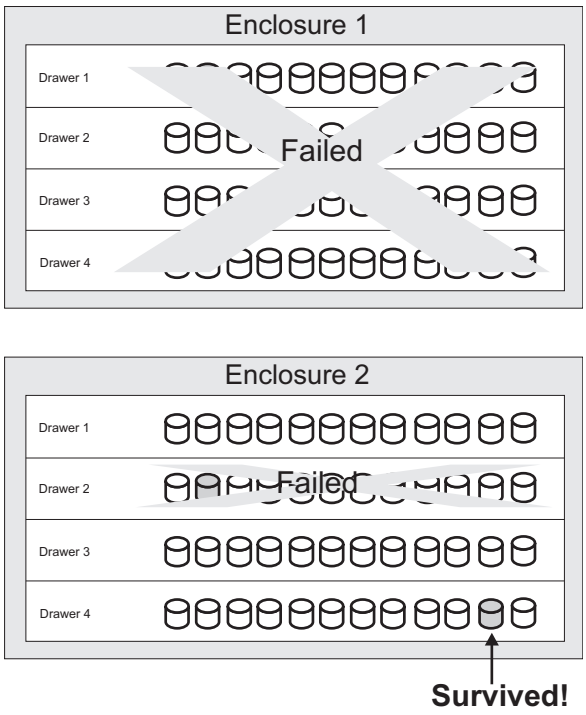



*Figure 7. Strips across JBOD enclosures after failure*

In this example, the pdisk-group fault-tolerant placement of individual strips across multiple enclosures and multiple drawers has ensured that at least one of the four duplicate copies has survived the multiple disk failures that occurred when an enclosure and a separate drawer failed.

## Disk hospital

The disk hospital is a key feature of IBM Spectrum Scale RAID that asynchronously diagnoses errors and faults in the storage subsystem. IBM Spectrum Scale RAID times out an individual pdisk I/O operation after about ten seconds, thereby limiting the impact from a faulty pdisk on a client I/O operation. When a pdisk I/O operation results in a timeout, an I/O error, or a checksum mismatch, the suspect pdisk is immediately admitted into the disk hospital. When a pdisk is first admitted, the hospital determines whether the error was caused by the pdisk itself or by the paths to it. While the hospital diagnoses the error, IBM Spectrum Scale RAID, if possible, uses vdisk redundancy codes to reconstruct lost or erased strips for I/O operations that would otherwise have used the suspect pdisk.

## Health metrics

The disk hospital maintains the following internal health assessment metrics for each pdisk. When one of these metrics exceeds the threshold, the pdisk is marked for replacement according to the disk maintenance replacement policy for the declustered array.

**relativePerformance**
Characterizes response times. Values greater than one indicate that the disk is performing above average speed; values less than one indicate that the disk is performing below average speed. Values within a range of 0.800 to 1.250 are considered normal. Examples of typical values are: 0.932, 0.978, 1.039, and 1.095. If the **relativePerformance** of a disk falls below a particular threshold (the default setting is 0.667), the hospital adds “slow” to the pdisk state, and the disk is prepared for replacement.

**dataBadness**
Characterizes media errors (hard errors) and checksum errors.

The disk hospital logs selected Self-Monitoring, Analysis and Reporting Technology (SMART) data, including the number of internal sector remapping events for each pdisk.

## Pdisk discovery

IBM Spectrum Scale RAID discovers all connected pdisks when it starts up, and then regularly schedules a process that will rediscover a pdisk that becomes newly accessible to the IBM Spectrum Scale RAID server. This allows pdisks to be physically connected or connection problems to be repaired without restarting the IBM Spectrum Scale RAID server.

## Disk replacement recording and reporting

The disk hospital keeps track of disks that require replacement according to the disk replacement policy of the declustered array, and it can be configured to report the need for replacement in a variety of ways. It records and reports the FRU number and physical hardware location of failed disks to help guide service personnel to the correct location with replacement disks.

If the storage JBOD supports multiple disks that are mounted on a removable carrier, such as the Power 775, disk replacement requires the hospital to suspend other disks in the same carrier temporarily. On the Power 775 storage JBOD, the disk carriers are also not removable until IBM Spectrum Scale RAID actuates a solenoid-controlled latch, in order to guard against human error.

In response to administrative commands, the hospital quiesces the appropriate disk (or multiple disks on a carrier), releases the carrier latch solenoid (if necessary), and turns on identify lights to guide replacement. After one or more disks are replaced and the disk or carrier is re-inserted, the hospital, in response to administrative commands, verifies that the repair has taken place and adds any new disks to the declustered array automatically, which causes IBM Spectrum Scale RAID to rebalance the tracks and spare space across all the disks of the declustered array. If service personnel fail to re-insert the disk or carrier within a reasonable period, the hospital declares the disks missing and starts rebuilding the affected data.

# Chapter 2. Administering IBM Spectrum Scale RAID

Before you perform IBM Spectrum Scale RAID administration tasks, review information about getting started with IBM Spectrum Scale, requirements for administering IBM Spectrum Scale RAID, and common command principles.

For information about getting started with IBM Spectrum Scale, see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

For information about the administration and maintenance of IBM Spectrum Scale and your GPFS file systems, see the *IBM Spectrum Scale: Administration Guide*.

## Requirements for administering IBM Spectrum Scale RAID

Root authority is required to perform all IBM Spectrum Scale RAID administration tasks.

IBM Spectrum Scale RAID commands maintain the appropriate environment across all nodes in the GPFS cluster. To achieve this, IBM Spectrum Scale RAID commands use the remote shell and remote file copy commands that you specify on the **mmcrcluster** command or the **mmchcluster** command. See the *IBM Spectrum Scale: Command and Programming Reference* for more information.

The default remote commands are **rsh** and **rcp**, but you can designate **ssh** and **scp** or any other remote commands with compatible syntax. The **rsh** and **rcp** commands that are provided by the Windows Cygwin environment do not support IBM Spectrum Scale. If your cluster includes Windows nodes, you must designate **ssh** and **scp** as the remote communication program.

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

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

## Common IBM Spectrum Scale RAID command principles

There are some common principles that you should keep in mind when you are running IBM Spectrum Scale RAID commands.

These principles include:

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

## Specifying nodes as input to IBM Spectrum Scale RAID commands

Many IBM Spectrum Scale RAID commands accept a node or multiple nodes as part of their input, using the **-N** flag. Nodes can be specified with IBM Spectrum Scale RAID commands in a variety of ways:

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

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

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

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

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

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

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

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

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

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

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

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

User-defined node classes are created with the **mmcrnodeclass** command. After a node class is created, it can be specified as an argument on commands that accept the **-N** *NodeClass* option. User-defined node classes are managed with the **mmchnodeclass**, **mmdelnodeclass**, and **mmlsnodeclass** commands. See the *IBM Spectrum Scale: Command and Programming Reference* for more information.

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

For commands operating on a file system, the stripe group manager node is always implicitly included in the node list. Not every IBM Spectrum Scale RAID command supports all of the node specification

options described in this topic. To learn which kinds of node specifications are supported by a particular IBM Spectrum Scale RAID command, see the relevant command description in Appendix B, "IBM Spectrum Scale RAID commands," on page 129.

## Stanza files

The input to a number of IBM Spectrum Scale RAID commands can be provided in a file organized in a stanza format.

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

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

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

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

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

IBM Spectrum Scale recognizes a number of stanzas:

**%nsd:**
NSD stanza

**%pdisk:**
Physical disk stanza

**%vdisk:**
Virtual disk stanza

**%da:**
Declustered array stanza

**%rg:**
Recovery group stanza

The details are documented under the corresponding commands.

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

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

Here is what a stanza file may look like:

```
# Sample file containing two NSD stanzas

# Example for an NSD stanza with imbedded comments
%nsd:  nsd=DATA5     # my name for this NSD
 device=/dev/hdisk5  # device name on node k145n05
```

```
  usage=dataOnly
 # List of server nodes for this disk
 servers=k145n05,k145n06
 failureGroup=2
 pool=dataPoolA

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

# Chapter 3. Managing IBM Spectrum Scale RAID with the mmvdisk command

The **mmvdisk** command provides a unified conceptual framework that simplifies many of the more complicated aspects of IBM Spectrum Scale RAID administration.

The **mmvdisk** command enforces and encourages IBM Spectrum Scale RAID best practices with respect to the following tasks:

- IBM Spectrum Scale RAID server configuration (**mmvdisk server**)
- Recovery group creation (**mmvdisk recoverygroup**)
- The definition, sizing, and creation of vdisk NSDs (**mmvdisk vdiskset**)
- The creation of vdisk-based file systems (**mmvdisk filesystem**)

The **mmvdisk** command does this without the manual stanza file editing required by many of the legacy IBM Spectrum Scale RAID commands.

The central concept in **mmvdisk** is that of a vdisk set. A vdisk set is a collection of uniform vdisk NSDs from one or more recovery groups. The member vdisk NSDs of a vdisk set all have the same attributes in each of the vdisk set's recovery groups. Vdisk sets provide a scalable specification template, which is applicable across multiple recovery groups, for sizing and creating uniform vdisk NSDs. The vdisk NSDs are then managed as a unit.

A vdisk-based file system is constructed from one or more vdisk sets. Each vdisk set in a file system contributes all of its member vdisk NSDs to the file system. A vdisk set must belong in its entirety to exactly one file system, but a file system can be constructed from multiple vdisk sets.

Compatibility between **mmvdisk** and the legacy IBM Spectrum Scale RAID commands is strictly limited.

The **mmvdisk** command provides a command for converting legacy IBM Spectrum Scale RAID recovery groups, servers, vdisks, and file systems to **mmvdisk** management.

Until a recovery group is converted to **mmvdisk** management, only the legacy IBM Spectrum Scale RAID commands can be used to manage the recovery group and its servers and vdisks; **mmvdisk** recognizes the recovery group as non-**mmvdisk** and refuses to manage it.

Until all existing legacy recovery groups are converted to **mmvdisk** management, **mmvdisk** refuses to create new recovery groups.

The legacy IBM Spectrum Scale RAID commands also refuse to manage a recovery group that is created by or converted to **mmvdisk** management.

When all recovery groups in an IBM Spectrum Scale are managed by **mmvdisk**, the legacy IBM Spectrum Scale RAID commands refuse to create new recovery groups. All further IBM Spectrum Scale RAID administration must be performed using **mmvdisk**.

## The mmvdisk administration methodology

The **mmvdisk** command organizes IBM Spectrum Scale RAID around a well-defined collection of objects: node classes, the servers within them, the recovery groups that they serve, the user vdisk NSDs across the recovery groups, and the file systems constructed from the user vdisk NSDs.

IBM Spectrum Scale RAID administration with **mmvdisk** begins with recovery group server node classes, servers, and recovery groups.

A recovery group server node class is called an **mmvdisk** node class. This is a regular IBM Spectrum Scale node class with the restriction that it can only be altered by the **mmvdisk** command.

The **mmvdisk** node class contains the pair of servers responsible for an ESS recovery group pair. The IBM Spectrum Scale RAID configuration parameters for the servers are maintained under the node class, and each of their two ESS recovery groups are associated with the node class.

File system vdisk NSDs are managed collectively as members of vdisk sets. A vdisk set is a collection of identical vdisk NSDs from one or more recovery groups. Each recovery group included in a vdisk set contributes one member vdisk NSD. A vdisk set is managed as a unit. All member vdisk NSDs of a vdisk set must belong to the same **mmvdisk** file system.

An mmvdisk file system is an IBM Spectrum Scale file system that is constructed from one or more vdisk sets.

The **mmvdisk** command can query certain version and configuration information for the servers in an **mmvdisk** node class, and can also validate that the servers have a consistent and correct supported server disk topology.

ESS recovery group pairs are created by supplying **mmvdisk** with two recovery group names and the associated two-server **mmvdisk** node class.

The **mmvdisk** command automatically creates the appropriate log vdisks for each recovery group as part of recovery group creation.

Once recovery groups are created, **mmvdisk** is used to define vdisk sets across the recovery groups. A vdisk set definition is a specification template that permits administrators to preview and evaluate how the vdisk sets are sized within the servers and declustered arrays of the recovery groups.

When the vdisk sets have been defined and sized satisfactorily in the desired recovery groups, **mmvdisk** is then used to create the vdisk sets. This instantiates a vdisk set definition into a real collection of vdisk NSDs.

The created vdisk sets are then used as the units from which **mmvdisk** builds IBM Spectrum Scale file systems.

The following definitions are relevant to using the **mmvdisk** command to manage IBM Spectrum Scale RAID:

- **Node class**: An **mmvdisk** node class is a special IBM Spectrum Scale node class that enumerates the cluster nodes that are, or are intended to be, the IBM Spectrum Scale RAID servers for an ESS recovery group pair. An **mmvdisk** node class can only be changed by **mmvdisk** commands.
- **Server**: A server is an IBM Spectrum Scale cluster node that is a member of an **mmvdisk** node class, or that has a non-zero **nsdRAIDTracks** IBM Spectrum Scale RAID configuration attribute.
- **Recovery group**: When an IBM Spectrum Scale RAID recovery group is managed by **mmvdisk**, it is called an **mmvdisk** recovery group. An **mmvdisk** recovery group has an associated **mmvdisk** node class, which is defined in one or more vdisk sets. For most purposes, an **mmvdisk** recovery group includes its log vdisks. The log vdisks of an **mmvdisk** recovery group are created and deleted together with the recovery group.
- **Vdisk NSD**: A vdisk NSD is an IBM Spectrum Scale NSD built from an IBM Spectrum Scale RAID user vdisk. For most purposes, **mmvdisk** treats a user vdisk and its NSD as a single entity.
- **Vdisk set**: A vdisk set is a collection of user vdisk NSDs from one or more **mmvdisk** recovery groups. Each vdisk NSD in a given vdisk set has an identical specification. A vdisk set defines and collects

identically specified vdisk NSDs from one or more **mmvdisk** recovery groups, and creates and deletes and assigns them to IBM Spectrum Scale file systems as a group.

- **File system**: An **mmvdisk** file system is an IBM Spectrum Scale file system that contains one or more vdisk NSDs where each vdisk NSD in the file system is from a vdisk set.

  **Note:** Regarding the definition of **mmvdisk file system**, it is possible for a file system to contain vdisk NSDs that are not from **mmvdisk** vdisk sets; a cluster could be in the process of being converted to **mmvdisk** administration, so that some of the vdisk NSDs are from converted recovery groups and some are not. By this definition, the file system is not an **mmvdisk** file system until the last legacy recovery group represented in the file system is converted to **mmvdisk** administration.

## Outline of an mmvdisk use case

The **mmvdisk** command assumes that the IBM Spectrum Scale cluster is installed and configured properly with respect to licenses, quorum and node designations, and network connectivity.

The intended IBM Spectrum Scale RAID recovery group server nodes should be properly connected to the disks that will make up the recovery groups. The **mmvdisk** command can be used to validate that these servers detect a supported disk topology.

The basic use case for creating vdisk-based file systems with **mmvdisk** follows this sequence of steps:

1. Creating recovery group server node classes. For ESS recovery group pairs, **mmvdisk** is used to create an **mmvdisk** node class for each ESS server pair. Each node class contains exactly one ESS server pair.

   ```
   # mmvdisk nodeclass create --node-class ESS01 -N server01,server02
   ```

2. Verifying recovery group server disk topologies. Next, **mmvdisk** is used to verify that the intended supported server disk topology is detected on the node class.

   ```
   # mmvdisk server list --node-class ESS01 --disk-topology
   ```

3. Configuring recovery group servers. Next, **mmvdisk** is used to configure each node class as GNR recovery group servers. Each server node class is configured independently of other nodes or node classes.

   ```
   # mmvdisk server configure --node-class ESS01 --recycle all
   ```

4. Creating recovery groups. Once an **mmvdisk** node class is configured and verified to have the correct disk topology, **mmvdisk** is used to create the two ESS recovery groups for the node class.

   ```
   # mmvdisk recoverygroup create --recovery-group ESS01L,ESS01R --node-class ESS01
   ```

5. Defining vdisk sets. Once the recovery groups are created, **mmvdisk** is used to define vdisk sets in the recovery groups. The vdisk set definitions are specification templates that are sized within one or more recovery groups, which allows the disk space and server memory demands to be previewed and evaluated before any vdisk NSDs are actually created.

   ```
   # mmvdisk vdiskset define --vdisk-set vs1 --recovery-group ESS01L,ESS01R --code 8+3p --block-size 4m --set-size 50%
   # mmvdisk vdiskset define --vdisk-set vs2 --recovery-group ESS01L,ESS01R --code 4+2p --block-size 512k --set-size 25%
   ```

   As long as the actual vdisk NSDs have not been created, vdisk sets can be undefined and defined again until the attributes and disk and memory sizes are satisfactory.

6. Creating vdisk NSDs. When the vdisk set definitions have been satisfactorily sized, **mmvdisk** is used to create the actual vdisk NSDs in the recovery groups.

   ```
   # mmvdisk vdiskset create --vdisk-set vs1,vs2
   ```

7. Creating vdisk-based file systems. Once vdisk sets are created, **mmvdisk** is used to create file systems that use the vdisk NSDs from one or more vdisk sets.

   ```
   # mmvdisk filesystem create --file-system fs1 --vdisk-set vs1
   # mmvdisk filesystem create --file-system fs2 --vdisk-set vs2
   ```

## Elements of a vdisk set definition

The definition of a vdisk set requires seven attributes and one or more recovery groups:

1. **Vdisk set name**: A name for the vdisk set that is unique within the IBM Spectrum Scale cluster.
2. **Declustered array**: The declustered array in the recovery groups where the member vdisk NSDs are located.
3. **RAID code**: The vdisk RAID code used for the member vdisk NSDs.
4. **Block size**: The file system block size (equivalently, the vdisk track size) of the member vdisk NSDs.
5. **Vdisk NSD size**: The usable size of a member vdisk NSD.
6. **NSD usage**: The file system usage of a member vdisk NSD.
7. **Storage pool**: The file system storage pool of a member vdisk NSD.

   **Recovery groups**: One or more recovery groups, each of which contributes one member vdisk NSD to the vdisk set.

Defining a vdisk set requires specifying at least one recovery group, the vdisk set name, the RAID code, the block size, and the vdisk set size (from which the member vdisk NSD size is calculated). The declustered array, NSD usage, and the storage pool attributes can be omitted when **mmvdisk** can determine suitable default values.

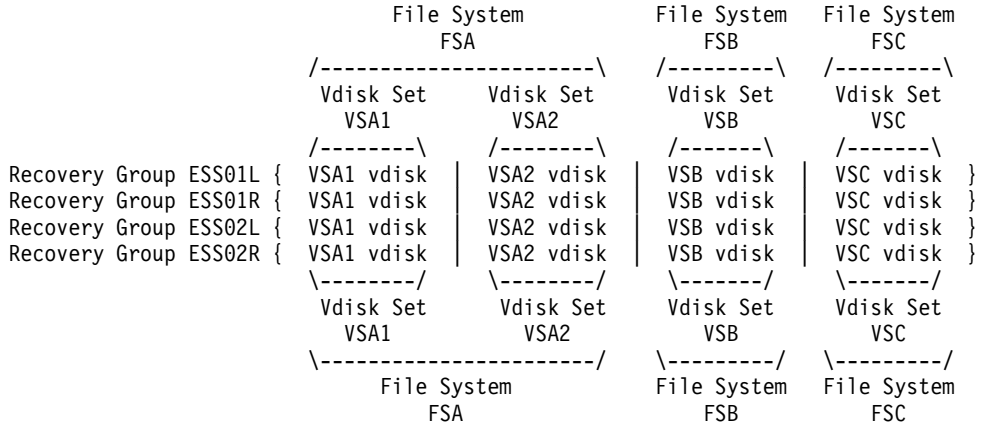
*Table 2.*

| Vdisk set attribute | Default value | Valid values |
|---|---|---|
| vdisk set name | must be specified | The vdisk set name must be unique within the IBM Spectrum Scale cluster. The name is limited to the same character set as IBM Spectrum Scale file system device names and IBM Spectrum Scale RAID recovery group names. |
| declustered array name | DA1 (if only one DA is present) | All recovery groups for which this vdisk set is defined must have a declustered array with this name where this vdisk set's members are created. The expectation is that a vdisk set extends across structurally identical recovery groups where the named declustered array has the same characteristics in each recovery group. If there is only one user declustered array in each recovery group, it is named DA1 and this is the default. If there is more than one user declustered array in a recovery group, there is no default and a declustered array name must be specified. |
| RAID code | must be specified | This is the vdisk RAID code for members of the vdisk set. Valid values are: 3WayReplication, 4WayReplication, 4+2P, 4+3P, 8+2P, or 8+3P |
| block size | must be specified | This is the file system block size (vdisk track size) for members of the vdisk set. It is constrained by the selected RAID code. Valid values for 3WayReplication and 4WayReplication are 256k, 512k, 1m, or 2m. Valid values for 4+2P and 4+3P are 512k, 1m, 2m, 4m, or 8m. Valid values for 8+2P and 8+3P are 512k, 1m, 2m, 4m, 8m, or 16m. |

*Table 2. (continued)*

| Vdisk set attribute | Default value | Valid values |
|---|---|---|
| vdisk set size | must be specified | The vdisk set size is the size of the vdisk set in the specified declustered array in one recovery group. If the vdisk set size is given as a percentage, it specifies the raw size to use from the DA, including RAID code redundancy. If the vdisk set size is given as a number of bytes, it specifies the desired usable size of the vdisk set, excluding RAID code redundancy. The vdisk set size is used to calculate the usable size of a single vdisk NSD member of the vdisk set in one recovery group. It is this calculated usable size that becomes part of the vdisk set definition, so if the size of a declustered array should ever change, the size of the individual member vdisk NSDs remains constant. |
| NSD usage | dataAndMetadata | This is the IBM Spectrum Scale file system data usage for the NSD. Valid values are dataAndMetadata, metadataOnly, and dataOnly. The default is dataAndMetadata. |
| storage pool | system | If the NSD usage is dataAndMetadata or metadataOnly, the storage pool value must be `system` and does not need to be specified. If the NSD usage is dataOnly, the storage pool must be specified and the value may not be `system`. |

The following illustration is a visualization of how four vdisk sets might span across four ESS recovery groups, and how they are used to form three file systems:




There are four structurally identical recovery groups built from two ESS building blocks. There are three file systems all uniformly striped across the recovery groups using four vdisk sets. An identical member of each of the four sets vdisk sets is present in each recovery group.

File system FSA uses two vdisk sets, VSA1 and VSA2. File system FSB uses one vdisk set, VSB. File system FSC uses one vdisk set, VSC.

For the high-level purpose of visualizing how four vdisk sets might span four recovery groups to construct three file systems, details such as the RAID code and block size for the four vdisk sets are omitted. It is enough to know that within a vdisk set, the details are the same for each member vdisk NSD.

This illustration also shows how the concept of vdisk sets helps to accomplish the IBM Spectrum Scale RAID best practice of constructing file systems from identical vdisk NSDs equally distributed across all recovery groups.

## Vdisk set definition sizing and preview

Defining vdisk sets before actually creating the member vdisk NSDs permits the IBM Spectrum Scale administrator to preview the declustered array size and server memory requirements of the vdisk sets.

The **mmvdisk** command refuses to define a vdisk set that exceeds declustered array disk capacity in a recovery group or server memory capacity in the recovery group server node class.

Newly created recovery groups have no vdisk sets. To see what capacity the newly created recovery groups have for defining vdisk sets, use **mmvdisk vdiskset list --recovery-group all**. This command shows all recovery group declustered array capacities where vdisk sets can be defined, and the server memory capacities for the recovery group server node classes.

In this example, the two recovery groups, which are newly created, have 100% of their raw declustered array capacity free. Their servers are using a baseline 386 MiB of 30 GiB available memory for internal server vdisk set maps.

```
# mmvdisk recoverygroup list

                                                          needs    user
recovery group  active   current or master server        service  vdisks  remarks
--------------  -------  -------------------------------  -------  ------  -------
BB01L           yes      server01.gpfs.net                no            0
BB01R           yes      server02.gpfs.net                no            0

# mmvdisk vdiskset list --recovery-group all

                declustered           capacity            all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
BB01L           DA1             36 TiB    36 TiB   100%  -
BB01R           DA1             36 TiB    36 TiB   100%  -

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
BB01           30 GiB   386 MiB  -

#
```

As vdisk sets are defined in these recovery groups, **mmvdisk** shows the cumulative claim made by the vdisk sets upon the declustered array capacity and server memory resources. It does so without actually having to create the vdisk NSDs.

The next example shows the accumulating declustered array space and memory claims associated with the definition of three vdisk sets in the BB01L and BB01R recovery groups and the BB01 server node class.

```
# mmvdisk vdiskset define --vdisk-set fs1data --recovery-group BB01L,BB01R --set-size 45% \
                          --code 8+3p --block-size 2m --nsd-usage dataOnly --storage-pool data
mmvdisk: Vdisk set 'fs1data' has been defined.
mmvdisk: Recovery group 'BB01L' has been defined in vdisk set 'fs1data'.
mmvdisk: Recovery group 'BB01R' has been defined in vdisk set 'fs1data'.

                     member vdisks
vdisk set       count   size   raw size  created  file system and attributes
--------------  -----  --------  --------  -------  -------------------------
fs1data             2   11 TiB   16 TiB  no       -, DA1, 8+3p, 2 MiB, dataOnly, data

                declustered           capacity            all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
BB01L           DA1             36 TiB    20 TiB    55%  fs1data
BB01R           DA1             36 TiB    20 TiB    55%  fs1data
```

```
                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
BB01           30 GiB   869 MiB  fs1data (483 MiB)

# mmvdisk vdiskset define --vdisk-set fs1meta --recovery-group BB01L,BB01R --set-size 5% \
                          --code 4wayReplication --block-size 512k --nsd-usage metadataOnly
mmvdisk: Vdisk set 'fs1meta' has been defined.
mmvdisk: Recovery group 'BB01L' has been defined in vdisk set 'fs1meta'.
mmvdisk: Recovery group 'BB01R' has been defined in vdisk set 'fs1meta'.

                     member vdisks
vdisk set       count   size   raw size  created  file system and attributes
--------------  ----- -------- --------  -------  -------------------------
fs1meta             2  463 GiB 1872 GiB  no       -, DA1, 4WayReplication, 512 KiB, metadataOnly, system

                declustered           capacity           all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
BB01L           DA1             36 TiB    18 TiB    50%  fs1data, fs1meta
BB01R           DA1             36 TiB    18 TiB    50%  fs1data, fs1meta

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
BB01           30 GiB   908 MiB  fs1data (483 MiB), fs1meta (38 MiB)

# mmvdisk vdiskset define --vdisk-set fs2 --recovery-group BB01L,BB01R --set-size 50% \
                          --code 8+3p --block-size 2m
mmvdisk: Vdisk set 'fs2' has been defined.
mmvdisk: Recovery group 'BB01L' has been defined in vdisk set 'fs2'.
mmvdisk: Recovery group 'BB01R' has been defined in vdisk set 'fs2'.

                     member vdisks
vdisk set       count   size   raw size  created  file system and attributes
--------------  ----- -------- --------  -------  -------------------------
fs2                 2   13 TiB   18 TiB  no       -, DA1, 8+3p, 2 MiB, dataAndMetadata, system

                declustered           capacity           all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
BB01L           DA1             36 TiB  4096 MiB     0%  fs1data, fs1meta, fs2
BB01R           DA1             36 TiB  4096 MiB     0%  fs1data, fs1meta, fs2

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
BB01           30 GiB  1445 MiB  fs1data (483 MiB), fs1meta (38 MiB), fs2 (537 MiB)

#
```

At this point, 100% of the disk capacity in DA1 has been claimed by these vdisk set definitions. The server memory demand is well under the available limit. None of the vdisk sets have been created yet. A vdisk set is only created when all of the expected vdisk NSDs in all the vdisk set's recovery groups exist in reality and not merely as a claim upon resource. If none, or only some but not all, of the expected vdisk NSDs actually exist, the value in the `created` column for a vdisk set is `no`.

Assuming these definitions are what is desired, the next example shows actually creating these three vdisk sets.

```
# mmvdisk vdiskset create --vdisk-set all
mmvdisk: 2 vdisks and 2 NSDs will be created in vdisk set 'fs2'.
mmvdisk: 2 vdisks and 2 NSDs will be created in vdisk set 'fs1data'.
mmvdisk: 2 vdisks and 2 NSDs will be created in vdisk set 'fs1meta'.
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001VS003
```

```
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002VS003
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002VS001
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001VS001
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002VS002
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001VS002
mmvdisk: Created all vdisks in vdisk set 'fs2'.
mmvdisk: Created all vdisks in vdisk set 'fs1data'.
mmvdisk: Created all vdisks in vdisk set 'fs1meta'.
mmvdisk: (mmcrnsd) Processing disk RG001VS003
mmvdisk: (mmcrnsd) Processing disk RG002VS003
mmvdisk: (mmcrnsd) Processing disk RG001VS001
mmvdisk: (mmcrnsd) Processing disk RG002VS001
mmvdisk: (mmcrnsd) Processing disk RG001VS002
mmvdisk: (mmcrnsd) Processing disk RG002VS002
mmvdisk: Created all NSDs in vdisk set 'fs2'.
mmvdisk: Created all NSDs in vdisk set 'fs1data'.
mmvdisk: Created all NSDs in vdisk set 'fs1meta'.
# mmvdisk vdiskset list --vdisk-set all

                     member vdisks
vdisk set       count   size   raw size  created  file system and attributes
--------------  ----- -------- --------  -------  -------------------------
fs1data             2   11 TiB   16 TiB  yes      -, DA1, 8+3p, 2 MiB, dataOnly, data
fs1meta             2  463 GiB 1872 GiB  yes      -, DA1, 4WayReplication, 512 KiB, metadataOnly, system
fs2                 2   13 TiB   18 TiB  yes      -, DA1, 8+3p, 2 MiB, dataAndMetadata, system

                 declustered           capacity           all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
BB01L           DA1             36 TiB  4096 MiB     0%  fs1data, fs1meta, fs2
BB01R           DA1             36 TiB  4096 MiB     0%  fs1data, fs1meta, fs2

                   vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
BB01           30 GiB  1445 MiB  fs1data (483 MiB), fs1meta (38 MiB), fs2 (537 MiB)

#
```

The **mmvdisk vdiskset create** command shows the creation of the individual member vdisk NSDs in which the automatically generated names of the individual member vdisks and NSDs are seen. The names of individual vdisk NSDs are generally unimportant since normal **mmvdisk** usage manages them collectively as vdisk set members.

**Note:** If it is ever required to know the details of individual member vdisk NSDs, the **mmvdisk vdisk list** and **mmvdisk recoverygroup list --vdisk** commands can be used.

After vdisk creation, when the vdisk sets are listed again, the `created` column shows `yes` for all three vdisk sets. The actual member vdisk NSDs have been created and are ready to be used to create file systems.

```
# mmvdisk filesystem create --file-system fs1 --vdisk-set fs1meta,fs1data
# mmvdisk filesystem create --file-system fs2 --vdisk-set fs2
```

After file system creation, listing the vdisk sets shows the file system device name with the vdisk set attributes.

```
# mmvdisk vdiskset list --vdisk-set all

                     member vdisks
vdisk set       count   size   raw size  created  file system and attributes
--------------  ----- -------- --------  -------  -------------------------
```

```
fs1data             2   11 TiB   16 TiB  yes      fs1, DA1, 8+3p, 2 MiB, dataOnly, data
fs1meta             2  463 GiB 1872 GiB  yes      fs1, DA1, 4WayReplication, 512 KiB, metadataOnly, system
fs2                 2   13 TiB   18 TiB  yes      fs2, DA1, 8+3p, 2 MiB, dataAndMetadata, system
...
```

## Overview of the mmvdisk commands

An overview of the primary functions of the **mmvdisk** commands.

For a complete description of the **mmvdisk** commands, see "mmvdisk command" on page 218.

**mmvdisk nodeclass**

The **mmvdisk nodeclass** command is used for the following functions:

- To create the **mmvdisk** node class for the two servers of an ESS recovery group pair.
- To list **mmvdisk** node classes.

Once created, an **mmvdisk** node class can be configured to be IBM Spectrum Scale RAID recovery group servers with **mmvdisk server configure**. Once configured, it can be associated with a pair of ESS recovery groups with **mmvdisk recoverygroup create**.

The **mmvdisk nodeclass** command performs functions similar to the **mmcrnodeclass**, **mmlsnodeclass**, **mmchnodeclass**, and **mmdelnodeclass** commands.

**mmvdisk server**

The **mmvdisk server** command is used for the following functions:

- To list and validate the supported ESS server disk topology of an **mmvdisk** node class.
- To configure an **mmvdisk** node class to be IBM Spectrum Scale RAID recovery group servers.

The **mmvdisk server** command performs functions similar to the **gssServerConfig**, **mmchconfig**, **mmlsconfig**, **mmshutdown**, **mmstartup**, **mmgetpdisktopology**, and **topsummary** commands.

**mmvdisk recoverygroup**

The **mmvdisk recoverygroup** command is used for the following functions:

- To create an ESS recovery group pair on a configured **mmvdisk** node class.
- To list recovery group information.
- To convert existing non-**mmvdisk** recovery groups to **mmvdisk** management.

The **mmvdisk recoverygroup** command performs functions similar to the **mmgetpdisktopology**, **mkrginput**, **mmcrrecoverygroup**, **mmcrvdisk**, **mmlsrecoverygroup**, **mmchrecoverygroup**, **mmdelvdisk**, and **mmdelrecoverygroup** commands.

**mmvdisk vdiskset**

The **mmvdisk vdiskset** command is used for the following functions:

- To define and size vdisk sets across recovery groups.
- To create the member vdisk NSDs of a vdisk set according to its definition.
- To list vdisk set attribute and sizing information.

The **mmvdisk vdiskset** command performs functions similar to the **mmcrvdisk**, **mmdelvdisk**, and **gssgenvdisks** commands.

**mmvdisk filesystem**

The **mmvdisk filesystem** command is used for the following functions:

- To create an **mmvdisk** file system using vdisk sets.
- To add vdisk sets to **mmvdisk** file systems.
- To delete vdisk sets from **mmvdisk** file systems.
- To list **mmvdisk** file system information.

The **mmvdisk filesystem** command performs functions similar to the **mmcrfs**, **mmadddisk**, **mmdeldisk**, and **mmdelfs** commands.

**mmvdisk pdisk**

The **mmvdisk pdisk** command is used for the following functions:

- To list **mmvdisk** recovery group pdisk information.
- To list pdisks that are out of service or in need of replacement.
- To replace pdisks that are in need of replacement.

The **mmvdisk pdisk** command performs functions similar to the **mmlspdisk**, **mmchcarrier**, and **mmchpdisk** commands.

**mmvdisk vdisk**

The primary function of the **mmvdisk vdisk** command is to list basic information about individual vdisks.

All other vdisk management is performed through the **mmvdisk vdiskset** command for file system vdisk NSDs, and through the**mmvdisk recoverygroup** command for recovery group log vdisks.

The **mmvdisk vdisk** command performs functions similar to the **mmlsvdisk** command.

## Use case for mmvdisk

This example assumes that there is a properly installed three-node IBM Spectrum Scale cluster with an ESS GL6 building block on the server pair ess01io1 and ess01io2:

```
# mmgetstate -a

 Node number  Node name        GPFS state
------------------------------------------
       1      ess01io1         active
       2      ess01io2         active
       3      ems              active
```

Define an **mmvdisk** node class for the server pair:

```
# mmvdisk nodeclass create --node-class ESS01 -N ess01io1,ess01io2
mmvdisk: Node class 'ESS01' created.
```

Verify that the correct ESS GL6 supported disk topology is detected on the node class:

```
# mmvdisk server list --node-class ESS01 --disk-topology

 node                                 needs    matching
number  server                      attention   metric   disk topology
------  ------------------------------  ---------  --------  -------------
     1  ess01io1.gpfs.net               no          100/100  ESS GL6
     2  ess01io2.gpfs.net               no          100/100  ESS GL6
```

Since a correct disk topology has been detected, configure the node class to be recovery group servers:

```
# mmvdisk server configure --node-class ESS01 --recycle one
mmvdisk: Checking resources for specified nodes.
mmvdisk: Node 'ess01io1.gpfs.net' has a paired recovery group disk topology.
mmvdisk: Node 'ess01io2.gpfs.net' has a paired recovery group disk topology.
mmvdisk: Node class 'ESS01' has a paired recovery group disk topology.
mmvdisk: Setting configuration for node class 'ESS01'.
mmvdisk: Node class 'ESS01' is now configured to be recovery group servers.
mmvdisk: Restarting GPFS daemon on node 'ess01io1.gpfs.net'.
mmvdisk: Restarting GPFS daemon on node 'ess01io2.gpfs.net'.
```

With the node class configured, create a recovery group pair:

```
# mmvdisk recoverygroup create --node-class ESS01 --recovery-group ESS01L,ESS01R
mmvdisk: Checking node class configuration.
mmvdisk: Checking daemon status on node 'ess01io1.gpfs.net'.
mmvdisk: Checking daemon status on node 'ess01io2.gpfs.net'.
mmvdisk: Analyzing disk topology for node 'ess01io1.gpfs.net'.
mmvdisk: Analyzing disk topology for node 'ess01io2.gpfs.net'.
mmvdisk: Creating recovery group 'ESS01L'.
mmvdisk: Creating recovery group 'ESS01R'.
mmvdisk: Creating log vdisks for recovery groups.
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001LOGTIP
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001LOGTIPBACKUP
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001LOGHOME
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002LOGTIP
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002LOGTIPBACKUP
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002LOGHOME
mmvdisk: Created recovery groups 'ESS01L' and 'ESS01R'.
```

With the recovery group pair created, the sizing available for vdisk sets can be listed:

```
# mmvdisk vdiskset list --recovery-group all

                declustered           capacity            all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
ESS01L          DA1            911 TiB   911 TiB   100%  -
ESS01R          DA1            911 TiB   911 TiB   100%  -

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
ESS01          29 GiB   387 MiB  -
```

For this example, assume that it is desired to use 50% of the space in declustered array DA1 to create a file system using RAID code 8+3p and a 16M block size.

Define vdisk set VS1 in recovery groups ESS01L and ESS01R using RAID code 8+3p, block size 16m, and a vdisk set size of 50%:

```
# mmvdisk vdiskset define --vdisk-set VS1 --recovery-group ESS01L,ESS01R --code 8+3p --block-size 16m --set-size 50%
mmvdisk: Vdisk set 'VS1' has been defined.
mmvdisk: Recovery group 'ESS01L' has been defined in vdisk set 'VS1'.
mmvdisk: Recovery group 'ESS01R' has been defined in vdisk set 'VS1'.

                     member vdisks
vdisk set       count   size   raw size  created  file system and attributes
--------------  ----- -------- --------  -------  -------------------------
VS1                 2  330 TiB  455 TiB  no       -, DA1, 8+3p, 16 MiB, dataAndMetadata, system

                declustered           capacity            all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
ESS01L          DA1            911 TiB   455 TiB    50%  VS1
ESS01R          DA1            911 TiB   455 TiB    50%  VS1

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
ESS01          29 GiB  8443 MiB  VS1 (8055 MiB)
```

Assuming that the sizes that result from this definition are satisfactory, create the actual member vdisk NSDs for vdisk set VS1:

```
# mmvdisk vdiskset create --vdisk-set VS1
mmvdisk: 2 vdisks and 2 NSDs will be created in vdisk set 'VS1'.
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001VS001
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002VS001
```

```
mmvdisk: Created all vdisks in vdisk set 'VS1'.
mmvdisk: (mmcrnsd) Processing disk RG001VS001
mmvdisk: (mmcrnsd) Processing disk RG002VS001
mmvdisk: Created all NSDs in vdisk set 'VS1'.
```

Then use **mmvdisk filesystem create** to create file system FS1 using vdisk set VS1. In this example, the **--mmcrfs** option is used to pass all remaining command line parameters to the underlying **mmcrfs** command to enable quotas and select a custom file system mount point:

```
# mmvdisk filesystem create --file-system FS1 --vdisk-set VS1 --mmcrfs -Q yes -T /FS1
mmvdisk: Creating file system 'FS1'.
mmvdisk: The following disks of FS1 will be formatted on node ess01io1:
mmvdisk:     RG001VS001: size 346179584 MB
mmvdisk:     RG002VS001: size 346179584 MB
mmvdisk: Formatting file system ...
mmvdisk: Disks up to size 2.58 PB can be added to storage pool system.
mmvdisk: Creating Inode File
mmvdisk:   68 % complete on Tue Jun 12 11:29:10 2018
mmvdisk:  100 % complete on Tue Jun 12 11:29:12 2018
mmvdisk: Creating Allocation Maps
mmvdisk: Creating Log Files
mmvdisk: Clearing Inode Allocation Map
mmvdisk: Clearing Block Allocation Map
mmvdisk: Formatting Allocation Map for storage pool system
mmvdisk: Completed creation of file system /dev/FS1.
```

The file system is now ready to be mounted and used:

```
# mmvdisk filesystem list --file-system FS1

                                vdisk     list of       holds    holds  storage
vdisk set       recovery group  count  failure groups  metadata  data    pool
--------------  --------------  -----  --------------  --------  -----  -------
VS1             ESS01L              1               1  yes       yes    system
VS1             ESS01R              1               2  yes       yes    system

# mmmount FS1 -a
Tue Jun 12 11:37:28 EDT 2018: mmmount: Mounting file systems ...
# sync; df /FS1
Filesystem        1K-blocks    Used    Available Use% Mounted on
FS1            708975788032 5718016 708970070016   1% /FS1
```

## Reporting file system orphans with mmvdisk

A vdisk set must belong to a single file system in its entirety.

If a file system vdisk set is extended into additional recovery groups, a situation is created where some of the member vdisk NSDs are in the file system and some are not. The vdisk NSDs that are not yet in the file system to which they belong are considered "orphaned" from the file system.

A similar but less common situation arises when a recovery group is in the process of being removed from a vdisk set. The member vdisk NSDs from that recovery group are first deleted from a file system, but they remain in the vdisk set until they are deleted from the recovery group and the recovery group is removed from the vdisk set definition.

The **mmvdisk** command detects these situations and reminds the administrator that more work is required to bring the vdisk set and the file system into alignment.

The following example illustrates how **mmvdisk** reports file system orphans after a vdisk set has been defined and created in one recovery group and used to create a file system:

```
# mmvdisk vdiskset list --vdisk-set VS1

                    member vdisks
```

```
vdisk set       count   size   raw size  created  file system and attributes
--------------  ----- -------- --------  -------  -------------------------
VS1                 1  132 TiB  182 TiB  yes      FS1, DA1, 8+3p, 16 MiB, dataAndMetadata, system

                declustered           capacity           all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
ESS01L          DA1            911 TiB   729 TiB    80%  VS1

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
ESS01          29 GiB  1998 MiB  VS1 (1610 MiB)

# mmvdisk filesystem list --file-system FS1

                                 vdisk     list of       holds    holds  storage
vdisk set       recovery group   count  failure groups  metadata  data    pool
--------------  --------------  -----  --------------  --------  -----  -------
VS1             ESS01L              1               1  yes       yes    system
```

It is now desired to extend the vdisk set VS1 and the file system FS1 into recovery group ESS01R.

When recovery group ESS01R is added to the VS1 vdisk set definition, **mmvdisk** immediately notices that the file system FS1 is no longer in sync with vdisk set VS1:

```
# mmvdisk vdiskset define --vdisk-set VS1 --recovery-group ESS01R
mmvdisk: Recovery group 'ESS01R' has been defined in vdisk set 'VS1'.

                    member vdisks
vdisk set       count   size   raw size  created  file system and attributes
--------------  ----- -------- --------  -------  -------------------------
VS1                 2  132 TiB  182 TiB  no       FS1, DA1, 8+3p, 16 MiB, dataAndMetadata, system

                declustered           capacity           all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
ESS01L          DA1            911 TiB   729 TiB    80%  VS1
ESS01R          DA1            911 TiB   729 TiB    80%  VS1

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
ESS01          29 GiB  3609 MiB  VS1 (3221 MiB)

mmvdisk: Attention: There are incomplete vdisk set or file system changes.
mmvdisk: Members of vdisk set 'VS1' are orphaned from file system 'FS1'.
mmvdisk: Complete any vdisk set or file system changes to dismiss this notice.
```

When a file system or vdisk set in this state is listed, **mmvdisk** reminds the administrator that work remains to be done. Listing file system FS1 provides another illustration:

```
# mmvdisk filesystem list --file-system FS1

                                 vdisk     list of       holds    holds  storage
vdisk set       recovery group   count  failure groups  metadata  data    pool
--------------  --------------  -----  --------------  --------  -----  -------
VS1             ESS01L              1               1  yes       yes    system
VS1             ESS01R              0               -  yes       yes    system

mmvdisk: Attention: There are incomplete vdisk set or file system changes.
mmvdisk: Members of vdisk set 'VS1' are orphaned from file system 'FS1'.
mmvdisk: Complete any vdisk set or file system changes to dismiss this notice.
```

In this case, since the intention is to extend file system FS1 into recovery group ESS01R, completing the vdisk set and file system changes means creating the new member vdisk NSDs for VS1 and adding them to the file system:

```
# mmvdisk vdiskset create --vdisk-set VS1
mmvdisk: 1 vdisk and 1 NSD will be created in vdisk set 'VS1'.
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002VS001
mmvdisk: Created all vdisks in vdisk set 'VS1'.
mmvdisk: (mmcrnsd) Processing disk RG002VS001
mmvdisk: Created all NSDs in vdisk set 'VS1'.

mmvdisk: Attention: There are incomplete vdisk set or file system changes.
mmvdisk: Members of vdisk set 'VS1' are orphaned from file system 'FS1'.
mmvdisk: Complete any vdisk set or file system changes to dismiss this notice.
```

The new member vdisk NSDs have been created, but they are still orphaned because they must also be added to the file system:

```
# mmvdisk filesystem add --file-system FS1 --vdisk-set VS1
mmvdisk: The following disks of FS1 will be formatted on node ess01io1:
mmvdisk:     RG002VS001: size 138425984 MB
mmvdisk: Extending Allocation Map
mmvdisk: Checking Allocation Map for storage pool system
mmvdisk: Completed adding disks to file system FS1.
```

Now that the new member vdisk NSDs have been added to the file system, listing the file system no longer shows the notice about orphaned members:

```
# mmvdisk filesystem list --file-system FS1

                                vdisk     list of       holds    holds  storage
vdisk set       recovery group  count  failure groups  metadata  data    pool
--------------  --------------  -----  --------------  --------  -----  -------
VS1             ESS01L              1               1  yes       yes    system
VS1             ESS01R              1               2  yes       yes    system
```

**Note:** This output showed that **mmvdisk** automatically followed file system best practices in using a different file system NSD failure group number for the new vdisk NSD from recovery group ESS01R.

## Converting existing recovery groups to mmvdisk management

To manage existing ESS recovery group pairs using **mmvdisk**, you must convert them to the **mmvdisk** IBM Spectrum Scale RAID configuration structure. This task is accomplished with the **mmvdisk recoverygroup convert** command.

The **mmvdisk recoverygroup list** command displays in the remarks column which existing recovery groups are not managed by **mmvdisk**:

```
# mmvdisk recoverygroup list

                                                          needs    user
recovery group  active   current or master server        service  vdisks  remarks
--------------  -------  -------------------------------  -------  ------  -------
ESS01L          yes      ess01io1.gpfs.net                no            5  non-mmvdisk
ESS01R          yes      ess01io2.gpfs.net                no            5  non-mmvdisk
```

A recovery group pair must be converted together, and the two recovery groups must share an **mmvdisk** node class. The node class can be an existing node class that contains exactly the two servers for the recovery group pair, which converts the node class to an **mmvdisk** node class. Alternatively, an unused node class name can be specified. This creates an **mmvdisk** node class with the two servers for the recovery group pair.

The IBM Spectrum Scale RAID configuration for the servers is set under the **mmvdisk** node class using the same settings that would be used for new **mmvdisk** recovery groups.

The user vdisk NSDs in the recovery group pair are matched into vdisk sets according to their file systems and attributes. The converted vdisk sets are given generated names, which can be changed later to something more meaningful using the **mmvdisk vdiskset rename** command.

To convert existing recovery groups ESS01L and ESS01R with a new **mmvdisk** node class ESS01, run the following command:

```
# mmvdisk recoverygroup convert --recovery-group ESS01L,ESS01R --node-class ESS01 --recycle one

mmvdisk: This command will permanently change the GNR configuration
mmvdisk: attributes and disable the legacy GNR command set for the
mmvdisk: servers and recovery groups involved, and their subsequent
mmvdisk: administration must be performed with the mmvdisk command.

mmvdisk: Do you wish to continue (yes or no)? yes

mmvdisk: Converting recovery groups 'ESS01L' and 'ESS01R'.
mmvdisk: Creating node class 'ESS01'.
mmvdisk: Adding 'ess01io1' to node class 'ESS01'.
mmvdisk: Adding 'ess01io2' to node class 'ESS01'.
mmvdisk: Associating recovery group 'ESS01L' with node class 'ESS01'.
mmvdisk: Associating recovery group 'ESS01R' with node class 'ESS01'.
mmvdisk: Recording pre-conversion cluster configuration in /var/mmfs/tmp/mmvdisk.convert.ESS01L.ESS01R.before.m24
mmvdisk: Updating server configuration attributes.
mmvdisk: Checking resources for specified nodes.
mmvdisk: Setting configuration for node class 'ESS01'.
mmvdisk: Defining vdisk set 'VS001_fs3' with recovery group 'ESS01L' (vdisk 'ESS01Lv3fs3').
mmvdisk: Defining vdisk set 'VS002_fs4' with recovery group 'ESS01L' (vdisk 'ESS01Lv4fs4').
mmvdisk: Defining vdisk set 'VS003_fs1' with recovery group 'ESS01L' (vdisk 'ESS01Lv1fs1data').
mmvdisk: Defining vdisk set 'VS004_fs2' with recovery group 'ESS01L' (vdisk 'ESS01Lv2fs2').
mmvdisk: Defining vdisk set 'VS005_fs1' with recovery group 'ESS01L' (vdisk 'ESS01Lv1fs1meta').
mmvdisk: Adding recovery group 'ESS01R' to vdisk set 'VS003_fs1' (vdisk 'ESS01Rv1fs1data').
mmvdisk: Adding recovery group 'ESS01R' to vdisk set 'VS004_fs2' (vdisk 'ESS01Rv2fs2').
mmvdisk: Adding recovery group 'ESS01R' to vdisk set 'VS001_fs3' (vdisk 'ESS01Rv3fs3').
mmvdisk: Adding recovery group 'ESS01R' to vdisk set 'VS005_fs1' (vdisk 'ESS01Rv1fs1meta').
mmvdisk: Adding recovery group 'ESS01R' to vdisk set 'VS002_fs4' (vdisk 'ESS01Rv4fs4').
mmvdisk: Committing cluster configuration changes.
mmvdisk: Recording post-conversion cluster configuration in /var/mmfs/tmp/mmvdisk.convert.ESS01L.ESS01R.after.m24

mmvdisk: Restarting GPFS on the following nodes:
mmvdisk:     ess01io1.gpfs.net
mmvdisk: Restarting GPFS on the following nodes:
mmvdisk:     ess01io2.gpfs.net
```

The recovery groups ESS01L and ESS01R are now under **mmvdisk** management. The remarks column of **mmvdisk recoverygroup list** is blank:

```
# mmvdisk recoverygroup list

                                                          needs    user
recovery group  active   current or master server         service  vdisks  remarks
--------------  -------  -------------------------------  -------  ------  -------
ESS01L          yes      ess01io1.gpfs.net                no            5
ESS01R          yes      ess01io2.gpfs.net                no            5
```

The converted vdisk sets can be listed to examine their attributes and sizing:

```
# mmvdisk vdiskset list --vdisk-set all

                     member vdisks
vdisk set       count   size   raw size  created  file system and attributes
--------------  ----- -------- --------  -------  -------------------------
VS001_fs3           2   59 TiB   82 TiB  yes      fs3, DA1, 8+3p, 16 MiB, dataAndMetadata, system
VS002_fs4           2  204 TiB  282 TiB  yes      fs4, DA1, 8+3p, 16 MiB, dataAndMetadata, system
VS003_fs1           2  297 TiB  410 TiB  yes      fs1, DA1, 8+3p, 16 MiB, dataOnly, data
VS004_fs2           2   51 TiB   91 TiB  yes      fs2, DA1, 4+3p, 4 MiB, dataAndMetadata, system
VS005_fs1           2   11 TiB   45 TiB  yes      fs1, DA1, 4WayReplication, 1 MiB, metadataOnly, system
```

```
                declustered           capacity            all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
ESS01L          DA1            911 TiB    64 GiB     0%  VS001_fs3, VS002_fs4, VS003_fs1, VS004_fs2,
                                                          VS005_fs1
ESS01R          DA1            911 TiB    64 GiB     0%  VS001_fs3, VS002_fs4, VS003_fs1, VS004_fs2,
                                                          VS005_fs1

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
ESS01          29 GiB    16 GiB  VS001_fs3 (1448 MiB), VS002_fs4 (4994 MiB), VS003_fs1 (7249 MiB),
                                 VS004_fs2 (1921 MiB), VS005_fs1 (568 MiB)
```

The vdisk sets can be renamed to suit the administrator's preferences, and the **mmvdisk** file system characteristics can be listed:

```
# mmvdisk vdiskset rename --vdisk-set VS005_fs1 --new-name fs1meta
mmvdisk: Vdisk set 'VS005_fs1' renamed to 'fs1meta'.
# mmvdisk vdiskset rename --vdisk-set VS003_fs1 --new-name fs1data
mmvdisk: Vdisk set 'VS003_fs1' renamed to 'fs1data'.
# mmvdisk filesystem list --file-system fs1

                                vdisk     list of       holds    holds  storage
vdisk set       recovery group  count  failure groups  metadata  data    pool
--------------  --------------  -----  --------------  --------  -----  -------
fs1data         ESS01L              1               1  no        yes    data
fs1data         ESS01R              1               2  no        yes    data
fs1meta         ESS01L              1               1  yes       no     system
fs1meta         ESS01R              1               2  yes       no     system
```

Multiple ESS building blocks, with multiple recovery group pairs, must be converted one pair at a time. If a pre-**mmvdisk** file system spans multiple recovery group pairs, it is not recognized as an **mmvdisk** file system until all of the recovery groups with vdisk NSDs for the file system are converted.

## Replacing a pdisk using mmvdisk

If one or more pdisks in the recovery group is marked for replacement, the **mmvdisk recoverygroup list** command reports it with a `yes` in the `needs service` column.

In the following example, the BB01L recovery group needs service:

```
# mmvdisk recoverygroup list

                                                          needs    user
recovery group  active   current or master server         service  vdisks  remarks
--------------  -------  -------------------------------  -------  ------  -------
BB01L           yes      server01.gpfs.net                yes           3
BB01R           yes      server02.gpfs.net                no            3
```

This happens when the number of failed pdisks in one of the recovery group's declustered arrays reaches or exceeds the replacement threshold for the declustered array.

Pdisks that have reached the threshold for replacement are listed with **mmvdisk pdisk list --replace**:

```
# mmvdisk pdisk list --recovery-group all --replace

recovery group  pdisk         priority  FRU (type)       location
--------------  ------------  --------  ---------------  --------
BB01L           e2s11             1.15  00W1240          Enclosure 2 Drive 11
BB01L           e3s01             1.15  00W1240          Enclosure 3 Drive 1

mmvdisk: A lower priority value means a higher need for replacement.
```

To replace the physical disk of a failed recovery group pdisk, complete the following tasks:

1. Prepare the pdisk for replacement.
2. Remove the failed physical disk.
3. Insert a new physical disk.
4. Replace the pdisk with the newly inserted physical disk.

To prepare pdisk e2s11 of recovery group BB01L for replacement, run the following command:

```
# mmvdisk pdisk replace --prepare --recovery-group BB01L --pdisk e2s11
mmvdisk: Suspending pdisk e2s11 of RG BB01L in location SX32901810-11.
mmvdisk: Location SX32901810-11 is Enclosure 2 Drive 11.
mmvdisk: Carrier released.
mmvdisk:
mmvdisk:   - Remove carrier.
mmvdisk:   - Replace disk in location SX32901810-11 with type '00W1240'.
mmvdisk:   - Reinsert carrier.
mmvdisk:   - Issue the following command:
mmvdisk:
mmvdisk:   mmvdisk pdisk replace --recovery-group BB01L --pdisk 'e2s11'
```

Then, remove the failed physical disk and insert a new physical disk of the same FRU/type.

Finish replacing pdisk e2s11 with the new physical disk by running the following command:

```
# mmvdisk pdisk replace --recovery-group BB01L --pdisk e2s11
mmvdisk:
mmvdisk: Preparing a new pdisk for use may take many minutes.
mmvdisk:
mmvdisk: The following pdisks will be formatted on node ess01io1:
mmvdisk:       /dev/sdrk
mmvdisk:
mmvdisk: Location SX32901810-11 is Enclosure 2 Drive 11.

mmvdisk: Pdisk e2s11 of RG BB01L successfully replaced.
mmvdisk: Carrier resumed.
```

Repeat this procedure for any other pdisk that is marked for replacement.

## Limitations of mmvdisk

The following limitations apply to the **mmvdisk** command:

- The ESS GUI does not support creating new file systems if there are recovery groups that are under **mmvdisk** management.
- The **mmvdisk** command does not integrate the following IBM Spectrum Scale RAID features:
  - Component database management
  - Firmware management

  **Note:** These features must be managed using the component database management commands (for example, the **mmlscomp** command) and firmware management commands (for example, the **mmlsfirmware** command).

# Chapter 4. Managing IBM Spectrum Scale RAID

This section includes information about the overall management of IBM Spectrum Scale RAID. It also describes, in more detail, the characteristics and behaviors of these IBM Spectrum Scale RAID concepts and entities: declustered arrays, recovery groups, pdisks, pdisk-group fault tolerance, and vdisks.

You can use the ESS GUI to perform various IBM Spectrum Scale RAID management tasks.

## Recovery groups

A *recovery group* is the fundamental organizing structure employed by IBM Spectrum Scale RAID. A recovery group is conceptually the internal GPFS equivalent of a hardware disk controller. Within a recovery group, individual JBOD disks are defined as *pdisks* and assigned to *declustered arrays*. Each pdisk belongs to exactly one declustered array within one recovery group. Within a declustered array of pdisks, *vdisks* are defined. The vdisks are the equivalent of the RAID logical unit numbers (LUNs) for a hardware disk controller. One or two GPFS cluster nodes must be defined as the servers for a recovery group, and these servers must have direct hardware connections to the JBOD disks in the recovery group. Two servers are recommended for high availability server failover, but only one server will actively manage the recovery group at any given time. One server is the preferred and *primary* server, and the other server, if defined, is the *backup* server.

Multiple recovery groups can be defined, and a GPFS cluster node can be the primary or backup server for more than one recovery group. The name of a recovery group must be unique within a GPFS cluster.

## Recovery group server parameters

To enable a GPFS cluster node as a recovery group server, it must have the **mmchconfig** configuration parameter **nsdRAIDTracks** set to a nonzero value, and the GPFS daemon must be restarted on the node. The **nsdRAIDTracks** parameter defines the maximum number of vdisk track descriptors that the server can have in memory at a given time. The volume of actual vdisk data that the server can cache in memory is governed by the size of the GPFS page pool on the server and the value of the **nsdRAIDBufferPoolSizePct** configuration parameter. The **nsdRAIDBufferPoolSizePct** parameter defaults to 80% of the page pool on the server. A recovery group server should be configured with a substantial amount of page pool, on the order of tens of gigabytes. A recovery group server becomes an NSD server after NSDs are defined on the vdisks in the recovery group, so the **nsdBufSpace** parameter also applies. The default for **nsdBufSpace** is 30% of the page pool, and it can be decreased to its minimum value of 10% because the vdisk data buffer pool is used directly to serve the vdisk NSDs.

The vdisk track descriptors, as governed by **nsdRAIDTracks**, include such information as the RAID code, track number, and status. The descriptors also contain pointers to vdisk data buffers in the GPFS page pool, as governed by **nsdRAIDBufferPoolSizePct**. It is these buffers that hold the actual vdisk data and redundancy information.

For more information on how to set the **nsdRAIDTracks** and **nsdRAIDBufferPoolSizePct** parameters, see Appendix A, “Best-practice recommendations for IBM Spectrum Scale RAID,” on page 127.

For more information on the **nsdRAIDTracks**, **nsdRAIDBufferPoolSizePct**, and **nsdBufSpace** parameters, see the **mmchconfig** command in the *IBM Spectrum Scale: Command and Programming Reference*.

## Recovery group creation

Recovery groups are created using the **mmcrrecoverygroup** command, which takes the following arguments:

- The name of the recovery group to create.
- The name of a stanza file describing the declustered arrays and pdisks within the recovery group.
- The names of the GPFS cluster nodes that will be the primary and, if specified, backup servers for the recovery group.

When a recovery group is created, the GPFS daemon must be running with the **nsdRAIDTracks** configuration parameter in effect on the specified servers.

See the "mmcrrecoverygroup command" on page 159 for more information.

## Recovery group server failover

When, as is recommended, a recovery group is assigned two servers, one server is the preferred and *primary* server for the recovery group and the other server is the *backup* server. Only one server can serve the recovery group at any given time; this server is known as the *active* recovery group server. The server that is not currently serving the recovery group is the *standby* server. If the active recovery group server is unable to serve a recovery group, it will relinquish control of the recovery group and pass it to the standby server, if available. The failover from the active to the standby server should be transparent to any GPFS file system using the vdisk NSDs in the recovery group. There will be a pause in access to the file system data in the vdisk NSDs of the recovery group while the recovery operation takes place on the new server. This server failover recovery operation involves the new server opening the component disks of the recovery group and playing back any logged RAID transactions.

The active server for a recovery group can be changed by the IBM Spectrum Scale RAID administrator using the **mmchrecoverygroup** command. This command can also be used to change the primary and backup servers for a recovery group.

See the "mmchrecoverygroup command" on page 156 for more information.

# Pdisks

The IBM Spectrum Scale RAID *pdisk* is an abstraction of a physical disk. A pdisk corresponds to exactly one physical disk, and belongs to exactly one declustered array within exactly one recovery group. Before discussing how declustered arrays collect pdisks into groups, it will be useful to describe the characteristics of pdisks.

A recovery group can contain a maximum of 512 pdisks. A declustered array within a recovery group can contain a maximum of 256 pdisks. The name of a pdisk must be unique within a recovery group; that is, two recovery groups can each contain a pdisk named `disk10`, but a recovery group cannot contain two pdisks named `disk10`, even if they are in different declustered arrays.

A pdisk is usually created using the **mmcrrecoverygroup** command, whereby it is assigned to a declustered array within a newly created recovery group. In unusual situations, pdisks can also be created and assigned to a declustered array of an existing recovery group by using the **mmaddpdisk** command.

To create a pdisk, a stanza must be supplied to the **mmcrrecoverygroup** or **mmaddpdisk** commands specifying the pdisk name, the declustered array name to which it is assigned, and a block device special file name for the entire physical disk as it is configured by the operating system on the active recovery group server. A sample pdisk creation stanza follows:

```
%pdisk: pdiskName=c073d1
        device=/dev/hdisk192
        da=DA1
        nPathActive=2
        nPathTotal=4
```

Other stanza parameters might be present. For more information about pdisk stanza parameters, see "Pdisk stanza format" on page 36.

The device name for a pdisk must refer to the entirety of a single physical disk; pdisks should not be created using virtualized or software-based disks (for example, logical volumes, disk partitions, logical units from other RAID controllers, or network-attached disks). The exception to this rule are non-volatile RAM (NVRAM) volumes used for the log tip vdisk, which is described in "Log vdisks" on page 42. For a pdisk to be created successfully, the physical disk must be present and functional at the specified device name on the active server. The physical disk must also be present on the standby recovery group server, if one is configured. The physical disk block device special name on the standby server will almost certainly be different, and will be discovered automatically by IBM Spectrum Scale.

The attributes of a pdisk include the physical disk's unique worldwide name (WWN), its field replaceable unit (FRU) code, and its physical location code. Pdisk attributes can be displayed using the **mmlspdisk** command; of particular interest here are the pdisk device *paths* and the pdisk *states*.

Pdisks that have failed and have been marked for replacement by the disk hospital are replaced using the **mmchcarrier** command. In unusual situations, pdisks can be added or deleted using the **mmaddpdisk** or **mmdelpdisk** commands. When deleted, either through replacement or the **mmdelpdisk** command, the pdisk abstraction will only cease to exist when all of the data it contained has been rebuilt onto spare space (even though the physical disk might have been removed from the system).

Pdisks are normally under the control of IBM Spectrum Scale RAID and the disk hospital. In some situations, however, the **mmchpdisk** command can be used to manipulate pdisks directly. For example, if a pdisk has to be removed temporarily to allow for hardware maintenance on other parts of the system, you can use the **mmchpdisk --begin-service-drain** command to drain the data before removing the pdisk. After bringing the pdisk back online, you can use the **mmchpdisk --end-service-drain** command to return the drained data to the pdisk.

**Note:** This process requires that there be sufficient spare space in the declustered array for the data that is to be drained. If the available spare space is insufficient, it can be increased with the **mmchrecoverygroup** command.

## Pdisk paths

To the operating system, physical disks are made visible as block devices with device special file names, such as `/dev/sdbc` (on Linux) or `/dev/hdisk32` (on AIX®). Most pdisks that IBM Spectrum Scale RAID uses are located in JBOD arrays, except for the NVRAM pdisk that is used for the log tip vdisk. To achieve high availability and throughput, the physical disks of a JBOD array are connected to each server by multiple (usually two) interfaces in a configuration known as *multipath* (or *dualpath*). When two operating system block devices are visible for each physical disk, IBM Spectrum Scale RAID refers to them as the *paths* to the pdisk.

In normal operation, the paths to individual pdisks are discovered by IBM Spectrum Scale RAID automatically. There are only two instances when a pdisk must be referred to by its explicit block device path name: during recovery group creation using the **mmcrrecoverygroup** command, and when adding new pdisks to an existing recovery group with the **mmaddpdisk** command. In both of these cases, only one of the block device path names as seen on the active server needs to be specified; any other paths on the active and standby servers will be discovered automatically. For each pdisk, the `nPathActive` and

nPathTotal stanza parameters can be used to specify the expected number of paths to that pdisk, from the active server and from all servers. This allows the disk hospital to verify that all expected paths are present and functioning.

The operating system could have the ability to internally merge multiple paths to a physical disk into a single block device. When IBM Spectrum Scale RAID is in use, the operating system multipath merge function must be disabled because IBM Spectrum Scale RAID itself manages the individual paths to the disk. For more information, see "Configuring GNR recovery groups: a sample scenario" on page 275.

## Pdisk stanza format

Pdisk stanzas have three mandatory parameters and six optional parameters, and they look like this:

```
%pdisk: pdiskName=PdiskName
        device=BlockDeviceName
        da=DeclusteredArrayName
       [nPathActive=ExpectedNumberActivePaths]
       [nPathTotal=ExpectedNumberTotalPaths]
       [rotationRate=HardwareRotationRate]
       [fruNumber=FieldReplaceableUnitNumber]
       [location=PdiskLocation]
       [nsdFormatVersion=DesiredNsdFormatVersion]
```

where:

**pdiskName**=*PdiskName*
: Specifies the name of a pdisk.

**device**=*BlockDeviceName*
: Specifies the name of a block device. The value provided for *BlockDeviceName* must refer to the block device as configured by the operating system on the primary recovery group server or have the node name prefixed to the device block name.

  Sample values for *BlockDeviceName* are `/dev/sdbc` and `//nodename/dev/sdbc` (on Linux), and `hdisk32`, `/dev/hdisk32` and `//nodename/dev/hdisk32` (on AIX).

  Only one *BlockDeviceName* needs to be used, even if the device uses multipath and has multiple device names.

**da**=*DeclusteredArrayName*
: Specifies the *DeclusteredArrayName* in the pdisk stanza, which implicitly creates the declustered array with default parameters.

**nPathActive**=*ExpectedNumberActivePaths*
: Specifies the expected number of paths for the connection from the active server to this pdisk. If this parameter is specified, the **mmlsrecoverygroup** and **mmlspdisk** commands will display warnings if the number of paths does not match the expected number for a pdisk that should be functioning normally. If this parameter is not specified, the default is 0, which means "do not issue such warnings".

  Sample values are 2 for all pdisks that are located in an ESS disk enclosure (or the IBM Power 775 Disk Enclosure) and 1 for the NVRAM pdisk that is used for the log tip vdisk.

**nPathTotal**=*ExpectedNumberTotalPaths*
: Specifies the expected number of paths for the connection from all active and backup servers to this pdisk. If this parameter is specified, the **mmlsrecoverygroup** and **mmlspdisk** commands will display warnings if the number of paths does not match the expected number, for a pdisk that should be functioning normally. If this parameter is not specified, the default is 0, which means "do not issue such warnings".

  Sample values are 4 for all pdisks located in an ESS disk enclosure (or the IBM Power 775 Disk Enclosure) and 1 for the NVRAM pdisk used for the log tip vdisk.

**rotationRate**=*HardwareRotationRate*
Specifies the hardware type of the pdisk: NVRAM, SSD, or a rotating HDD. The only valid values are the string `NVRAM`, the string `SSD`, or a number between 1025 and 65535 (inclusive) indicating the rotation rate in revolutions per minute for HDDs. For all pdisks that are used in an ESS disk enclosure (or the IBM Power 775 Disk Enclosure), there is no need to specify this parameter, as the hardware type and rotation rate will be determined from the hardware automatically. This parameter should only be specified for the NVRAM pdisk on the ESS. The default is to rely on the hardware to identify itself, or leave the hardware type and rotation rate unknown if the hardware does not have the ability to identify itself.

A sample value is the string `NVRAM` for the NVRAM pdisk used for the log tip vdisk.

**fruNumber**=*FieldReplaceableUnitNumber*
Specifies the unit number for the field replaceable unit that is needed to repair this pdisk if it fails. For all pdisks used in an ESS disk enclosure (or the IBM Power 775 Disk Enclosure), there is no need to specify this parameter, as it is automatically determined from the hardware. For the NVRAM pdisk used in the log tip vdisk, the user can enter a string here, which will be displayed to service personnel when replacement of that pdisk is performed. However, setting this value for the NVRAM pdisk is not required, as the service replacement procedure for that pdisk is specific to that particular type of hardware. The default is to rely on the hardware to identify itself, or to leave the FRU number unknown if the hardware does not have the ability to identify itself.

**location**=*PdiskLocation*
Specifies the physical location of this pdisk. For all pdisks used in an ESS disk enclosure (or the IBM Power 775 Disk Enclosure), there is no need to specify this parameter, as it automatically determined from the hardware. For the NVRAM pdisk used in the log tip vdisk, the user can enter a string here, which will be displayed in the output of mmlspdisk. The default is to rely on the location reported by the hardware, or leave the location unknown.

A sample value is SV21314035-5-1, which describes a pdisk in enclosure serial number SV21314035, drawer 5, slot 1.

**nsdFormatVersion**=*DesiredNsdFormatVersion*
Specifies the desired Nsd Format version for this pdisk. The value can be either 1 or 2. This parameter is only effective with recovery group version 4.2.0.1 or later. If this parameter is not specified, the pdisk Nsd version will be 2 for recovery group version 4.2.0.1 or later. For recovery group version 4.1.0.1 or earlier, the Nsd version can only be 1.

## Pdisk states

IBM Spectrum Scale RAID maintains its view of a pdisk and its corresponding physical disk by using a *pdisk state*. The pdisk state consists of multiple keyword flags, which can be displayed by using the **mmlsrecoverygroup** or **mmlspdisk** commands. You can also use the ESS GUI to display pdisk states. The state of pdisks is displayed in these views: **Arrays > Physical**, **Monitoring > System**, and **Monitoring > System Details**. In addition, information about pdisks with a negative state (disks that should be replaced, for example) is displayed in the **Monitoring > Events** view.

The pdisk state flags indicate in detail how IBM Spectrum Scale RAID is using or managing a disk.

In normal circumstances, the state of most of the pdisks is represented by the sole keyword `ok`. The `ok` keyword means that IBM Spectrum Scale RAID considers the pdisk to be healthy: the recovery group server is able to communicate with the disk, the disk is functioning normally, and the disk can be used to store data. The `diagnosing` flag is present in the pdisk state when the IBM Spectrum Scale RAID disk hospital suspects, or attempts to correct, a problem. If IBM Spectrum Scale RAID is unable to communicate with a disk, the pdisk state includes the `missing` keyword. If a `missing` disk becomes reconnected and functions properly, its state changes back to `ok`. The `readonly` flag means that a disk is indicating that it can no longer safely write data. A disk can also be marked by the disk hospital as `failing`, which might be due to an excessive number of media or checksum errors. When the disk

hospital concludes that a disk is no longer operating effectively, it declares the disk `dead`. If the number of non-functioning (`dead`, `missing`, `failing`, or `slow`) pdisks reaches or exceeds the replacement threshold of their declustered array, the disk hospital adds the `replace` flag to the pdisk state, which indicates that physical disk replacement should be performed as soon as possible.

When the state of a pdisk indicates that it can no longer behave reliably, IBM Spectrum Scale RAID rebuilds the pdisk's data onto spare space on the other pdisks in the same declustered array. This is called *draining* the pdisk. Flags indicate whether a pdisk is draining or was drained. The `draining` flag means that IBM Spectrum Scale RAID will rebuild the data from the pdisk. The `deleting` flag means that the IBM Spectrum Scale RAID administrator issued the **mmdelpdisk** command to delete the pdisk.

To summarize, most of the pdisks are in the `ok` state during normal operation. The `ok` state indicates that the disk is reachable, functioning, not draining, and that the disk contains user data as well as IBM Spectrum Scale RAID recovery group and vdisk configuration information. A more complex example of a pdisk state is `dead/drained` for a single pdisk that has failed. This set of pdisk state flags indicates that the pdisk was declared dead by the system, was marked to be drained, and that all of its data (recovery group, vdisk configuration, and user) was successfully rebuilt onto the spare space on other pdisks.

In addition to the states discussed here, there are some transient pdisk states that have little impact on normal operations. Table 3 lists the complete set of states.

*Table 3. Pdisk states*

| State | Description |
|---|---|
| `ok` | The disk is available. |
| `dead` | The disk completely failed. |
| `simulatedDead` | The disk is being treated as if it were dead for error injection (see **mmchpdisk --simulate-dead**). |
| `missing` | The disk hospital determined that the system cannot connect to the drive. |
| `readonly` | The disk has failed; it can still be read but not written. |
| `failing` | The disk needs to be drained and replaced due to a SMART trip or high uncorrectable error rate. |
| `simulatedFailing` | The disk is being treated as if it were failing for error injection (see **mmchpdisk --simulate-failing**). |
| `slow` | The disk needs to be drained and replaced due to poor performance. |
| `diagnosing` | The disk hospital is checking the disk after an error. |
| `PTOW` | The disk is temporarily unavailable because of a pending timed-out write. |
| `suspended` | The disk is temporarily offline for service (see **mmchpdisk** and **mmchcarrier**). |
| `serviceDrain` | The disk is being drained of data for service (see **mmchpdisk --begin-service-drain**). |
| `draining` | The data is being drained from the disk and moved to distributed spare space on other disks. |
| `deleting` | The disk is being deleted from the system through the **mmdelpdisk**, **mmaddpdisk/--replace**, or **mmchcarrier** command. |
| `drained` | All of the data was successfully drained from the disk and the disk is replaceable, but the replace threshold was not met. |
| `undrainable` | As much of the data as possible was drained from the disk and moved to distributed spare space. |
| `replace` | The disk is ready for replacement. |

### Related information

- "mmdelpdisk command" on page 173
- "mmlspdisk command" on page 199
- "mmlsrecoverygroup command" on page 202

## Declustered arrays

**Note:** The features available for actual declustered arrays will depend on the specific supported disk configuration in an IBM Spectrum Scale RAID installation.

Declustered arrays are disjoint subsets of the pdisks in a recovery group. Vdisks are created within declustered arrays, and vdisk tracks are declustered across all of an array's pdisks. The number of declustered arrays in a recovery group is determined by the enclosure and disk hardware configuration of the IBM Spectrum Scale RAID server for the recovery group. Depending on the specific configuration, an individual declustered array may contain up to 256 pdisks; however, the total number of pdisks in all declustered arrays within a recovery group may not exceed 512. A pdisk can belong to only one declustered array. The name of a declustered array must be unique within a recovery group; that is, two recovery groups can each contain a declustered array named `DA3`, but a recovery group cannot contain two declustered arrays named `DA3`. The pdisks within a declustered array must all be of the same size and should all have similar performance characteristics.

A declustered array is usually created together with its member pdisks and its containing recovery group through the use of the **mmchrecoverygroup** command. A declustered array can also be created using the **mmaddpdisk** command to add pdisks to a declustered array that does not yet exist in a recovery group. A declustered array may be deleted by deleting its last member pdisk, or by deleting the recovery group in which it resides. Any vdisk NSDs and vdisks within the declustered array must already have been deleted. There are no explicit commands to create or delete declustered arrays.

The main purpose of a declustered array is to segregate pdisks of similar performance characteristics and similar use. Because vdisks are contained within a single declustered array, mixing pdisks of varying performance within a declustered array would not use the disks optimally. In a typical IBM Spectrum Scale RAID system, the first declustered array contains SSD pdisks that are used for the log vdisk, or the log backup vdisk if configured. If the system is configured to use a log tip vdisk (see "Log vdisks" on page 42), another declustered array contains NVRAM pdisks for that vdisk. Vdisks that are GPFS NSDs are then contained in one or more declustered arrays using high-capacity HDDs or SSDs.

A secondary purpose of declustered arrays is to partition disks that share a common point of failure or unavailability, such as removable carriers that hold multiple disks. This comes into play when one considers that removing a multi-disk carrier to perform disk replacement also temporarily removes some good disks, perhaps a number in excess of the fault tolerance of the vdisk NSDs. This would cause temporary suspension of file system activity until the disks are restored. To avoid this, each disk position in a removable carrier should be used to define a separate declustered array, such that disk position one defines `DA1`, disk position two defines `DA2`, and so on. Then when a disk carrier is removed, each declustered array will suffer the loss of just one disk, which is within the fault tolerance of any IBM Spectrum Scale RAID vdisk NSD.

## Declustered array parameters

Declustered arrays have four parameters that can be set using stanza parameters when creating a declustered array, and can be changed using the **mmchrecoverygroup** command with the **--declustered-array** option. These are:

**dataSpares**
The number of disks' worth of equivalent spare space used for rebuilding vdisk data if pdisks fail. This defaults to one for arrays with nine or fewer pdisks, and two for arrays with 10 or more pdisks.

**vcdSpares**
The number of disks that can be unavailable while the IBM Spectrum Scale RAID server continues to function with full replication of vdisk configuration data (VCD). This value defaults to the number of data spares. To enable pdisk-group fault tolerance, this parameter is typically set to a larger value during initial system configuration (half of the number of pdisks in the declustered array + 1, for example).

**replaceThreshold**
The number of disks that must fail before the declustered array is marked as needing to have disks replaced. The default is the number of data spares.

**scrubDuration**
The number of days over which all the vdisks in the declustered array are scrubbed for errors. The default is 14 days.

## Declustered array size

IBM Spectrum Scale RAID distinguishes between large and small declustered arrays. A declustered array is considered *large* if, at the time of its creation, the number of its pdisks is at least the setting of the `vcdSpares` parameter + 9. When using the default values for the `vcdSpares` and `dataSpares` parameters, this means that a declustered array is considered large if it contains at least 11 pdisks. All other declustered arrays are considered *small*. At least one declustered array in each recovery group must be large, because only large declustered arrays have enough pdisks to safely store an adequate number of replicas of the IBM Spectrum Scale RAID configuration data for the recovery group.

Because the narrowest RAID code that IBM Spectrum Scale RAID supports for user data is 3-way replication, the smallest possible declustered array contains four pdisks, including the minimum required equivalent spare space of one disk. The RAID code width of the intended vdisk NSDs and the amount of equivalent spare space also affect declustered array size; if Reed-Solomon 8 + 3p vdisks, which have a code width of 11, are required, and two disks of equivalent spare space is also required, the declustered array must have at least 13 member pdisks. Declustered arrays that contain only log vdisks can be smaller than these limits.

## Data spare space and VCD spares

While operating with a failed pdisk in a declustered array, IBM Spectrum Scale RAID continues to serve file system I/O requests by using redundancy information on other pdisks to reconstruct data that cannot be read, and by marking data that cannot be written to the failed pdisk as stale. Meanwhile, to restore full redundancy and fault tolerance, the data on the failed pdisk is rebuilt onto *data spare space*, reserved unused portions of the declustered array that are declustered over all of the member pdisks. The failed disk is thereby *drained* of its data by copying it to the data spare space.

The amount of data spare space in a declustered array is set at creation time and can be changed later. The data spare space is expressed in whole units equivalent to the capacity of a member pdisk of the declustered array, but is spread among all of the member pdisks. There are no dedicated spare pdisks. This implies that a number of pdisks equal to the specified data spare space could fail, and the full redundancy of all of the data in the declustered array can be restored through a rebuild operation. If the user chooses to not fill the space in the declustered array with vdisks, and wants to use the unallocated space as extra data spare space, the user can increase the setting of the `dataSpares` parameter to the desired level of resilience against pdisk failures.

At a minimum, each declustered array normally requires data spare space that is equivalent to the size of one member pdisk. The exceptions, which have zero data spares and zero VCD spares, are declustered arrays that consist of the following:

- Non-volatile RAM disks used for a log tip vdisk
- SSDs used for a log tip backup vdisk.

Because large declustered arrays have a greater probability of disk failure, the default amount of data spare space depends on the size of the declustered array. A declustered array with nine or fewer pdisks defaults to having one disk of equivalent data spare space. A declustered array with 10 or more disks defaults to having two disks of equivalent data spare space. These defaults can be overridden, especially at declustered array creation. However, if at a later point too much of the declustered array is already allocated for use by vdisks, it might not be possible to increase the amount of data spare space.

IBM Spectrum Scale RAID *vdisk configuration data (VCD)* is stored more redundantly than vdisk content, typically 5-way replicated. When a pdisk fails, this configuration data is rebuilt at the highest priority, onto functioning pdisks. The redundancy of configuration data always has to be maintained, and IBM Spectrum Scale RAID will not serve a declustered array that does not have sufficient pdisks to store all configuration data at full redundancy. The declustered array parameter `vcdSpares` determines how many pdisks can fail and have full VCD redundancy restored, by reserving room on each pdisk for vdisk configuration data. When using pdisk-group fault tolerance, the value of `vcdSpares` should be set higher than the value of the `dataSpares` parameter to account for the expected failure of hardware failure domains.

## Increasing VCD spares

When new recovery groups are created, the **mkrginput** script sets recommended values for VCD spares.

To increase the VCD spares for existing recovery groups, use the **mmchrecoverygroup** command. See the "mmchrecoverygroup command" on page 156 for more information.

## Declustered array free space

The declustered array free space reported by the **mmlsrecoverygroup** command reflects the space available for creating vdisks. Spare space is not included in this value since it is not available for creating new vdisks.

## Pdisk free space

The pdisk free space reported by the **mmlsrecoverygroup** command reflects the actual number of unused data partitions on the disk. This includes spare space, so if a pdisk fails, these values will decrease as data is moved to the spare space.

---

## Vdisks

Vdisks are created across the pdisks within a declustered array. Each recovery group requires a special *log home vdisk* to function (along with other log-type vdisks, as appropriate for specific environments); see "Log vdisks" on page 42. All other vdisks are created for use as GPFS file system NSDs.

A recovery group can contain at most 64 vdisks. Vdisks can be allocated arbitrarily among declustered arrays. Vdisks are created with the **mmcrvdisk** command. The **mmdelvdisk** command destroys vdisks and all their contained data.

When creating a vdisk, you must specify the RAID code, block size, vdisk size, and a name that is unique within the recovery group and the GPFS cluster. There are no adjustable parameters available for vdisks.

## RAID code

The type, performance, and space efficiency of the RAID codes that are used for vdisks, discussed in "RAID codes" on page 2, should be considered when choosing the RAID code for a particular set of user data. GPFS storage pools and policy-based data placement can be used to ensure that data is stored with appropriate RAID codes.

If an IBM Spectrum Scale RAID server cannot serve a vdisk due to too many unreachable pdisks (pdisks in the missing state), the server fails over the recovery group to the backup server in hopes that the backup server has better I/O connectivity. If the backup server cannot serve vdisks either due to too many missing pdisks, it fails the recovery group back to the primary server, which repeats the cycle until I/O connectivity improves. This approach is aimed at making the system robust to temporary pdisk connectivity problems by allowing the administrator to restore pdisk connectivity before the system fails clients I/Os due to missing pdisks.

If the problem is caused by too many pdisks in other non-functioning states (for example, dead state), the recovery group will not failover to the partner node because the pdisk state is not expected to improve. It is expected to remain in the same state even if tried by the partner node. In this case, the client I/Os that are affected by too many such pdisks will fail.

A recovery group can contain vdisks of different levels of fault tolerances. In this case, the recovery group might contain some vdisks with sufficient fault tolerance that could be served despite the missing pdisks. Nevertheless, the vdisks might fail over to the partner server because the unit of failover is a recovery group with all its associated vdisks. Mixing vdisks of varying fault tolerance levels in the same recovery group is allowed; however, in the presence of too many unreachable pdisks, service of vdisks with higher fault tolerance might be limited by vdisks with lower fault tolerance.

## Block size

The vdisk block size must equal the GPFS file system block size of the storage pool where the vdisk is assigned. For replication codes, the supported vdisk block sizes are 256 KiB, 512 KiB, 1 MiB, and 2 MiB. For Reed-Solomon codes, the supported vdisk block sizes are 512 KiB, 1 MiB, 2 MiB, 4 MiB, 8 MiB, and 16 MiB. See Appendix A, "Best-practice recommendations for IBM Spectrum Scale RAID," on page 127 for some vdisk configuration considerations.

The **maxblocksize** configuration attribute of the IBM Spectrum Scale **mmchconfig** command must be set appropriately for all nodes. The value of **maxblocksize** must be greater than or equal to the maximum block size of the vdisks. For more information about this attribute, see the **mmchconfig** command description in the *IBM Spectrum Scale: Command and Programming Reference*.

## Vdisk size

The maximum vdisk size is the total space available on the pdisks in the declustered array, taking into account the overhead of the RAID code, minus spare space, minus vdisk configuration data, and minus a small amount of space reserved as a buffer for write operations. IBM Spectrum Scale RAID will round up the requested vdisk size as required. When creating a vdisk, the user can specify to use all remaining space in the declustered array for that vdisk.

## Log vdisks

IBM Spectrum Scale RAID uses log vdisks to store such internal information as event log entries, updates to vdisk configuration data, and certain data write operations quickly. There are four types of log vdisks, as follows. Among them, they can be created and destroyed in any order.

**log home vdisk**
Every recovery group requires one log home vdisk to function. The log home vdisk must be created before any other non-log vdisks in the recovery group, and it can only be deleted after all other non-log vdisks in the recovery group have been deleted. The log home vdisk is divided into four sublogs: long-term event log, short-term event log, metadata log, and fast-write log to log small write operations.

**log tip vdisk**
The log tip vdisk (appropriate for certain environments, but not required for all) is a vdisk to which log records are initially written, then migrated to the log home vdisk. The intent is to use a small, high-performance NVRAM device for the log tip, and a larger vdisk on conventional spinning disks for the log home vdisk. The fast writes to the log tip hide the latency of the spinning disks used for the main body of the log.

**log tip backup vdisk**
The log tip backup vdisk (appropriate for certain environments, but not required for all) is used as an additional replica of the log tip vdisk when the log tip vdisk is two-way replicated on nonvolatile RAM disks. Ideally, the log tip backup vdisk provides a level of performance between that of NVRAM disks and that of spinning disks.

**log reserved vdisk**
Log reserved vdisks are optional vdisks that are used when the log home disk is not allocated in its own declustered array. Log reserved vdisks have the same size as the log home vdisk and are used to equalize the space consumption on the data declustered arrays, but they are otherwise unused.

## Typical configurations

The following are descriptions of typical vdisk configurations in various recovery group environments:

**ESS** *without* **NVRAM disks**
In this configuration, a three-way replicated log tip vdisk is allocated on a declustered array made up of three SSDs. A four-way replicated log home vdisk is allocated in the first declustered array of HDDs.

**ESS** *with* **NVRAM disks**
In this configuration, a two-way replicated log tip vdisk is allocated on NVRAM disks, one from each of the servers.

A log tip backup vdisk is allocated on a declustered array of one or more SSDs. This provides an additional copy of the log tip data when one of the NVRAM disks is unavailable. If only one SSD is used, then the log tip backup uses a RAID code of **Unreplicated**.

A four-way replicated log home vdisk is allocated in the first declustered array of HDDs. A four-way replicated log reserved vdisk for each of the data declustered arrays that do not contain the log home vdisk.

**ESS** *with* **NVRAM disks, using SSDs for data**
In this configuration, a two-way replicated log tip vdisk is allocated on NVRAM disks, one from each of the servers.

All SSDs for a recovery group form a single declustered array, containing the log home vdisk and user data vdisks. No log tip backup disk is used.

**Power 775 configuration**
In this configuration, the log home vdisk is allocated on a declustered array made up of four SSDs. Only three-way and four-way replication codes are supported for the log home vdisk. In the typical system with four SSDs and with spare space equal to the size of one disk, the three-way replication code would be used for the log home vdisk

## Creating vdisks and NSDs

You can create vdisks and NSDs using IBM Spectrum Scale RAID commands or using the ESS GUI.

After you create a vdisk using the **mmcrvdisk** command, you can create NSDs using the **mmcrnsd** command.

Alternatively, when you use the ESS GUI to create a file system, the GUI creates the vdisks and NSDs as well. NSDs and vdisks that are created using IBM Spectrum Scale RAID commands are ignored by the ESS GUI and cannot be used for creating a file system using the ESS GUI file system creation wizard, which is launched by using the **Create File System** action in the **Files > File Systems** view. Users who plan to create file systems using the ESS GUI must not create vdisks and NSDs using IBM Spectrum Scale RAID commands.

The relationship between vdisks and NSDs is as follows:

- GPFS file systems are built from vdisk NSDs in the same way as they are built from any other NSDs.
- While an NSD exists for a vdisk, that vdisk cannot be deleted.
- A node cannot serve vdisk-based NSDs and non-vdisk-based NSDs.
- Vdisk NSDs should not be used as tiebreaker disks.

For more information about:

- The **mmcrvdisk** command, see "mmcrvdisk command" on page 162
- The **mmcrnsd** command, see the *IBM Spectrum Scale: Command and Programming Reference*
- Vdisk and NSD best practices, see Appendix A, "Best-practice recommendations for IBM Spectrum Scale RAID," on page 127.

## Vdisk states

IBM Spectrum Scale RAID reports vdisk states, which are displayed using the **mmlsvdisk** command and also on the **mmlsrecoverygroup** command if the **-L** option is used. You can also use the ESS GUI to display vdisk states. The state of vdisks is displayed in the **Arrays > Volumes** view.

Vdisks are normally in the **ok** state, which indicates that the vdisk is fully functional with full redundancy. When a pdisk failure affects a specific vdisk, the vdisk state will be reported as degraded until the affected tracks have been rebuilt onto spare space.

When enough pdisks have failed that the specific vdisk has no redundancy left, the vdisk state will be reported as `critical` until the critically affected tracks have been rebuilt onto spare space. If the system uses up all the available spare space while a vdisk is in the degraded or critical state, the state will be followed by `(need spare)`, which indicates that rebuild activity is stalled, and will resume once the failed pdisks have been replaced.

*Table 4. Vdisk states*

| State | Description |
|---|---|
| `ok` | The vdisk is functioning normally. |
| *m*`/`*n* `-degraded` | The vdisk is currently running in degraded mode:<br>*m* is the number of pdisks that are draining<br>*n* is the fault-tolerance of the vdisk. |
| `critical` | The vdisk is currently running in degraded mode and can tolerate no more pdisk losses. |
| `(need spare)` | Rebuild will resume when more spare space is available; applies to degraded and critical states. |

## Determining pdisk-group fault-tolerance

You can obtain a synopsis of the current layout for user and system data by running this command:
`mmlsrecoverygroup RecoveryGroupName -L`

The output of this command includes a synopsis for the following:
- configuration data rebuild space
- configuration data recovery group descriptor layout
- configuration data system index layout
- user data vdisk layout

**Note:** The actual and maximum pdisk-group fault-tolerance values are point-in-time calculations based on the available disk space and current disk hardware configuration. These values could change whenever a rebuild or rebalance operation occurs.

Here is some sample output:

```
config data         declustered array   VCD spares     actual rebuild spare space         remarks
-----------------   -----------------   ------------   --------------------------------   ----------------
rebuild space       da0                 2              1 enclosure                        limited by VCD spares

config data         max disk group fault tolerance     actual disk group fault tolerance  remarks
-----------------   --------------------------------   --------------------------------   ----------------
rg descriptor       1 enclosure + 1 pdisk              1 enclosure + 1 pdisk
system index        2 enclosure                        1 enclosure                        limited by rebuild space

vdisk               max disk group fault tolerance     actual disk group fault tolerance  remarks
-----------------   --------------------------------   --------------------------------   ----------------
rg00log             2 enclosure                        1 enclosure                        limited by rebuild space
rg00meta            3 enclosure                        1 enclosure                        limited by rebuild space
rg00data            1 enclosure                        1 enclosure
```

This sample output from the **mmlsrecoverygroup** *RecoveryGroupName* **-L** command includes:
- a `config data` section with a `rebuild space` entry, which shows the available rebuild space that can be used to restore redundancy of the configuration data after disk failure
- a `config data` section with:
  - an `rg descriptor` entry, which shows the recovery group descriptor layout
  - a `system index` entry, which shows the system index layout
- a `vdisk` section, which shows a set of user-defined vdisk layouts.

This sample output lists the exact type of failure we would have survived in the `actual disk group fault tolerance` column:

```
vdisk         max disk group fault tolerance  actual disk group fault tolerance  remarks
----------  --------------------------------   --------------------------  -------------
.
.
.
rg00data       1 enclosure                       1 enclosure
```

For the `rg00data` vdisk, the output shows that we could survive one enclosure failure in the `actual disk group fault tolerance` column.

Note that for some vdisks in the sample output, there is a difference in maximum versus actual disk group fault tolerance:

```
vdisk       max disk group fault tolerance   actual disk group fault tolerance  remarks
----------  ------------------------------  --------------------------------  ----------------
.
```

```
.
.
rg00meta   3 enclosure                    1 enclosure                        limited by rebuild space
```

This example indicates that even though the `rg00meta` vdisk has a layout that ensures its data is protected from three enclosure failures, as shown in the `max disk group fault tolerance` column, its actual disk group fault tolerance is one enclosure and it is being limited by rebuild space dependency. Effectively, the configuration data rebuild space is not sufficient and is limiting the disk group fault tolerance of its vdisk dependent.

In cases where the actual disk group fault tolerance is less than the maximum disk group fault tolerance, it is best to examine the dependency change that is shown in the `remarks` column. For this example, you would examine the `rebuild space` entry under the `config data` section:

```
config data    declustered array   VCD spares     actual rebuild spare space      remarks
-----------  ------------------  -------------  -----------------------------  ---------------
rebuild space    da0               2              1 enclosure                    limited by VCD spares
```

This section indicates that we are being limited by the VCD spares. It is possible to increase the VCD spares and thus increase the actual rebuild spare space. If the rebuild space increases above 1 enclosure, it would no longer be the limiting dependency factor for the `rg00meta` vdisk. See "Increasing VCD spares" on page 41 for more information.

Note that the initial allocation during vdisk creation and subsequent redistribution of strips of the redundancy code in the event of failure or disk group changes is a dynamic operation. As such, it must work within the space available to it during the operation. The goals of this operation are always to minimize unnecessary movement, while at the same time balancing fault tolerance for all user and configuration data. Because of the vagaries of available space during initial allocation versus redistribution and because of the wide possible range of vdisk redundancy codes within the system, an initial fault tolerance state might not be repeatable after disk group faults and rebuilds occur.

For more information about **mmlsrecoverygroup** command output, see "mmlsrecoverygroup command" on page 202.

# Chapter 5. Configuring components on the Elastic Storage Server

In an ESS system, IBM Spectrum Scale uses component information in the cluster configuration data to perform configuration and validation tasks. For example, when a disk needs to be replaced, the system reports the rack location of the enclosure containing the disk along with a description of the disk's slot within the enclosure.

A component in this context refers to some resource, usually hardware, that is part of the GPFS cluster; for example, **rack** or **storage enclosure**. Each component has an associated component specification that is identified by its part number. You can use the **mmlscompspec** command to see the defined component specifications.

You will normally configure components when deploying a cluster or when adding new hardware to a cluster. The cluster must exist before you can run any of the component-related commands.

If you use IBM Spectrum Scale RAID commands to configure components, the basic configuration steps are as follows:

1. Add components to the configuration using **mmdiscovercomp** and **mmaddcomp**.
2. Define the location of storage enclosures and servers using **mmchcomploc**.
3. Set the two-digit ID visible on the back of some storage enclosures using **mmsyncdisplayid**.
4. Update the component names and other attributes using **mmchcomp**.

The following commands use or change the cluster's component configuration:

**mmaddcomp**
: Adds new components

**mmchcomp**
: Change component attributes

**mmchcomploc**
: Change component locations

**mmdelcomp**
: Delete components

**mmdiscovercomp**
: Discover components and add them to the configuration

**mmlscomp**
: List components.

**mmlscomploc**
: List component locations

**mmlscompspec**
: List component specifications

**mmsyncdisplayid**
: Synchronize enclosure display IDs with the cluster configuration

For information about the options that these commands support, see Appendix B, “IBM Spectrum Scale RAID commands,” on page 129. Most commands allow you to make a single change using command-line arguments. They also support stanza file input for making any number of changes with a single command.

Alternatively, instead of using the IBM Spectrum Scale RAID commands described in this chapter, you can optionally perform initial component configuration using the ESS GUI's system setup wizard, which is launched automatically after you log in to the ESS GUI for the first time.

You can use the ESS GUI to edit component configuration at a later time. You can do this in the **Actions > Edit Rack Components** view or the **Monitoring > System** view.

## Adding components to the cluster's configuration

This section discusses how to add components to the cluster's configuration in an ESS system. Sample output in this topic might not match the output produced on your system.

Before adding any components, you might want to view the available component specifications. To do so, you can use the **mmlscompspec** command, which lists the defined component specifications, identified by part number.

```
$ mmlscompspec

    Rack Specifications

Part Number  Height  Description
-----------  ------  -------------------------------------------------
1410HEA          42  42U 1200mm Deep Expansion Rack
1410HPA          42  42U 1200mm Deep Primary Rack

    Server Specifications

Part Number  Height  Description
-----------  ------  ------------------
824722L           2 IBM Power System S822L

    Storage Enclosure Specifications

Part Number  Height  Description                 Vendor ID  Product ID  Drive Slots  Has Display ID
-----------  ------  --------------------------  ---------  ----------  -----------  --------------
1818-80E          4  DCS3700 Expansion Enclosure  IBM        DCS3700              60               1
```

If this was the first component command run on the cluster, you will see some initialization messages before the command output. The list of component specifications will be short because it only includes supported ESS hardware.

You can use **mmdiscovercomp** to define some of the cluster's components. This command finds supported storage enclosures attached to any of the cluster nodes and adds them to the cluster's configuration. (In your version of IBM Spectrum Scale, **mmdiscovercomp** might find additional component types.) The output below resembles a cluster that includes two GL4 units.

```
$ mmdiscovercomp all
Adding enclosure: serial number = SV12345001; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345007; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345003; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345005; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345004; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345002; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345008; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345006; vendor ID = IBM; product ID = DCS3700

Storage Enclosures

Status  Comp ID  Component Type    Part Number  Serial Number  Name                Display ID
------  -------  ----------------  -----------  -------------  ------------------  ----------
new           1  storageEnclosure  1818-80E     SV12345001     1818-80E-SV12345001  00
new           2  storageEnclosure  1818-80E     SV12345007     1818-80E-SV12345007  00
new           3  storageEnclosure  1818-80E     SV12345003     1818-80E-SV12345003  00
new           4  storageEnclosure  1818-80E     SV12345005     1818-80E-SV12345005  00
```

```
new           5  storageEnclosure  1818-80E     SV12345004     1818-80E-SV12345004  00
new           6  storageEnclosure  1818-80E     SV12345002     1818-80E-SV12345002  00
new           7  storageEnclosure  1818-80E     SV12345008     1818-80E-SV12345008  00
new           8  storageEnclosure  1818-80E     SV12345006     1818-80E-SV12345006  00
```

In this example, **mmdiscovercomp** found eight storage enclosures that were not already in the cluster configuration. Notice that each component gets a default name. Below we will discuss how you can change the name to some identifier that is more suitable for your environment. You may run **mmdiscovercomp** a second time, in which case it should not find any new components.

**Note:** The serial numbers used by IBM Spectrum Scale to identify 1818-80E storage enclosures do not match the product serial numbers shown on the outside of the enclosure. You may want to record this visible serial number as part of the enclosure's name. For example, you could rename an enclosure to **S1E3-SX98760044**, which would stand for "storage server 1" (**S1**), "enclosure 3" (**E3**), with product serial number SX98760044. For details about how to change the name of a component using the **mmchcomp** command, see "Updating component attributes" on page 50.

The other essential components are racks to hold the storage enclosures. Use the **mmlscompspec** command to see the available rack part numbers, then use the **mmaddcomp** command to define the racks. This example shows how to define a "42U 1200mm Deep Primary Rack" and give it the name **R01C01**.

```
$ mmaddcomp 1410HPA --name R01C01
$ mmlscomp

    Rack Components

Comp ID  Part Number  Serial Number  Name
-------  -----------  -------------  ------
      9  1410HPA                     R01C01

    Storage Enclosure Components

Comp ID  Part Number  Serial Number  Name                Display ID
-------  -----------  -------------  ------------------  ----------
      1  1818-80E     SV12345001     1818-80E-SV12345001
      2  1818-80E     SV12345007     1818-80E-SV12345007
      3  1818-80E     SV12345003     1818-80E-SV12345003
      4  1818-80E     SV12345005     1818-80E-SV12345005
      5  1818-80E     SV12345004     1818-80E-SV12345004
      6  1818-80E     SV12345002     1818-80E-SV12345002
      7  1818-80E     SV12345008     1818-80E-SV12345008
      8  1818-80E     SV12345006     1818-80E-SV12345006
```

## Defining component locations

This section discusses how to define component locations in an ESS system. Sample output in this topic might not match the output produced on your system.

Use the **mmchcomploc** command to place a component in a rack or change its location. This example puts the storage enclosure SV12345003 in rack R01C01 at U location 13.

```
$ mmchcomploc SV12345003 R01C01 13
$ mmlscomploc
Component           Location
------------------  ------------------
1818-80E-SV12345003  Rack R01C01 U13-16
```

The first argument, which identifies the component, can be either the component ID, serial number, or name. The second argument is the rack (container) and the third argument is the U position in the rack.

Changing one component at a time can be slow because each change is propagated to all nodes in the cluster. You may want to use a stanza file to make multiple changes with a single command. The following example uses a stanza file to position the remaining enclosures:

```
$ cat comploc.stanza
%compLoc: compId=SV12345007 containerId=R01C01 position=1
%compLoc: compId=SV12345005 containerId=R01C01 position=5
%compLoc: compId=SV12345003 containerId=R01C01 position=13
%compLoc: compId=SV12345008 containerId=R01C01 position=17
%compLoc: compId=SV12345001 containerId=R01C01 position=21
%compLoc: compId=SV12345002 containerId=R01C01 position=25
%compLoc: compId=SV12345006 containerId=R01C01 position=33
%compLoc: compId=SV12345004 containerId=R01C01 position=37

$ mmchcomploc -F comploc.stanza
$ mmlscomploc
Component            Location
------------------   ------------------
1818-80E-SV12345007  Rack R01C01 U01-04
1818-80E-SV12345005  Rack R01C01 U05-08
1818-80E-SV12345003  Rack R01C01 U13-16
1818-80E-SV12345008  Rack R01C01 U17-20
1818-80E-SV12345001  Rack R01C01 U21-24
1818-80E-SV12345002  Rack R01C01 U25-28
1818-80E-SV12345006  Rack R01C01 U33-36
1818-80E-SV12345004  Rack R01C01 U37-40
```

## Synchronizing display IDs

This section discusses how to synchronize display IDs in an ESS system. Sample output in this topic might not match the output produced on your system.

Some ESS disk enclosures have a two-digit display on their environmental service modules (ESMs). This display is visible from the back of the enclosure. When configured correctly, the two-digit display will be the same as the U location of the enclosure within its rack. **mmsyncdisplayid** sets the display on the ESMs to match the locations of the enclosure as defined by the component configuration. Here is an example:

```
$ mmsyncdisplayid all
$ mmlscomp --type storageenclosure

    Storage Enclosure Components

Comp ID  Part Number   Serial Number  Name                Display ID
-------  -----------   -------------  ------------------  ----------
      1  1818-80E      SV12345001     1818-80E-SV12345001  21
      2  1818-80E      SV12345007     1818-80E-SV12345007  01
      3  1818-80E      SV12345003     1818-80E-SV12345003  13
      4  1818-80E      SV12345005     1818-80E-SV12345005  05
      5  1818-80E      SV12345004     1818-80E-SV12345004  37
      6  1818-80E      SV12345002     1818-80E-SV12345002  25
      7  1818-80E      SV12345008     1818-80E-SV12345008  17
      8  1818-80E      SV12345006     1818-80E-SV12345006  33
```

With the display ID set on the actual hardware, you should now verify that the locations defined in the cluster's configuration actually match the enclosure's position in its rack. If you find that the enclosures are not numbered 01, 05, 13, ... starting from the bottom of the rack, then use the **mmchcomploc** command to correct the configuration. Then rerun **mmsyncdisplayid** and recheck the hardware.

## Updating component attributes

This section discusses how to update component attributes in an ESS system. Sample output in this topic might not match the output produced on your system.

You can change some component attributes such as the name or serial number. This example updates the name of the third enclosure so that it includes the product serial number as recommended:

```
$ mmchcomp 3 --name S1E3-SX98760044
$ mmlscomp --type storageenclosure

    Storage Enclosure Components

Comp ID  Part Number  Serial Number   Name                 Display ID
-------  -----------  -------------   ------------------   ----------
      1  1818-80E      SV12345001     1818-80E-SV12345001  21
      2  1818-80E      SV12345007     1818-80E-SV12345007  01
      3  1818-80E      SV12345003     S1E3-SX98760044      13
      4  1818-80E      SV12345005     1818-80E-SV12345005  05
      5  1818-80E      SV12345004     1818-80E-SV12345004  37
      6  1818-80E      SV12345002     1818-80E-SV12345002  25
      7  1818-80E      SV12345008     1818-80E-SV12345008  17
      8  1818-80E      SV12345006     1818-80E-SV12345006  33
```

The **mmchcomp** *Component* argument may be either the component ID, the component serial number, or the component name. This example uses the component ID (the value 3).

Changing one component at a time can be slow, so you can use a stanza file to make multiple changes with a single command. As an aid, **mmlscomp** can list components in stanza format. You can capture this output into a file, edit the file to make the desired updates, then use the stanza file as input to **mmchcomp**. The following example uses this procedure to rename the remaining enclosures:

```
$ mmlscomp --format stanza > rename.stanza
$ cat rename.stanza
%comp:
  compId=9
  compType=rack
  partNumber=1410HEA

%comp:
  compId=1
  compType=storageEnclosure
  displayId=21
  name='1818-80E-SV12345001'
  partNumber=1818-80E
  serialNumber=SV12345001

%comp:
  compId=2
  compType=storageEnclosure
  displayId=1
  name='1818-80E-SV12345007'
  partNumber=1818-80E
  serialNumber=SV12345007

%comp:
  compId=3
  compType=storageEnclosure
  displayId=13
  name='S1E3-SX98760044'
  partNumber=1818-80E
  serialNumber=SV12345003

...
```

Edit `rename.stanza` to define new enclosure names, then apply the changes using **mmchcomp**.

```
$ vi rename.stanza
$ cat rename.stanza
%comp:
  compId=9
  compType=rack
```

```
  name=R01C01
  partNumber=1410HEA

%comp:
  compId=1
  compType=storageEnclosure
  displayId=21
  name='S1E5-SX98760048'
  partNumber=1818-80E
  serialNumber=SV12345001

%comp:
  compId=2
  compType=storageEnclosure
  displayId=1
  name='S1E1-SX98760044'
  partNumber=1818-80E
  serialNumber=SV12345007

%comp:
  compId=3
  compType=storageEnclosure
  displayId=13
  name='S1E3-SX98760041'
  partNumber=1818-80E
  serialNumber=SV12345003

...

$ mmchcomp -F comp.stanza
$ mmlscomp --sort name

    Rack Components

Comp ID  Part Number  Serial Number  Name
-------  -----------  -------------  ------
      9  1410HEA                     R01C01

    Storage Enclosure Components

Comp ID  Part Number  Serial Number   Name                Display ID
-------  -----------  -------------   ------------------  ----------
      2  1818-80E      SV12345007     S1E1-SX98760044     01
      4  1818-80E      SV12345005     S1E2-SX98760049     05
      3  1818-80E      SV12345003     S1E3-SX98760041     13
      7  1818-80E      SV12345008     S1E4-SX98760036     17
      1  1818-80E      SV12345001     S1E5-SX98760048     21
      6  1818-80E      SV12345002     S1E6-SX98760050     25
      8  1818-80E      SV12345006     S1E7-SX98760039     33
      5  1818-80E      SV12345004     S1E8-SX98760052     37
```

# Chapter 6. Monitoring IBM Spectrum Scale RAID

Monitoring the ESS system includes system health, performance, and capacity monitoring. You can monitor the system either through ESS GUI or with the help of CLI.

The following table lists the CLI options that are available to monitor the system.

*Table 5. CLI options that are available to monitor the system.* CLI options that are available to monitor the system

| CLI commands | Function |
|---|---|
| mmhealth | Single command that can monitor the health of nodes, services, logical components, events, list thresholds, and so on. |
| mmperfmon | Configures the performance monitoring tool and lists the performance metrics. |
| mmpmon | Manages performance monitoring of GPFS and displays performance information. |
| mmlsrecoverygroup | Lists information about IBM Spectrum Scale RAID recovery groups. |
| mmlspdisk | Lists information for one or more IBM Spectrum Scale RAID pdisks. |
| mmlsvdisk | Lists information for one or more IBM Spectrum Scale RAID vdisks. |
| mmlsenclosure | Displays the environmental status of IBM Spectrum Scale RAID disk enclosures. |
| mmaddcallback | Registers a user-defined command that GPFS will execute when certain events occur. |

The following topics describe the monitoring options that are available in the ESS GUI:


- "Monitoring system health using ESS GUI" on page 60
- "Performance monitoring using ESS GUI" on page 67
- "Monitoring capacity through GUI" on page 80


## System health monitoring

The monitoring framework that is designed in the system takes care of the system health monitoring and you can use the **mmhealth** command to get the details collected by the monitoring framework.

Use the **mmhealth** command to get details of the following aspects:

- Health of individual nodes and entire set of nodes at the cluster level
- Health of the logical components
- Health of services that are hosted in the system
- Events that are reported in the system
- Thresholds that are defined in the system

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

Hardware components are monitored through xtreme Cluster/Cloud Administration Toolkit (xCAT). It is a scalable and open-source cluster management software. The management infrastructure of ESS is deployed by xCAT.

## Monitoring framework

Every service that is hosted on an ESS node has its own health monitoring service. The monitoring service works based on the roles that are assigned to the nodes in the cluster. The role of a node in monitoring determines the components that need to be monitored. The node roles are hardcoded. It also determines the monitoring service that is required on a specific node. For example, you do not need a CES monitoring on a non-CES node. The monitoring services are only started if a specific node role is assigned to the node. Every monitoring service includes at least one monitor.

The following list provides the details of the monitoring services available in the IBM Spectrum Scale system:

**GPFS** Node role: Active on all ESS nodes.

Tasks: Monitors all GPFS daemon-related functions. For example, **mmfsd** process and *gpfs port accessibility*.

**Network**
Node role: Every ESS node has the node role that is required for the network monitoring service.

**CES** Node role: Nodes with node class *cesNodeperforms* the monitoring services for all CES services. The CES service does not have its own monitoring service or events. The status of the CES is an aggregation of the status of the subservices.

**File authentication**
Tasks: Monitors LDAP, AD, or NIS-based authentication services.

**Object authentication**
Tasks: Monitors the OpenStack identity service functions.

**Block** Tasks: Checks whether the iSCSI daemon is functioning properly.

**CES network**
Tasks: Monitors CES network-related adapters and IP addresses.

**NFS** Tasks: Monitoring NFS-related functions.

**Object**
Tasks: Monitors the ESS for object functions. Especially, the status of relevant system services and accessibility to ports are checked.

**SMB** Tasks: Monitoring SMB-related functions like the smbd process, the ports, and ctdb processes.

**Cloud gateway**
Node role: A node gets the cloud gateway node role if it is identified as a transparent cloud tiering node. These nodes are listed when you issue the **mmcloudgateway nodelist** command.

Tasks: Checks whether the cloud gateway service functions as expected.

**Disk** Node role: Nodes with node class *nsdNodes* monitor the disk service.

Tasks: Checks whether the ESS disks are available and running.

**File system**
Node role: This node role is active on all ESS nodes.

Tasks: Monitors different aspects of IBM Spectrum Scale file systems.

**GUI** Node role: Nodes with node class *GUI_MGMT_SERVERS* monitor the GUI service.

Tasks: Verifies whether the GUI services are functioning properly.

**Hadoop connector**

Node role: Nodes where the Hadoop service is configured get the Hadoop connector node role.

Tasks: Monitors the Hadoop data node and name node services.

**Performance monitoring**

Node role: Nodes where *PerfmonSensors* or *PerfmonCollectorservices* are installed get the performance monitoring node role. Performance monitoring sensors are determined through the *perfmon* designation in the **mmlscluster**. Performance monitoring collectors are determined through the *colCandidates* line in the configuration file.

Tasks: Monitors whether the performance monitoring collectors and sensors are running as expected.

For more information on system health monitoring options that are available in GUI, see "Monitoring system health using ESS GUI" on page 60.

# Monitoring events

The recorded events are stored in local database on each node. The user can get a list of recorded events by using the **mmhealth node eventlog** command. You can also use the **mmces** command to get the details of the events that are recoded in the system.

For more information about the **mmhealth** and **mmces** commands, see the *IBM Spectrum Scale: Command and Programming Reference*.

## Monitoring events through callhome

The call home feature automatically notifies IBM Support if certain types of events occur in the system. Using this information, IBM Support can contact the system administrator in case of any issues. Configuring call home reduces the response time for IBM Support to address the issues.

The details are collected from individual nodes that are marked as call home child nodes in the cluster. The details from each child node are collected by the call home node. You need to create a call home group by grouping call home child nodes. One of the nodes in the group is configured as the call home node and it performs data collection and upload.

The data gathering and upload can be configured individually on each group. Use the groups to reflect logical units in the cluster. For example, it is easier to manage when you create a group for all CES nodes and another group for all non-CES nodes.

Use the **mmcallhome** command to configure the call home feature in the system. For more information about the **mmcallhome** command, see the *IBM Spectrum Scale: Command and Programming Reference*.

## IBM Spectrum Scale RAID callbacks

IBM Spectrum Scale RAID includes callbacks for events that can occur during recovery group operations. These callbacks can be installed by the system administrator using the **mmaddcallback** command.

The callbacks are provided primarily as a method for system administrators to take notice when important IBM Spectrum Scale RAID events occur. For example, an IBM Spectrum Scale RAID administrator can use the **pdReplacePdisk** callback to send an e-mail to notify system operators that the replacement threshold for a declustered array was reached and that pdisks must be replaced. Similarly, the **preRGTakeover** callback can be used to inform system administrators of a possible server failover.

As notification methods, no real processing should occur in the callback scripts. IBM Spectrum Scale RAID callbacks should not be installed for synchronous execution; the default of asynchronous callback

execution should be used in all cases. Synchronous or complicated processing within a callback might delay GPFS daemon execution pathways and cause unexpected and undesired results, including loss of file system availability.

The IBM Spectrum Scale RAID callbacks and their corresponding parameters follow:

**preRGTakeover**
The **preRGTakeover** callback is invoked on a recovery group server prior to attempting to open and serve recovery groups. The **rgName** parameter may be passed into the callback as the keyword value **_ALL_**, indicating that the recovery group server is about to open multiple recovery groups; this is typically at server startup, and the parameter **rgCount** will be equal to the number of recovery groups being processed. Additionally, the callback will be invoked with the **rgName** of each individual recovery group and an **rgCount** of 1 whenever the server checks to determine whether it should open and serve recovery group **rgName**.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%rgErr**, **%rgCount**, and **%rgReason**.

**postRGTakeover**
The **postRGTakeover** callback is invoked on a recovery group server after it has checked, attempted, or begun to serve a recovery group. If multiple recovery groups have been taken over, the callback will be invoked with **rgName** keyword **_ALL_** and an **rgCount** equal to the total number of involved recovery groups. The callback will also be triggered for each individual recovery group.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%rgErr**, **%rgCount**, and **%rgReason**.

**preRGRelinquish**
The **preRGRelinquish** callback is invoked on a recovery group server prior to relinquishing service of recovery groups. The **rgName** parameter may be passed into the callback as the keyword value **_ALL_**, indicating that the recovery group server is about to relinquish service for all recovery groups it is serving; the **rgCount** parameter will be equal to the number of recovery groups being relinquished. Additionally, the callback will be invoked with the **rgName** of each individual recovery group and an **rgCount** of 1 whenever the server relinquishes serving recovery group **rgName**.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%rgErr**, **%rgCount**, and **%rgReason**.

**postRGRelinquish**
The **postRGRelinquish** callback is invoked on a recovery group server after it has relinquished serving recovery groups. If multiple recovery groups have been relinquished, the callback will be invoked with **rgName** keyword **_ALL_** and an **rgCount** equal to the total number of involved recovery groups. The callback will also be triggered for each individual recovery group.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%rgErr**, **%rgCount**, and **%rgReason**.

**rgOpenFailed**
The **rgOpenFailed** callback will be invoked on a recovery group server when it fails to open a recovery group that it is attempting to serve. This may be due to loss of connectivity to some or all of the disks in the recovery group; the **rgReason** string will indicate why the recovery group could not be opened.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%rgErr**, and **%rgReason**.

**rgPanic**
The **rgPanic** callback will be invoked on a recovery group server when it is no longer able to continue serving a recovery group. This may be due to loss of connectivity to some or all of the disks in the recovery group; the **rgReason** string will indicate why the recovery group can no longer be served.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%rgErr**, and **%rgReason**.

**pdFailed**
The **pdFailed** callback is generated whenever a pdisk in a recovery group is marked as `dead`, `missing`, `failed`, or `readonly`.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%daName**, **%pdName**, **%pdLocation**, **%pdFru**, **%pdWwn**, and **%pdState**.

**pdRecovered**
The **pdRecovered** callback is generated whenever a missing pdisk is rediscovered.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%daName**, **%pdName**, **%pdLocation**, **%pdFru**, and **%pdWwn**.

**pdReplacePdisk**
The **pdReplacePdisk** callback is generated whenever a pdisk is marked for replacement according to the replace threshold setting of the declustered array in which it resides.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%daName**, **%pdName**, **%pdLocation**, **%pdFru**, **%pdWwn**, **%pdState**, and **%pdPriority**.

**pdPathDown**
The **pdPathDown** callback is generated whenever one of the block device paths to a pdisk disappears or becomes inoperative. The occurrence of this event can indicate connectivity problems with the JBOD array in which the pdisk resides.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%daName**, **%pdName**, **%pdPath**, **%pdLocation**, **%pdFru**, and **%pdWwn**.

**daRebuildFailed**
The **daRebuildFailed** callback is generated when the spare space in a declustered array has been exhausted, and vdisk tracks involving damaged pdisks can no longer be rebuilt. The occurrence of this event indicates that fault tolerance in the declustered array has become degraded and that disk maintenance should be performed immediately. The **daRemainingRedundancy** parameter indicates how much fault tolerance remains in the declustered array.

The following parameters are available to this callback: **%myNode**, **%rgName**, **%daName**, and **%daRemainingRedundancy**.

**nsdCksumMismatch**
The **nsdCksumMismatch** callback is generated whenever transmission of vdisk data by the NSD network layer fails to verify the data checksum. This can indicate problems in the network between the GPFS client node and a recovery group server. The first error between a given client and server generates the callback; subsequent callbacks are generated for each **ckReportingInterval** occurrence.

The following parameters are available to this callback: **%myNode**, **%ckRole**, **%ckOtherNode**, **%ckNSD**, **%ckReason**, **%ckStartSector**, **%ckDataLen**, **%ckErrorCountClient**, **%ckErrorCountNSD**, and **%ckReportingInterval**.

IBM Spectrum Scale RAID recognizes the following variables (in addition to the **%myNode** variable, which specifies the node where the callback script is invoked):

**%ckDataLen**
The length of data involved in a checksum mismatch.

**%ckErrorCountClient**
The cumulative number of errors for the client side in a checksum mismatch.

**%ckErrorCountServer**
The cumulative number of errors for the server side in a checksum mismatch.

**%ckErrorCountNSD**
The cumulative number of errors for the NSD side in a checksum mismatch.

**%ckNSD**
The NSD involved.

**%ckOtherNode**
The IP address of the other node in an NSD checksum event.

**%ckReason**
The reason string indicating why a checksum mismatch callback was invoked.

**%ckReportingInterval**
The error-reporting interval in effect at the time of a checksum mismatch.

**%ckRole**
The role (client or server) of a GPFS node.

**%ckStartSector**
The starting sector of a checksum mismatch.

**%daName**
The name of the declustered array involved.

**%daRemainingRedundancy**
The remaining fault tolerance in a declustered array.

**%pdFru**
The FRU (field replaceable unit) number of the pdisk.

**%pdLocation**
The physical location code of a pdisk.

**%pdName**
The name of the pdisk involved.

**%pdPath**
The block device path of the pdisk.

**%pdPriority**
The replacement priority of the pdisk.

**%pdState**
The state of the pdisk involved.

**%pdWwn**
The worldwide name of the pdisk.

**%rgCount**
The number of recovery groups involved.

**%rgErr**
A code from a recovery group, where 0 indicates no error.

**%rgName**
The name of the recovery group involved.

**%rgReason**
The reason string indicating why a recovery group callback was invoked.

All IBM Spectrum Scale RAID callbacks are local, which means that the event triggering the callback occurs only on the involved node or nodes, in the case of **nsdCksumMismatch**, rather than on every node in the GPFS cluster. The nodes where IBM Spectrum Scale RAID callbacks should be installed are, by definition, the recovery group server nodes. An exception is the case of **nsdCksumMismatch**, where it makes sense to install the callback on GPFS client nodes as well as recovery group servers.

A sample callback script, /usr/lpp/mmfs/samples/vdisk/gnrcallback.sh, is available to demonstrate how callbacks can be used to log events or email an administrator when IBM Spectrum Scale RAID events occur.

Logged events would look something like:

```
Fri Feb 28 10:22:17 EST 2014:  mmfsd: [W] event=pdFailed node=c45f01n01-ib0.gpfs.net
rgName=BB1RGL daName=DA1 pdName=e4d5s03 pdLocation=SV13306129-5-3 pdFru=46W6911
pdWwn=naa.5000C50055D4D437 pdState=dead/systemDrain
Fri Feb 28 10:22:39 EST 2014:  mmfsd: [I] event=pdRecovered node=c45f01n01-ib0.gpfs.net
rgName=BB1RGL daName=DA1 pdName=e4d5s03 pdLocation=SV13306129-5-3 pdFru=46W6911
pdWwn=naa.5000C50055D4D437 pdState=UNDEFINED
Fri Feb 28 10:23:59 EST 2014:  mmfsd: [E] event=rgPanic node=c45f01n01-ib0.gpfs.net
rgName=BB1RGL rgErr=756 rgReason=missing_pdisk_causes_unavailability
Fri Feb 28 10:24:00 EST 2014:  mmfsd: [I] event=postRGRelinquish node=c45f01n01-ib0.gpfs.net
rgName=BB1RGL rgErr=0 rgReason=unable_to_continue_serving
Fri Feb 28 10:24:00 EST 2014:  mmfsd: [I] event=postRGRelinquish node=c45f01n01-ib0.gpfs.net
rgName=_ALL_ rgErr=0 rgReason=unable_to_continue_serving
Fri Feb 28 10:35:06 EST 2014: mmfsd: [I] event=postRGTakeover node=c45f01n01-ib0.gpfs.net
rgName=BB1RGL rgErr=0 rgReason=retry_takeover
Fri Feb 28 10:35:06 EST 2014: mmfsd: [I] event=postRGTakeover node=c45f01n01-ib0.gpfs.net
rgName=_ALL_ rgErr=0 rgReason=none
```

An email notification would look something like this:

```
> mail
Heirloom Mail version 12.4 7/29/08.  Type ? for help.
"/var/spool/mail/root": 7 messages 7 new
>N  1 root                  Fri Feb 28 10:22  18/817   "[W] pdFailed"
 N  2 root                  Fri Feb 28 10:22  18/816   "[I] pdRecovered"
 N  3 root                  Fri Feb 28 10:23  18/752   "[E] rgPanic"
 N  4 root                  Fri Feb 28 10:24  18/759   "[I] postRGRelinquish"
 N  5 root                  Fri Feb 28 10:24  18/758   "[I] postRGRelinquish"
 N  6 root                  Fri Feb 28 10:35  18/743   "[I] postRGTakeover"
 N  7 root                  Fri Feb 28 10:35  18/732   "[I} postRGTakeover"

From root@c45f01n01.localdomain  Wed Mar  5 12:27:04 2014
Return-Path: <root@c45f01n01.localdomain>
X-Original-To: root
Delivered-To: root@c45f01n01.localdomain
Date: Wed, 05 Mar 2014 12:27:04 -0500
To: root@c45f01n01.localdomain
Subject: [W] pdFailed
User-Agent: Heirloom mailx 12.4 7/29/08
Content-Type: text/plain; charset=us-ascii
From: root@c45f01n01.localdomain (root)
Status: R

Wed Mar 5 12:27:04 EST 2014: mmfsd: [W] event=pdFailed node=c45f01n01-ib0.gpfs.net
rgName=BB1RGL daName=DA1 pdName=e4d5s03 pdLocation=SV13306129-5-3 pdFru=46W6911
pdWwn=naa.50 00C50055D4D437 pdState=dead/systemDrain
```

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

## IBM Spectrum Scale RAID events in syslog

If the linux **syslog** facility is enabled, IBM Spectrum Scale RAID will log IBM Spectrum Scale RAID events to the syslog. This can be controlled using the IBM Spectrum Scale RAID **systemLogLevel** configuration variable. By default, all informational messages and higher-level error messages are logged.

Log events would look something like this:

```
Feb 27 15:43:20 c45f01n01 mmfsd: Error=MMFS_PDISK_FAILED, ID=0x157E5F74, Tag=1024872:
Pdisk Failed: Location=SV13306129-5-3, FRU=46W6911, WWID=5000c50055d4d437, RecoveryGrou
p=BB1RGL, DeclusteredArray=DA1, Pdisk=e4d5s03, PdiskState=dead/systemDrain
Feb 27 15:48:10 c45f01n01 mmfsd: Error=MMFS_PDISK_RECOVERED, ID=0x5DCF6F0A, Tag=1024873:
Pdisk Recovered: Location=SV13306129-5-3, FRU=46W6911, WWID=5000c50055d4d437, Recove
ryGroup=BB1RGL, DeclusteredArray=DA1, Pdisk=e4d5s03
```

# Monitoring system health using ESS GUI

The following table lists the system health monitoring options that are available in the ESS GUI.

*Table 6. System health monitoring options available in ESS GUI*

| Option | Function |
|---|---|
| **Monitoring** > **Hardware** | Displays the status of all servers, enclosures, and drives in an enclosure. Click a server or enclosure to view the details of that particular hardware component. |
| **Monitoring** > **Hardware Details** | Displays status and details of the servers and enclosures that are a part of the system. Click any of the system components to view its details. This view also provides a text search and filtering for unhealthy hardware. |
| **Monitoring** > **Events** | Lists the events that are reported in the system. You can monitor and troubleshoot errors on your system from the Events page. |
| **Monitoring** > **Tips** | Lists the tips reported in the system and allows to hide or show tips. The tip events give recommendations to the user to avoid certain issues that might occur in the future. |
| **Home** | Provides overall system health of the ESS system. This page is displayed in the GUI only if the minimum release level of ESS is 4.2.2 or later. |
| **Monitoring** > **Nodes** | Lists the events reported at the node level. |
| **Files** > **File Systems** | Lists the events reported at the file system level. |
| **Files** > **Transparent Cloud Tiering** | Lists the events reported for the Transparent Cloud Tiering service. The GUI displays this page only if the transparent cloud tiering feature is enabled in the system. |
| **Files** > **Filesets** | Lists events reported for filesets. |
| **Files** > **Active File Management** | Displays health status and lists events reported for AFM cache relationship, AFM disaster recovery (AFMDR) relationship, and gateway nodes. |
| **Storage** > **Pools** | Displays health status and lists events reported for storage pools. |
| **Storage** > **NSDs** | Lists the events reported at the NSD level. |
| **Storage** > **Physical** | Displays health information and properties of physical disks and the corresponding declustered arrays. |
| **Storage** > **Volumes** | Displays health information and properties of logical volumes and the corresponding declustered arrays. |
| Health indicator that is available in the upper right corner of the GUI. | Displays the number of events with warning and error status. |
| **System overview** widget in the **Monitoring** > **Dashboard** page. | Displays the number of events reported against each component. |
| **System health events** widget in the **Monitoring** > **Dashboard** page. | Provides an overview of the events reported in the system. |

*Table 6. System health monitoring options available in ESS GUI (continued)*

| Option | Function |
|---|---|
| **Timeline** widget in the **Monitoring** > **Dashboard** page. | Displays the events that are reported in a particular time frame on the selected performance chart. |

## Monitoring events using GUI

You can primarily use the **Monitoring** > **Events** page to review the entire set of events that are reported in the ESS system.

The events are raised against the respective component, for example, GPFS, NFS, SMB, and so on. Same events might occur multiple times in the system. Such events are grouped together under the **Event Groups** tab and the number of occurrences of the events are indicated in the **Occurrences** column. The **Individual Events** tab lists all the events irrespective of the multiple occurrences.

You can further filter the events listed in the Events page with the help of the following filter options:

- **Current Issues** displays all unfixed errors and warnings.
- **Unread Messages** displays all unfixed errors and warnings and information messages that are not marked as read.
- **All Events** displays every event, no matter if it is fixed or marked as read.

The status icons help to quickly determine whether the event is informational, a warning, or an error. Click an event and select **Properties** from the **Action** menu to see the detailed information of that event. The event table displays the most recent events first.

**Marking events as read**

You can mark certain events as read to change the status of the event in the events view. The status icons become gray in case an error or warning is fixed or if it is marked as read.

**Running fix procedure**

Some issues can be resolved by running a fix procedure. Use action **Run Fix Procedure** to do so. The Events page provides a recommendation for which fix procedure to run next.

For more information on how to set up event notifications, see "Set up event notifications"

### Tips events

You can monitor events of type "Tips" from the **Monitoring** > **Tips** page of the GUI. The tip events give recommendations to the user to avoid certain issues that might occur in the future. The system detects the entities with tip event as healthy. A tip disappears from the GUI when the problem behind the tip event is resolved.

Select **Properties** from the **Actions** menu to view the details of the tip. After you review the tip, decide whether it requires attention or can be ignored. Select **Hide** from the **Actions** menu to ignore the events that are not important and select **Show** to mark the tips that require attention.

## Set up event notifications

The system can use Simple Network Management Protocol (SNMP) traps and emails to notify you when significant events are detected. Any combination of these notification methods can be used simultaneously. Use **Settings** > **Event Notifications** page in the GUI to configure event notifications.

Notifications are normally sent immediately after an event is raised.

In email notification method, you can also define whether a recipient needs to get a report of events that are reported in the system. These reports are sent only once in a day. Based on the seriousness of the issue, each event that is reported in the system gets a severity level associated with it.

The following table describes the severity levels of event notifications.

*Table 7. Notification levels*

| Notification level | Description |
|---|---|
| Error | Error notification is sent to indicate a problem that must be corrected as soon as possible.<br><br>This notification indicates a serious problem with the system. For example, the event that is being reported might indicate a loss of redundancy in the system, and it is possible that another failure might result in loss of access to data. The most typical reason that this type of notification is because of a hardware failure, but some configuration errors or fabric errors also are included in this notification level. |
| Warning | A warning notification is sent to indicate a problem or unexpected condition with the system. Always immediately investigate this type of notification to determine the effect that it might have on your operation, and make any necessary corrections.<br><br>Therefore, a warning notification does not require any replacement parts and it does not require IBM Support Center involvement. |
| Information | An informational notification is sent to indicate that an expected event is occurred. For example, a NAS service is started. No remedial action is required when these notifications are sent. |

**Configuring email notifications:**

The email feature transmits operational and error-related data in the form of an event notification email.

To configure an email server, from the Event Notifications page, select **Email Server**. Select **Edit** and then click **Enable email notifications**. Enter required details and when you are ready, click **OK**.

Email notifications can be customized by setting a custom header and footer for the emails and customizing the subject by selecting and combining from the following variables: *&message*, *&messageId*, *&severity*, *&dateAndTime*, *&cluster* and *&component*.

Emails containing the quota reports and other events reported in the following functional areas are sent to the recipients:

- AFM and AFM DR
- Authentication
- CES network
- Transparent Cloud Tiering
- NSD
- File system
- GPFS
- GUI
- Hadoop connector
- iSCSI
- Keystone
- Network
- NFS

- Object
- Performance monitoring
- SMB
- Object authentication
- Node
- CES

You can specify the severity level of events and whether to send a report that contains a summary of the events received.

To create email recipients, select **Email Recipients** from the **Event Notifications** page, and then click **Create Recipient**.

**Note:** You can change the email notification configuration or disable the email service at any time.

**Configuring SNMP manager:**

Simple Network Management Protocol (SNMP) is a standard protocol for managing networks and exchanging messages. The system can send SNMP messages that notify personnel about an event. You can use an SNMP manager to view the SNMP messages that the system sends.

With an SNMP manager, such as IBM Systems Director, you can view, and act on the messages that the SNMP agent sends. The SNMP manager can send SNMP notifications, which are also known as traps, when an event occurs in the system. Select **Settings** > **Event Notifications** > **SNMP Manager** to configure SNMP managers for event notifications. You can specify up to a maximum of six SNMP managers.

In the SNMP mode of event notification, one SNMP notification (trap) with object identifiers (OID) .1.3.6.1.4.1.2.6.212.10.0.1 is sent by the GUI for each event. The following table provides the SNMP objects included in the event notifications.

*Table 8. SNMP objects included in event notifications*

| OID | Description | Examples |
|---|---|---|
| .1.3.6.1.4.1.2.6.212.10.1.1 | Cluster ID | 317908494245422510 |
| .1.3.6.1.4.1.2.6.212.10.1.2 | Entity type | SERVER, FILESYSTEM |
| .1.3.6.1.4.1.2.6.212.10.1.3 | Entity name | gss-11, fs01 |
| .1.3.6.1.4.1.2.6.212.10.1.4 | Component | SMB, AUTHENTICATION |
| .1.3.6.1.4.1.2.6.212.10.1.5 | Severity | SEVERE, WARN, INFO |
| .1.3.6.1.4.1.2.6.212.10.1.6 | Date and time | 17.02.2016 13:27:42.516 |
| .1.3.6.1.4.1.2.6.212.10.1.7 | Event name | MS1014 |
| .1.3.6.1.4.1.2.6.212.10.1.8 | Message | At least one CPU of "gss-11" is failed. |
| 1.3.6.1.4.1.2.6.212.10.1.9 | Reporting node | The node where the problem is reported. |

### Understanding the SNMP OID ranges

The following table gives the description of the SNMP OID ranges.

*Table 9. SNMP OID ranges*

| OID range | Description |
|---|---|
| .1.3.6.1.4.1.2.6.212 | IBM Spectrum Scale |
| .1.3.6.1.4.1.2.6.212.10 | IBM Spectrum Scale GUI |
| .1.3.6.1.4.1.2.6.212.10.0.1 | IBM Spectrum Scale GUI event notification (trap) |
| .1.3.6.1.4.1.2.6.212.10.1.x | IBM Spectrum Scale GUI event notification parameters (objects) |

The traps for the core IBM Spectrum Scale and those trap objects are not included in the SNMP notifications that are configured through the IBM Spectrum Scale management GUI.

### Example for SNMP traps

The following example shows the SNMP event notification that is sent when performance monitoring sensor is shut down on a node:

```
SNMPv2-MIB::snmpTrapOID.0 = OID: SNMPv2-SMI::enterprises.2.6.212.10.0.1
SNMPv2-SMI::enterprises.2.6.212.10.1.1 = STRING: "317908494245422510"
SNMPv2-SMI::enterprises.2.6.212.10.1.2 = STRING: "NODE"
SNMPv2-SMI::enterprises.2.6.212.10.1.3 = STRING: "gss-11"
SNMPv2-SMI::enterprises.2.6.212.10.1.4 = STRING: "PERFMON"
SNMPv2-SMI::enterprises.2.6.212.10.1.5 = STRING: "ERROR"
SNMPv2-SMI::enterprises.2.6.212.10.1.6 = STRING: "18.02.2016 12:46:44.839"
SNMPv2-SMI::enterprises.2.6.212.10.1.7 = STRING: "pmsensors_down"
SNMPv2-SMI::enterprises.2.6.212.10.1.8 = STRING: "pmsensors service should be started and is stopped"
SNMPv2-SMI::enterprises.2.6.212.10.1.9 = STRING: "gss-11"
```

The following example shows the SNMP event notification that is sent for an SNMP test message:

```
SNMPv2-MIB::snmpTrapOID.0 = OID: SNMPv2-SMI::enterprises.2.6.212.10.0.1
SNMPv2-SMI::enterprises.2.6.212.10.1.1 = STRING: "317908494245422510"
SNMPv2-SMI::enterprises.2.6.212.10.1.2 = STRING: "CLUSTER"
SNMPv2-SMI::enterprises.2.6.212.10.1.3 = STRING: "UNKNOWN"
SNMPv2-SMI::enterprises.2.6.212.10.1.4 = STRING: "GUI"
SNMPv2-SMI::enterprises.2.6.212.10.1.5 = STRING: "INFO"
SNMPv2-SMI::enterprises.2.6.212.10.1.6 = STRING: "18.02.2016 12:47:10.851"
SNMPv2-SMI::enterprises.2.6.212.10.1.7 = STRING: "snmp_test"
SNMPv2-SMI::enterprises.2.6.212.10.1.8 = STRING: "This is a SNMP test message."
SNMPv2-SMI::enterprises.2.6.212.10.1.9 = STRING: "gss-11"
```

### SNMP MIBs

The SNMP Management Information Base (MIB) is a collection of definitions that define the properties of the managed objects.

The IBM Spectrum Scale GUI MIB OID range starts with 1.3.6.1.4.1.2.6.212.10. The OID range 1.3.6.1.4.1.2.6.212.10.0.1 denotes IBM Spectrum Scale GUI event notification (trap) and .1.3.6.1.4.1.2.6.212.10.1.x denotes IBM Spectrum Scale GUI event notification parameters (objects). Use the following text to configure IBM Spectrum Scale GUI MIB:While configuring SNMP, use the MIB file that is available at the following location of each GUI node: `/usr/lpp/mmfs/gui/IBM-SPECTRUM-SCALE-GUI-MIB.txt`.

```
IBM-SPECTRUM-SCALE-GUI-MIB DEFINITIONS ::= BEGIN

IMPORTS
        MODULE-IDENTITY,
```

```
        OBJECT-TYPE,
        NOTIFICATION-TYPE,
        Counter64,
        enterprises
                FROM SNMPv2-SMI
        DisplayString
                FROM RFC1213-MIB;

ibmSpectrumScaleGUI MODULE-IDENTITY
        LAST-UPDATED "201607080000Z"    -- July 08, 2016
        ORGANIZATION "International Business Machines Corp."
        CONTACT-INFO ""
        DESCRIPTION  "Definition of Spectrum Scale GUI Notifications for Spectrum Scale product.
                      These objects are subject to modification by IBM as product specifications require."

 -- Revision log, in reverse chronological order

 REVISION "201607080000Z"    -- July 08, 2016
        DESCRIPTION "Version 0.2."
                            ::= { ibmGPFS 10 }

-- ibmGPFS is copied from GPFS MIB (/usr/lpp/mmfs/data/GPFS-MIB.txt)
ibm                    OBJECT IDENTIFIER ::= { enterprises 2 }
ibmProd                OBJECT IDENTIFIER ::= { ibm 6 }
ibmGPFS                OBJECT IDENTIFIER ::= { ibmProd 212 }

ibmSpectrumScaleGuiNotification OBJECT IDENTIFIER ::= { ibmSpectrumScaleGUI 0}
ibmSpectrumScaleGuiEventObject OBJECT IDENTIFIER ::= { ibmSpectrumScaleGUI 1}

--- ****************************************************************************
--  IBM Spectrum Scale GUI Scalar object declarations - accessible for notifications
--- ****************************************************************************
ibmSpectrumScaleGuiEventCluster OBJECT-TYPE
        SYNTAX DisplayString (SIZE (0..255))
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
                "The cluster where the notification occurred."
      ::= { ibmSpectrumScaleGuiEventObject 1 }

ibmSpectrumScaleGuiEventEntityType OBJECT-TYPE
        SYNTAX DisplayString (SIZE (0..255))
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
                "The type of entity for which the notification occurred."
      ::= { ibmSpectrumScaleGuiEventObject 2 }

ibmSpectrumScaleGuiEventEntityName OBJECT-TYPE
        SYNTAX DisplayString (SIZE (0..255))
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
                "The name of the entity for which the notification occurred."
      ::= { ibmSpectrumScaleGuiEventObject 3 }

ibmSpectrumScaleGuiEventComponent OBJECT-TYPE
        SYNTAX DisplayString (SIZE (0..255))
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
                "The component for which the notification occurred."
      ::= { ibmSpectrumScaleGuiEventObject 4 }

ibmSpectrumScaleGuiEventSeverity OBJECT-TYPE
        SYNTAX DisplayString (SIZE (0..255))
        MAX-ACCESS accessible-for-notify
```

```
        STATUS current
        DESCRIPTION
                "The severity."
      ::= { ibmSpectrumScaleGuiEventObject 5 }

ibmSpectrumScaleGuiEventTime OBJECT-TYPE
        SYNTAX  DisplayString (SIZE (0..255))
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
                "A representation of the date and time when the notification occurred."
      ::= { ibmSpectrumScaleGuiEventObject 6 }

ibmSpectrumScaleGuiEventName OBJECT-TYPE
        SYNTAX DisplayString (SIZE (0..255))
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
                "The event name."
      ::= { ibmSpectrumScaleGuiEventObject 7 }

ibmSpectrumScaleGuiEventMessage OBJECT-TYPE
        SYNTAX DisplayString (SIZE (0..1492))
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
                "The human readable message of the notification."
      ::= { ibmSpectrumScaleGuiEventObject 8 }

ibmSpectrumScaleGuiEventReportingNode OBJECT-TYPE
        SYNTAX DisplayString (SIZE (0..255))
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
                "The node that reported the event."
      ::= { ibmSpectrumScaleGuiEventObject 9 }

ibmSpectrumScaleGuiNotificationEvent NOTIFICATION-TYPE
        OBJECTS {
            ibmSpectrumScaleGuiEventCluster,
            ibmSpectrumScaleGuiEventEntityType,
            ibmSpectrumScaleGuiEventEntityName,
            ibmSpectrumScaleGuiEventComponent,
            ibmSpectrumScaleGuiEventSeverity,
            ibmSpectrumScaleGuiEventTime,
            ibmSpectrumScaleGuiEventName,
            ibmSpectrumScaleGuiEventMessage,
            ibmSpectrumScaleGuiEventReportingNode
         }
         STATUS  current
         DESCRIPTION
           "This notification indicates a Health event as reported by the Spectrum Scale GUI."
        ::= { ibmSpectrumScaleGuiNotification 1 }


END
```

## Performance monitoring

The performance monitoring tool helps to view the metrics that are associated with GPFS and the associated protocols, get a graphical representation of the status and trends of the key performance indicators, and analyze ESS performance problems. You can use both CLI and GUI to monitor the system performance based on various aspects.

The following options are available to monitor system performance:

- **mmperfmon** command that helps to fetch system performance details that are collected through the performance monitoring tools.
- **mmpmon** command that helps to monitor the GPFS performance on the node in which it is run, and other specified nodes.
- ESS GUI
- Open source tool that is called Grafana can be used to monitor performance details that are collected through the performance monitoring tools.

For more information on how to monitor system performance by using mmperfmon and mmpmon commands and Grafana, see Peformance monitoring documentation in *IBM Spectrum Scale: Problem Determination Guide*.

For more information on performance monitoring options that are available in GUI, see "Performance monitoring using ESS GUI."

## Performance monitoring using ESS GUI

The ESS GUI provides a graphical representation of the status and historical trends of the key performance indicators. This helps the users to make decisions easily without wasting time.

The following table lists the performance monitoring options that are available in the ESS GUI.

*Table 10. Performance monitoring options available in ESS GUI*

| Option | Function |
|---|---|
| **Monitoring** > **Statistics** | Displays performance of system resources and file and object storage in various performance charts. You can select the required charts and monitor the performance based on the filter criteria.<br><br>The pre-defined performance widgets and metrics help in investigating every node or any particular node that is collecting the metrics. |
| **Monitoring** > **Dashboards** | Provides an easy to read and real-time user interface that shows a graphical representation of the status and historical trends of key performance indicators. This helps the users to make decisions easily without wasting time. |
| **Monitoring** > **NodesNodes** | Provides an easy way to monitor the performance, health status, and configuration aspects of all available nodes in the ESS cluster. |
| **Network** | Provides an easy way to monitor the performance and health status of various types of networks and network adapters. |
| **Monitoring** > **Thresholds** | Provides an option to 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. |
| **Files** > **File Systems** | Provides a detailed view of the performance and health aspects of file systems. |
| **Files** > **Filesets** | Provides a detailed view of the fileset performance. |
| **Storage** > **Pools** | Provides a detailed view of the performance and health aspects of storage pools. |
| **Storage** > **NSDs** | Provides a detailed view of the performance and health aspects of individual NSDs. |

*Table 10. Performance monitoring options available in ESS GUI (continued)*

| Option | Function |
|---|---|
| **Files** > **Transparent Cloud Tiering** | Provides insight into health, performance and configuration of the transparent cloud tiering service. |
| **Files** > **Active File Management** | Provides a detailed view of the configuration, performance, and health status of AFM cache relationship, AFM disaster recovery (AFMDR) relationship, and gateway nodes. |

The **Statistics** page is used for selecting the attributes based on which the performance of the system needs to be monitored and comparing the performance based on the selected metrics. You can also mark charts as favorite charts and these charts become available for selection when you add widgets in the dashboard. You can display only two charts at a time in the **Statistics** page.

Favorite charts that are defined in the **Statistics** page and the predefined charts are available for selection in the **Dashboard**.

You can configure the system to monitor the performance of the following functional areas in the system:

- Network
- System resources
- Native RAID
- NSD server
- IBM Spectrum Scale client
- NFS
- SMB
- Object
- CTDB
- Transparent cloud tiering. This option is available only when the cluster is configured to work with the transparent cloud tiering service.
- Waiters
- AFM

**Note:** The functional areas such as NFS, SMB, Object, CTDB, and Transparent cloud tiering are available only if the feature is enabled in the system.

The performance and capacity data are collected with the help of the following two components:

- **Sensor:** The sensors are placed on all the nodes and they share the data with the collector. The sensors run on any node that is required to collect metrics. Sensors are started by default only on the protocol nodes.
- **Collector:** Collects data from the sensors. The metric collector runs on a single node and gathers metrics from all the nodes that are running the associated sensors. The metrics are stored in a database on the collector node. The collector ensures aggregation of data once data gets older. The collector can run on any node in the system. By default, the collector runs on the management node. You can configure multiple collectors in the system. To configure performance monitoring through GUI, it is mandatory to configure a collector on each GUI node.

The following picture provides a graphical representation of the performance monitoring configuration for GUI.

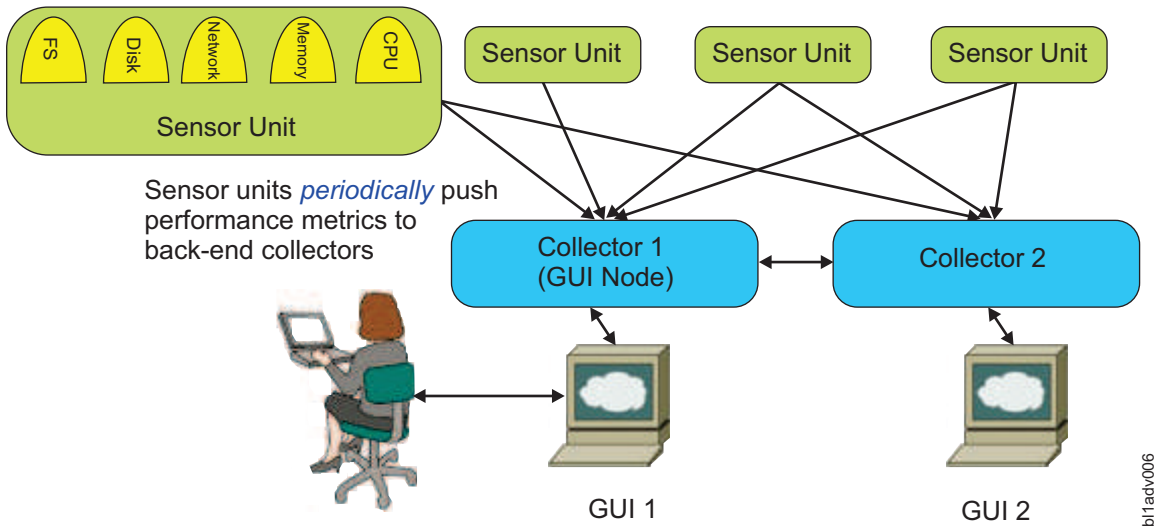



*Figure 8. Performance monitoring configuration for GUI*

The **mmperfmon** command can be used to query performance data through CLI, and configure the performance data collection. The GUI displays a subset of the available metrics.

## Configuring performance monitoring options in GUI

You need to configure and enable the performance monitoring for GUI to view the performance data in the GUI.

### Enabling performance tools in management GUI

You need to enable performance tools in the management GUI to display performance data in the management GUI. For more information on how to enable performance tools in GUI, see *Enabling performance tools in management GUI* section in the *IBM Spectrum Scale: Administration Guide*.

### Verifying sensor and collector configurations

Do the following to verify whether collectors are working properly:

1. Issue **systemctl status pmcollector** on the GUI node to confirm that the collector is running. Start collector it if it is not started already.
2. If you cannot start the service, verify the log file that is located at the following location to fix the issue: `/var/log/zimon/ZIMonCollector.log`.
3. Use a sample CLI query to test if data collection works properly. For example:

   ```
   mmperfmon query cpu_user
   ```

Do the following to verify whether sensors are working properly:

1. Confirm that the sensor is configured correctly by issuing the **mmperfmon config show** command. This command lists the content of the sensor configuration that is located at the following location: `/opt/IBM/zimon/ZIMonSensors.cfg`. The configuration must point to the node where the collector is running and all the expected sensors must be enabled. An enabled sensor has a period greater than `0` in the same config file.
2. Issue **systemctl status pmsensors** to verify the status of the sensors.

## Configuring performance metrics and display options in the Statistics page of the GUI

Use the **Monitoring** > **Statistics** page to monitor the performance of system resources and file and object storage. Performance of the system can be monitored by using various pre-defined charts. You can select the required charts and monitor the performance based on the filter criteria.

The pre-defined performance charts and metrics help in investigating every node or any particular node that is collecting the metrics. The following figure shows various configuration options that are available in the Statistics page of the management GUI.

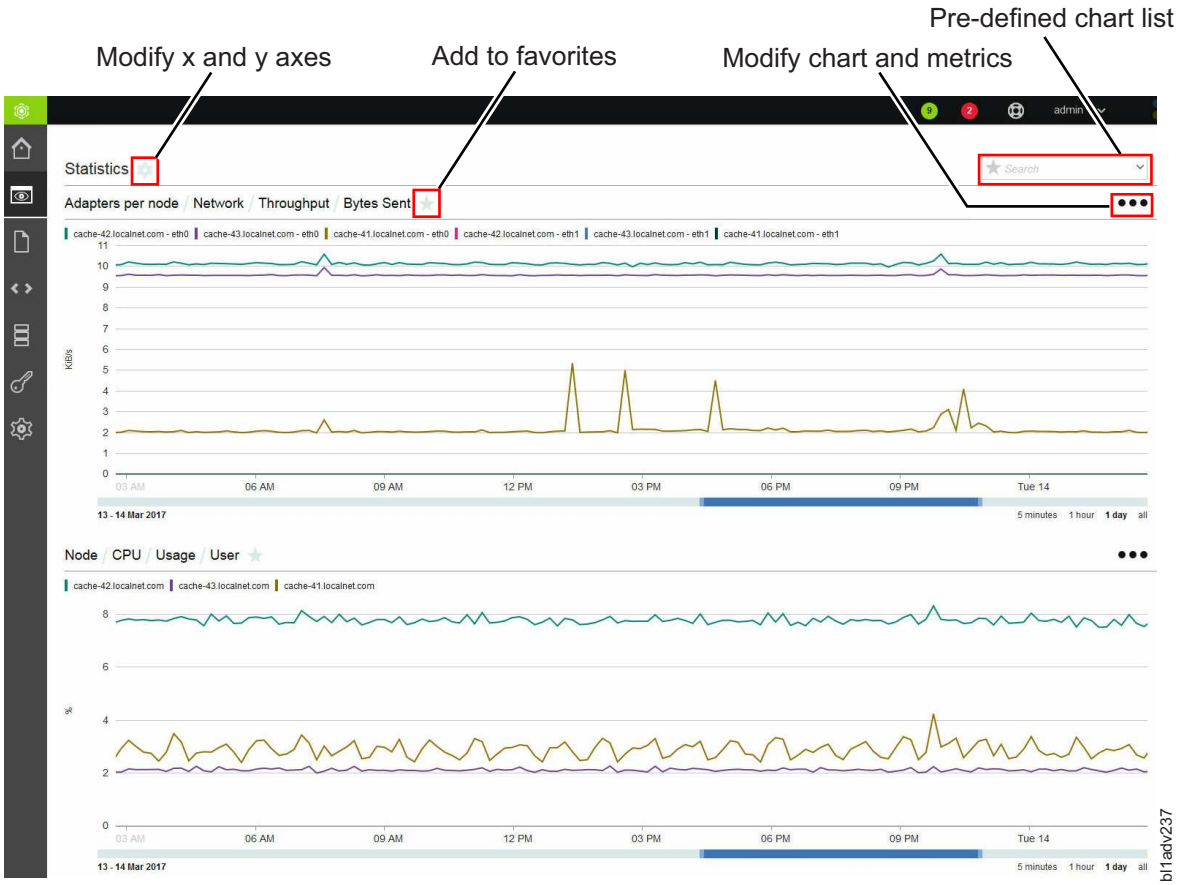



*Figure 9. Statistics page in the IBM Spectrum Scale management GUI*

You can select pre-defined charts that are available for selection from pre-defined chart list. You can display up to two charts at a time.

### Display options in performance charts

The charting section displays the performance details based on various aspects. The GUI provides a rich set of controls to view performance charts. You can use these controls to perform the following actions on the charts that are displayed on the page:

- Zoom the chart by using the mouse wheel or resizing the timeline control. Y-axis can be automatically adjusted during zooming.
- Click and drag the chart or the timeline control at the bottom. Y-axis can be automatically adjusted during panning.
- Compare charts side by side. You can synchronize y-axis and bind x-axis. To modify the x and y axes of the chart, click the configuration symbol next to the title *Statistics* and select the required options.

- Link the timelines of the two charts together by using the display options that are available.
- The Dashboard helps to access all single graph charts, which are either predefined or custom created favorites.

## Selecting performance and capacity metrics

To monitor the performance of the system, you need to select the appropriate metrics to be displayed in the performance charts. Metrics are grouped under the combination of resource types and aggregation levels. The resource types determine the area from which the data is taken to create the performance analysis and aggregation level determines the level at which the data is aggregated. The aggregation levels that are available for selection varies based on the resource type.

Sensors are configured against each resource type. The following table provides a mapping between resource types and sensors under the Performance category.

*Table 11. Sensors available for each resource type*

| Resource type | Sensor name | Candidate nodes |
|---|---|---|
| Network | Network | All nodes |
| System Resources | CPU | All nodes |
| | Load | |
| | Memory | |
| NSD Server | GPFSNSDDisk | NSD server nodes |
| IBM Spectrum Scale Client | GPFSFilesystem | IBM Spectrum Scale Client nodes |
| | GPFSVFS | |
| | GPFSFilesystemAPI | |
| Native RAID | GPFSPDDisk | Elastic Storage Server (ESS) building block nodes |
| NFS | NFSIO | Protocol nodes running NFS service |
| SMB | SMBStats | Protocol nodes running SMB service |
| | SMBGlobalStats | |
| CTDB | CTDBStats | Protocol nodes running SMB service |
| Waiters | GPFSWaiters | All nodes |
| Object | SwiftAccount | Protocol nodes running Object service |
| | SwiftContainer | |
| | SwiftObject | |
| | SwiftProxy | |
| AFM | GPFSAFM | AFM gateway nodes |
| | GPFSAFMFS | |
| | GPFSAFMFSET | |
| Transparent Cloud Tiering | MCStoreGPFSStats | Cloud gateway nodes |
| | MCStoreIcstoreStats | |
| | MCStoreLWEStats | |

The resource type *Waiters* are used to monitor the long running file system threads. Waiters are characterized by the purpose of the corresponding file system threads. For example, an RPC call waiter that is waiting for Network I/O threads or a waiter that is waiting for a local disk I/O file system

operation. Each waiter has a wait time associated with it and it defines how long the waiter is already waiting. With some exceptions, long waiters typically indicate that something in the system is not healthy.

The *Waiters* performance chart shows the aggregation of the total count of waiters of all nodes in the cluster above a certain threshold. Different thresholds from 100 milliseconds to 60 seconds can be selected in the list below the aggregation level. By default, the value shown in the graph is the sum of the number of waiters that exceed threshold in all nodes of the cluster at that point in time. The filter functionality can be used to display waiters data only for some selected nodes or file systems. Furthermore, there are separate metrics for different waiter types such as Local Disk I/O, Network I/O, ThCond, ThMutex, Delay, and Syscall.

You can also monitor the capacity details that are aggregated at the following levels:

- NSD
- Node
- File system
- Pool
- Fileset
- Cluster

The following table lists the sensors that are used for capturing the capacity details.

*Table 12. Sensors available to capture capacity details*

| Sensor name | Candidate nodes |
|---|---|
| DiskFree | All nodes |
| GPFSFilesetQuota | Only a single node |
| GPFSDiskCap | Only a single node |
| GPFSPool | Only a single node where all GPFS file systems are mounted. The GUI does not display any values based on this sensor but it displays warnings or errors due to thresholds based on this sensor. |
| GPFSFileset | Only a single node. The GUI does not display any values based on this sensor but it displays warnings or errors due to thresholds based on this sensor. |

You can edit an existing chart by clicking the icon that is available on the upper right corner of the performance chart and select Edit to modify the metrics selections. Do the following to drill down to the metric you are interested in:

1. Select the cluster to be monitored from the **Cluster** field. You can either select the local cluster or the remote cluster.
2. Select **Resource type**. This is the area from which the data is taken to create the performance analysis.
3. Select **Aggregation level**. The aggregation level determines the level at which the data is aggregated. The aggregation levels that are available for selection varies based on the resource type.
4. Select the entities that need to be graphed. The table lists all entities that are available for the chosen resource type and aggregation level. When a metric is selected, you can also see the selected metrics in the same grid and use methods like sorting, filtering, or adjusting the time frame to select the entities that you want to select.
5. Select **Metrics**. Metrics is the type of data that need to be included in the performance chart. The list of metrics that is available for selection varies based on the resource type and aggregation type.
6. Use the filter option to further narrow down in addition to the objects and metrics selection by using filters. Depending on the selected object category and aggregation level, the "Filter" section can be

displayed underneath the aggregation level, allowing one or more filters to be set. Filters are specified as regular expressions as shown in the following examples:

- As a single entity:
  *node1*
  *eth0*
- Filter metrics applicable to multiple nodes as shown in the following examples:
  - To select a range of nodes such as node1, node2 and node3:
    *node1|node2|node3*
    *node[1-3]*
  - To filter based on a string of text. For example, all nodes starting with 'nod' or ending with 'int':
    *nod.+|.+int*
  - To filter network interfaces eth0 through eth6, bond0 and eno0 through eno6:
    *eth[0-6]|bond0|eno[0-6]*
  - To filter nodes starting with 'strg' or 'int' and ending with 'nx':
    *(strg)|(int).+nx*

### Creating favorite charts

Favorite charts are nothing but customized predefined charts. Favorite charts along with the predefined charts are available for selection when you add widgets in the Dashboard page.

To create favorite charts, click the ‘star’ symbol that is placed next to the chart title and enter the label.

## Configuring the dashboard to view performance charts

The **Monitoring** > **Dashboard** page provides an easy to read, single page, and real-time user interface that provides a quick overview of the system performance.

The dashboard consists of several dashboard widgets and the associated favorite charts that can be displayed within a chosen layout. Currently, the following important widget types are available in the dashboard:

- Performance
- File system capacity by fileset
- System health events
- System overview
- Filesets with the largest growth rate in last week
- Timeline

The following picture highlights the configuration options that are available in the edit mode of the dashboard.

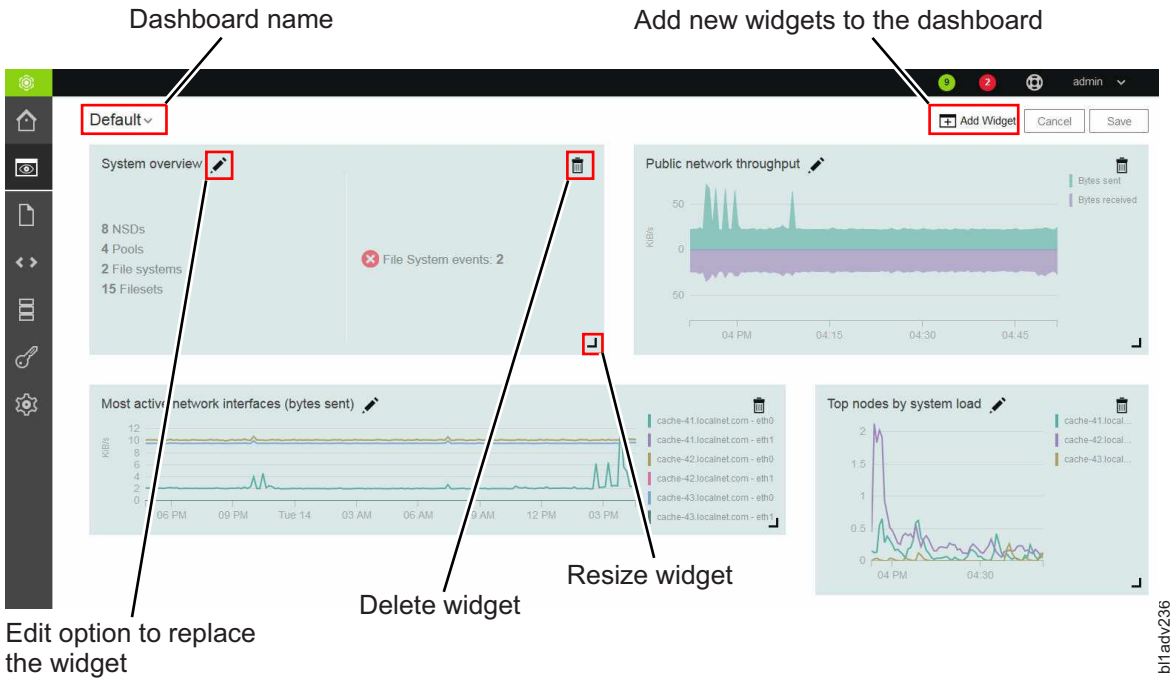



*Figure 10. Dashboard page in the edit mode*

## Layout options

The highly customizable dashboard layout options helps to add or remove widgets and change its display options. Select **Layout Options** option from the menu that is available in the upper right corner of the Dashboard GUI page to change the layout options. While selecting the layout options, you can either select the basic layouts that are available for selection or create a new layout by selecting an empty layout as the starting point.

You can also save the dashboard so that it can be used by other users. Select **Create Dashboard** and **Delete Dashboard** options from the menu that is available in the upper right corner of the Dashboard page to create and delete dashboards respectively. If several GUIs are running by using CCR, saved dashboards are available on all nodes.

When you open the IBM Spectrum Scale GUI after the installation or upgrade, you can see the default dashboards that are shipped with the product. You can further modify or delete the default dashboards to suit your requirements.

## Widget options

Several dashboard widgets can be added in the selected dashboard layout. Select **Edit Widgets** option from the menu that is available in the upper right corner of the Dashboard GUI page to edit or remove widgets in the dashboard. You can also modify the size of the widget in the edit mode. Use the **Add Widget** option that is available in the edit mode to add widgets in the dashboard.

The widgets with type *Performance* lists the charts that are marked as favorite charts in the Statistics page of the GUI. Favorite charts along with the predefined charts are available for selection when you add widgets in the dashboard.

To create favorite charts, click the 'star' symbol that is placed next to the chart title in the **Monitoring** > **Statistics** page.

## Querying performance data shown in the GUI through CLI

You can query the performance data that is displayed in the GUI through the CLI. This is usually used for external system integration or to troubleshoot any issues with the performance data displayed in the GUI.

The following example shows how to query the performance data through CLI:

```
# mmperfmon query "sum(netdev_bytes_r)"
```

This query displays the following output:

```
Legend:
 1:     mr-31.localnet.com|Network|eth0|netdev_bytes_r
 2:     mr-31.localnet.com|Network|eth1|netdev_bytes_r
 3:     mr-31.localnet.com|Network|lo|netdev_bytes_r

Row           Timestamp netdev_bytes_r netdev_bytes_r netdev_bytes_r
  1 2016-03-15-14:52:09          10024
  2 2016-03-15-14:52:10           9456
  3 2016-03-15-14:52:11           9456
  4 2016-03-15-14:52:12           9456
  5 2016-03-15-14:52:13           9456
  6 2016-03-15-14:52:14           9456
  7 2016-03-15-14:52:15          27320
  8 2016-03-15-14:52:16           9456
  9 2016-03-15-14:52:17           9456
 10 2016-03-15-14:52:18          11387
```

The sensor gets the performance data for the collector and the collector passes it to the performance monitoring tool to display it in the CLI and GUI. If sensors and collectors are not enabled in the system, the system does not display the performance data and when you try to query data from a system resource, it returns an error message. For example, if performance monitoring tools are not configured properly for the resource type *Transparent Cloud Tiering*, the system displays the following output while querying the performance data:

```
mmperfmon query "sum(mcs_total_requests)" number_buckets 1
Error: No data available for query: 3169

mmperfmon: Command failed. Examine previous error messages to determine cause.
```

## Monitoring performance of nodes

The **Monitoring** > **Nodes** page provides an easy way to monitor the performance, health status, and configuration aspects of all available nodes in the IBM Spectrum Scale cluster.

The Nodes page provides the following options to analyze performance of nodes:

1. A quick view that gives the number of nodes in the system, and the overall performance of nodes based on CPU and memory usages.

   You can access this view by selecting the expand button that is placed next to the title of the page. You can close this view if not required.

   The graphs in the overview show the nodes that have the highest average performance metric over a past period. These graphs are refreshed regularly. The refresh intervals of the top three entities are depended on the displayed time frame as shown below:

   - Every minute for the 5 minutes time frame
   - Every 15 minutes for the 1 hour time frame
   - Every six hours for the 24 hours time frame
   - Every two days for the 7 days time frame
   - Every seven days for the 30 days time frame
   - Every four months for the 365 days time frame
2. A nodes table that displays many different performance metrics.

To find nodes with extreme values, you can sort the values displayed in the nodes table by different performance metrics. Click the performance metric in the table header to sort the data based on that metric.

You can select the time range that determines the averaging of the values that are displayed in the table and the time range of the charts in the overview from the time range selector, which is placed in the upper right corner. The metrics in the table do not update automatically. The refresh button above the table allows to refresh the table content with more recent data.

You can group the nodes to be monitored based on the following criteria:

- All nodes
- NSD server nodes
- Protocol nodes

3. A detailed view of the performance and health aspects of individual nodes that are listed in the Nodes page.

   Select the node for which you need to view the performance details and select **View Details**. The system displays various performance charts on the right pane.

   The detailed performance view helps to drill-down to various performance aspects. The following list provides the performance details that can be obtained from each tab of the performance view:

   - **Overview** tab provides performance chart for the following:
     - Client IOPS
     - Client data rate
     - Server data rate
     - Server IOPS
     - Network
     - CPU
     - Load
     - Memory
   - **Events** tab helps to monitor the events that are reported in the node. Three filter options are available to filter the events by their status; such as **Current Issues**, **Unread Messages**, and **All Events** displays every event, no matter if it is fixed or marked as read. Similar to the Events page, you can also perform the operations like marking events as read and running fix procedure from this events view.
   - **File Systems** tab provides performance details of the file systems mounted on the node. You can view the file system read or write throughput, average read or write transactions size, and file system read or write latency.
   - **NSDs** tab gives status of the disks that are attached to the node. The NSD tab appears only if the node is configured as an NSD server.
   - **SMB** and **NFS** tabs provide the performance details of the SMB and NFS services hosted on the node. These tabs appear in the chart only if the node is configured as a protocol node.
   - **Network** tab displays the network performance details.

## Monitoring performance of file systems

The File Systems page provides an easy way to monitor the performance, health status, and configuration aspects of the all available file systems in the ESS cluster.

The following options are available to analyze the file system performance:

1. A quick view that gives the number of protocol nodes, NSD servers, and NSDs that are part of the available file systems that are mounted on the GUI server. It also provides overall capacity and total throughput details of these file systems. You can access this view by selecting the expand button that is placed next to the title of the page. You can close this view if not required.

The graphs displayed in the quick view are refreshed regularly. The refresh intervals are depended on the displayed time frame as shown below:

- Every minute for the 5 minutes time frame
- Every 15 minutes for the 1 hour time frame
- Every six hours for the 24 hours time frame
- Every two days for the 7 days time frame
- Every seven days for the 30 days time frame
- Every four months for the 365 days time frame

2. A file systems table that displays many different performance metrics. To find file systems with extreme values, you can sort the values displayed in the file systems table by different performance metrics. Click the performance metric in the table header to sort the data based on that metric. You can select the time range that determines the averaging of the values that are displayed in the table and the time range of the charts in the overview from the time range selector, which is placed in the upper right corner. The metrics in the table do not update automatically. The refresh button above the table allows to refresh the table with more recent data.
3. A detailed view of the performance and health aspects of individual file systems. To see the detailed view, you can either double-click on the file system for which you need to view the details or select the file system and click **View Details**.

   The detailed performance view helps to drill-down to various performance aspects. The following list provides the performance details that can be obtained from each tab of the performance view:

   - **Overview**: Provides an overview of the file system, performance, and properties.
   - **Events**: System health events reported for the file system.
   - **NSDs**: Details of the NSDs that are part of the file system.
   - **Pools**: Details of the pools that are part of the file system.
   - **Nodes**: Details of the nodes on which the file system is mounted.
   - **Filesets**: Details of the filesets that are part of the file system.
   - **NFS**: Details of the NFS exports created in the file system.
   - **SMB**: Details of the SMB shares created in the file system.
   - **Object**: Details of the ESS object storage on the file system.

## Monitoring performance of NSDs

The NSDs page 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 ESS cluster.

The following options are available in the NSDs page to analyze the NSD performance:

1. An NSD table that displays the available NSDs and many different performance metrics. To find NSDs with extreme values, you can sort the values that are displayed in the table by different performance metrics. Click the performance metric in the table header to sort the data based on that metric. You can select the time range that determines the averaging of the values that are displayed in the table from the time range selector, which is placed in the upper right corner. The metrics in the table are refreshed based on the selected time frame. You can refresh it manually to see the latest data.
2. A detailed view of the performance and health aspects of individual NSDs are also available in the NSDs page. Select the NSD for which you need to view the performance details and select **View Details.** The system displays various performance charts on the right pane.

   The detailed performance view helps to drill-down to various performance aspects. The following list provides the performance details that can be obtained from each tab of the performance view:

   - **Overview:** Provides an overview of the NSD performance details and related attributes.
   - **Events:** System health events reported for the NSD.
   - **Nodes:** Details of the nodes that serve the NSD.
   - **Vdisk:** Displays the ESS aspects of NSD.

# Monitoring IBM Spectrum Scale RAID I/O performance

You can use the IBM Spectrum Scale **mmpmon** command to monitor IBM Spectrum Scale RAID I/O performance.

The **mmpmon** command includes input requests for displaying and resetting vdisk I/O statistics.

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

For more information about using **mmpmon** to monitor IBM Spectrum Scale I/O performance, see the *IBM Spectrum Scale: : Administration Guide*.

## Displaying vdisk I/O statistics

To display vdisk I/O statistics, run **mmpmon** with the following command included in the input file:

**vio_s** [**f** [**rg** *RecoveryGroupName* [ **da** *DeclusteredArrayName* [ **v** *VdiskName*]]]] [**reset**]

This request returns strings containing vdisk I/O statistics as seen by that node. The values are presented as total values for the node, or they can be filtered with the **f** option. The reset option indicates that the statistics should be reset after the data is sampled.

If the **-p** option is specified when running **mmpmon**, the vdisk I/O statistics are provided in the form of keywords and values in the **vio_s** response. Table 13 lists and describes these keywords in the order in which they appear in the output.

*Table 13. Keywords and descriptions of values provided in the* **mmpmon vio_s** *response*

| Keyword | Description |
|---|---|
| **_n_** | The IP address of the node that is responding. This is the address by which IBM Spectrum Scale knows the node. |
| **_nn_** | The name by which IBM Spectrum Scale knows the node. |
| **_rc_** | The reason or error code. In this case, the reply value is 0 (OK). |
| **_t_** | The current time of day in seconds (absolute seconds since Epoch (1970)). |
| **_tu_** | The microseconds part of the current time of day. |
| **_rg_** | The name of the recovery group. |
| **_da_** | The name of the declustered array. |
| **_v_** | The name of the disk. |
| **_r_** | The total number of read operations. |
| **_sw_** | The total number of short write operations. |
| **_mw_** | The total number of medium write operations. |
| **_pfw_** | The total number of promoted full track write operations. |
| **_ftw_** | The total number of full track write operations. |
| **_fuw_** | The total number of flushed update write operations. |
| **_fpw_** | The total number of flushed promoted full track write operations. |
| **_m_** | The total number of migrate operations. |
| **_s_** | The total number of scrub operations. |
| **_l_** | The total number log write operations. |
| **_fc_** | The total number of force consistency operations. |
| **_fix_** | The total number of buffer range fix operations. |

*Table 13. Keywords and descriptions of values provided in the* **mmpmon vio_s** *response (continued)*

| Keyword | Description |
|---|---|
| **_ltr_** | The total number of log tip read operations. |
| **_lhr_** | The total number of log home read operations. |
| **_rgd_** | The total number of recovery group descriptor write operations. |
| **_meta_** | The total number metadata block write operations. |

To display these statistics, use the sample script `/usr/lpp/mmfs/samples/vdisk/viostat`. The following shows the usage of the `viostat` script:

```
viostat  [-F NodeFile | [--recovery-group RecoveryGroupName
                         [--declustered-array  DeclusteredArrayName
                         [--vdisk VdiskName]]]]
         [Interval [Count]]
```

**Example of mmpmon vio_s request:**
Suppose **commandFile** contains this line:

```
vio_s
```

and this command is issued:

```
mmpmon -i commandFile
```

The output is similar to this:

```
mmpmon node 172.28.213.53 name kibgpfs003 vio_s OK VIOPS per second
timestamp:                          1399010914/597072
recovery group:                     *
declustered array:                  *
vdisk:                              *
client reads:                       207267
client short writes:                26162
client medium writes:               2
client promoted full track writes:  1499
client full track writes:           864339
flushed update writes:              0
flushed promoted full track writes: 0
migrate operations:                 24
scrub operations:                   5307
log writes:                         878084
force consistency operations:       0
fixit operations:                   0
logTip read operations:             48
logHome read operations:            52
rgdesc writes:                      12
metadata writes:                    5153
```

## Resetting vdisk I/O statistics

The **vio_s_reset** request resets the statistics that are displayed with **vio_s** requests.

Table 14 describes the keywords for the **vio_s_reset** response, in the order that they appear in the output. These keywords are used only when **mmpmon** is invoked with the **-p** flag. The response is a single string.

*Table 14. Keywords and values for the* **mmpmon vio_s_reset** *response*

| Keyword | Description |
|---|---|
| **_n_** | IP address of the node that is responding. This is the address by which IBM Spectrum Scale knows the node. |
| **_nn_** | The hostname that corresponds to the IP address (the **_n_** value). |

*Table 14. Keywords and values for the* **mmpmon vio_s_reset** *response (continued)*

| Keyword | Description |
|---|---|
| **_rc_** | Indicates the status of the operation. |
| **_t_** | Indicates the current time of day in seconds (absolute seconds since Epoch (1970)). |
| **_tu_** | Microseconds part of the current time of day. |

**Example of mmpmon vio_s_reset request:**
Suppose **commandFile** contains this line:

```
vio_s_reset
```

and this command is issued:

```
mmpmon -p -i commandFile
```

The output is similar to this:

```
_vio_s_reset_ _n_ 199.18.1.8 _nn_ node1 _rc_ 0 _t_ 1066660148 _tu_ 407431
```

If the **-p** flag is not specified, the output is similar to this:

```
mmpmon node 199.18.1.8 name node1 reset OK
```

# Monitoring capacity through GUI

You can monitor the capacity of the file system, pools, filesets, NSDs, users, and user groups in the IBM Spectrum Scale system.

The capacity details displayed in the GUI are obtained from the following sources:

- GPFS quota database. The system collects the quota details and stores it in the postgres database.
- Performance monitoring tool. The performance monitoring tool collects the capacity data and displays it in the various pages in the GUI.

Based on the source of the capacity information, different procedures need to be performed to enable capacity and quota data collection.

For both GPFS quota database and performance monitoring tool-based capacity and quota collection, you need to use the **Files** > **Quotas** page to enable quota data collection per file system and enforce quota limit checking. If quota is not enabled for a file system:

- No capacity and inode data is collected for users, groups, and filesets.
- Quota limits for users, groups, and filesets cannot be defined.
- No alerts are sent and the data writes are not restricted.

To enable capacity data collection from the performance monitoring tool, the **GPFSFilesetQuota** sensor must be enabled. For more information on how to enable the performance monitoring sensor for capacity data collection, see *Manual installation of IBM Spectrum Scale GUI* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

## Capacity data obtained from the GPFS quota database

The capacity and quota information collected from the GPFS quota database is displayed on the Files > Quotas and Files > User Capacity pages in the management GUI.

1. **Files** > **Quotas page**

Use quotas to control the allocation of files and data blocks in a file system. You can create default, user, group, and fileset quotas through the Quotas page.

A quota is the amount of disk space and the amount of metadata that is assigned as upper limits for a specified user, group of users, or fileset. Use the **Actions** menu to create or modify quotas. The management GUI allows you to only manage capacity-related quota. The inode-related quota management is only possible in the command-line interface.

You can specify a soft limit, a hard limit, or both. When you set a soft limit quota, a warning is sent to the administrator when the file system is close to reaching its storage limit. A grace period starts when the soft quota limit is reached. Data is written until the grace period expires, or until the hard quota limit is reached. Grace time resets when used capacity goes below the soft limit. If you set a hard limit quota, you cannot save data after the quota is reached. If the quota is exceeded, you must delete files or raise the quota limit to store more data.

**Note:**

- User or user group quotas for filesets are only supported if the *Per Fileset* option is enabled at the file system level. Use the command-line interface to set the option. See the manpages of **mmcrfs** and **mmchfs** commands for more detail.
- You need to unmount a file system to change the quota enablement method from per file system to per fileset or vice versa.

You can set default user quotas at the file system level rather than defining user quotas explicitly for each user. Default quota limits can be set for users. You can specify the general quota collection scope such as per file system or per fileset to define whether the default quota needs to be defined at file system level or fileset level and set the default user quota. After this value is set, all child objects that are created under the file system or fileset will be configured with the default soft and hard limits. You can assign a custom quota limit to individual child objects, but the default limits remain the same unless changed at the file system or fileset level.

After reconfiguring quota settings, it is recommended to run the **mmcheckquota** command for the affected file system to verify the changes.

For more information on how to manage quotas, see *Managing GPFS quotas* section in the *IBM Spectrum Scale: Administration Guide*.

Capacity data from users, groups, and filesets with no quota limit set are not listed in the Quotas page. Use the **Files** > **User Capacity** page to see capacity information of such users and groups. Use the **Files** > **Filesets** page to view current and historic capacity information of filesets.

2. **Files** > **User Capacity page**

The **Files** > **User Capacity** page provides predefined capacity reports for users and groups. While capacity information of file systems, pools, and filesets is available in the respective areas of the GUI, the **Files** > **User Capacity** page is the only place where information on used capacity per user or group is available.

The User Capacity page depends on the quota accounting method of the file system. You need to enable quota for a file system to display the user capacity data. If quota is not enabled, you can follow the fix procedure in the **Files** > **Quotas** page or use the **mmchfs <Device> -Q yes** CLI command to enable quota. Even if the capacity limits are not set, the User Capacity page shows data as soon as the quota accounting is enabled and users write data. This is different in the Quotas page, where only users and groups with quota limits defined are listed. The user and group capacity quota information is automatically collected once a day by the GUI.

For users and user groups, you can see the total capacity and whether quotas are set for these objects. you can also see the percentage of soft limit and hard limit usage. When the hard limit is exceeded, no more files belong to the respective user, user group, or fileset can be written. However, exceeding the hard limit allows a certain grace period before disallowing more file writes. Soft and hard limits for disk capacity are measured in units of kilobytes (KiB), megabytes (MiB), or gigabytes (GiB). Use the **Files** > **Quotas** page to change the quota limits.

## Capacity data collected through the performance monitoring tool

The historical capacity data collection for file systems, pools, and filesets depend on the correctly configured data collection sensors for fileset quota and disk capacity. When the IBM Spectrum Scale system is installed through the installation toolkit, the capacity data collection is configured by default. In other cases, you need to enable capacity sensors manually.

If the capacity data collection is not configured correctly you can use **mmperfmon** CLI command or the **Services** > **Performance Monitoring** > **Sensors** page.

The **Services** > **Performance Monitoring** > **Sensors** page allows to view and edit the sensor settings. By default, the collection periods of capacity collection sensors are set to collect data with a period of up to one day. Therefore, it might take a while until the data is refreshed in the GUI.

The following sensors are collecting capacity related information and are used by the GUI.

**GPFSDiskCap**
: NSD, Pool and File system level capacity. Uses the **mmdf** command in the background and typically runs once per day as it is resource intensive. Should be restricted to run on a single node only.

**GPFSPool**
: Pool and file system level capacity. Requires a mounted file system and typically runs every 5 minutes. Should be restricted to run on a single node only.

**GPFSFilesetQuota**
: Fileset capacity based on the quota collection mechanism. Typically, runs every hour. Should be restricted to run only on a single node.

**GPFSFileset**
: Inode space (independent filest) capacity and limits. Typically runs every 5 minutes. Should be restricted to run only on a single node.

**DiskFree**
: Overall capacity and local node capacity. Can run on every node.

The **Monitoring** > **Statistics** page allows to create customized capacity reports for file systems, pools and filesets. You can store these reports as favorites and add them to the dashboard as well.

The dedicated GUI pages combine information about configuration, health, performance, and capacity in one place. The following GUI pages provide the corresponding capacity views:

- **Files** > **File Systems**
- **Files** > **Filesets**
- **Storage** > **Pools**
- **Storage** > **NSDs**

The Filesets grid and details depend on quota that is obtained from the GPFS quota database and the performance monitoring sensor *GPFSFilesetQuota*. If quota is disabled, the system displays a warning dialog in the Filesets page.

## Troubleshooting issues with capacity data displayed in the GUI

Due to the impact that capacity data collection can have on the system, different capacity values are collected on a different schedule and are provided by different system components. The following list provides insight on the issues that can arise from the multitude of schedules and subsystems that provide capacity data:

**Capacity in the file system view and the total amount of the capacity for pools and volumes view do not match.**

The capacity data in the file system view is collected every 10 minutes by performance monitoring collector, but the capacity data for pools and Network Shared Disks (NSD) are not updated. By default, NSD data is only collected once per day by performance monitoring collector and it is cached. Clicking the refresh icon gathers the last two records from performance monitoring tool and it displays the last record values if they are not null. If the last record has null values, the system displays the previous one. If the values of both records are null, the system displays N/A and the check box for displaying a time chart is disabled. The last update date is the record date that is fetched from performance monitoring tool if the values are not null.

**Capacity in the file system view and the total amount of used capacity for all filesets in that file system do not match.**

There are differences both in the collection schedule as well as in the collection mechanism that contributes to the fact that the fileset capacities do not add up to the file system used capacity.

**Scheduling differences:**

Capacity information that is shown for filesets in the GUI is collected once per hour by performance monitoring collector and displayed on Filesets page. When you click the refresh icon you get the information of the last record from performance monitoring. If the last two records have null values, you get a 'Not collected' warning for used capacity. The file system capacity information on the file systems view is collected every 10 minutes by performance monitoring collector and when you click the refresh icon you get the information of the last record from performance monitoring.

**Data collection differences:**

Quota values show the sum of the size of all files and are reported asynchronously. The quota reporting does not consider metadata, snapshots, or capacity that cannot be allocated within a subblock. Therefore, the sum of the fileset quota values can be lower than the data shown in the file system view. You can use the CLI command **mmlsfileset** with the **-d** and **-i** options to view capacity information. The GUI does not provide a means to display this values because of the performance impact due to data collection.

**The sum of all fileset inode values on the view quota window does not match the number of inodes that are displayed on the file system properties window.**

The quota value only accounts for user-created inodes while the properties for the file system also display inodes that are used internally. Refresh the quota data to update these values.

**No capacity data shown on a new system or for a newly created file system**

Capacity data may show up with a delay of up to 1 day. The capacity data for file systems, NSDs, and pools is collected once a day as this is a resource intensive operation. Line charts do not show a line if only a single data point exists. You can use the hover function in order to see the first data point in the chart.

**The management GUI displays negative fileset capacity or an extremely high used capacity like millions of Petabytes or 4000000000 used inodes.**

This problem can be seen in the quota and filesets views. This problem is caused when the quota accounting is out of sync. To fix this error, issue the **mmcheckquota** command. This command recounts inode and capacity usage in a file system by user, user group, and fileset, and writes the collected data into the database. It also checks quota limits for users, user groups, and filesets in a file system. Running this command can impact performance of I/O operations.

**No capacity data is displayed on the performance monitoring charts**

Verify whether the GPFSFilesetQuota sensor is enabled. You can check the sensor status form the Services > Performance Monitoring page in the GUI. For more information on how to enable the performance monitoring sensor for capacity data collection, see *Manual installation of IBM Spectrum Scale GUI* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

# Chapter 7. Checking the health of an ESS configuration: a sample scenario

The scenario presented here shows how to use the **gnrhealthcheck** sample script to check the general health of an ESS configuration.

1. In this example, all checks are successful.

   To run a health check on the local server nodes and place output in `/tmp/gnrhealthcheck.out`, issue the following command:

   ```
   gnrhealthcheck --local | tee /tmp/gnrhealthcheck.out
   ```

   The system displays information similar to this:

   ```
   ############################################################
   # Beginning topology checks.
   ############################################################
   Topology checks successful.

   ############################################################
   # Beginning enclosure checks.
   ############################################################
   Enclosure checks successful.

   ############################################################
   # Beginning recovery group checks.
   ############################################################
   Recovery group checks successful.

   ############################################################
   # Beginning pdisk checks.
   ############################################################
   Pdisk group checks successful.

   ################################
   Beginning IBM Power RAID checks.
   ################################
   IBM Power RAID checks successful.
   ```

2. In this example, several issues need to be investigated.

   To run a health check on the local server nodes and place output in `/tmp/gnrhealthcheck.out`, issue the following command:

   ```
   gnrhealthcheck --local | tee /tmp/gnrhealthcheck.out
   ```

   The system displays information similar to this:

   ```
   ############################################################
   # Beginning topology checks.
   ############################################################
   Found topology problems on node c45f01n01-ib0.gpfs.net

   DCS3700 enclosures found: 0123456789AB SV11812206 SV12616296 SV13306129
   Enclosure 0123456789AB (number 1):
   Enclosure 0123456789AB ESM A sg244[0379][scsi8 port 4] ESM B sg4[0379][scsi7 port 4]
   Enclosure 0123456789AB Drawer 1 ESM sg244 12 disks diskset "19968" ESM sg4 12 disks diskset "19968"
   Enclosure 0123456789AB Drawer 2 ESM sg244 12 disks diskset "11294" ESM sg4 12 disks diskset "11294"
   Enclosure 0123456789AB Drawer 3 ESM sg244 12 disks diskset "60155" ESM sg4 12 disks diskset "60155"
   Enclosure 0123456789AB Drawer 4 ESM sg244 12 disks diskset "03345" ESM sg4 12 disks diskset "03345"
   Enclosure 0123456789AB Drawer 5 ESM sg244 11 disks diskset "33625" ESM sg4 11 disks diskset "33625"
   Enclosure 0123456789AB sees 59 disks

   Enclosure SV12616296 (number 2):
   Enclosure SV12616296 ESM A sg63[0379][scsi7 port 3] ESM B sg3[0379][scsi9 port 4]
   Enclosure SV12616296 Drawer 1 ESM sg63 11 disks diskset "51519" ESM sg3 11 disks diskset "51519"
   ```

```
Enclosure SV12616296 Drawer 2 ESM sg63 12 disks diskset "36246" ESM sg3 12 disks diskset "36246"
Enclosure SV12616296 Drawer 3 ESM sg63 12 disks diskset "53750" ESM sg3 12 disks diskset "53750"
Enclosure SV12616296 Drawer 4 ESM sg63 12 disks diskset "07471" ESM sg3 12 disks diskset "07471"
Enclosure SV12616296 Drawer 5 ESM sg63 11 disks diskset "16033" ESM sg3 11 disks diskset "16033"
Enclosure SV12616296 sees 58 disks

Enclosure SV11812206 (number 3):
Enclosure SV11812206 ESM A sg66[0379][scsi9 port 3] ESM B sg6[0379][scsi8 port 3]
Enclosure SV11812206 Drawer 1 ESM sg66 11 disks diskset "23334" ESM sg6 11 disks diskset "23334"
Enclosure SV11812206 Drawer 2 ESM sg66 12 disks diskset "16332" ESM sg6 12 disks diskset "16332"
Enclosure SV11812206 Drawer 3 ESM sg66 12 disks diskset "52806" ESM sg6 12 disks diskset "52806"
Enclosure SV11812206 Drawer 4 ESM sg66 12 disks diskset "28492" ESM sg6 12 disks diskset "28492"
Enclosure SV11812206 Drawer 5 ESM sg66 11 disks diskset "24964" ESM sg6 11 disks diskset "24964"
Enclosure SV11812206 sees 58 disks

Enclosure SV13306129 (number 4):
Enclosure SV13306129 ESM A sg64[0379][scsi8 port 2] ESM B sg353[0379][scsi7 port 2]
Enclosure SV13306129 Drawer 1 ESM sg64 11 disks diskset "47887" ESM sg353 11 disks diskset "47887"
Enclosure SV13306129 Drawer 2 ESM sg64 12 disks diskset "53906" ESM sg353 12 disks diskset "53906"
Enclosure SV13306129 Drawer 3 ESM sg64 12 disks diskset "35322" ESM sg353 12 disks diskset "35322"
Enclosure SV13306129 Drawer 4 ESM sg64 12 disks diskset "37055" ESM sg353 12 disks diskset "37055"
Enclosure SV13306129 Drawer 5 ESM sg64 11 disks diskset "16025" ESM sg353 11 disks diskset "16025"
Enclosure SV13306129 sees 58 disks

DCS3700 configuration: 4 enclosures, 1 SSD, 7 empty slots, 233 disks total
Location 0123456789AB-5-12 appears empty but should have an SSD
Location SV12616296-1-3 appears empty but should have an SSD
Location SV12616296-5-12 appears empty but should have an SSD
Location SV11812206-1-3 appears empty but should have an SSD
Location SV11812206-5-12 appears empty but should have an SSD

scsi7[07.00.00.00] 0000:11:00.0 [P2 SV13306129 ESM B (sg353)] [P3 SV12616296 ESM A (sg63)]
[P4 0123456789AB ESM B (sg4)]
scsi8[07.00.00.00] 0000:8b:00.0 [P2 SV13306129 ESM A (sg64)] [P3 SV11812206 ESM B (sg6)]
[P4 0123456789AB ESM A (sg244)]
scsi9[07.00.00.00] 0000:90:00.0 [P3 SV11812206 ESM A (sg66)] [P4 SV12616296 ESM B (sg3)]

############################################################
# Beginning enclosure checks.
############################################################
Enclosure checks successful.

############################################################
# Beginning recovery group checks.
############################################################
Found recovery group BB1RGR, primary server is not the active server.

############################################################
# Beginning pdisk checks.
############################################################
Found recovery group BB1RGL pdisk e4d5s06 has 0 paths.

################################
Beginning IBM Power RAID checks.
################################
IBM Power RAID Array is running in degraded mode.
Name   PCI/SCSI Location          Description               Status
------ ------------------------  ------------------------- -----------------
       0007:90:00.0/0:           PCI-E SAS RAID Adapter    Operational
       0007:90:00.0/0:0:1:0      Advanced Function Disk    Failed
       0007:90:00.0/0:0:2:0      Advanced Function Disk    Active
sda    0007:90:00.0/0:2:0:0      RAID 10  Disk Array       Degraded
       0007:90:00.0/0:0:0:0      RAID 10  Array Member     Active
       0007:90:00.0/0:0:3:0      RAID 10  Array Member     Failed
       0007:90:00.0/0:0:4:0      Enclosure                 Active
       0007:90:00.0/0:0:6:0      Enclosure                 Active
       0007:90:00.0/0:0:7:0      Enclosure                 Active
```

See the “gnrhealthcheck script” on page 262 for more information.

# Chapter 8. Setting up IBM Spectrum Scale RAID on the Elastic Storage Server

This section includes information about setting IBM Spectrum Scale RAID up on the Elastic Storage Server.

It provides information about preparing ESS recovery group servers and creating recovery groups on the ESS.

## Configuring IBM Spectrum Scale RAID recovery groups on the ESS: a sample scenario

This topic provides a detailed example of configuring IBM Spectrum Scale RAID on an ESS building block. The example considers one GL4 building block, which consists of two recovery group servers cabled to four DCS3700 JBOD disk enclosures, and shows how recovery groups are defined on the disk enclosures. Throughout this topic, it might be helpful to have ESS documentation close at hand.

## Preparing ESS recovery group servers

This topic includes information about disk enclosure and host bus adapter (HBA) cabling and provides a procdeure for verifying that a GL4 building block is configured correctly.

### Disk enclosure and HBA cabling

The GL4 DCS3700 JBOD disk enclosures should be cabled to the intended recovery group servers according to the ESS hardware installation instructions. The GL4 building block contains four disk enclosures. Each disk enclosure contains five disk drawers. In each disk enclosure, drawers 2, 3, and 4 contain 12 HDDs. In disk enclosure 1 only, drawers 1 and 5 have one SSD and 11 HDDs. In disk enclosures 2, 3, and 4 (and, in the case of GL6, 5 and 6), drawers 1 and 5 have 11 HDDs and 1 empty slot. In total, a GL4 building block has 4 enclosures, 20 drawers, 232 HDDs, 2 SSDs, and 6 empty slots. (For comparison, a GL6 building block has 6 enclosures, 30 drawers, 348 HDDs, 2 SSDs, and 10 empty slots.) Each disk enclosure provides redundant A and B environmental service modules (ESM) port connections for host server HBA connections. To ensure proper multi-pathing and redundancy, each recovery group server must be connected to ESM A and ESM B using different HBAs. The ESS hardware installation documentation describes precisely which HBA ports must be connected to which disk enclosure ESM ports.

IBM Spectrum Scale RAID provides system administration tools for verifying the correct connectivity of GL4 and GL6 disk enclosures, which will be seen later during the operating system preparation.

When the ESM ports of the GL4 disk enclosures have been cabled to the appropriate HBAs of the two recovery group servers, the disk enclosures should be powered on and the recovery group servers should be rebooted.

### Verifying that the GL4 building block is configured correctly

Preparation then continues at the operating system level on the recovery group servers, which must be installed with the Red Hat Enterprise Linux distribution that is provided with ESS. After the cabling is done and the servers have been rebooted, it is necessary to perform a thorough discovery of the disk topology on each server.

To proceed, IBM Spectrum Scale must be installed on the recovery group servers, but these servers do not need to be added to a GPFS cluster yet. See *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* for information about installing IBM Spectrum Scale.

IBM Spectrum Scale RAID provides tools in `/usr/lpp/mmfs/samples/vdisk` (linked for convenience in the regular `/usr/lpp/mmfs/bin directory`) for collecting and collating information on any attached GL4 disk enclosures and for verifying that the detected topology is correct. The **mmgetpdisktopology** command examines the list of connected devices for the operating system and produces a colon-delimited database with a line for each discovered GL4 physical disk, disk enclosure ESM expander device, and HBA. **mmgetpdisktopology** should be run on each of the two intended recovery group server nodes, and the results examined to verify that the disk enclosure hardware and software configuration is as expected. An additional tool called **topsummary** concisely summarizes the output of the **mmgetpdisktopology** command.

Create a directory in which to work, and then capture the output of the **mmgetpdisktopology** command from each of the two intended recovery group server nodes:

```
# mkdir gl4
# cd gl4
# ssh server1 /usr/lpp/mmfs/bin/mmgetpdisktopology > server1.top
# ssh server2 /usr/lpp/mmfs/bin/mmgetpdisktopology > server2.top
```

Then view the summary for each of the nodes; the following example is for `server1`:

```
# /usr/lpp/mmfs/samples/vdisk/topsummary server1.top
DCS3700 enclosures found: SV34604344 SV34615757 SV34617244 SV35229088
Enclosure SV35229088 (number 1):
Enclosure SV35229088 ESM A sg419[0396][scsi6 port 2] ESM B sg244[0396][scsi4 port 2]
Enclosure SV35229088 Drawer 1 ESM sg419 12 disks diskset "11082" ESM sg244 12 disks diskset "11082"
Enclosure SV35229088 Drawer 2 ESM sg419 12 disks diskset "16380" ESM sg244 12 disks diskset "16380"
Enclosure SV35229088 Drawer 3 ESM sg419 12 disks diskset "11864" ESM sg244 12 disks diskset "11864"
Enclosure SV35229088 Drawer 4 ESM sg419 12 disks diskset "31717" ESM sg244 12 disks diskset "31717"
Enclosure SV35229088 Drawer 5 ESM sg419 12 disks diskset "11824" ESM sg244 12 disks diskset "11824"
Enclosure SV35229088 sees 60 disks
Enclosure SV34604344 (number 2):
Enclosure SV34604344 ESM A sg185[0396][scsi4 port 1] ESM B sg57[0396][scsi2 port 2]
Enclosure SV34604344 Drawer 1 ESM sg185 11 disks diskset "60508" ESM sg57 11 disks diskset "60508"
Enclosure SV34604344 Drawer 2 ESM sg185 12 disks diskset "29045" ESM sg57 12 disks diskset "29045"
Enclosure SV34604344 Drawer 3 ESM sg185 12 disks diskset "08276" ESM sg57 12 disks diskset "08276"
Enclosure SV34604344 Drawer 4 ESM sg185 12 disks diskset "28750" ESM sg57 12 disks diskset "28750"
Enclosure SV34604344 Drawer 5 ESM sg185 11 disks diskset "34265" ESM sg57 11 disks diskset "34265"
Enclosure SV34604344 sees 58 disks
Enclosure SV34617244 (number 3):
Enclosure SV34617244 ESM A sg7[0396][scsi2 port 1] ESM B sg365[0396][scsi6 port 1]
Enclosure SV34617244 Drawer 1 ESM sg7 11 disks diskset "36679" ESM sg365 11 disks diskset "36679"
Enclosure SV34617244 Drawer 2 ESM sg7 12 disks diskset "18808" ESM sg365 12 disks diskset "18808"
Enclosure SV34617244 Drawer 3 ESM sg7 12 disks diskset "55525" ESM sg365 12 disks diskset "55525"
Enclosure SV34617244 Drawer 4 ESM sg7 12 disks diskset "51485" ESM sg365 12 disks diskset "51485"
Enclosure SV34617244 Drawer 5 ESM sg7 11 disks diskset "24274" ESM sg365 11 disks diskset "24274"
Enclosure SV34617244 sees 58 disks
Enclosure SV34615757 (number 4):
Enclosure SV34615757 ESM A sg305[0396][scsi5 port 2] ESM B sg125[0396][scsi3 port 2]
Enclosure SV34615757 Drawer 1 ESM sg305 11 disks diskset "59007" ESM sg125 11 disks diskset "59007"
Enclosure SV34615757 Drawer 2 ESM sg305 12 disks diskset "01848" ESM sg125 12 disks diskset "01848"
Enclosure SV34615757 Drawer 3 ESM sg305 12 disks diskset "49359" ESM sg125 12 disks diskset "49359"
Enclosure SV34615757 Drawer 4 ESM sg305 12 disks diskset "62742" ESM sg125 12 disks diskset "62742"
Enclosure SV34615757 Drawer 5 ESM sg305 11 disks diskset "62485" ESM sg125 11 disks diskset "62485"
Enclosure SV34615757 sees 58 disks

GSS configuration: 4 enclosures, 2 SSDs, 6 empty slots, 234 disks total, 6 NVRAM partitions

scsi3[19.00.00.00] U78CB.001.WZS00KG-P1-C2-T1  [P2 SV34615757 ESM B (sg125)]
scsi4[19.00.00.00] U78CB.001.WZS00KG-P1-C2-T2  [P1 SV34604344 ESM A (sg185)] [P2 SV35229088 ESM B (sg244)]
scsi5[19.00.00.00] U78CB.001.WZS00KG-P1-C3-T1  [P2 SV34615757 ESM A (sg305)]
```

```
scsi6[19.00.00.00] U78CB.001.WZS00KG-P1-C3-T2  [P1 SV34617244 ESM B (sg365)] [P2 SV35229088 ESM A (sg419)]
scsi0[19.00.00.00] U78CB.001.WZS00KG-P1-C11-T1
scsi2[19.00.00.00] U78CB.001.WZS00KG-P1-C11-T2 [P1 SV34617244 ESM A (sg7)]   [P2 SV34604344 ESM B (sg57)]
```

This output shows a correctly cabled and populated GL4 installation. Four disk enclosures with serial numbers `SV34615757`, `SV34617244`, `SV35229088`, and `SV34604344` are found cabled in the correct order and with the correct numbers of disks.

The section for each enclosure begins with the enclosure number as determined by the cabling, followed by which ESM and HBA ports provide connectivity to the enclosure; the following line indicates that `/dev/sg305` is the SCSI/SES expander device representing the enclosure's A ESM, which is at firmware level 0396 and is connected to port 2 of the HBA that the operating system configured as SCSI host 5:

```
Enclosure SV34615757 ESM A sg305[0396][scsi5 port 2] ESM B sg125[0396][scsi3 port 2]
```

Each drawer of an enclosure is represented by a line indicating how many disks are seen over each ESM and a "diskset" checksum of the disk World Wide Names (WWNs), which is used to verify that each ESM connection sees the same set of disks over different device paths.

If the cabling were incorrect in any way, one or more of the enclosures would indicate "number undetermined", and the following message would be printed:

```
Unable to determine enclosure order; check the HBA to enclosure cabling.
```

If any slots are unexpectedly empty or contain the wrong type of disk, the **topsummary** output will include additional messages such as these:

```
Location SV34615757-2-10 appears only on the sg305 path
Location SV34617244-1-3 appears empty but should have an SSD
Location SV34615757-5-12 has an HDD but should be empty
```

The HBAs are also listed (firmware levels in brackets) with their bus addresses and port connections. For example, `[P2 SV34615757 ESM B (sg125)]` indicates that HBA port 2 is connected to ESM B of enclosure `SV34615757`; ESM B is represented by the `/dev/sg125` SCSI/SES expander device. The HBA location codes and port connections are standard in an ESS building block and are used by **topsummary** to validate the correctness of the cabling.

If the cabling is called out as incorrect and the enclosure order cannot be determined, or if other discrepancies are noted (for example, physical disks that are expected but do not show up at all, or SSDs or HDDs in the wrong locations), corrections should be made and verified before proceeding. This would probably involve shutting down the servers and the disk enclosures and fixing the cabling and disk placement.

The ESS configuration line for the server topology should indicate the presence of six NVRAM partitions. These are not in the disk enclosures, but are local to the server. Two of them will be used to provide high-speed update logging for IBM Spectrum Scale RAID (the other four are reserved for possible later use).

The `server2.top` topology file should also be examined using the **topsummary** sample script and verified to be correct. It should also be verified to have found the same enclosures with the same serial numbers and in the same order. One way to do this is to run the following commands and verify that they give exactly the same output:

```
# /usr/lpp/mmfs/samples/vdisk/topsummary server1.top | grep number
# /usr/lpp/mmfs/samples/vdisk/topsummary server2.top | grep number
```

After the GL4 disk enclosure topologies are verified to be correct on both intended recovery group server nodes, it is highly recommended that the disk enclosures be entered into the IBM Spectrum Scale RAID

component database. This allows the system administrator to provide a meaningful name for each component and to record each of their physical rack locations in a manner intended to make maintenance actions easier.

To proceed (either to populate the IBM Spectrum Scale RAID component database or to create recovery groups), the recovery group servers must now be members of the same GPFS cluster. See the *IBM Spectrum Scale: Administration Guide* for information about creating a GPFS cluster.

After the intended recovery group servers are added to a GPFS cluster, the component database can be updated to identify the equipment rack and the U compartments in which the disk enclosures reside.

In this scenario, a GL4 building block will occupy a 42U Primary Rack. Using the component database, the four disk enclosures and the rack in which they reside may be given meaningful names and associated with each other.

To create the component database entry for a 42U Primary Rack, which has part number 1410HPA, and to give it the descriptive name BB1, use the **mmaddcomp** command:

```
# mmaddcomp 1410HPA --name BB1
```

Do the same for each of the four disk enclosures as identified by the **topsummary** command. In this example, the enclosures were identified according to the cabling in this order:

```
# /usr/lpp/mmfs/samples/vdisk/topsummary server1.top | grep number
Enclosure SV35229088 (number 1):
Enclosure SV34604344 (number 2):
Enclosure SV34617244 (number 3):
Enclosure SV34615757 (number 4):
```

Suppose this is GL4 building block 1, and the desired descriptive names for the disk enclosures are BB1ENC1, BB1ENC2, BB1ENC3, and BB1ENC4. To define these to the component database, use the disk enclosure part number 1818-80E as follows:

```
# mmaddcomp 1818-80E --serial-number SV35229088 --name BB1ENC1
# mmaddcomp 1818-80E --serial-number SV34604344 --name BB1ENC2
# mmaddcomp 1818-80E --serial-number SV34617244 --name BB1ENC3
# mmaddcomp 1818-80E --serial-number SV34615757 --name BB1ENC4
```

You can also use the **mmdiscovercomp** command in place of **mmaddcomp**, then use **mmchcomp** to assign descriptive names to the enclosures.

At this point, the building block rack and disk enclosures can be listed from the component database using the **mmlscomp** command:

```
# mmlscomp
Rack Components
Comp ID Part Number Serial Number Name
------- ----------- ------------- ------
      1 1410HPA                   BB1

Storage Enclosure Components

Comp ID Part Number Serial Number Name    Display ID
------- ----------- ------------- ------- ----------
      2 1818-80E    SV35229088    BB1ENC1
      3 1818-80E    SV34604344    BB1ENC2
      4 1818-80E    SV34617244    BB1ENC3
      5 1818-80E    SV34615757    BB1ENC4
```

Each of the defined components has an ID. The location of the enclosures can be defined according to the four U compartments they occupy in the rack. To do this, use the **mmchcomploc** command to specify the component IDs of the enclosures, the rack that contains them, and the starting U compartment (position) within the rack. The syntax of the **mmchcomploc** command is:

mmchcomploc *Component Container Position*

To define enclosure BB1ENC1 as occupying U compartments 1 through 4 in rack BB1, use this command:

```
# mmchcomploc BB1ENC1 BB1 1
```

Suppose BB1ENC2, BB1ENC3, and BB1ENC4 have starting U compartments 5, 13, and 17, respectively, in rack BB1. Use the following commands to define these relationships:

```
# mmchcomploc BB1ENC2 BB1 5
# mmchcomploc BB1ENC3 BB1 13
# mmchcomploc BB1ENC4 BB1 17
```

The defined component locations can be listed with the **mmlscomploc** command:

```
# mmlscomploc
Component Location
--------- -----------------
BB1ENC1 Rack BB1 U01-04
BB1ENC2 Rack BB1 U05-08
BB1ENC3 Rack BB1 U13-16
BB1ENC4 Rack BB1 U17-20
```

U compartments 9 - 10 and 11 - 12 are presumably occupied by the recovery group servers. They are omitted here as extraneous to this set of examples. They can be named and defined if desired by using component part number 7915AC1 with the **mmaddcomp** command, and then using their component IDs and starting U compartments with the **mmchcomploc** command, as was done with the disk enclosures.

With the component names and locations entered into the IBM Spectrum Scale RAID component database as shown, a common maintenance action such as replacing a disk will be able to supply meaningful direction to the user in locating the equipment in a crowded machine room.

## Creating recovery groups on the ESS

You can create recovery groups using IBM Spectrum Scale RAID commands. You can create vdisks, NSDs, and file systems using IBM Spectrum Scale commands or using the ESS GUI.

To create vdisks, NSDs, and file systems using the ESS GUI, use the **Create File System** action in the **Files > File Systems** view. You must decide whether you will use IBM Spectrum Scale commands or the ESS GUI to create vdisks, NSDs, and file systems. A combination of the two is not supported. The ESS GUI cannot create a file system on existing NSDs.

### Configuring GPFS nodes to be recovery group servers

Having verified the disk enclosure connectivity of the GL4 building block, and having optionally also created a component database to associate names and machine room locations with the storage hardware, the recovery groups may now be created on the GL4 building block disk enclosures and servers.

The servers must be members of the same GPFS cluster and must be configured for IBM Spectrum Scale RAID.

IBM Spectrum Scale must be running on the servers, and the servers should not have been rebooted or had their disk configuration changed since the verified disk topology files were acquired.

### Defining the recovery group layout

The definition of recovery groups on a GL4 building block is accomplished by dividing the drawers of the enclosures into left and right halves. The sharing of GL4 disk enclosures by two servers implies two recovery groups; one is served by one node and one by the other, and each server acts as the other's backup. Half the disks in each enclosure and drawer should belong to one recovery group, and half to

the other. One recovery group will therefore be defined on the disks in the left half of each drawer, slots 1 though 6, and one on the disks in the right half of each drawer, slots 7 through 12. The SSD in drawer 1, slot 3 of the first enclosure will make up the SSD declustered array for the left recovery group, and the SSD in drawer 5, slot 12 of the first enclosure will make up the SSD declustered array of the right recovery group. The remaining 116 HDDs in each half are divided into two vdisk data declustered arrays of 58 disks.

IBM Spectrum Scale RAID provides a script, **mkrginput**, that understands the layout of ESS building blocks and will automatically generate the **mmcrrecoverygroup** stanza files for creating the left and right recovery groups. The **mkrginput** script, when supplied with the output of the **mmgetpdisktopology** command from the two servers, will create recovery group stanza files for the left and right sets of disks.

In the same directory in which the verified correct topology files of the two servers are stored, run the **mkrginput** command on the two topology files:

```
# mkrginput server1.top server2.top
```

(In ESS 3.5, the **-s** parameter can be used with **mkrginput** to create a single data declustered array in GL4 and GL6 building blocks. Do this only if all GL4 and GL6 recovery groups in the cluster are to be treated the same. See the "mkrginput script" on page 265 for more information.)

This will create two files, one for the left set of disks and one for the right set of disks found in the `server1` topology. The files will be named after the serial number of the enclosure determined to be first in the topology, but each will contain disks from all four enclosures. In this case, the resulting stanza files will be `SV35229088L.stanza` for the left half and `SV35229088R.stanza` for the right half:

```
# ls -l SV35229088L.stanza SV35229088R.stanza
-rw-r--r-- 1 root root 7244 Nov 25 09:18 SV35229088L.stanza
-rw-r--r-- 1 root root 7243 Nov 25 09:18 SV35229088R.stanza
```

The recovery group stanza files will follow the recommended best practice for a GL4 building block of defining in each half a declustered array called `NVR` with two NVRAM partitions (one from each server) for fast recovery group RAID update logging; a declustered array called `SSD` with either the left or right SSD to act as a backup for RAID update logging; and two file system data declustered arrays called `DA1` and `DA2` using the regular HDDs. (If the **-s** parameter was used with **mkrginput**, there will be no `DA2` data declustered array.)

The declustered array definitions and their required parameters can be seen by grepping for `daName` in the stanza files:

```
# grep daName SV35229088L.stanza
%da: daName=SSD spares=0 replaceThreshold=1 auLogSize=120m
%da: daName=NVR spares=0 replaceThreshold=1 auLogSize=120m nspdEnable=yes
%da: daName=DA1 VCDSpares=31
%da: daName=DA2 VCDSpares=31
```

The parameters after the declustered array names are required exactly as shown. (If the **-s** parameter was used with **mkrginput**, there will be no `DA2` data declustered array, and the parameters for the `DA1` data declustered array will instead be `spares=4 VCDSpares=60`.)

The disks that have been placed in each declustered array can be seen by grepping for example for `da=NVR` in the stanza file:

```
# grep da=NVR SV35229088L.stanza
%pdisk: pdiskName=n1s01 device=//server1/dev/sda5 da=NVR rotationRate=NVRAM
%pdisk: pdiskName=n2s01 device=//server2/dev/sda5 da=NVR rotationRate=NVRAM
```

This shows that a pdisk called `n1s01` will be created using the `/dev/sda5` NVRAM partition from `server1`, and a pdisk called `n2s01` will use the `/dev/sda5` NVRAM partition on `server2`. The parameters again are required exactly as shown. The name `n1s01` means node 1, slot 1, referring to the server node and the NVRAM partition, or "slot." Similarly, `n2s01` means server node 2, NVRAM slot 1.

The disks in the SSD, DA1, and DA2 declustered arrays can be found using similar grep invocations on the stanza files.

If, as was recommended, the IBM Spectrum Scale RAID component database was used to provide meaningful names for the GL4 building block components, the names of the recovery groups should be chosen to follow that convention. In this example, the building block rack was named BB1 and the enclosures were named BB1ENC1, BB1ENC2, BB1ENC3, and BB1ENC4. It would make sense then to name the left and right recovery groups BB1RGL and BB1RGR. Other conventions are possible as well.

The left and right recovery group stanza files can then be supplied to the **mmcrrecoverygroup** command:

```
# mmcrrecoverygroup BB1RGL -F SV35229088L.stanza --servers server1,server2
mmcrrecoverygroup: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
# mmcrrecoverygroup BB1RGR -F SV35229088R.stanza --servers server2,server1
mmcrrecoverygroup: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

The left recovery group is created with server1 as primary and server2 as backup, and the right recovery group is created with server2 as primary and server1 as backup.

## Verifying recovery group creation

Use the **mmlsrecoverygroup** command to verify that each recovery group was created (BB1RGL shown):

```
# mmlsrecoverygroup BB1RGL -L

                   declustered
recovery group       arrays     vdisks  pdisks  format version
-----------------  -----------  ------  ------  --------------
BB1RGL                       4       0     119  4.1.0.1

declustered   needs                             replace                scrub       background activity
   array     service  vdisks  pdisks  spares  threshold  free space  duration  task   progress  priority
-----------  -------  ------  ------  ------  ---------  ----------  --------  -------------------------
SSD          no            0       1     0,0          1     186 GiB   14 days  repair-RGD/VCD 10%  low
NVR          no            0       2     0,0          1    3744 MiB   14 days  repair-RGD/VCD 10%  low
DA1          no            0      58    2,31          2     138 TiB   14 days  repair-RGD/VCD 10%  low
DA2          no            0      58    2,31          2     138 TiB   14 days  repair-RGD/VCD 10%  low

                                      declustered                           checksum
vdisk                RAID code              array     vdisk size  block size  granularity  state remarks
-------------------  ------------------  -----------  ----------  ----------  -----------  ----- -------

config data          declustered array   VCD spares     actual rebuild spare space         remarks
-------------------  ------------------  -------------  ---------------------------------  ----------------
rebuild space        DA1                 31             36 pdisk
rebuild space        DA2                 31             36 pdisk

config data     max disk group fault tolerance  actual disk group fault tolerance  remarks
--------------  ------------------------------ ---------------------------------- -----------
rg descriptor  1 enclosure + 1 drawer        1 enclosure + 1 drawer        limiting fault tolerance
system index   2 enclosure                   1 enclosure + 1 drawer        limited by rg descriptor

active recovery group server                    servers
----------------------------------------------  -------
server1                                         server1,server2
```

Notice that the vdisk information for the newly-created recovery group is indicated with 0s or is missing; the next step is to create the vdisks.

## Defining and creating the vdisks

Once the recovery groups are created and being served by their respective servers, it is time to create the vdisks using the **mmcrvdisk** command.

The internal RAID transaction and update log vdisks must be created first.

Each of the GL4 left and right recovery groups will now require:

- A log tip vdisk (type `vdiskLogTip`) in the `NVR` declustered array
- A log tip backup vdisk (type `vdiskLogTipBackup`) in the `SSD` declustered array
- A log home vdisk (type `vdiskLog`) in the `DA1` declustered array
- A log reserved vdisk (type `vdiskLogReserved` in the `DA2` declustered array (and in the `DA3` declustered array, in the case of GL6)

These all need to be specified in a log vdisk creation stanza file. (If the **-s** parameter was used with **mkrginput**, disregard references to the log reserved vdisk and `DA2` in the remainder of this example.)

On Power Systems servers, the **checksumGranularity=4k** parameter is required for the various log vdisks in the log vdisk stanza file. This parameter should be omitted on non-Power servers.

The log vdisk creation file for a GL4 building block with NVRAM partitions will look like this:

```
# cat mmcrvdisklog.BB1
%vdisk:
  vdiskName=BB1RGLLOGTIP
  rg=BB1RGL
  daName=NVR
  blocksize=2m
  size=48m
  raidCode=2WayReplication
  checksumGranularity=4k        # Power only
  diskUsage=vdiskLogTip

%vdisk:
  vdiskName=BB1RGLLOGTIPBACKUP
  rg=BB1RGL
  daName=SSD
  blocksize=2m
  size=48m
  raidCode=Unreplicated
  checksumGranularity=4k        # Power only
  diskUsage=vdiskLogTipBackup

%vdisk:
  vdiskName=BB1RGLLOGHOME
  rg=BB1RGL
  daName=DA1
  blocksize=2m
  size=20g
  raidCode=4WayReplication
  checksumGranularity=4k        # Power only
  diskUsage=vdiskLog
  longTermEventLogSize=4m
  shortTermEventLogSize=4m
  fastWriteLogPct=90

%vdisk:
  vdiskName=BB1RGLDA2RESERVED
  rg=BB1RGL
  daName=DA2
  blocksize=2m
  size=20g
  raidCode=4WayReplication
```

```
  checksumGranularity=4k        # Power only
  diskUsage=vdiskLogReserved

%vdisk:
  vdiskName=BB1RGRLOGTIP
  rg=BB1RGR
  daName=NVR
  blocksize=2m
  size=48m
  raidCode=2WayReplication
  checksumGranularity=4k        # Power only
  diskUsage=vdiskLogTip

%vdisk:
  vdiskName=BB1RGRLOGTIPBACKUP
  rg=BB1RGR
  daName=SSD
  blocksize=2m
  size=48m
  raidCode=3WayReplication
  checksumGranularity=4k        # Power only
  diskUsage=vdiskLogTipBackup

%vdisk:
  vdiskName=BB1RGRLOGHOME
  rg=BB1RGR
  daName=DA1
  blocksize=2m
  size=20g
  raidCode=4WayReplication
  checksumGranularity=4k        # Power only
  diskUsage=vdiskLog
  longTermEventLogSize=4m
  shortTermEventLogSize=4m
  fastWriteLogPct=90

%vdisk:
  vdiskName=BB1RGRDA2RESERVED
  rg=BB1RGR
  daName=DA2
  blocksize=2m
  size=20g
  raidCode=4WayReplication
  checksumGranularity=4k        # Power only
  diskUsage=vdiskLogReserved
```

The parameters chosen for size, blocksize, raidCode, fastWriteLogPct, and the event log sizes are standard and have been carefully calculated, and they should not be changed. The only difference in the vdisk log stanza files between two building blocks will be in the recovery group and vdisk names. (In the case of a GL6 building block with NVRAM partitions, there will be an additional vdiskLogReserved for DA3, with parameters otherwise identical to the DA2 log reserved vdisk.)

The **checksumGranularity=4k** parameter is required for Power Systems servers. It should be omitted on non-Power servers.

The log vdisks for the sample GL4 building block BB1 can now be created using the **mmcrvdisk** command:

```
# mmcrvdisk -F mmcrvdisklog.BB1
mmcrvdisk: [I] Processing vdisk BB1RGLLOGTIP
mmcrvdisk: [I] Processing vdisk BB1RGLLOGTIPBACKUP
mmcrvdisk: [I] Processing vdisk BB1RGLLOGHOME
mmcrvdisk: [I] Processing vdisk BB1RGLDA2RESERVED
mmcrvdisk: [I] Processing vdisk BB1RGRLOGTIP
mmcrvdisk: [I] Processing vdisk BB1RGRLOGTIPBACKUP
```

```
mmcrvdisk: [I] Processing vdisk BB1RGRLOGHOME
mmcrvdisk: [I] Processing vdisk BB1RGRDA2RESERVED
mmcrvdisk: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

You can use the **mmlsvdisk** command (or the **mmlsrecoverygroup** command) to verify that the log vdisks have been created:

```
# mmlsvdisk
                                                  declustered  block size
 vdisk name          RAID code        recovery group         array       in KiB    remarks
 ------------------  ---------------  ------------------  -----------  ----------  -------
 BB1RGLDA2RESERVED   4WayReplication  BB1RGL              DA2                2048  logRsvd
 BB1RGLLOGHOME       4WayReplication  BB1RGL              DA1                2048  log
 BB1RGLLOGTIP        2WayReplication  BB1RGL              NVR                2048  logTip
 BB1RGLLOGTIPBACKUP  Unreplicated     BB1RGL              SSD                2048  logTipBackup
 BB1RGRDA2RESERVED   4WayReplication  BB1RGR              DA2                2048  logRsvd
 BB1RGRLOGHOME       4WayReplication  BB1RGR              DA1                2048  log
 BB1RGRLOGTIP        2WayReplication  BB1RGR              NVR                2048  logTip
 BB1RGRLOGTIPBACKUP  Unreplicated     BB1RGR              SSD                2048  logTipBackup
```

Now the file system vdisks may be created.

Data vdisks are required to be defined in the two data declustered arrays for use as file system NSDs. In this example, each of the declustered arrays for file system data is divided into two vdisks with different characteristics:

- one using 4-way replication and a 1 MiB block size and a total vdisk size of 2048 GiB suitable for file system metadata
- one using Reed-Solomon 8 + 3p encoding and an 16 MiB block size suitable for file system data

The vdisk size is omitted for the Reed-Solomon vdisks, meaning that they will default to use the remaining non-spare space in the declustered array (for this to work, any vdisks with specified total sizes must of course be defined first).

The possibilities for the vdisk creation stanza file are quite great, depending on the number and type of vdisk NSDs required for the number and type of file systems desired, so the vdisk stanza file will need to be created by hand, possibly following a template. The sample vdisk stanza file that is supplied in `/usr/lpp/mmfs/samples/vdisk/vdisk.stanza` can be used for this purpose and adapted to specific file system requirements.

The file system NSD vdisk stanza file in this example looks like this:

```
# cat mmcrvdisknsd.BB1
%vdisk: vdiskName=BB1RGLMETA1
  rg=BB1RGL
  da=DA1
  blocksize=1m
  size=2048g
  raidCode=4WayReplication
  diskUsage=metadataOnly
  failureGroup=1
  pool=system

%vdisk: vdiskName=BB1RGLMETA2
  rg=BB1RGL
  da=DA2
  blocksize=1m
  size=2048g
  raidCode=4WayReplication
  diskUsage=metadataOnly
  failureGroup=1
  pool=system

%vdisk: vdiskName=BB1RGRMETA1
```

```
  rg=BB1RGR
  da=DA1
  blocksize=1m
  size=2048g
  raidCode=4WayReplication
  diskUsage=metadataOnly
  failureGroup=1
  pool=system

%vdisk: vdiskName=BB1RGRMETA2
  rg=BB1RGR
  da=DA2
  blocksize=1m
  size=2048g
  raidCode=4WayReplication
  diskUsage=metadataOnly
  failureGroup=1
  pool=system

%vdisk: vdiskName=BB1RGLDATA1
  rg=BB1RGL
  da=DA1
  blocksize=16m
  raidCode=8+3p
  diskUsage=dataOnly
  failureGroup=1
  pool=data

%vdisk: vdiskName=BB1RGLDATA2
  rg=BB1RGL
  da=DA2
  blocksize=16m
  raidCode=8+3p
  diskUsage=dataOnly
  failureGroup=1
  pool=data

%vdisk: vdiskName=BB1RGRDATA1
  rg=BB1RGR
  da=DA1
  blocksize=16m
  raidCode=8+3p
  diskUsage=dataOnly
  failureGroup=1
  pool=data

%vdisk: vdiskName=BB1RGRDATA2
  rg=BB1RGR
  da=DA2
  blocksize=16m
  raidCode=8+3p
  diskUsage=dataOnly
  failureGroup=1
  pool=data
```

Notice how the file system metadata vdisks are flagged for eventual file system usage as metadataOnly and for placement in the system storage pool, and the file system data vdisks are flagged for eventual dataOnly usage in the data storage pool. (After the file system is created, a policy will be required to allocate file system data to the correct storage pools; see “Creating the GPFS file system” on page 98.)

Importantly, also notice that block sizes for the file system metadata and file system data vdisks must be specified at this time, may not later be changed, and must match the block sizes supplied to the eventual **mmcrfs** command.

Notice also that the eventual failureGroup=1 value for the NSDs on the file system vdisks is the same for vdisks in both the BB1RGL and BB1RGR recovery groups. This is because the recovery groups, although they have different servers, still share a common point of failure in the four GL4 disk enclosures, and IBM Spectrum Scale should be informed of this through a distinct failure group designation for each disk enclosure. It is up to the IBM Spectrum Scale system administrator to decide upon the failure group numbers for each ESS building block in the GPFS cluster. In this example, the failure group number 1 has been chosen to match the example building block number.

To create the file system NSD vdisks specified in the mmcrvdisknsd.BB1 file, use the following **mmcrvdisk** command:

```
# mmcrvdisk -F mmcrvdisknsd.BB1
mmcrvdisk: [I] Processing vdisk BB1RGLMETA1
mmcrvdisk: [I] Processing vdisk BB1RGLMETA2
mmcrvdisk: [I] Processing vdisk BB1RGRMETA1
mmcrvdisk: [I] Processing vdisk BB1RGRMETA2
mmcrvdisk: [I] Processing vdisk BB1RGLDATA1
mmcrvdisk: [I] Processing vdisk BB1RGLDATA2
mmcrvdisk: [I] Processing vdisk BB1RGRDATA1
mmcrvdisk: [I] Processing vdisk BB1RGRDATA2
mmcrvdisk: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

You can use the **mmlsvdisk** command or the **mmlsrecoverygroup** command to verify that the vdisks have been created.

## Creating NSDs from vdisks

The **mmcrvdisk** command rewrites the input file so that it is ready to be passed to the **mmcrnsd** command that creates the NSDs from which IBM Spectrum Scale builds file systems. To create the vdisk NSDs, run the **mmcrnsd** command on the rewritten **mmcrvdisk** stanza file:

```
# mmcrnsd -F mmcrvdisknsd.BB1
mmcrnsd: Processing disk BB1RGLMETA1
mmcrnsd: Processing disk BB1RGLMETA2
mmcrnsd: Processing disk BB1RGRMETA1
mmcrnsd: Processing disk BB1RGRMETA2
mmcrnsd: Processing disk BB1RGLDATA1
mmcrnsd: Processing disk BB1RGLDATA2
mmcrnsd: Processing disk BB1RGRDATA1
mmcrnsd: Processing disk BB1RGRDATA2
mmcrnsd: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

The **mmcrnsd** command then once again rewrites the stanza file in preparation for use as input to the **mmcrfs** command.

## Creating the GPFS file system

Run the **mmcrfs** command to create the file system:

```
# mmcrfs gpfsbb1 -F mmcrvdisknsd.BB1 -B 16m --metadata-block-size 1m -T /gpfsbb1 -n 256
The following disks of gpfsbb1 will be formatted on node server1:
BB1RGLMETA1: size 269213696 KB
BB1RGRMETA1: size 269213696 KB
BB1RGLDATA1: size 8593965056 KB
BB1RGRDATA1: size 8593965056 KB
BB1RGLMETA2: size 269213696 KB
BB1RGRMETA2: size 269213696 KB
BB1RGLDATA2: size 8593965056 KB
BB1RGRDATA2: size 8593965056 KB
Formatting file system ...
Disks up to size 3.3 TB can be added to storage pool system.
Disks up to size 82 TB can be added to storage pool data.
```

```
Creating Inode File
Creating Allocation Maps
Creating Log Files
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool system
98 % complete on Tue Nov 25 13:27:00 2014
100 % complete on Tue Nov 25 13:27:00 2014
Formatting Allocation Map for storage pool data
85 % complete on Tue Nov 25 13:27:06 2014
100 % complete on Tue Nov 25 13:27:06 2014
Completed creation of file system /dev/gpfsbb1.
mmcrfs: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

The **-n 256** parameter specifies that the allocation maps should account for 256 nodes mounting the file system. This is an example only and should be adjusted to actual cluster expectations.

Notice how the 16 MiB data block size is specified with the traditional **-B** parameter and the 1 MiB metadata block size is specified with the **--metadata-block-size** parameter. Because a file system with different metadata and data block sizes requires the use of multiple GPFS storage pools, a file system placement policy is needed to direct user file data to the data storage pool. In this example, the file placement policy is very simple:

```
# cat policy
rule 'default' set pool 'data'
```

The policy must then be installed in the file system using the **mmchpolicy** command:

```
# mmchpolicy gpfsbb1 policy -I yes
Validated policy 'policy': parsed 1 Placement Rules, 0 Restore Rules, 0 Migrate/Delete/Exclude Rules,
0 List Rules, 0 External Pool/List Rules
Policy 'policy'. installed and broadcast to all nodes.
```

If a policy is not placed in a file system with multiple storage pools, attempts to place data into files will return ENOSPC as if the file system were full.

This file system, built on a GL4 building block using two recovery groups, two recovery group servers, four file system metadata vdisk NSDs and four file system data vdisk NSDs, can now be mounted and placed into service:

```
# mmmount gpfsbb1 -a
```

# Chapter 9. IBM Spectrum Scale RAID management API

With the IBM Spectrum Scale RAID management API, you can develop scripts to automate labor-intensive cluster management tasks. These APIs provide a useful way to integrate and use the ESS system.

The IBM Spectrum Scale RAID management API is a REST-style interface for managing ESS cluster resources. It runs on HTTPS and uses JSON syntax to frame data inside HTTP requests and responses.

**Note:** This section covers only the IBM Spectrum Scale RAID-specific API commands. The API commands that are available with the IBM Spectrum Scale system are also available to the ESS users. For more information about the API architecture, and the list of IBM Spectrum Scale management API commands that are available, see IBM Spectrum Scale management API.

The following IBM Spectrum Scale RAID management API commands are currently available:

# Gnr/recoverygroups: GET

Gets a list of recovery groups that are configured in the cluster.

## Availability

Available with the Elastic Storage Server.

## Description

The **GET gnr/recoverygroups** request gets a list of recovery groups that are configured in the cluster. For more information about the fields in the data structures that are returned, see "mmlsrecoverygroup command" on page 202.

## Request URL

```
https://IP address of API server:<port>/scalemgmt/v2/gnr/recoverygroups
```

where

**gnr/recoverygroups**
Specifies that recovery groups are the resource of this GET call. Required.

## Request headers

```
Content-Type: application/json
Accept: application/json
```

## Request parameters

The following parameters can be used in the request URL to customize the request:

*Table 15. List of request parameters*

| Parameter name | Description and applicable keywords | Required/optional |
|---|---|---|
| fields | Comma-separated list of fields to be included in response. ':all:' selects all available fields. | |
| filter | Filter objects by expression. For example, 'status=HEALTHY,entityType=FILESET' | |

## Request data

No request data.

## Response data

```
{
   "status": {
      "code":ReturnCode,
      "message":"ReturnMessage"
   },
    "paging":
    {
    "next": "URL"
    },
   "recoveryGroups": [
      {
```

```
            "name":"Name"
            "declusteredArrayCount":"Number",
            "capacity":"Capacity",
            "freeSpace":"Available space",
            "usedSpace":"Used space",
            "nodes":"List of nodes"
            "oid":"ID",
            "vdiskCount":"Number of Vdisks",
            "pdiskCount":"Number of Pdisks",
      }
   ]
}
```

For more information about the fields in the following data structures, see the links at the end of this topic.

**"status":**
Return status.

**"message": "***ReturnMessage***"**
The return message.

**"code":** *ReturnCode*
The return code.

**"paging"**
The URL to retrieve the next page. Paging is enabled when more than 1000 objects are returned by the query.

**"recoveryGroups":**
Details of the recovery group.

**"name":"***Name***"**
The name of the recovery group.

**"declusteredArrayCount":"***Number***"**
Number of declustered arrays present in the recovery group.

**"capacity":"***Capacity***"**
Capacity of the recovery group.

**"freeSpace":"***Available space***"**
Available space that is left in the recovery group.

**"usedSpace":"***Used space***"**
The space used in the recovery group.

**"nodes":"***List of nodes***"**
The nodes that are part of the recovery group.

**"oid":"***ID***"**
The internal identifier of the recovery group that is used for paging.

**"vdiskCount":"***Number of Vdisks***"**
Number of Vdisks that are part of the recovery group.

**"pdiskCount":"***Number of Pdisks***"**
Number of Pdisks that are part of the recovery group.

## Examples

The following example gets information about the recovery groups that are configured in the cluster.

Request data:

```
curl -k -u admin:admin001 -X GET --header 'accept:application/json'
'https://198.51.100.1:443/scalemgmt/v2/gnr/recoverygroups
```

Response data:

**Note:** In the JSON data that is returned, the return code indicates whether the command is successful. The response code 200 indicates that the command successfully retrieved the information. Error code 400 represents an invalid request and 500 represents internal server error.

```
{
  "status": {
    "code": 200,
    "message": "..."
  },
  "paging": {
    "next": "https://localhost:443/scalemgmt/v2/gnr/recoverygroups"
  },
  "recoveryGroups": [
    {
      "name": "RG1",
      "declusteredArrayCount": 2,
      "capacity": 12878459437056,
      "freeSpace": 0,
      "usedSpace": 0,
      "nodes": ["node1"],
      "oid": 0,
      "vdiskCount": 0,
      "pdiskCount": 24
    }
  ]
}
```

"mmlsrecoverygroup command" on page 202
Lists information about IBM Spectrum Scale RAID recovery groups.

"mmcrrecoverygroup command" on page 159
Creates an IBM Spectrum Scale RAID recovery group and its component declustered arrays and pdisks and specifies the servers.

# Gnr/recoverygroups/{recoveryGroupName}: GET

Gets the details of a specific recovery group.

## Availability

Available with the Elastic Storage Server.

## Description

The **GET gnr/recoverygroup/{recoveryGroupName}** request gets the details of a specific recovery group that is configured in the cluster. For more information about the fields in the data structures that are returned, see "mmlsrecoverygroup command" on page 202.

## Request URL

```
https://IP address of API server:<port>/scalemgmt/v2/gnr/recoverygroups/recoveryGroupName
```

where

**gnr/recoverygroups/recoveryGroupName**
Specifies a particular recovery group as the resource of this GET call. Required.

## Request headers

```
Content-Type: application/json
Accept: application/json
```

## Request parameters

The following parameters can be used in the request URL to customize the request:

*Table 16. List of request parameters*

| Parameter name | Description and applicable keywords | Required/optional |
|---|---|---|
| fields | Comma-separated list of fields to be included in response. ':all:' selects all available fields. | Optional. |
| filter | Filter objects by expression. For example, 'status=HEALTHY,entityType=FILESET' | Optional. |
| recoveryGroupName | The name of the recovery group about which you need the details. | Required. |

## Request data

No request data.

## Response data

```
{
   "status": {
      "code":ReturnCode,
      "message":"ReturnMessage"
   },
    "paging":
    {
    "next": "URL"
```

```
    },
   "recoveryGroups": [
      {
            "name":"Name"
            "declusteredArrayCount":"Declustered array count",
            "capacity":"Capacity",
            "freeSpace":"Available space",
            "usedSpace":"Used space",
            "nodes":"List of nodes"
            "oid":"ID",
            "vdiskCount":"Number of Vdisks",
            "pdiskCount":"Number of Pdisks",
      }
   ]
}
```

For more information about the fields in the following data structures, see the links at the end of this topic.

**"status":**
Return status.

**"message": "***ReturnMessage***"**
The return message.

**"code":** *ReturnCode*
The return code.

**"paging"**
The URL to retrieve the next page. Paging is enabled when more than 1000 objects are returned by the query.

**"recoveryGroups":**
Details of the recovery group.

**"name":"***Name***"**
The name of the recovery group.

**"declusteredArrayCount":"***Number***"**
Number of declustered arrays present in the recovery group.

**"capacity":"***Capacity***"**
Capacity of the recovery group.

**"freeSpace":"***Available space***"**
Available space that is left in the recovery group.

**"usedSpace":"***Used space***"**
The space used in the recovery group.

**"nodes":"***List of nodes***"**
The nodes that are part of the recovery group.

**"oid":"***ID***"**
The internal identifier of the recovery group that is used for paging.

**"vdiskCount":"***Number of Vdisks***"**
Number of Vdisks that are part of the recovery group.

**"pdiskCount":"***Number of Pdisks***"**
Number of Pdisks that are part of the recovery group.

## Examples

The following example gets information about the recovery group *RG1* that is configured in the cluster.

Request data:

```
curl -k -u admin:admin001 -X GET --header 'accept:application/json'
'https://198.51.100.1:443/scalemgmt/v2/gnr/recoverygroups/RG1
```

Response data:

**Note:** In the JSON data that is returned, the return code indicates whether the command is successful. The response code 200 indicates that the command successfully retrieved the information. Error code 400 represents an invalid request and 500 represents internal server error.

```
{
  "status": {
    "code": 200,
    "message": "..."
  },
  "paging": {
    "next": "https://localhost:443/scalemgmt/v2/gnr/recoverygroups/RG1"
  },
  "recoveryGroups": [
    {
      "name": "RG1",
      "declusteredArrayCount": 2,
      "capacity": 12878459437056,
      "freeSpace": 0,
      "usedSpace": 0,
      "nodes": ["node1", "node2"],
      "oid": 0,
      "vdiskCount": 0,
      "pdiskCount": 24
    }
  ]
}
```

"mmlsrecoverygroup command" on page 202
Lists information about IBM Spectrum Scale RAID recovery groups.

"mmcrrecoverygroup command" on page 159
Creates an IBM Spectrum Scale RAID recovery group and its component declustered arrays and pdisks and specifies the servers.

# Gnr/recoverygroups/{recoveryGroupName}/pdisks: GET

Gets the details of the physical disks that are available in a particular recovery group.

## Availability

Available with the Elastic Storage Server.

## Description

The **GET gnr/recoverygroup/{recoveryGroupName}/pdisks** request gets the details of the pdisks that are available in a recovery group. For more information about the fields in the data structures that are returned, see "mmlspdisk command" on page 199.

## Request URL

`https://`*IP address of API server*`:<port>/`**scalemgmt/v2/gnr/recoverygroups**`/`*recoveryGroupName*`/`**pdisks**

where

**gnr/recoverygroups/recoveryGroupName**
Specifies the recovery group to which the pdisks belong. Required.

**pdisks**
Specifies pdisks as the source of the GET call. Required.

## Request headers

```
Content-Type: application/json
Accept: application/json
```

## Request parameters

The following parameters can be used in the request URL to customize the request:

*Table 17. List of request parameters*

| Parameter name | Description and applicable keywords | Required/optional |
|---|---|---|
| fields | Comma-separated list of fields to be included in response. ':all:' selects all available fields. | Optional. |
| filter | Filter objects by expression. For example, 'status=HEALTHY,entityType=FILESET' | Optional. |
| recoveryGroupName | The name of the recovery group to which the pdisks belong. | Required. |

## Request data

No request data.

## Response data

```
{
   "status": {
      "code":ReturnCode,
      "message":"ReturnMessage"
   },
```

```
    "paging":
    {
    "next": "URL"
    },
   "pdisks": [
      {
          {
            "details":"Pdisk details"
             {
            "replacementPriority":"Replacement priority"
            "fru":"Field replaceable unit",
            "location":"Location",
            "wwn":"World wide name",
            "server":"Server",
            "rgIndex":"Recovery group index"
            "vendor":"Vendor name",
            "product":"Product name",
            "revision":"Revision",
            "serial":"Serial",
            "hardwareType":"Hardware type",
            "speed":"Speed",
      }
          "metrics":"Performance metrics"
            {
            "reads":"Reads"
            "writes":"Writes",
            "bytesRead":"Bytes read",
            "bytesWritten":"Bytes written",
            "ioErrors":"Number of I/O errors",
            "ioTimeOuts":"Number of I/O timeouts"
            "mediaErrors":"Media errors",
            "checksumErrors":"Cehcksum errors",
            "pathErrors":"Path errors",
            "relativePerformance":"Relative performance"
            "dataBadness":"Data badness",
      }
      "paths":"Path details"
             {
            "paths":"List of paths"
            "noOfPathsActive":"Number of active paths",
            "noOfPathsTotal":"Total number of paths",
            "expectedPathsActive":"Expected acive paths",
            "expectedPathsTotal":"Expected total paths",
            "noOfPathsActiveWrong":"Number of active wrong paths"
            "noOfPathsTotalWrong":"Total number of wrong paths",
           }
            "oid":"ID"
            "name":"Name",
            "declusteredArray":"Declustered array",
            "recoveryGroup":"Recovery group",
            "state":"State",
            "stateInfo":"State information"
            "healthState":"Health state",
            "freeSpace":"Free space",
            "capacity":"Capacity"
            "pathCount":"Path count",
  }
 ]
}
```

For more information about the fields in the following data structures, see the links at the end of this topic.

**"status":**
Return status.

**"message": "***ReturnMessage***"**
The return message.

**"code": *ReturnCode***
The return code.

**"paging"**
The URL to retrieve the next page. Paging is enabled when more than 1000 objects are returned by the query.

**"pdisks":"***Physical disk details***"**
An array of information that lists the details of the pdisk.

**"details":"***Pdisk details***"**
Details of pdisk.

**"replacementPriority":"***Replacement priority***"**
Replacement priority that is defined for the pdisk.

**"fru":"***Field replaceable unit***",**
The field replaceable unit (FRU) of the pdisk.

**"location":"***Location***",**
Location of the pdisk in an enclosure.

**"wwn":"***World wide name***",**
The world wide name of the pdisk.

**"server":"***Server***",**
The server name of the ESS.

**"rgIndex":"***Recovery group index***"**
Recovery group index of the pdisk.

**"vendor":"***Vendor name***",**
Vendor name of the pdisk.

**"product":"***Product name***",**
Product name of the pdisk.

**"revision":"***Revision***",**
Revision of the pdisk.

**"serial":"***Serial***",**
The serial of the pdisk.

**"hardwareType":"***Hardware type***",**
Hardware type of the pdisk.

**"speed":"***Speed***"**
Speed of the pdisk.

**"metrics":"***Performance metrics***"**

**"reads":"***Reads***"**
Number of reads from the pdisk.

**"writes":"***Writes***",**
Number of writes to the pdisk.

**"bytesRead":"***Bytes read***",**
Number of bytes read from the pdisk.

**"bytesWritten":"***Bytes written***",**
Number of writes to the pdisk.

**"ioErrors":"***Number of I/O errors***",**
Number of I/O errors reported for the pdisk.

**"ioTimeOuts":"***Number of I/O timeouts***"**
Number of I/O timeouts reported for the pdisk.

**"mediaErrors":"***Media errors***",**
Number of media errors reported for the pdisk.

**"checksumErrors":"***Cehcksum errors***",**
Number of checksum errors reported for the pdisk.

**"pathErrors":"***Path errors***",**
Number of path errors reported for the pdisk.

**"relativePerformance":"***Relative performance***"**
Relative performance of the pdisk.

**"dataBadness":"***Data badness***"**
Data badness of the pdisk.

**"paths":"***Path details***"**

**"paths":"***List of paths***"**
List of paths of the pdisk.

**"noOfPathsActive":"***Number of active paths***",**
The number of active paths for the pdisk

**"noOfPathsTotal":"***Total number of paths***",**
The total number of paths for the pdisk.

**"expectedPathsActive":"***Expected acive paths***",**
The number of expected active paths for the pdisk.

**"expectedPathsTotal":"***Expected total paths***",**
The total number of expected paths for the pdisk.

**"noOfPathsActiveWrong":"***Number of active wrong paths***"**
The number of active wrong paths for the pdisk.

**"noOfPathsTotalWrong":"***Total number of wrong paths***"**
The total number of wrong paths for the pdisk.

**"oid":"***ID***"**
The ID used for paging.

**"name":"***Name***",**
Name of the pdisk.

**"declusteredArray":"***Declustered array***",**
The name of the declustered array of the pdisk.

**"recoveryGroup":"***Recovery group***",**
The name of the recovery group of the pdisk.

**"state":"***State***",**
The state of the pdisk

**"stateInfo":"***State information***"**
The state information about the pdisk.

**"healthState":"***Health state***",**
The health state of the pdisk.

**"freeSpace":"***Free space***",**
The free space of the pdisk in bytes.

**"capacity":"***Capacity***"**
The capacity of the pdisk in bytes.

**"pathCount":"***Path count***"**
The number of paths for the pdisk.

## Examples

The following example gets information about the pdisks that are available in the recovery group *RG1*.

Request data:

```
curl -k -u admin:admin001 -X GET --header 'accept:application/json'
'https://198.51.100.1:443/scalemgmt/v2/gnr/recoverygroups/RG1/pdisks
```

Response data:

**Note:** In the JSON data that is returned, the return code indicates whether the command is successful. The response code 200 indicates that the command successfully retrieved the information. Error code 400 represents an invalid request and 500 represents internal server error.

```
{
  "status": {
    "code": 200,
    "message": "..."
  },
  "paging": {
    "next": "https://localhost:443/scalemgmt/v2/filesystems/gpfs0/filesets?lastId=1001"
  },
  "pdisks": [
    {
      "details": {
        "replacementPriority": 1000,
        "fru": "QEMU HARDDISK",
        "location": "24",
        "wwn": "naa.5000CCA01C514D48",
        "server": "gss-22.localnet.com",
        "rgIndex": 23,
        "vendor": "QEMU",
        "product": "QEMU HARDDISK",
        "revision": "1.5",
        "serial": "48",
        "hardwareType": "ROTATING",
        "speed": 7200
      },
      "metrics": {
        "reads": 3484,
        "writes": 698,
        "bytesRead": 546534588,
        "bytesWritten": 612032839,
        "ioErrors": 0,
        "ioTimeOuts": 0,
        "mediaErrors": 0,
        "checksumErrors": 0,
        "pathErrors": 0,
        "relativePerformance": 0.968,
        "dataBadness": 0
      },
      "paths": {
        "paths": "//gss-22/dev/sdar",
        "noOfPathsActive": 1,
        "noOfPathsTotal": 2,
        "expectedPathsActive": 1,
        "expectedPathsTotal": 2,
        "noOfPathsActiveWrong": 0,
        "noOfPathsTotalWrong": 0
```

```
      },
      "oid": 0,
      "name": "pdisk1",
      "declusteredArray": "DA1",
      "recoveryGroup": "RG1",
      "state": "NORMAL",
      "stateInfo": "ok",
      "healthState": "HEALTHY",
      "freeSpace": 533649686528,
      "capacity": 536602476544,
      "pathCount": 2
    }
  ]
}
```

"mmlspdisk command" on page 199
Lists information for one or more IBM Spectrum Scale RAID pdisks.

# Gnr/recoverygroups/{recoveryGroupName}/pdisks/{pdiskName}: GET

Gets the details of a specific physical disk that is available in a recovery group.

## Availability

Available with the Elastic Storage Server.

## Description

The **GET gnr/recoverygroup/{recoveryGroupName}/pdisks/{pdiskName}** request gets the details of a specific pdisk that is part of a particular recovery group. For more information about the fields in the data structures that are returned, see "mmlspdisk command" on page 199.

## Request URL

`https://`*`IP address of API server`*`:<port>/`**`scalemgmt/v2/gnr/recoverygroups`**`/`*`recoveryGroupName`*`/`**`pdisks`**`/`*`pdiskName`*

where

**`gnr/recoverygroups/recoveryGroupName`**
Specifies a particular recovery group that contains the pdisk. Required.

**`/pdisks/pdiskName`**
Specifies a particular pdisk as the resource of this GET call. Required.

## Request headers

```
Content-Type: application/json
Accept: application/json
```

## Request parameters

The following parameters can be used in the request URL to customize the request:

*Table 18. List of request parameters*

| Parameter name | Description and applicable keywords | Required/optional |
|---|---|---|
| fields | Comma-separated list of fields to be included in response. ':all:' selects all available fields. | Optional. |
| filter | Filter objects by expression. For example, 'status=HEALTHY,entityType=FILESET' | Optional. |
| recoveryGroupName | The name of the recovery group that contains the pdisk. | Required. |
| pdiskName | The name of the pdisk about which you need the details. | Required. |

## Request data

No request data.

## Response data

```
{
   "status": {
      "code":ReturnCode,
      "message":"ReturnMessage"
   },
    "paging":
    {
    "next": "URL"
    },
   "pdisks": [
      {
          {
            "details":"Pdisk details"
             {
            "replacementPriority":"Replacement priority"
            "fru":"Field replaceable unit",
            "location":"Location",
            "wwn":"World wide name",
            "server":"Server",
            "rgIndex":"Recovery group index"
            "vendor":"Vendor name",
            "product":"Product name",
            "revision":"Revision",
            "serial":"Serial",
            "hardwareType":"Hardware type",
            "speed":"Speed",
      }
          "metrics":"Performance metrics"
             {
            "reads":"Reads"
            "writes":"Writes",
            "bytesRead":"Bytes read",
            "bytesWritten":"Bytes written",
            "ioErrors":"Number of I/O errors",
            "ioTimeOuts":"Number of I/O timeouts"
            "mediaErrors":"Media errors",
            "checksumErrors":"Cehcksum errors",
            "pathErrors":"Path errors",
            "relativePerformance":"Relative performance"
            "dataBadness":"Data badness",
      }
      "paths":"Path details"
             {
            "paths":"List of paths"
            "noOfPathsActive":"Number of active paths",
            "noOfPathsTotal":"Total number of paths",
            "expectedPathsActive":"Expected acive paths",
            "expectedPathsTotal":"Expected total paths",
            "noOfPathsActiveWrong":"Number of active wrong paths"
            "noOfPathsTotalWrong":"Total number of wrong paths",
           }
            "oid":"ID"
            "name":"Name",
            "declusteredArray":"Declustered array",
            "recoveryGroup":"Recovery group",
            "state":"State",
            "stateInfo":"State information"
            "healthState":"Health state",
            "freeSpace":"Free space",
            "capacity":"Capacity"
            "pathCount":"Path count",
  }
 ]
}
```

For more information about the fields in the following data structures, see the links at the end of this topic.

**"status":**
Return status.

**"message": "***ReturnMessage***"**
The return message.

**"code":** *ReturnCode*
The return code.

**"paging"**
The URL to retrieve the next page. Paging is enabled when more than 1000 objects are returned by the query.

**"pdisks":"***Physical disk details***"**
Array of information that lists the details of the pdisk.

**"details":"***Pdisk details***"**
Details of pdisk.

**"replacementPriority":"***Replacement priority***"**
Replacement priority defined for the pdisk.

**"fru":"***Field replaceable unit***",**
The field replaceable unit (FRU) of the pdisk.

**"location":"***Location***",**
Location of the pdisk in an enclosure.

**"wwn":"***World wide name***",**
The world wide name of the pdisk.

**"server":"***Server***",**
The server name of the ESS.

**"rgIndex":"***Recovery group index***"**
Recovery group index of the pdisk.

**"vendor":"***Vendor name***",**
Vendor name of the pdisk.

**"product":"***Product name***",**
Product name of the pdisk.

**"revision":"***Revision***",**
Revision of the pdisk.

**"serial":"***Serial***",**
The serial of the pdisk.

**"hardwareType":"***Hardware type***",**
Hardware type of the pdisk.

**"speed":"***Speed***"**
Speed of the pdisk.

**"metrics":"***Performance metrics***"**

**"reads":"***Reads***"**
Number of reads from the pdisk.

**"writes":"***Writes***",**
Number of writes to the pdisk.

**"bytesRead":"***Bytes read***",**
Number of bytes read from the pdisk.

**"bytesWritten":"***Bytes written***",**
Number of writes to the pdisk.

**"ioErrors":"***Number of I/O errors***",**
Number of I/O errors reported for the pdisk.

**"ioTimeOuts":"***Number of I/O timeouts***"**
Number of I/O timeouts reported for the pdisk.

**"mediaErrors":"***Media errors***",**
Number of media errors reported for the pdisk.

**"checksumErrors":"***Cehcksum errors***",**
Number of checksum errors reported for the pdisk.

**"pathErrors":"***Path errors***",**
Number of path errors reported for the pdisk.

**"relativePerformance":"***Relative performance***"**
Relative performance of the pdisk.

**"dataBadness":"***Data badness***"**
Data badness of the pdisk.

**"paths":"***Path details***"**

**"paths":"***List of paths***"**
List of paths of the pdisk.

**"noOfPathsActive":"***Number of active paths***",**
The number of active paths for the pdisk

**"noOfPathsTotal":"***Total number of paths***",**
The total number of paths for the pdisk.

**"expectedPathsActive":"***Expected acive paths***",**
The number of expected active paths for the pdisk.

**"expectedPathsTotal":"***Expected total paths***",**
The total number of expected paths for the pdisk.

**"noOfPathsActiveWrong":"***Number of active wrong paths***"**
The number of active wrong paths for the pdisk.

**"noOfPathsTotalWrong":"***Total number of wrong paths***"**
The total number of wrong paths for the pdisk.

**"oid":"***ID***"**
The ID used for paging.

**"name":"***Name***",**
Name of the pdisk.

**"declusteredArray":"***Declustered array***",**
The name of the declustered array of the pdisk.

**"recoveryGroup":"***Recovery group***",**
The name of the recovery group of the pdisk.

**"state":"***State***",**
The state of the pdisk

**"stateInfo":"***State information***"**
The state information about the pdisk.

**"healthState":"***Health state***",**
The health state of the pdisk.

**"freeSpace":"***Free space***",**
The free space of the pdisk in bytes.

**"capacity":"***Capacity***"**
The capacity of the pdisk in bytes.

**"pathCount":"***Path count***"**
The number of paths for the pdisk.

## Examples

The following example gets information about the pdisk *pdisk1* that is available in the recovery group *RG1*.

Request data:

```
curl -k -u admin:admin001 -X GET --header 'accept:application/json'
'https://198.51.100.1:443/scalemgmt/v2/gnr/recoverygroups/RG1/pdisks/pdisk1
```

Response data:

**Note:** In the JSON data that is returned, the return code indicates whether the command is successful. The response code 200 indicates that the command successfully retrieved the information. Error code 400 represents an invalid request and 500 represents internal server error.

```
{
  "status": {
    "code": 200,
    "message": "..."
  },
  "paging": {
    "next": "https://localhost:443/scalemgmt/v2/filesystems/gpfs0/filesets?lastId=1001"
  },
  "pdisks": [
    {
      "details": {
        "replacementPriority": 1000,
        "fru": "QEMU HARDDISK",
        "location": "24",
        "wwn": "naa.5000CCA01C514D48",
        "server": "gss-22.localnet.com",
        "rgIndex": 23,
        "vendor": "QEMU",
        "product": "QEMU HARDDISK",
        "revision": "1.5",
        "serial": "48",
        "hardwareType": "ROTATING",
        "speed": 7200
      },
      "metrics": {
        "reads": 3484,
        "writes": 698,
        "bytesRead": 546534588,
        "bytesWritten": 612032839,
        "ioErrors": 0,
        "ioTimeOuts": 0,
        "mediaErrors": 0,
        "checksumErrors": 0,
        "pathErrors": 0,
        "relativePerformance": 0.968,
        "dataBadness": 0
      },
```

```
      "paths": {
        "paths": "//gss-22/dev/sdar",
        "noOfPathsActive": 1,
        "noOfPathsTotal": 2,
        "expectedPathsActive": 1,
        "expectedPathsTotal": 2,
        "noOfPathsActiveWrong": 0,
        "noOfPathsTotalWrong": 0
      },
      "oid": 0,
      "name": "pdisk1",
      "usteredArray": "DA1",
      "recoveryGroup": "RG1",
      "state": "NORMAL",
      "stateInfo": "ok",
      "healthState": "HEALTHY",
      "freeSpace": 533649686528,
      "capacity": 536602476544,
      "pathCount": 2
    }
  ]
}
```


"mmlspdisk command" on page 199
Lists information for one or more IBM Spectrum Scale RAID pdisks.

# Gnr/recoverygroups/{recoveryGroupName}/vdisks: GET

Returns the details of the virtual disks that are available in a specified recovery group.

## Availability

Available with the Elastic Storage Server.

## Description

The **GET gnr/recoverygroup/{recoveryGroupName}/vdisks** request gets the details of the vdisks that are part of a particular recovery group. For more information about the fields in the data structures that are returned, see "mmlsvdisk command" on page 213.

## Request URL

```
https://IP address of API server:<port>/scalemgmt/v2/gnr/recoverygroups/recoveryGroupName/vdisks
```

where

**gnr/recoverygroups/recoveryGroupName**
Specifies a particular recovery group that contains the vdisk. Required.

**vdisks**
Specifies the vdisks as the resource of this GET call. Required.

## Request headers

```
Content-Type: application/json
Accept: application/json
```

## Request parameters

The following parameters can be used in the request URL to customize the request:

*Table 19. List of request parameters*

| Parameter name | Description and applicable keywords | Required/optional |
|---|---|---|
| fields | Comma-separated list of fields to be included in response. ':all:' selects all available fields. | Optional. |
| filter | Filter objects by expression. For example, 'status=HEALTHY,entityType=FILESET' | Optional. |
| recoveryGroupName | The name of the recovery group that contains the vdisks. | Required. |

## Request data

No request data.

## Response data

```
{
   "status": {
      "code":ReturnCode,
      "message":"ReturnMessage"
   },
```

```
    "paging":
    {
    "next": "URL"
    },
   "vdisks": [
      {

             "name":"Name",
             "recoveryGroup":"Recovery group",
             "declusteredArray":"Declustered array",
             "size":"Size",
             "raidCode":"RAID code"
             "trackSize":"Track size",
             "checksumGranularity":"Checksum granularity",
             "remarks":"Remarks",
             "state":"State",
             "healthState":"Health status",
             "oid":"ID",
  }
 ]
}
```

For more information about the fields in the following data structures, see the links at the end of this topic.

**"status":**
Return status.

**"message": "***ReturnMessage***"**
The return message.

**"code": "***ReturnCode***"**
The return code.

**"paging"**
The URL to retrieve the next page. Paging is enabled when more than 1000 objects are returned by the query.

**"vdisks":"***Virtual disk details***"**
Array of information that lists the details of the vdisk that is available in the specified recovery group.

**"name":"***Name***"**
Name of the vdisk.

**"recoveryGroup":"***Recovery group***"**
The name of the recovery group of the vdisk.

**"declusteredArray":"***Declustered array***"**
The name of the declustered array of the vdisk.

**"size":"***Size***"**
Size of the vdisk.

**"raidCode":"***RAID code***"**
RAID code of the vdisk.

**"trackSize":"***Track size***"**
Size of the track.

**"checksumGranularity":"***Checksum granularity***"**
The granularity of checksum.

**"remarks":"***Remarks***"**
Remarks about the vdisk.

**"state":"***State***"**
The state of the vdisk.

**"healthState":"***Health status***"**
The health status of the vdisk.

**"oid":"***ID***"**
The ID used for paging.

## Examples

The following example gets information about the vdisks that are available in the recovery group *RG1*.

Request data:

```
curl -k -u admin:admin001 -X GET --header 'accept:application/json'
'https://198.51.100.1:443/scalemgmt/v2/gnr/recoverygroups/RG1/vdisks
```

Response data:

**Note:** In the JSON data that is returned, the return code indicates whether the command is successful. The response code 200 indicates that the command successfully retrieved the information. Error code 400 represents an invalid request and 500 represents internal server error.

```
{
  "status": {
    "code": 200,
    "message": "..."
  },
  "paging": {
    "next": "https://localhost:443/scalemgmt/v2/filesystems/gpfs0/filesets?lastId=1001"
  },
  "vdisks": [
    {
      "name": "vdisk1",
      "recoveryGroup": "RG1",
      "declusteredArray": "DA1",
      "size": 262144,
      "raidCode": "2WayReplication",
      "trackSize": 262144,
      "checksumGranularity": 32768,
      "remarks": "logTip",
      "state": "ok",
      "healthState": "UNKNOWN",
      "oid": 0
    }
  ]
}
```

"mmlsvdisk command" on page 213
Lists information for one or more IBM Spectrum Scale RAID vdisks.

"mmcrvdisk command" on page 162
Creates a vdisk within a declustered array of an IBM Spectrum Scale RAID recovery group.

# Gnr/recoverygroups/{recoveryGroupName}/vdisks/{vdiskName}: GET

Returns the details a virtual disk that is available in a specified recovery group.

## Availability

Available with the Elastic Storage Server.

## Description

The **GET gnr/recoverygroup/{recoveryGroupName}/vdisks/{vdiskName}** request gets the details of a specific vdisk that is part of a particular recovery group. For more information about the fields in the data structures that are returned, see "mmlsvdisk command" on page 213.

## Request URL

`https://`*`IP address of API server`*`:<port>/`**`scalemgmt/v2/gnr/recoverygroups`**`/`*`recoveryGroupName`*`/`**`vdisks`**`/`*`vdiskName`*

where

**`gnr/recoverygroups/recoveryGroupName`**
Specifies a particular recovery group that contains the vdisk. Required.

**`/vdisks/vdiskName`**
Specifies a particular vdisk as the resource of this GET call. Required.

## Request headers

```
Content-Type: application/json
Accept: application/json
```

## Request parameters

The following parameters can be used in the request URL to customize the request:

*Table 20. List of request parameters*

| Parameter name | Description and applicable keywords | Required/optional |
|---|---|---|
| fields | Comma-separated list of fields to be included in response. ':all:' selects all available fields. | Optional. |
| filter | Filter objects by expression. For example, 'status=HEALTHY,entityType=FILESET' | Optional. |
| recoveryGroupName | The name of the recovery group that contains the vdisk. | Required. |
| vdiskName | The name of the vdisk about which you need the details. | Required. |

## Request data

No request data.

## Response data

```
{
   "status": {
      "code":ReturnCode,
      "message":"ReturnMessage"
   },
    "paging":
    {
    "next": "URL"
    },
   "vdisks": [
      {

            "name":"Name",
            "recoveryGroup":"Recovery group",
            "declusteredArray":"Declustered array",
            "size":"Size",
            "raidCode":"RAID code"
            "trackSize":"Track size",
            "checksumGranularity":"Checksum granularity",
            "remarks":"Remarks",
            "state":"State",
            "healthState":"Health status",
            "oid":"ID",
  }
 ]
}
```

For more information about the fields in the following data structures, see the links at the end of this topic.

**"status":**
Return status.

**"message": "***ReturnMessage***"**
The return message.

**"code":** *ReturnCode*
The return code.

**"paging"**
The URL to retrieve the next page. Paging is enabled when more than 1000 objects are returned by the query.

**"vdisks":"***Virtual disk details***"**
Array of information that lists the details of the vdisk that is available in the specified recovery group.

**"name":"***Name***",**
Name of the vdisk.

**"recoveryGroup":"***Recovery group***"**
The name of the recovery group of the vdisk.

**"declusteredArray":"***Declustered array***"**
The name of the declustered array of the vdisk.

**"size":"***Size***"**
Size of the vdisk.

**"raidCode":"***RAID code***"**
RAID code of the vdisk.

**"trackSize":"***Track size***"**
Size of the track.

**"checksumGranularity":"***Checksum granularity***"**
The granularity of checksum.

**"remarks":"***Remarks***"**
Remarks about the vdisk.

**"state":"***State***"**
The state of the vdisk.

**"healthState":"***Health status***"**
The health status of the vdisk.

**"oid":"***ID***"**
The ID used for paging.

## Examples

The following example gets information about the vdisk *vdisk1* that is available in the recovery group *RG1*.

Request data:

```
curl -k -u admin:admin001 -X GET --header 'accept:application/json'
'https://198.51.100.1:443/scalemgmt/v2/gnr/recoverygroups/RG1/vdisks/vdisk1
```

Response data:

**Note:** In the JSON data that is returned, the return code indicates whether the command is successful. The response code 200 indicates that the command successfully retrieved the information. Error code 400® represents an invalid request and 500 represents internal server error.

```
{
  "status": {
    "code": 200,
    "message": "..."
  },
  "paging": {
    "next": "https://localhost:443/scalemgmt/v2/filesystems/gpfs0/filesets?lastId=1001"
  },
  "vdisks": [
    {
      "name": "vdisk1",
      "recoveryGroup": "RG1",
      "declusteredArray": "DA1",
      "size": 262144,
      "raidCode": "2WayReplication",
      "trackSize": 262144,
      "checksumGranularity": 32768,
      "remarks": "logTip",
      "state": "ok",
      "healthState": "UNKNOWN",
      "oid": 0
    }
  ]
}
```

"mmlsvdisk command" on page 213
Lists information for one or more IBM Spectrum Scale RAID vdisks.

"mmcrvdisk command" on page 162
Creates a vdisk within a declustered array of an IBM Spectrum Scale RAID recovery group.

# Appendix A. Best-practice recommendations for IBM Spectrum Scale RAID

This topic includes some best-practice recommendations for using IBM Spectrum Scale RAID.

Planning a IBM Spectrum Scale RAID implementation requires consideration of the nature of the JBOD arrays being used, the required redundancy protection and usable disk capacity, the required spare capacity and maintenance strategy, and the ultimate GPFS file system configuration.

- Assign a primary and backup server to each recovery group.

  Each JBOD array should be connected to two servers to protect against server failure. Each server should also have two independent paths to each physical disk to protect against path failure and provide higher throughput to the individual disks.

  Define multiple recovery groups on a JBOD array, if the architecture suggests it, and use mutually reinforcing primary and backup servers to spread the processing evenly across the servers and the JBOD array.

  Recovery group server nodes can be designated GPFS quorum or manager nodes, but they should otherwise be dedicated to IBM Spectrum Scale RAID and not run application workload.

- Configure recovery group servers with a large vdisk track cache and a large page pool.

  The **nsdRAIDTracks** configuration parameter tells IBM Spectrum Scale RAID how many vdisk track descriptors, not including the actual track data, to cache in memory.

  In general, a large number of vdisk track descriptors should be cached. The **nsdRAIDTracks** value for the recovery group servers should be on the order of 100000, or even more if server memory exceeds 128 GiB. If the expected vdisk NSD access pattern is random across all defined vdisks and within individual vdisks, a larger value for **nsdRAIDTracks** might be warranted. If the expected access pattern is sequential, a smaller value can be sufficient.

  The amount of actual vdisk data (including user data, parity, and checksums) that can be cached depends on the size of the GPFS page pool on the recovery group servers and the percentage of page pool reserved for IBM Spectrum Scale RAID. The **nsdRAIDBufferPoolSizePct** parameter specifies what percentage of the page pool should be used for vdisk data. The default is 80%, but it can be set as high as 90% or as low as 10%. Because a recovery group server is also an NSD server and the vdisk buffer pool also acts as the NSD buffer pool, the configuration parameter **nsdBufSpace** should be reduced to its minimum value of 10%.

  As an example, to have a recovery group server cache 20000 vdisk track descriptors (**nsdRAIDTracks**), where the data size of each track is 4 MiB, using 80% (**nsdRAIDBufferPoolSizePct**) of the page pool, an approximate page pool size of 20000 * 4 MiB * (100/80) ≈ 100000 MiB ≈ 98 GiB would be required. It is not necessary to configure the page pool to cache all the data for every cached vdisk track descriptor, but this example calculation can provide some guidance in determining appropriate values for **nsdRAIDTracks** and **nsdRAIDBufferPoolSizePct**.

- Define each recovery group with at least one large declustered array.

  A large declustered array contains enough pdisks to store the required redundancy of IBM Spectrum Scale RAID vdisk configuration data. This is defined as at least nine pdisks plus the effective spare capacity. A minimum spare capacity equivalent to two pdisks is strongly recommended in each large declustered array. The code width of the vdisks must also be considered. The effective number of non-spare pdisks must be at least as great as the largest vdisk code width. A declustered array with two effective spares where 11 is the largest code width (8 + 3p Reed-Solomon vdisks) must contain at least 13 pdisks. A declustered array with two effective spares in which 10 is the largest code width (8 + 2p Reed-Solomon vdisks) must contain at least 12 pdisks.

- Define the log vdisks based on the type of configuration.

See "Typical configurations" under “Log vdisks” on page 42 and Chapter 8, “Setting up IBM Spectrum Scale RAID on the Elastic Storage Server,” on page 87 for log vdisk considerations.

- Determine the declustered array maintenance strategy.

  Disks will fail and need replacement, so a general strategy of deferred maintenance can be used. For example, failed pdisks in a declustered array are only replaced when the spare capacity of the declustered array is exhausted. This is implemented with the replacement threshold for the declustered array set equal to the effective spare capacity. This strategy is useful in installations with a large number of recovery groups where disk replacement might be scheduled on a weekly basis. Smaller installations can have IBM Spectrum Scale RAID require disk replacement as disks fail, which means the declustered array replacement threshold can be set to 1.

- Choose the vdisk RAID codes based on GPFS file system usage.

  The choice of vdisk RAID codes depends on the level of redundancy protection required versus the amount of actual space required for user data, and the ultimate intended use of the vdisk NSDs in a GPFS file system.

  Reed-Solomon vdisks are more space efficient. An 8 + 3p vdisk uses approximately 27% of actual disk space for redundancy protection and 73% for user data. An 8 + 2p vdisk uses 20% for redundancy and 80% for user data. Reed-Solomon vdisks perform best when writing whole tracks (the GPFS block size) at once. When partial tracks of a Reed-Solomon vdisk are written, parity recalculation must occur.

  Replicated vdisks are less space efficient. A vdisk with 3-way replication uses approximately 67% of actual disk space for redundancy protection and 33% for user data. A vdisk with 4-way replication uses 75% of actual disk space for redundancy and 25% for user data. The advantage of vdisks with *N*-way replication is that small or partial write operations can complete faster.

  For file system applications where write performance must be optimized, the preceding considerations make replicated vdisks most suitable for use as GPFS file system **metadataOnly** NSDs, and Reed-Solomon vdisks most suitable for use as GPFS file system **dataOnly** NSDs. The volume of GPFS file system metadata is usually small (1% - 3%) relative to file system data, so the impact of the space inefficiency of a replicated RAID code is minimized. The file system metadata is typically written in small chunks, which takes advantage of the faster small and partial write operations of the replicated RAID code. Applications are often tuned to write file system user data in whole multiples of the file system block size, which works to the strengths of the Reed-Solomon RAID codes both in terms of space efficiency and speed.

  When segregating vdisk NSDs for file system **metadataOnly** and **dataOnly** disk usage, the **metadataOnly** replicated vdisks can be created with a smaller block size and assigned to the GPFS file system storage pool. The **dataOnly** Reed-Solomon vdisks can be created with a larger block size and assigned to GPFS file system data storage pools. When using multiple storage pools, a GPFS placement policy must be installed to direct file system data to non-system storage pools.

  When write performance optimization is not important, it is acceptable to use Reed-Solomon vdisks as **dataAndMetadata** NSDs for better space efficiency.

- When assigning the failure groups to vdisk NSDs in a GPFS file system, the ESS building block should be considered the common point of failure. All vdisks within all recovery groups in a given ESS building block should be assigned the same failure group number. An exception to this is when the cluster consists of only one ESS building block. In this case, failure groups should be associated with recovery groups rather than with the entire ESS building block.

  Within a recovery group, all file system vdisk NSDs should be assigned the same failure group. If there is more than one ESS building block, all file system vdisk NSDs within both recovery groups of a building block should be assigned the same failure group.

- Attaching storage that is not associated with IBM Spectrum Scale RAID (SAN-attached disks, for example) to ESS NSD server nodes is *not* supported.

- Because of possible differences in performance and availability characteristics, mixing IBM Spectrum Scale RAID vdisks with disks that are not associated with IBM Spectrum Scale RAID in the same storage pool in a file system is *not* recommended. This recommendation applies when mixing different types of storage in the same storage pool. A good IBM Spectrum Scale planning practice is to put storage with similar characteristics in the same storage pool.

# Appendix B. IBM Spectrum Scale RAID commands

Descriptions of these IBM Spectrum Scale RAID commands follow:



For information about other IBM Spectrum Scale commands, see the *IBM Spectrum Scale: Command and Programming Reference*.

# mmaddcomp command

Adds one or more storage components.

## Synopsis

```
mmaddcomp PartNumber
            [--serial-number SerialNumber] [--name Name]
            [--display-id DisplayId] [--replace] [--dry-run]
```

or

```
mmaddcomp -F StanzaFile [--replace] [--dry-run]
```

## Availability

Available with the Elastic Storage Server.

## Description

The **mmaddcomp** command adds one or more new storage components. A default name is assigned if one is not specified.

## Parameters

*PartNumber*
Specifies the part number or model type of the component to be added.

**--serial-number** *SerialNumber*
Specifies a serial number to be assigned to the component. If this serial number matches that of an existing component, the command will fail unless **--replace** is also specified.

**--name** *Name*
Specifies a name to be assigned to the component. A default name is assigned if one is not specified.

**--display-id** *DisplayID*
Specifies a display ID to be assigned to the component. This applies to storage enclosures only.

**-F** *StanzaFile*
Specifies the name of a stanza file from which input values are to be read. The stanza file can include stanzas for multiple components.

The correct format for the component definitions in the stanza file is as follows:

```
%comp:
 partNumber=PartNumber
 serialNumber=SerialNumber
 name=Name
 displayId=DisplayId
```

where:

**partNumber=***PartNumber*
Specifies the part number or model type of the component to be added. This is a required parameter.

**serialNumber=***SerialNumber*
Specifies a serial number to be assigned to the component. If this serial number matches that of an existing component, the command will fail unless **--replace** is also specified.

**name=***Name*
Specifies a name to be assigned to the component. A default name is assigned if one is not specified.

**displayId=***DisplayId*
Specifies a display ID to be assigned to the component. This applies to storage enclosures only.

**--replace**
Indicates that the command is to replace an existing component that has a matching serial number or component ID.

**--dry-run**
Indicates that the changes resulting from the command are not to be recorded in the configuration file.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmaddcomp** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

To define a rack and see the result, enter the following commands:

```
mmaddcomp 1410HPA --name R01C01
mmlscomp
```

The system displays output similar to this:

```
    Rack Components

Comp ID  Part Number  Serial Number  Name
-------  -----------  -------------  ------
      9  1410HPA                     R01C01

    Storage Enclosure Components

Comp ID  Part Number  Serial Number  Name                Display ID
-------  -----------  -------------  ------------------  ----------
      1  1818-80E     SV12345001     1818-80E-SV12345001
      2  1818-80E     SV12345007     1818-80E-SV12345007
      3  1818-80E     SV12345003     1818-80E-SV12345003
      4  1818-80E     SV12345005     1818-80E-SV12345005
      5  1818-80E     SV12345004     1818-80E-SV12345004
      6  1818-80E     SV12345002     1818-80E-SV12345002
      7  1818-80E     SV12345008     1818-80E-SV12345008
      8  1818-80E     SV12345006     1818-80E-SV12345006
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddcompspec command" on page 133
- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144
- "mmchenclosure command" on page 146

- "mmchfirmware command" on page 149
- "mmdelcomp command" on page 167
- "mmdelcomploc command" on page 169
- "mmdelcompspec command" on page 171
- "mmdiscovercomp command" on page 179
- "mmlscomp command" on page 184
- "mmlscomploc command" on page 187
- "mmlscompspec command" on page 190
- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196
- "mmsyncdisplayid command" on page 216
- Chapter 5, "Configuring components on the Elastic Storage Server," on page 47.

## Location

`/usr/lpp/mmfs/bin`

# mmaddcompspec command

Adds one or more new storage component specifications.

## Synopsis

```
mmaddcompspec PartNumber CompType
              [--height Height] [--pci-slots PciSlots] [--drive-slots DriveSlots]
              [--vendor-id VendorId] [--product-id ProductId]
              [--has-display-id { yes | no } ] [--description Description]
              [--replace] [--dry-run]
```

or

```
mmaddcompspec -F StanzaFile [--ignore-unknown] [--replace] [--dry-run]
```

or

```
mmaddcompspec default [--replace] [--dry-run]
```

## Availability

Available with the Elastic Storage Server.

## Description

The **mmaddcompspec** command adds one or more new storage component specifications. A default name is assigned if one is not specified.

## Parameters

*PartNumber*
Specifies the part number or model type of the component to be added.

*CompType*
Specifies the component type. Valid values are: **rack**, **server**, **storageEnclosure**, and **storageServer**.

**--height** *Height*
Specifies the component height in rack units (U).

**--pci-slots** *PciSlots*
Specifies the number of PCI slots. This parameter applies to servers only.

**--drive-slots** *DriveSlots*
Specifies the number of drive slots. This parameter applies to storage enclosures only.

**--vendor-id** *VendorId*
Specifies the vendor ID reported by SCSI inquiry. This parameter applies to storage enclosures only.

**--product-id** *ProductId*
Specifies the product ID reported by SCSI inquiry. This parameter applies to storage enclosures only.

**--has-display-id yes | no**
Indicates whether the component has a programmable display ID. This parameter applies to storage enclosures only.

**--description** *Description*
Specifies a brief description of the component.

**--replace**
Replaces an existing specification with a matching part number.

**--dry-run**
Runs the command without making any permanent changes.

**-F** *StanzaFile*
Specifies the name of a stanza file from which input values are to be read. The stanza file can include stanzas for multiple components.

The correct format for the component definitions in the stanza file is as follows:

```
%compSpec:
 partNumber=PartNumber
 compType=CompType
 height=Height
 description=Description
 vendorId=VendorId
 productId=ProductId
 driveSlots=DriveSlots
 hasDisplayId=HasDisplayId
```

**--ignore-unknown**
Ignores unknown attributes in compSpec stanzas.

**default**
Add all of the predefined component specifications.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmaddcompspec** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

To define a generic 60U rack and see the result, enter the following commands:

```
mmaddcompspec RACK60U rack --height 60 --description "Generic 60u rack"

mmlscompspec --type rack
```

The system displays output similar to this:

```
    Rack Specifications

Part Number  Height  Description
-----------  ------  -------------------------------------------------------
1410HEA          42  IBM Intelligent Cluster 42U 1200mm Deep Expansion Rack
1410HPA          42  IBM Intelligent Cluster 42U 1200mm Deep Primary Rack
9308RC4          42  IBM 42U Enterprise Rack
RACK42U          42  Generic 42U rack
RACK60U          60  Generic 60u rack
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144

- “mmdelcomp command” on page 167
- “mmdelcomploc command” on page 169
- “mmdelcompspec command” on page 171
- “mmdiscovercomp command” on page 179
- “mmlscomp command” on page 184
- “mmlscomploc command” on page 187
- “mmlscompspec command” on page 190
- “mmsyncdisplayid command” on page 216
- Chapter 5, “Configuring components on the Elastic Storage Server,” on page 47.

## Location

`/usr/lpp/mmfs/bin`

# mmaddpdisk command

Adds a pdisk to an IBM Spectrum Scale RAID recovery group.

## Synopsis

**mmaddpdisk** *RecoveryGroupName* **-F** *StanzaFile* [**--replace**] [**-v** {**yes** | **no**}]

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmaddpdisk** command adds a pdisk to a recovery group.

**Note:** The GPFS daemon must be running on both the primary and backup servers to run this command.

## Parameters

*RecoveryGroupName*
Specifies the recovery group to which the pdisks are being added.

**-F** *StanzaFile*
Specifies a file containing pdisk stanzas that identify the pdisks to be added. For more information about pdisk stanzas, see the the following *IBM Spectrum Scale RAID: Administration* topic: "Pdisk stanza format" on page 36.

**--replace**
Indicates that any existing pdisk that has the same name as a pdisk listed in the stanza file is to be replaced. (This is an atomic deletion and addition.)

**-v {yes | no}**
Verifies that specified pdisks do not belong to an existing recovery group. The default is **-v yes**.

Specify **-v no** only when you are certain that the specified disk does not belong to an existing recovery group. Using **-v no** on a disk that already belongs to a recovery group will corrupt that recovery group.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmaddpdisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

In this example, suppose the input stanza file, `pdisk.c033d2`, contains the following lines:

```
%pdisk: pdiskName=c033d2
        device=/dev/hdisk674
        da=DA2
        nPathActive=2
        nPathTotal=4
```

This command example shows how to add the pdisk described in stanza file `pdisk.c033d2` to recovery group `000DE37BOT`:

```
mmaddpdisk 000DE37BOT -F pdisk.c033d2
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmchpdisk command" on page 153
- "mmdelpdisk command" on page 173
- "mmlspdisk command" on page 199
- "mmlsrecoverygroup command" on page 202.

## Location

`/usr/lpp/mmfs/bin`

# mmchcarrier command

Allows IBM Spectrum Scale RAID pdisks to be physically removed and replaced.

## Synopsis

```
mmchcarrier RecoveryGroupName --release
               {[--pdisk "Pdisk[;Pdisk]" [--location "Location[;Location]"]}
               [--force-release] [--force-rg]
```

or

```
mmchcarrier RecoveryGroupName --resume
               {[--pdisk "Pdisk[;Pdisk]" [--location "Location[;Location]"]}
               [--force-rg]
```

or

```
mmchcarrier RecoveryGroupName --replace
               {[--pdisk "Pdisk[;Pdisk]" [--location "Location[;Location]"]}
               [-v {yes|no}] [--force-fru] [--force-rg] [--nsd-version {1|2}]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmchcarrier** command is used to control disk carriers and replace failed pdisks.

Replacing a pdisk requires the following three steps:

1. Run the **mmchcarrier --release** command to prepare the carrier for removal.

   The **mmchcarrier --release** command suspends I/O to all disks in the carrier, turns off power to the disks, illuminates identify lights on the carrier, and unlocks the carrier latch (if applicable).
2. Remove the carrier from the disk drawer, replace the failed disk or disks with new disks, and reinsert the carrier into the disk drawer.
3. Run the **mmchcarrier --replace** command to complete the replacement.

   The **mmchcarrier --replace** command powers on the disks, verifies that the new disks have been installed, resumes I/O, and begins the rebuilding and rebalancing process onto the new disks.

   **Note:** New disks will take the name of the replaced pdisks. In the event that replaced pdisks have not completely drained, they will be given a temporary name consisting of the old pdisk name with a suffix of the form **#***nnnn*. The temporary pdisk will have the `adminDrain` pdisk state flag set and will be deleted once drained. For example, a pdisk named `p25` will receive a temporary name similar to `p25#0010` when the `adminDrain` state flag is set. This allows the new disk that is replacing it to be named `p25` immediately rather than waiting for the old disk to be completely drained and deleted. Until the draining and deleting process completes, both the new pdisk `p25` and the old pdisk `p25#0010` will show up in the output of the **mmlsrecoverygroup** and **mmlspdisk** commands.

Both the release and replace commands require either a recovery group name and a location code, or a recovery group name and a pdisk name to identify the carrier and particular disk slot within the carrier. It is acceptable to provide more than one location code or pdisk name to replace multiple disks within the same carrier.

The **mmchcarrier --resume** command reverses the effect of the release command without doing disk replacements. It can be used to cancel the disk replacement procedure after running the **mmchcarrier --release** command.

## Parameters

*RecoveryGroupName*
Specifies the name of the recovery group to which the carrier belongs. This is used to identify the active server where the low level commands will be issued.

**--release**
Suspends all disks in the carrier, activates identify lights, and unlocks the carrier.

**--resume**
Resumes all disks in the carrier without doing disk replacements.

**--replace**
Formats the replacement disks for use and resumes all disks in the carrier.

**--pdisk**
Specifies the target pdisk or pdisks and identifies the carrier. All specified pdisks must belong to the same carrier.

**--location**
Specifies the target pdisk or pdisks and identifies the carrier by location code. All specified pdisks must belong to the same carrier. If this option is used, the **location** code must be obtained from the output of the **mmlspdisk** command. There is a field **location** listed for each pdisk.

**--force-release**
This is a force flag for the **--release** option, to release the carrier even if the target is not marked for replacement. Disks marked for replacement are identified via the **mmlspdisk --replace** command.

**--force-fru**
This is a force flag for the **--replace** option, to allow the replacement even if the field replaceable unit (FRU) number of the new disk does not match that of the old disk.

**--force-rg**
This is a force flag for the **--release**, **--resume**, and **--replace** options to allow actions on the carrier even if all the pdisks do not belong to the same recovery group.

**--nsd-version**
Specifies the desired Nsd version for the replacement disks. The value can be either 1 or 2. This parameter is only effective with recovery group version 4.2.0.1 or up. If the Nsd version for the disks marked for replacement is known, this parameter will be ignored. If the Nsd version for the disk marked for replacement is not known, and if this parameter is not specified, the pdisk Nsd version will be 2 for recovery group version 4.2.0.1 or up. For recovery group version 4.1.0.1 or lower, the Nsd version can only be 1.

**-v {yes | no}**
Verification flag for the **--replace** option; indicates whether or not to verify that the new disk does not already have a valid pdisk descriptor. The default is **-v yes**.

Specify **-v no** to allow a disk that was formerly part of some other recovery group to be reused.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmchcarrier** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For

additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. The following command example shows how to release the carrier containing failed pdisk `c014d3` in recovery group `000DE37BOT`:

   ```
   mmchcarrier 000DE37BOT --release --pdisk c014d3
   ```

   The system displays output similar to the following:

   ```
   [I] Suspending pdisk c014d1 of RG 000DE37TOP in location 78AD.001.000DE37-C14-D1.
   [I] Suspending pdisk c014d2 of RG 000DE37TOP in location 78AD.001.000DE37-C14-D2.
   [I] Suspending pdisk c014d3 of RG 000DE37TOP in location 78AD.001.000DE37-C14-D3.
   [I] Suspending pdisk c014d4 of RG 000DE37TOP in location 78AD.001.000DE37-C14-D4.
   [I] Carrier released.

     - Remove carrier.
     - Replace disk in location 78AD.001.000DE37-C14-D3 with FRU 74Y4936.
     - Reinsert carrier.
     - Issue the following command:

         mmchcarrier 000DE37TOP --replace --pdisk 'c014d3'
   ```

2. The following command example shows how to tell IBM Spectrum Scale that the carrier containing pdisk `c014d3` in recovery group `000DE37BOT` has been reinserted and is ready to be brought back online:

   ```
   mmchcarrier 000DE37BOT --replace --pdisk c014d3
   ```

   The system displays output similar to the following:

   ```
   [I] The following pdisks will be formatted on node server1:
       /dev/rhdisk354
   [I] Pdisk c014d3 of RG 000DE37TOP successfully replaced.
   [I] Resuming pdisk c014d1 of RG 000DE37TOP.
   [I] Resuming pdisk c014d2 of RG 000DE37TOP.
   [I] Resuming pdisk c014d3#162 of RG 000DE37TOP.
   [I] Resuming pdisk c014d4 of RG 000DE37TOP.
   [I] Carrier resumed.
   ```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmchpdisk command" on page 153
- "mmchrecoverygroup command" on page 156.
- "mmlspdisk command" on page 199

## Location

`/usr/lpp/mmfs/bin`

# mmchcomp command

Changes attributes associated with one or more storage components.

## Synopsis

```
mmchcomp Component
           [--part-number PartNumber] [--serial-number SerialNumber]
           [--display-id DisplayId] [--name Name] [--dry-run]
```

or

```
mmchcomp -F StanzaFile [--dry-run]
```

## Availability

Available with the Elastic Storage Server.

## Description

The **mmchcomp** command changes the various attributes associated with one or more components.

## Parameters

*Component*
: Specifies the current name, serial number, or ID of a single component for which one or more attributes are to be changed.

**--part-number** *PartNumber*
: Specifies a new part number to be assigned to the component. You cannot specify a part number that would change the component type.

**--serial-number** *SerialNumber*
: Specifies a new serial number to be assigned to the component.

**--display-id** *DisplayID*
: Specifies a new display ID to be assigned to the component. This applies to storage enclosures only. After setting this attribute, use **mmsyncdisplayid** to update the storage enclosure hardware.

**--name** *Name*
: Specifies a new name to be assigned to the component.

**-F** *StanzaFile*
: Specifies the name of a stanza file from which input values are to be read. The stanza file can include stanzas for multiple components.

  The correct format for the component definitions in the stanza file is as follows:

```
%comp:
 compID=Component
 partNumber=PartNumber
 serialNumber=SerialNumber
 name=Name
 displayId=DisplayId
 nodeNumber=NodeNumber
```

  where:

  **compID=***Component*
  : Specifies the current name, serial number, or ID of a single component for which one or more attributes are to be changed. This is a required parameter.

  **partNumber=***PartNumber*
  : Specifies a new part number to be assigned to the component. You cannot specify a part number that would change the component type.

**serialNumber=***SerialNumber*
Specifies a new serial number to be assigned to the component.

**name=***Name*
Specifies a name to be assigned to the component. A default name is assigned if one is not specified.

**displayId=***DisplayId*
Specifies a new display ID to be assigned to the component. This applies to storage enclosures only. After setting this attribute, use **mmsyncdisplayid** to update the storage enclosure hardware.

**--dry-run**
Indicates that the changes resulting from the command are not to be recorded in the configuration file.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmchcomp** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

To update the name of an enclosure (in this case, the third enclosure) so that the name includes the product serial number, and to see the result, enter the following commands:

```
mmchcomp 3 --name G1E3-SX98760044
mmlscomp --type storageenclosure
```

The system displays output similar to this:

```
    Storage Enclosure Components

Comp ID  Part Number  Serial Number  Name                Display ID
-------  -----------  -------------  ------------------  ----------
      1  1818-80E     SV12345001     1818-80E-SV12345001  21
      2  1818-80E     SV12345007     1818-80E-SV12345007  01
      3  1818-80E     SV12345003     G1E3-SX98760044     13
      4  1818-80E     SV12345005     1818-80E-SV12345005  05
      5  1818-80E     SV12345004     1818-80E-SV12345004  37
      6  1818-80E     SV12345002     1818-80E-SV12345002  25
      7  1818-80E     SV12345008     1818-80E-SV12345008  17
      8  1818-80E     SV12345006     1818-80E-SV12345006  33
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddcomp command" on page 130
- "mmaddcompspec command" on page 133
- "mmchcomploc command" on page 144
- "mmchenclosure command" on page 146

- “mmchfirmware command” on page 149
- “mmdelcomp command” on page 167
- “mmdelcomploc command” on page 169
- “mmdelcompspec command” on page 171
- “mmdiscovercomp command” on page 179
- “mmlscomp command” on page 184
- “mmlscomploc command” on page 187
- “mmlscompspec command” on page 190
- “mmlsenclosure command” on page 192
- “mmlsfirmware command” on page 196
- “mmsyncdisplayid command” on page 216
- Chapter 5, “Configuring components on the Elastic Storage Server,” on page 47.

## Location

`/usr/lpp/mmfs/bin`

# mmchcomploc command

Changes the location of one or more storage components.

## Synopsis

**mmchcomploc** *Component Container Position* [**--dry-run**]

or

**mmchcomploc -F** *StanzaFile* [**--dry-run**]

## Availability

Available with the Elastic Storage Server.

## Description

The **mmchcomploc** command changes the location (container and position within the container) of one or more storage components.

## Parameters

*Component*
: Specifies the current name, serial number, or component ID of a single component for which the location is to be changed.

*Container*
: Specifies the name, serial number, or component ID of the container to be assigned to the component.

*Position*
: Specifies an integer value that identifies the position within the container to be assigned to the component.

**-F** *StanzaFile*
: Specifies the name of a stanza file from which input values are to be read. The stanza file can include stanzas for multiple components.

  The correct format for the component definitions in the stanza file is as follows:

```
%comp:
 compId=Component
 containerId=Container
 position=Position
```

  where:

  **compId=***Component*
  : Specifies the current name, serial number, or component ID of a single component for which the location is to be changed. This is a required parameter.

  **containerId=***Container*
  : Specifies the name, serial number, or component ID of the container to be assigned to the component. This is a required parameter.

  **position=***Position*
  : Specifies an integer value that identifies the position within the container to be assigned to the component. This is a required parameter.

**--dry-run**
: Indicates that the changes that result from the command are not to be recorded in the configuration file.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmchcomploc** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

To place a component in a rack and see the result, enter the following commands:

```
mmchcomploc SV12345003 R01C01 13
mmlscomploc
```

The system displays output similar to this:

```
Component           Location
------------------  ------------------
1818-80E-SV12345003 Rack R01C01 U13-16
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddcomp command" on page 130
- "mmaddcompspec command" on page 133
- "mmchcomp command" on page 141
- "mmchenclosure command" on page 146
- "mmchfirmware command" on page 149
- "mmdelcomp command" on page 167
- "mmdelcomploc command" on page 169
- "mmdelcompspec command" on page 171
- "mmdiscovercomp command" on page 179
- "mmlscomp command" on page 184
- "mmlscomploc command" on page 187
- "mmlscompspec command" on page 190
- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196
- "mmsyncdisplayid command" on page 216
- Chapter 5, "Configuring components on the Elastic Storage Server," on page 47.

## Location

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

# mmchenclosure command

A service aid to be run before replacing disk enclosure components. Identifies whether it is safe to complete the repair action or whether IBM Spectrum Scale needs to be shut down first.

## Synopsis

```
mmchenclosure SerialNumber
              --component { currentSensor | dcm | drawer | enclosure | esm | expander | fan |
                            powerSupply | sideplane | tempSensor | voltageSensor }
              --component-id Id
```

## Availability

Available with the Elastic Storage Server.

## Description

Use the **mmchenclosure** command to check whether it is safe to replace a component while the GPFS daemon is active. The command will indicate whether there are issues that would require the GPFS daemon to be stopped before proceeding.

In the case of failures, the field replaceable units are:

- Drives.
- Enclosure drawers. Any failure that is related to a drawer control module (DCM)'s component ID (component type: **dcm**) requires a drawer replacement.
- Environmental service modules (ESMs).
- Expanders.
- Fan assemblies.
- Power supplies.
- Sideplanes.

## Parameters

**--serial-number** *SerialNumber*
Specifies the serial number of the storage enclosure (as identified in the output of the **mmlsenclosure** command).

**--component { currentSensor | dcm | drawer | enclosure | esm | expander | fan | powerSupply | sideplane | tempSensor | voltageSensor }**
Specifies the type of component that needs to be changed. The component types follow:

**currentSensor**
Is a current sensor.

**dcm**
Is a drawer control module.

**drawer**
Is a drawer.

**enclosure**
Is a storage enclosure.

**esm**
Is an environmental service module.

**expander**
Is a sub-expander.

**fan**
Is a cooling fan.

**powerSupply**
Is a power supply.

**sideplane**
Is an expander board.

**tempSensor**
Is a temperature sensor.

**voltageSensor**
Is a voltage sensor.

**--component-id** *Id*
Specifies the component ID (as identified in the output of the **mmlsenclosure** command).

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmchenclosure** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. Before replacing a fan with a component ID of 2_BOT_RIGHT on enclosure SV13306129, run:

   ```
   mmchenclosure SV13306129 --component fan --component-id 2_BOT_RIGHT
   ```

   In this example, **mmlsenclosure** did not show the fan as needing to be replaced. The system displays information similar to this:

   ```
   mmchenclosure: [E] Storage enclosure SV13306129 component fan component id 2_BOT_RGHT is not failed.
    Service is not required.
   Storage enclosure SV13306129 component fan component id 2_BOT_RGHT is not failed. Service is not required.
    mmchenclosure: Command failed.  Examine previous error messages to determine cause.
   ```

2. Before replacing a fan with a component ID of 1_BOT_LEFT on enclosure SV13306129, run:

   ```
   mmchenclosure SV13306129 --component fan --component-id 1_BOT_LEFT
   ```

   The system displays output similar to this:

   ```
   mmchenclosure: Proceed with the replace operation.
   ```

3. Before replacing the drawer associated with DCM component ID DCM_0A on enclosure SV13306129, run:

   ```
   mmchenclosure SV13306129 --component dcm --component-id DCM_0A
   ```

   The system displays output similar to this:

   ```
   mmchenclosure: Shutdown GPFS on the following nodes and then proceed
   with the replace operation.
   Shutdown GPFS on the following nodes and then proceed with the replace
   operation.
   ```

4. Before replacing an ESM with a component ID of ESM_A on enclosure SV13306129, run:

   ```
   mmchenclosure SV13306129 --component esm --component-id ESM_A
   ```

   The system displays output similar to this:

```
mmchenclosure: Enclosure SV13306129 ESM A is currently redundant.
mmchenclosure: All in-use disks on enclosure SV13306129 ESM A have redundant paths.
mmchenclosure: Proceed with the replace operation.
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topic:

- "mmlsenclosure command" on page 192.

See also:

- The `/usr/lpp/mmfs/samples/vdisk/chdrawer` sample script.

## Location

`/usr/lpp/mmfs/bin`

# mmchfirmware command

Changes the firmware levels on one or more storage components.

## Synopsis

```
mmchfirmware --type {storage-enclosure | drive | host-adapter}
             [ --update-id PREVIOUS]
             [ --serial-number SerialNumber ]
             [ -N { Node[,Node...] | NodeFile | NodeClass } ]
             [--stop-on-error { yes | no } ]
             [ --fast-offline ]
             [ --dry-run ]
             [ -v {yes | no}
```

## Availability

Available with the Elastic Storage Server.

## Description

Use the **mmchfirmware** command to update the firmware level on one or more storage components.

A particular component is updated only if it satisfies the **--update-id**, **--type**, **--serial-number**, and **-N** options when they are specified.

Consequently, when you update drives on a cluster that has the IBM Spectrum Scale daemons active, you need to use the **-N** option to include both servers of the ESS building blocks that might be affected.

**Note:** The firmware on each of the two sides of an enclosure can be upgraded independently. During a serial update procedure, the two sides of an enclosure typically operate with mismatched firmware levels for a short period (a few minutes), during which time the enclosure presents a warning because of the firmware mismatch.

## Parameters

**--type {storage-enclosure | drive | host-adapter}**
Specifies the type of component for which firmware is to be updated.

**--update-id PREVIOUS**
Only the keyword **PREVIOUS** is supported. The default is to install the newest available firmware.

The keyword **PREVIOUS** assumes that you already loaded the latest firmware available and now want to back off to the previous level.

**--serial-number** *SerialNumber*
Specifies the serial number of a particular component to be updated.

**-N {***Node***[,***Node***...] |** *NodeFile* **|** *NodeClass***}**
Specifies the nodes to be included in the update. This command supports all defined node classes. Only components directly accessible from one of the specified nodes are included in the component selection list. The default is to run on the invoking node.

For general information about how to specify node names, see the following *IBM Spectrum Scale RAID: Administration* topic: "Specifying nodes as input to IBM Spectrum Scale RAID commands" on page 12.

**--stop-on-error {yes | no}**
Specifies whether the command is to stop when a firmware update fails. The default value is **yes**.

**--fast-offline**
This option is deprecated. It was intended to increase parallelism when updating drive firmware if IBM Spectrum Scale was shut down on all the nodes in the cluster. Parallelism automatically happens for the following cases:

- **host-adapter**: Runs in parallel on all of the nodes that are specified unless debugging is enabled.

  **Note:** IBM Spectrum Scale must be shut down on all of the specified nodes.
- **drive**: Parallelism automatically happens if IBM Spectrum Scale is shut down on all of the potentially affected nodes.

  **Note:** Potentially affected nodes include the input nodes and any other nodes that have access to the same storage enclosures and drives.
- **storage-enclosure**: Parallelism automatically happens if IBM Spectrum Scale is shut down on all of the potentially affected nodes.

  **Note:** Potentially affected nodes include the input nodes and any other nodes that have access to the same storage enclosures and drives.

**--dry-run**
Specifies that the command is to be run without updating any firmware.

**-v {yes | no}**
Specifies whether the command is to verify that the firmware version is newer than the current version. A value of **yes** specifies that the firmware is to be updated only if it is a newer version than the firmware already installed on the device. A value of **no** specifies that the firmware is to be updated in all cases, even if its version number indicates that it is an earlier level or the same level as the firmware already installed on the device. The default value is **yes**.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mmchfirmware** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. To update an enclosure (in this case, enclosure SV13306129) to the latest firmware that was supplied, issue the following command:

   ```
   mmchfirmware --type storage-enclosure --serial-number SV13306129
   ```

   The system displays output similar to this:

   ```
   Mon Dec  3 08:19:12 EST 2012: mmchfirmware: Processing node c45f01n01-ib0.gpfs.net
   c45f01n01-ib0: update-directory /usr/lpp/mmfs/updates/latest/firmware/enclosure/ProductId
   c45f01n01-ib0: Found storage-enclosure DCS3700 SV13306129, update-id esm0375.esm.
   c45f01n01-ib0: Updating enclosure firmware ESM_A. c45f01n01-ib0: Updating enclosure firmware ESM_B.
   ```

2. To update the drive firmware, issue the following command:

   ```
   mmchfirmware --type drive
   ```

   The system displays output similar to this:

```
Mon Dec  3 09:10:47 EST 2012: mmchfirmware: Processing node c45f02n01-ib0.gpfs.net
c45f02n01-ib0: update-directory /usr/lpp/mmfs/updates/latest/firmware/drive.
c45f02n01-ib0: Found drive firmware update-id ibm_fw_hdd_sas-9.99_linux_32-64_gn
r.zip.
c45f02n01-ib0: Updating firmware for drives in recovery group rg.
c45f02n01-ib0: Updating firmware for drives sg46,sg179,sg183.
c45f02n01-ib0: Updating firmware for drives sg48,sg158,sg171.
c45f02n01-ib0: Updating firmware for drives sg45,sg131,sg352.
c45f02n01-ib0: Updating firmware for drives sg304,sg123.
c45f02n01-ib0: Updating firmware for drives sg193,sg180.
...
c45f02n01-ib0: Updating firmware for drives not contained in a recovery group.
c45f02n01-ib0: Updating firmware for drives sg201,sg29,sg250,sg297,sg36.
c45f02n01-ib0: Updating firmware for drives sg345,sg97,sg200,sg249,sg199.
c45f02n01-ib0: Updating firmware for drives sg69,sg28,sg298,sg346.
```

3. To update the drive firmware for a specific drive, you first must determine the serial number of the drive. You can discover the drive serial number by using one of two options, as follows:
   a. Issue the following command:

      ```
      mmlspdisk RecoveryGroupName --pdisk
      ```

      The system displays output similar to this:

      ```
      >mmlspdisk BB1RGL --pdisk e4d5s04
      pdisk:
         replacementPriority = 1000
         name = "e4d5s04"
         device = "/dev/sdbf,/dev/sdhy"
         recoveryGroup = "BB1RGL"
         declusteredArray = "DA2"
         .
         .
         .
         hardware = "IBM-ESXS ST2000NM0001 BC4A Z1P4K3VF00009320N6RS"
      ```

      The drive serial number is the last part of the `hardware` value: `Z1P4K3VF00009320N6RS` in this example.

   b. Issue the following command:

      ```
      mmgetpdisktopology > OutputFile
      ```

      Edit your output file (`out1`, for example) and search for the pdisk name or device path that you want to update:

      ```
      cat out1 | grep e4d5s04
      ```

      The system displays output similar to this:

      ```
        >cat out1 | grep e4d5s04
         sdbf,sdhy:0:/dev/sdbf,/dev/sdhy:C0A82D0B5384A13C|e4d5s04|BB1RGL|C0A82D0B53849EE7|DA2||1:
      naa.5000C50055D5F0E7:0000%3a11%3a00.0/7.0.51.0,0000%3a8b%3a00.0/8.0.51.0:[7.0.51.0],
      [8.0.51.0]:/dev/sg59,/dev/sg237:IBM-ESXS:ST2000NM0001:BC4A:Z1P4K3VF00009320N6RS:46W6911:
      2000398934016:naa.500062B200422340,naa.500062B200422280:50080e5245ded03f.50080e5245dec03f:
      SV13306129:0/1:5-4:5000c50055d5f0e5.5000c50055d5f0e6:sg8,sg186:
      ```

      The drive serial number is field 12 of the colon-delimited fields. Again, you see that the drive serial number is `Z1P4K3VF00009320N6RS` in this example.

   Now issue the following command:

   ```
   mmchfirmware --type drive --serial-number Z1P4K3VF00009320N6RS
   ```

4. This example shows you how to update the drive firmware when the IBM Spectrum Scale RAID cluster is shut down. Updating the firmware during a maintenance shutdown is much faster than

updating the firmware while IBM Spectrum Scale is active. With IBM Spectrum Scale active, the parallelism is reduced, pdisks must be suspended and resumed, and declustered arrays must be rebalanced between each update iteration.

With IBM Spectrum Scale shut down in the cluster, run this command:

```
mmchfirmware --type drive --fast-offline
```

Progress messages are suppressed during this concurrent update because they are interspersed from different nodes. To display progress messages, run this command:

```
DEBUGmmchfirmware=1 mmchfirmware --type drive --fast-offline
```

5. To update the host adapter firmware, run this command:

```
mmchfirmware --type host-adapter
```

**Note:** The node must be down where the host adapter firmware is being updated.

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196.

## Location

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

# mmchpdisk command

Changes IBM Spectrum Scale RAID pdisk states. This command is to be used only in extreme situations under the guidance of IBM service personnel.

## Synopsis

```
mmchpdisk RecoveryGroupName --pdisk PdiskName
      { --simulate-dead | --simulate-failing | --kill | --revive | --revive-failing |
     --revive-slow | --diagnose |--suspend | --resume |
     --begin-service-drain | --end-service-drain }
     [ --identify {on|off}] [ --clear-error-counters ]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmchpdisk** command changes the states of pdisks.

**Attention:** This command is to be used only in extreme situations under the guidance of IBM service personnel.

## Parameters

*RecoveryGroupName*
Specifies the recovery group that contains the pdisk for which one or more states are to be changed.

**--pdisk** *PdiskName*
Specifies the target pdisk.

**--simulate-dead**
Specifies that the disk is being treated as if it were dead.

**Attention:**

- This command can cause permanent data loss, so do not use it on production systems.
- This option must be used with caution; if the total number of failures in a declustered array exceeds the fault tolerance of any vdisk in that array, permanent data loss might result.

**--simulate-failing**
Specifies that the disk is being treated as if it were failing.

**Attention:** This option must be used with caution; if the total number of failures in a declustered array exceeds the fault tolerance of any vdisk in that array, permanent data loss might result.

**--kill**
The **--kill** option is deprecated. If it is run, it acts like **--simulate-failing**.

**--revive**
Attempts to make a failed disk usable again by removing `dead`, `failing`, and `readonly` pdisk state flags. Data can become readable again if the disk was not rebuilt onto spare space; however, any data that was already reported as lost cannot be recovered.

**--revive-failing**
Clears the "failing" pdisk state flag and resets all of the disk hospital bit error rate history for the pdisk.

**Note:** It might take several months to rebuild this history. Do not use this option to clear `simulatedFailing`; use **--revive** instead.

**--revive-slow**
Clears the "slow" pdisk state flag and resets all of the disk hospital I/O performance history for the pdisk.

**--diagnose**
Runs basic tests on the pdisk. If no problems are found, the pdisk state automatically returns to `ok`.

**--suspend**
Suspends I/O to the pdisk until a subsequent resume command is given. If a pdisk remains in the `suspended` state for longer than a predefined timeout period, IBM Spectrum Scale RAID begins rebuilding the data from that pdisk into spare space.

**Attention:** Use this option with caution and only when performing maintenance on disks manually, bypassing the automatic system provided by **mmchcarrier**.

If a pdisk is removed by using this option, vdisks that store data on it are temporarily degraded, requiring data that was stored on the removed pdisk to be rebuilt from redundant data. Also, if you try to remove more pdisks than the redundancy level of the least redundant vdisk in that declustered array, data becomes inaccessible.

Therefore, when preparing to remove a pdisk, use the **--begin-service-drain** and **--end-service-drain** options instead of this option.

**--resume**
Cancels a previously run **mmchpdisk --suspend** command and resumes use of the pdisk.

Use this option only when performing maintenance on disks manually and bypassing the automatic system provided by **mmchcarrier**.

**--begin-service-drain**
Starts draining the specified pdisk so that it can be temporarily removed. After issuing the command with this option, wait until the pdisk is drained before removing it.

**Note:** This process requires sufficient spare space in the declustered array for the data that is to be drained. If the available spare space is insufficient, it can be increased with the **mmchrecoverygroup** command.

**--end-service-drain**
Returns drained data to a pdisk after the pdisk is brought back online.

**--identify {on | off}**
Turns on or off the disk identify light, if available.

**--clear-error-counters**
Resets the `IOErrors`, `IOTimeouts`, `mediaErrors`, `checksumErrors`, and `pathErrors` counters.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mmchpdisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

The following command example shows how to instruct IBM Spectrum Scale to try to revive the failed pdisk `c036d3` in recovery group `000DE37BOT`:

```
mmchpdisk 000DE37BOT --pdisk c036d3 --revive
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddpdisk command" on page 136
- "mmchcarrier command" on page 138
- "mmdelpdisk command" on page 173
- "mmlspdisk command" on page 199
- "mmlsrecoverygroup command" on page 202.

## Location

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

# mmchrecoverygroup command

Changes IBM Spectrum Scale RAID recovery group and declustered array attributes.

## Synopsis

```
mmchrecoverygroup RecoveryGroupName {--declustered-array DeclusteredArrayName
                                     {[--spares NumberOfSpares]
                                     [--vcd-spares NumberOfVCDSpares]
                                     [--scrub-duration NumberOfDays]
                                     [--replace-threshold NumberOfDisks]}
                                     [--upgrade-nsd-v2]
                                     [--version {VersionString | LATEST}]}
```

or

```
mmchrecoverygroup RecoveryGroupName --active ServerName
```

or

```
mmchrecoverygroup RecoveryGroupName --servers Primary[,Backup] [-v {yes | no}]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmchrecoverygroup** command changes recovery group and declustered array attributes.

**--version** is the only option that applies to the whole recovery group. All other options apply to a declustered array and must be used in conjunction with the **--declustered-array** option.

## Parameters

*RecoveryGroupName*
: Specifies the name of the recovery group being changed.

**--declustered-array** *DeclusteredArrayName*
: Specifies the name of the declustered array being changed.

**--spares** *NumberOfSpares*
: Specifies the number of disks' worth of spare space to set aside in the declustered array. This space is used to rebuild the declustered arrays when physical disks fail.

**--vcd-spares** *NumberOfVCDSpares*
: Specifies the number of disks that can be unavailable while full replication of vdisk configuration data (VCD) is still maintained.

**--scrub-duration** *NumberOfDays*
: Specifies the number of days, from 1 to 365 , for the duration of the scrub. The default value is 14.

**--replace-threshold** *NumberOfDisks*
: Specifies the number of pdisks that must fail in the declustered array before **mmlsrecoverygroup** will report that service (disk replacement) is needed.

**--version {***VersionString* **| LATEST}**
: Specifies the recovery group version.

  The valid version strings are:

  **LATEST**
  : Denotes the latest supported version. New recovery groups will default to the latest version.

**4.2.0-1**
Denotes the version with all features supported by Elastic Storage Server (ESS) 4.0.

**4.1.0.1**
Denotes the version with all features supported by GPFS Storage Server (GSS) 2.0.

**3.5.0.13**
Denotes the version with all features supported by GSS 1.5.

**3.5.0.5**
Denotes the version with all features supported prior to GSS 1.5.

**--upgrade-nsd-v2**
Changes pdisks' NSD version to version 2. This option is enabled when the recovery group version is 4.2.0.1.

**--active** *ServerName*
Changes the active server for the recovery group.

**--servers** *Primary*[,*Backup*]
Changes the defined list of recovery group servers.

**Note:** To use this option, all file systems that use this recovery group must be unmounted if the primary and backup servers are not just being swapped.

**-v {yes | no}**
Specifies whether the new server or servers should verify access to the pdisks of the recovery group. The default is **-v yes**. Use **-v no** to specify that configuration changes should be made without verifying pdisk access; for example, you could use this if all the servers were down.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmchrecoverygroup** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. The following command example shows how to change the number of spares to one and the replacement threshold to one for declustered array DA4 in recovery group 000DE37TOP:

   ```
   mmchrecoverygroup 000DE37TOP --declustered-array DA4 --spares 1 --replace-threshold 1
   ```

2. The following command example shows how to change the scrub duration to one day for declustered array DA2 of recovery group 000DE37BOT:

   ```
   mmchrecoverygroup 000DE37BOT --declustered-array DA2 --scrub-duration 1
   ```

3. The following command example shows how to replace the servers for a recovery group. In this example, assume that the two current servers for recovery group RG1 have been taken down for extended maintenance. IBM Spectrum Scale is shut down on these current RG1 servers. The new servers have already been configured to be recovery group servers, with **mmchconfig** parameter **nsdRAIDTracks** set to a nonzero value. The disks for RG1 have not yet been connected to the new servers, so verification of disk availability must be disabled by specifying **-v no** as shown here:

```
mmchrecoverygroup RG1 --servers newprimary,newbackup -v no
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmchpdisk command" on page 153
- "mmdelrecoverygroup command" on page 175
- "mmcrrecoverygroup command" on page 159
- "mmlsrecoverygroup command" on page 202.

## Location

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

# mmcrrecoverygroup command

Creates an IBM Spectrum Scale RAID recovery group and its component declustered arrays and pdisks and specifies the servers.

## Synopsis

```
mmcrrecoverygroup RecoveryGroupName -F StanzaFile --servers {Primary[,Backup]}
                  [--version {VersionString | LATEST}]
                  [-v {yes | no}]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmcrrecoverygroup** command is used to define a cluster-wide recovery group for use by IBM Spectrum Scale RAID. A recovery group is a set of physical disks shared by up to two server nodes. The set of disks must be partitioned into one or more declustered arrays.

See the following *IBM Spectrum Scale RAID: Administration* topic: Chapter 4, "Managing IBM Spectrum Scale RAID," on page 33.

The pdisk stanzas assigned to a recovery group must contain at least one declustered array that meets the definition of *large*.

While the **mmcrrecoverygroup** command creates the declustered arrays and pdisks and defines the servers, further processing with the **mmcrvdisk** and **mmcrnsd** commands is necessary to create IBM Spectrum Scale RAID vdisk NSDs within the recovery group.

**Note:** The recovery group servers must be active to run this command.

## Parameters

*RecoveryGroupName*
: Name of the recovery group being created.

**-F** *StanzaFile*
: Specifies a file that includes pdisk stanzas and declustered array stanzas that are used to create the recovery group. The declustered array stanzas are optional. For more information about pdisk stanzas, see the following *IBM Spectrum Scale RAID: Administration* topic: "Pdisk stanza format" on page 36.

  Declustered array stanzas look like the following:

```
%da: daName=DeclusteredArrayName
  spares=Number
  vcdSpares=Number
  replaceThreshold=Number
  scrubDuration=Number
  auLogSize=Number
  nspdEnable={yes|no}
```

  where:

  **daName**=*DeclusteredArrayName*
  : Specifies the name of the declustered array for which you are overriding the default values.

  **spares**=*Number*
  : Specifies the number of disks' worth of spare space to set aside in the declustered array. The number of spares can be 1 or higher. The default values are the following:

**1** for arrays with 9 or fewer disks

**2** for arrays with 10 or more disks

**vcdSpares**=*Number*
Specifies the number of disks that can be unavailable while full replication of vdisk configuration data (VCD) is still maintained. The default value is for this value to be the same as the number of spares.

**replaceThreshold**=*Number*
Specifies the number of pdisks that must fail in the declustered array before **mmlspdisk** will report that pdisks need to be replaced. The default is equal to the number of spares.

**scrubDuration**=*Number*
Specifies the length of time (in days) by which the scrubbing of entire array must be completed. Valid values are 1 to 60. The default value is 14 days.

**auLogSize**
Specifies the size of the atomic update log. This value must be chosen very carefully and typically only needs to be set for a declustered array consisting of non-volatile RAM disks. See the following *IBM Spectrum Scale RAID: Administration* topic: "Configuring IBM Spectrum Scale RAID recovery groups on the ESS: a sample scenario" on page 87.

**nspdEnable {yes|no}**
Specifies whether this declustered array should be enabled for network shared pdisks. Declustered arrays that contain non-volatile RAM disks must set **nspdEnable=yes**. The default value is no. See the following *IBM Spectrum Scale RAID: Administration* topic: "Configuring IBM Spectrum Scale RAID recovery groups on the ESS: a sample scenario" on page 87.

**--servers {***Primary***[,***Backup***]}**
Specifies the primary server and, optionally, a backup server.

**--version {***VersionString* **| LATEST}**
Specifies the recovery group version.

The valid version strings are:

**LATEST**
Denotes the latest supported version. New recovery groups will default to the latest version.

**4.2.0-1**
Denotes the version with all features supported by Elastic Storage Server (ESS) 4.0.

**4.1.0.1**
Denotes the version with all features supported by GPFS Storage Server (GSS) 2.0.

**3.5.0.13**
Denotes the version with all features supported by GSS 1.5.

**3.5.0.5**
Denotes the version with all features supported prior to GSS 1.5.

**-v {yes | no}**
Verification flag that specifies whether each pdisk in the stanza file should only be created if it has not been previously formatted as a pdisk (or NSD). The default is **-v yes**. Use **-v no** to specify that the disks should be created regardless of whether they have been previously formatted or not.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmcrrecoverygroup** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages.

## Examples

Suppose input stanza file `000DE37BOT` contains the following lines:

```
%pdisk: pdiskName=c034d1
        device=/dev/hdisk316
        da=DA1
        nPathActive=2
        nPathTotal=4
%pdisk: pdiskName=c034d2
        device=/dev/hdisk317
        da=DA2
        nPathActive=2
        nPathTotal=4
%pdisk: pdiskName=c034d3
        device=/dev/hdisk318
        da=DA3
        nPathActive=2
        nPathTotal=4
%pdisk: pdiskName=c034d4
        device=/dev/hdisk319
        da=DA4
        nPathActive=2
        nPathTotal=4
%pdisk: pdiskName=c033d1
        device=/dev/hdisk312
        da=LOG
        nPathActive=2
        nPathTotal=4
        nsdformatversion=1
[...]
```

The following command example shows how to create recovery group `000DE37BOT` using stanza file `000DE37BOT`, with `c250f10c08ap01-hf0` as the primary server and `c250f10c07ap01-hf0` as the backup server:

```
mmcrrecoverygroup 000DE37BOT -F 000DE37BOT --servers c250f10c08ap01-hf0,c250f10c07ap01-hf0
```

The system displays output similar to the following:

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

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmchrecoverygroup command" on page 156
- "mmcrvdisk command" on page 162
- "mmdelrecoverygroup command" on page 175
- "mmlsrecoverygroup command" on page 202.

## Location

`/usr/lpp/mmfs/bin`

# mmcrvdisk command

Creates a vdisk within a declustered array of an IBM Spectrum Scale RAID recovery group.

## Synopsis

```
mmcrvdisk -F StanzaFile [ -v { yes | no } ]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmcrvdisk** command creates one or more vdisks. Upon successful completion of the **mmcrvdisk** command, the vdisk stanza file is rewritten in a form that can be used as input to the **mmcrnsd** command.

The first vdisk that is created in a recovery group must be a log vdisk, which is indicated by a disk usage of **vdiskLog**.

**Note:** The recovery group must be active to run this command.

## Parameters

**-F** *StanzaFile*
Specifies a file that includes vdisk stanzas identifying the vdisks to be created.

Vdisk stanzas look like the following:

```
%vdisk: vdiskName=VdiskName
        rg=RecoveryGroupName
        da=DeclusteredArrayName
        blocksize=BlockSize
        size=Size
        raidCode=RAIDCode
        diskUsage=DiskUsage
        failureGroup=FailureGroup
        pool=StoragePool
        longTermEventLogSize=LongTermEventLogSize
        shortTermEventLogSize=ShortTermEventLogSize
        fastWriteLogPct=fastWriteLogPct
```

where:

**vdiskName**=*VdiskName*
Specifies the name you want to assign to the vdisk NSD that is to be created. This name must not already be used as another GPFS disk name, and it must not begin with the reserved string **gpfs**.

**Note:** This name can contain the following characters only:
Uppercase letters **A** through **Z**
Lowercase letters **a** through **z**
Numerals **0** through **9**
Underscore character (_)

All other characters are not valid.

**rg**=*RecoveryGroupName*
Specifies the recovery group where the vdisk is to be created.

**da**=*DeclusteredArrayName*
Specifies the declustered array within the recovery group where the vdisk is to be created.

**blocksize**=*BlockSize*
Specifies the size of the data blocks for the vdisk. This must match the block size planned for the file system. The valid values are:

**For nWayReplicated RAID codes:**
256 KiB, 512 KiB, 1 MiB, 2 MiB

**For 8+2p and 8+3p RAID codes:**
512 KiB, 1 MiB, 2 MiB, 4 MiB, 8 MiB, 16 MiB

Specify this value with the character **K** or **M** (for example, **512K**).

If you are using the system pool as metadata only and placing your data in a separate pool, you can specify your metadata-only vdisks with one block size and your data vdisks with a different block size. Since a file system with different metadata and data block sizes requires the use of multiple GPFS storage pools, a file system placement policy is needed to direct user file data to the data storage pool.

**size**=*Size*
Specifies the size of the vdisk. If **size**=*Size* is omitted, it defaults to using all the available space in the declustered array. The requested vdisk size is equally allocated across all of the pdisks within the declustered array.

**raidCode**=*RAIDCode*
Specifies the RAID code to be used for the vdisk. Valid codes are the following:

**Unreplicated**
Indicates no replication. This should only be used for a log tip backup vdisk.

**2WayReplication**
Indicates two-way replication. This should only be used for a log tip vdisk.

**3WayReplication**
Indicates three-way replication.

**4WayReplication**
Indicates four-way replication.

**8+2p**
Indicates Reed-Solomon 8 + 2p.

**8+3p**
Indicates Reed-Solomon 8 + 3p.

**diskUsage**=*DiskUsage*
Specifies a disk usage or accepts the default. With the exception of the **vdiskLog** value, this field is ignored by the **mmcrvdisk** command and is passed unchanged to the output stanza file.

Possible values are the following:

**dataAndMetadata**
Indicates that the disk contains both data and metadata. This is the default for disks in the system pool.

**dataOnly**
Indicates that the disk contains data and does not contain metadata. This is the default for disks in storage pools other than the system pool.

**metadataOnly**
Indicates that the disk contains metadata and does not contain data.

**descOnly**
Indicates that the disk contains no data and no file metadata. Such a disk is used solely to keep a copy of the file system descriptor, and can be used as a third failure group in certain

disaster recovery configurations. For more information, see the section called "Synchronous mirroring utilizing GPFS replication" in the chapter on "Establishing disaster recovery for your GPFS cluster" of the *IBM Spectrum Scale: : Administration Guide*.

**vdiskLog**
Indicates that this is the log home vdisk for the recovery group.

The log home vdisk must be created before any vdisks that are exposed through NSD.

**vdiskLogTip**
Indicates that this is the log tip vdisk for the recovery group.

**vdiskLogTipBackup**
Indicates that this is the log tip backup vdisk for the recovery group.

**vdiskLogReserved**
Indicates a vdisk that should have the same size and RAID code as the log home vdisk but is otherwise unused.

This is used to equalize the space consumption in the declustered arrays that do not contain the log home vdisk.

**failureGroup**=*FailureGroup*
This field is ignored by the **mmcrvdisk** command and is passed unchanged to the output stanza file. It identifies the failure group to which the vdisk NSD belongs.

**pool**=*StoragePool*
This field is ignored by the **mmcrvdisk** command and is passed unchanged to the output stanza file. It specifies the name of the storage pool to which the vdisk NSD is assigned.

**longTermEventLogSize**=*LongTermEventLogSize*
Specifies the size of the long-term event log, rounding up to the block size if not aligned. The default value is 4MB.

**longTermEventLogSize** applies only to the log vdisk.

**shortTermEventLogSize**=*ShortTermEventLogSize*
Specifies the size of the short-term event log, rounding up to the block size if not aligned. The default value is 1MB.

**shortTermEventLogSize** applies only to the log vdisk.

**fastWriteLogPct**=*FastWriteLogPct*
Specifies the fraction of the remaining log space to be used for the fast-write log, after allocating space for the event logs. The default value is 90%.

**fastWriteLogPct** applies only to the log vdisk.

**-v {yes | no }**
Verification flag; a value of **yes** specifies that vdisk creation will only start if the buffer space requirements of other recovery groups can be verified. The default value is **yes**.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmcrvdisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

Assume that input stanza file `000DE37TOP.vdisk` contains the following lines:

```
%vdisk: vdiskName=000DE37TOPLOG
        rg=000DE37TOP
        da=LOG
        blocksize=1m
        size=4g
        raidCode=3WayReplication
        diskUsage=vdiskLog
        longTermEventLogSize=4M
        shortTermEventLogSize=1M
        fastWriteLogPct=90
%vdisk: vdiskName=000DE37TOPDA1META
        rg=000DE37TOP
        da=DA1
        blocksize=1m
        size=250g
        raidCode=4WayReplication
        diskUsage=metadataOnly
        failureGroup=37
        pool=system
%vdisk: vdiskName=000DE37TOPDA1DATA
        rg=000DE37TOP
        da=DA1
        blocksize=8m
        raidCode=8+3p
        diskUsage=dataOnly
        failureGroup=37
        pool=data
[...]
```

The following command example shows how to create the vdisks described in the stanza file `000DE37TOP.vdisk`:

```
mmcrvdisk -F 000DE37TOP.vdisk
```

The system displays output similar to the following:

```
mmcrvdisk: [I] Processing vdisk 000DE37TOPLOG
mmcrvdisk: [I] Processing vdisk 000DE37TOPDA1META
mmcrvdisk: [I] Processing vdisk 000DE37TOPDA1DATA
mmcrvdisk: [I] Processing vdisk 000DE37TOPDA2META
mmcrvdisk: [I] Processing vdisk 000DE37TOPDA2DATA
mmcrvdisk: [I] Processing vdisk 000DE37TOPDA3META
mmcrvdisk: [I] Processing vdisk 000DE37TOPDA3DATA
mmcrvdisk: [I] Processing vdisk 000DE37TOPDA4META
mmcrvdisk: [I] Processing vdisk 000DE37TOPDA4DATA
mmcrvdisk: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmcrrecoverygroup command" on page 159
- "mmdelvdisk command" on page 177
- .

## Location

/usr/lpp/mmfs/bin

# mmdelcomp command

Deletes one or more storage components.

## Synopsis

**mmdelcomp** *Component* [**--dry-run**]

or

**mmdelcomp -F** *StanzaFile* [**--dry-run**]

## Availability

Available with the Elastic Storage Server.

## Description

The **mmdelcomp** command deletes one or more storage components.

## Parameters

*Component*
Specifies the name, serial number, or component ID of a single component to be deleted.

**-F** *StanzaFile*
Specifies the name of a stanza file from which input values are to be read. The stanza file can include stanzas for multiple components.

The correct format for each stanza is shown here. Each stanza must include at least one of the parameters listed. If more than one parameter is included in a stanza, the parameters must match the same component.

```
%comp:
 compId=Component
 serialNumber=SerialNumber
 name=Name
```

where:

**compId=***Component*
Specifies the component ID of a single component to be deleted.

**serialNumber=***SerialNumber*
Specifies the serial number of a single component to be deleted.

**name=***Name*
Specifies the name of a single component to be deleted.

**--dry-run**
Indicates that the changes resulting from the command are not to be recorded in the configuration file.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmdelcomp** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddcomp command" on page 130
- "mmaddcompspec command" on page 133
- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144
- "mmchenclosure command" on page 146
- "mmchfirmware command" on page 149
- "mmdelcomploc command" on page 169
- "mmdelcompspec command" on page 171
- "mmdiscovercomp command" on page 179
- "mmlscomp command" on page 184
- "mmlscomploc command" on page 187
- "mmlscompspec command" on page 190
- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196
- "mmsyncdisplayid command" on page 216
- Chapter 5, "Configuring components on the Elastic Storage Server," on page 47.

## Location

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

# mmdelcomploc command

Deletes one or more storage components from their locations.

## Synopsis

**mmdelcomploc** *Component Container* [**--dry-run**]

or

**mmdelcomploc -F** *StanzaFile* [**--dry-run**]

## Availability

Available with the Elastic Storage Server.

## Description

The **mmdelcomploc** command deletes one or more storage components from their locations.

## Parameters

*Component*
: Specifies the name, serial number, or component ID of a single component to be deleted from its location.

*Container*
: Specifies the name, serial number, or component ID of the container that holds the component.

**-F** *StanzaFile*
: Specifies the name of a stanza file from which input values are to be read. The stanza file can include stanzas for multiple components.

**--dry-run**
: Indicates that the changes resulting from the command are not to be recorded in the configuration file.

## Exit status

**0** Successful completion.

**nonzero**
: A failure has occurred.

## Security

You must have root authority to run the **mmdelcomploc** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddcomp command" on page 130
- "mmaddcompspec command" on page 133
- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144

- "mmchenclosure command" on page 146
- "mmchfirmware command" on page 149
- "mmdelcomp command" on page 167
- "mmdelcompspec command" on page 171
- "mmdiscovercomp command" on page 179
- "mmlscomp command" on page 184
- "mmlscomploc command" on page 187
- "mmlscompspec command" on page 190
- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196
- "mmsyncdisplayid command" on page 216.

## Location

`/usr/lpp/mmfs/bin`

# mmdelcompspec command

Deletes one or more storage component specifications.

## Synopsis

**mmdelcompspec** *PartNumber* [**--dry-run**]

or

**mmdelcompspec -F** *StanzaFile* [**--dry-run**]

## Availability

Available with the Elastic Storage Server.

## Description

The **mmdelcompspec** command deletes one or more storage component specifications.

## Parameters

*PartNumber*
Specifies the part number or model type of the component to be deleted.

**-F** *StanzaFile*
Specifies the name of a stanza file from which input values are to be read. The stanza file can include stanzas for multiple components.

The correct format for each stanza is shown here. Each stanza must include at least one of the parameters listed. If more than one parameter is included in a stanza, the parameters must match the same component.

```
%compSpec:
 partNumber=PartNumber
```

**--dry-run**
Runs the command without making any permanent changes.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmdelcompspec** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

To delete a generic 60U rack, which had been added previously with a part number of RACK60U, enter the following command:

```
mmdelcompspec RACK60U
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddcompspec command" on page 133
- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144
- "mmdelcomp command" on page 167
- "mmdelcomploc command" on page 169
- "mmdiscovercomp command" on page 179
- "mmlscomp command" on page 184
- "mmlscomploc command" on page 187
- "mmlscompspec command" on page 190
- "mmsyncdisplayid command" on page 216
- Chapter 5, "Configuring components on the Elastic Storage Server," on page 47.

## Location

`/usr/lpp/mmfs/bin`

# mmdelpdisk command

Deletes IBM Spectrum Scale RAID pdisks.

## Synopsis

**mmdelpdisk** *RecoveryGroupName* {**--pdisk** "*PdiskName*[;*PdiskName*...]" | **-F** *StanzaFile*} [**-a**]

or

**mmdelpdisk** *RecoveryGroupName* **--declustered-array** *DeclusteredArrayName*

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmdelpdisk** command deletes one or more pdisks. Deleting a pdisk causes any data allocated to that disk to be moved or *rebuilt* (drained) to spare space in the declustered array.

The **mmdelpdisk** command first renames each pdisk that is to be deleted, giving it a temporary name. The command then drains each renamed pdisk to remove all data from it. Finally, the command destroys each renamed pdisk once all data has been drained from it.

**Note:** The temporary name is obtained by appending a suffix in the form **#***nnnn* to the pdisk name. For example, a pdisk named `p25` will receive a temporary name similar to `p25#010`; this allows you to use the **mmaddpdisk** command to add a new pdisk with the name `p25` immediately rather than waiting for the old disk to be completely drained and removed. Until the draining and removing process is complete, both the new pdisk `p25` and the old pdisk `p25#0010` will show up in the output of the **mmlsrecoverygroup** and **mmlspdisk** commands.

If **mmdelpdisk** is interrupted (by an interrupt signal or by GPFS server failover), the deletion will proceed and will be completed as soon as another IBM Spectrum Scale RAID server becomes the active vdisk server of the recovery group.

If you wish to delete a declustered array and all pdisks in that declustered array, use the **--declustered-array** *DeclusteredArrayName* form of the command.

The **mmdelpdisk** command cannot be used if the declustered array does not have enough spare space to hold the data that needs to be drained, or if it attempts to reduce the size of a large declustered array below the limit for large declustered arrays. Normally, all of the space in a declustered array is allocated to vdisks and spares, and therefore the only times the **mmdelpdisk** command typically can be used is after adding pdisks, after deleting vdisks, or after reducing the designated number of spares. See the following topics: “mmaddpdisk command” on page 136 and “mmchcarrier command” on page 138.

**Note:** The recovery group must be active to run this command.

## Parameters

*RecoveryGroupName*
: Specifies the recovery group from which the pdisks are being deleted.

**--pdisk** "*PdiskName*[;*PdiskName*...]"
: Specifies a semicolon-separated list of pdisk names identifying the pdisks to be deleted.

**-F** *StanzaFile*
Specifies a file that contains pdisk stanzas identifying the pdisks to be deleted. For more information about pdisk stanzas, see the the following *IBM Spectrum Scale RAID: Administration* topic: "Pdisk stanza format" on page 36.

**-a** Indicates that the data on the deleted pdisks is to be drained asynchronously. The pdisk will continue to exist, with its name changed to a temporary name, while the deletion progresses.

**--declustered-array** *DeclusteredArrayName*
Specifies the name of the declustered array whose pdisks are to be deleted.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmdelpdisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

The following command example shows how to remove pdisk `c016d1` from recovery group `000DE37TOP` and have it be drained in the background, thereby returning the administrator immediately to the command prompt:

```
mmdelpdisk 000DE37TOP --pdisk c016d1 -a
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddpdisk command" on page 136
- "mmchpdisk command" on page 153
- "mmdelrecoverygroup command" on page 175
- "mmdelvdisk command" on page 177
- "mmlspdisk command" on page 199
- "mmlsrecoverygroup command" on page 202.

## Location

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

# mmdelrecoverygroup command

Deletes an IBM Spectrum Scale RAID recovery group.

## Synopsis

**mmdelrecoverygroup** *RecoveryGroupName* [**-p**]

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmdelrecoverygroup** command deletes the specified recovery group and the declustered arrays and pdisks that it contains. The recovery group must not contain any vdisks; use the **mmdelvdisk** command to delete vdisks prior to running this command.

**Note:** The recovery group must be active to run this command, unless the **-p** option is specified.

## Parameters

*RecoveryGroupName*
: Specifies the name of the recovery group to delete.

**-p** Indicates that the recovery group is permanently damaged and that the recovery group information should be removed from the GPFS cluster configuration data. The **-p** option may be used when the GPFS daemon is down on the recovery group servers.

## Exit status

**0** Successful completion.

**nonzero**
: A failure has occurred.

## Security

You must have root authority to run the **mmdelrecoverygroup** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

The following command example shows how to delete recovery group `000DE37BOT`:

```
mmdelrecoverygroup 000DE37BOT
```

The system displays output similar to the following:

```
mmdelrecoverygroup: [I] Recovery group 000DE37BOT deleted on node c250f10c08ap01-hf0.ppd.pok.ibm.com.
mmdelrecoverygroup: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmchrecoverygroup command" on page 156
- "mmcrrecoverygroup command" on page 159
- "mmlsrecoverygroup command" on page 202.

## Location

`/usr/lpp/mmfs/bin`

# mmdelvdisk command

Deletes vdisks from a declustered array in an IBM Spectrum Scale RAID recovery group.

## Synopsis

**mmdelvdisk** {"*VdiskName*[**;***VdiskName***...]" | **-F** *StanzaFile*} [**-p** | **--recovery-group** *RecoveryGroupName*]

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmdelvdisk** command is used to delete vdisks from the declustered arrays of recovery groups. There must be no NSD defined on the specified vdisk; if necessary, use the **mmdelnsd** command prior to running this command.

The log home vdisk in a recovery group cannot be deleted until all the other vdisks in the recovery group have been deleted. Log tip and log tip backup vdisks may be created and deleted at any time, even while work is active on those or other vdisks.

## Parameters

*VdiskName*[**;***VdiskName***...**]
: Specifies the vdisks to be deleted.

**-F** *StanzaFile*
: Specifies the name of a stanza file in which stanzas of the type **%vdisk** identify the vdisks to be deleted. Only the vdisk name is required to be included in the vdisk stanza; however, for a complete description of vdisk stanzas, see the following topic: "mmcrvdisk command" on page 162.

**-p** Indicates that the recovery group is permanently damaged, and therefore the vdisk information should be removed from the GPFS cluster configuration data. This option can be used when the GPFS daemon is down on the recovery group servers.

**--recovery-group** *RecoveryGroupName*
: Specifies the name of the recovery group that contains the vdisks. If **mmcrvdisk** fails in the middle of the operation, the vdisks will be present in the recovery group; however, they will not be present in the GPFS cluster configuration data.

  You can see the vdisks in the recovery group by issuing either of the following commands:

  ```
  mmlsvdisk --recovery-group RecoveryGroupName
  ```

  or

  ```
  mmlsrecoverygroup RecoveryGroupName -L
  ```

## Exit status

**0** Successful completion.

**nonzero**
: A failure has occurred.

## Security

You must have root authority to run the **mmdelvdisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For

additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

The following command example shows how to delete vdisk `000DE37BOTDA4DATA`:

```
mmdelvdisk 000DE37BOTDA4DATA
```

The system displays output similar to the following:

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

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmcrvdisk command" on page 162
- "mmdelpdisk command" on page 173
- "mmdelrecoverygroup command" on page 175
- "mmlsvdisk command" on page 213.

## Location

`/usr/lpp/mmfs/bin`

# mmdiscovercomp command

Discovers components and adds them to the GPFS cluster configuration.

## Synopsis

**mmdiscovercomp -N** {*Node*[,*Node*...] | *NodeFile* | *NodeClass*} [**--dry-run**]

## Availability

Available with the Elastic Storage Server.

## Description

The **mmdiscovercomp** command discovers components and adds them to the configuration.

## Parameters

**-N {***Node***[,***Node***...] | *NodeFile* | *NodeClass***}**
Specifies the nodes to which the command applies.

**--dry-run**
Indicates that the changes resulting from the command are not to be recorded in the configuration file.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmdiscovercomp** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

To find supported storage enclosures attached to the nodes of a cluster and add them to the configuration, enter the following command:

```
mmdiscovercomp all
```

The system displays output similar to this:

```
Adding enclosure: serial number = SV12345001; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345007; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345003; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345005; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345004; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345002; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345008; vendor ID = IBM; product ID = DCS3700
Adding enclosure: serial number = SV12345006; vendor ID = IBM; product ID = DCS3700

Storage Enclosures

Status  Comp ID  Component Type    Part Number  Serial Number  Name               Display ID
```

```
------  -------  ---------------  -----------  -------------  ------------------  ----------
new           1  storageEnclosure  1818-80E     SV12345001     1818-80E-SV12345001  00
new           2  storageEnclosure  1818-80E     SV12345007     1818-80E-SV12345007  00
new           3  storageEnclosure  1818-80E     SV12345003     1818-80E-SV12345003  00
new           4  storageEnclosure  1818-80E     SV12345005     1818-80E-SV12345005  00
new           5  storageEnclosure  1818-80E     SV12345004     1818-80E-SV12345004  00
new           6  storageEnclosure  1818-80E     SV12345002     1818-80E-SV12345002  00
new           7  storageEnclosure  1818-80E     SV12345008     1818-80E-SV12345008  00
new           8  storageEnclosure  1818-80E     SV12345006     1818-80E-SV12345006  00
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddcomp command" on page 130
- "mmaddcompspec command" on page 133
- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144
- "mmchenclosure command" on page 146
- "mmchfirmware command" on page 149
- "mmdelcomp command" on page 167
- "mmdelcomploc command" on page 169
- "mmdelcompspec command" on page 171
- "mmlscomp command" on page 184
- "mmlscomploc command" on page 187
- "mmlscompspec command" on page 190
- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196
- "mmsyncdisplayid command" on page 216
- Chapter 5, "Configuring components on the Elastic Storage Server," on page 47.

## Location

`/usr/lpp/mmfs/bin`

# mmgetpdisktopology command

Obtains the disk subsystem configuration on an IBM Spectrum Scale RAID server.

## Synopsis

```
mmgetpdisktopology
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmgetpdisktopology** command prints a colon-delimited database of the elements of the disk subsystem on a IBM Spectrum Scale RAID server. The information is obtained through operating system and device queries. The output is intended to be captured to a file for use by such IBM Spectrum Scale RAID configuration scripts as **topsummary**, **topselect**, and **mkrginput** in verifying the disk subsystem and creating IBM Spectrum Scale RAID recovery groups.

The *topology file* that is produced by **mmgetpdisktopology** might also be useful to IBM Service personnel and in diagnosing disk subsystem configuration problems.

IBM Spectrum Scale does not need to be running to acquire a topology file.

For the purpose of verifying the disk subsystem topology on a server, IBM Spectrum Scale needs merely to be installed to acquire a topology file.

For the purpose of creating IBM Spectrum Scale RAID recovery groups, the node where the topology file is acquired needs to be a configured member of a GPFS cluster when the topology is acquired.

It is sometimes useful to peer inside the topology file, even though it is usually considered an opaque object with the primary purpose of facilitating IBM Spectrum Scale RAID recovery group creation.

Within the topology file are lines of four types: disk, expander, adapter, and host. There is one host line, for the server that this topology describes. There is one line that describes the attributes and relationships for each disk, expander, and adapter present on the server at the time the topology was acquired.

Each line, regardless of type, contains 21 colon-delimited fields.

The host line is the simplest and is used mainly to record the server hostname and GPFS node name.

The disk, expander, and adapter lines capture information in all 21 fields.

*Table 21. Topology file fields*

| Field | Name | Description |
|---|---|---|
| 1 | **name** | Name for this device |
| 2 | **type** | SCSI type (0 for disk, 13 for expander, 31 for adapter) |
| 3 | **device** | **/dev** device |
| 4 | **nsdid** | GPFS NSD label information for disk devices |
| 5 | **wwid** | Unique world-wide identifier |
| 6 | **location** | Physical location code |
| 7 | **hctl** | Hostbus:controller:target:lun |
| 8 | **altdev** | Other **/dev** device |

*Table 21. Topology file fields (continued)*

| Field | Name | Description |
|---|---|---|
| 9 | **vendor** | Vendor SCSI query |
| 10 | **product** | Product name SCSI query |
| 11 | **firmware** | Firmware level SCSI query |
| 12 | **serial** | Serial number SCSI query |
| 13 | **fru** | FRU SCSI query |
| 14 | **size** | Size in bytes for disk devices |
| 15 | **adapters** | WWIDs of adapters that see this device |
| 16 | **expanders** | WWIDs of expanders that see this device |
| 17 | **enclosure** | Enclosure that contains this device |
| 18 | **port** | Enclosure ESM port that connects the device |
| 19 | **slot** | Enclosure slot where the disk resides |
| 20 | **sasaddrs** | SAS addresses for the device |
| 21 | **sesdev** | Device names of expanders that see the device |

Some fields, such as `size`, apply to disks only. Some, such as `enclosure`, do not apply to adapters. Some fields contain two comma-separated values; these are for multiple paths to the same disk or expander. In the case of multiple paths, the order of the comma-separated values is preserved across fields. If **"/dev/sdabc,/dev/sdxyz"** is the `device` field and **"sg57,sg195"** is the `sesdev` field, **sg57** is the expander through which **/dev/sdabc** is connected and **sg195** is the expander through which **/dev/sdxyz** is connected.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmgetpdisktopology** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Example

The following command example shows how to acquire the topology file on a IBM Spectrum Scale RAID server:

```
mmgetpdisktopology > server1.top
```

The **server1.top** topology file can now be used with the **topsummary**, **topselect**, and **mkrginput** commands.

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmcrrecoverygroup command" on page 159
- "mkrginput script" on page 265
- "topselect script" on page 268
- "topsummary script" on page 271.

## Location

`/usr/lpp/mmfs/bin`

# mmlscomp command

Displays storage components.

## Synopsis

```
mmlscomp [--type CompType] [--part-number PartNumber]
           [--serial-number SerialNumber] [--name Name] [--comp-id CompId]
           [--format Format] [--output-file OutputFile] [--sortSortKey]
```

## Availability

Available with the Elastic Storage Server.

## Description

The **mmlscomp** command displays a list of storage components with attributes that match the specified parameters.

## Parameters

**--type** *CompType*
: Specifies a component type, which may include wildcard characters. The resulting list includes those components whose component types (**rack**, **server**, or **storageEnclosure** ) match the specified value.

**--part-number** *PartNumber*
: Specifies a part number, which may include wildcard characters. The resulting list includes those components whose part numbers match the specified value.

**--serial-number** *SerialNumber*
: Specifies a serial number, which may include wildcard characters. The resulting list includes those components whose serial numbers match the specified value.

**--name** *Name*
: Specifies a name, which may include wildcard characters. The resulting list includes those components whose names match the specified value.

**--comp-id** *CompId*
: Specifies a component ID, which may include wildcard characters. The resulting list includes those components whose component IDs match the specified value.

**--format** *Format*
: Specifies the format in which the component data will be listed. Valid values are: **colon**, **column**, **stanza**, and **table**. The default value is **table**.

**--output-file** *OutputFile*
: Specifies the name of an output file in which the component data will listed.

**--sort** *SortKey*
: Indicates that the command output is to be sorted by the attribute specified in *SortKey*.

## Exit status

**0** Successful completion.

**nonzero**
: A failure has occurred.

## Security

You must have root authority to run the **mmlscomp** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. To list components sorted by name, enter the following command:

```
$ mmlscomp --sort name
```

The system displays output similar to this:

```
    Rack Components

Comp ID  Part Number  Serial Number  Name
-------  -----------  -------------  ------
      9  1410HEA                     R01C01

    Storage Enclosure Components

Comp ID  Part Number  Serial Number  Name                Display ID
-------  -----------  -------------  ------------------  ----------
      2  1818-80E     SV12345007     G1E1-SX98760044     01
      4  1818-80E     SV12345005     G1E2-SX98760049     05
      3  1818-80E     SV12345003     G1E3-SX98760041     13
      7  1818-80E     SV12345008     G1E4-SX98760036     17
      1  1818-80E     SV12345001     G1E5-SX98760048     21
      6  1818-80E     SV12345002     G1E6-SX98760050     25
      8  1818-80E     SV12345006     G1E7-SX98760039     33
      5  1818-80E     SV12345004     G1E8-SX98760052     37
```

2. To list components in stanza format, enter the following command:

```
mmlscomp --format stanza
```

The system displays output similar to this:

```
%rack:
  compId=9
  name=R01C01
  partNumber=1410HEA

%storageEnclosure:
  compId=1
  displayId=21
  name='1818-80E-SV12345001'
  partNumber=1818-80E
  serialNumber=SV12345001

%storageEnclosure:
  compId=2
  displayId=1
  name='1818-80E-SV12345007'
  partNumber=1818-80E
  serialNumber=SV12345007

%storageEnclosure:
  compId=3
  displayId=13
  name='G1E3-SX98760044'
  partNumber=1818-80E
  serialNumber=SV12345003
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddcomp command" on page 130
- "mmaddcompspec command" on page 133

- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144
- "mmchenclosure command" on page 146
- "mmchfirmware command" on page 149
- "mmdelcomp command" on page 167
- "mmdelcomploc command" on page 169
- "mmdelcompspec command" on page 171
- "mmdiscovercomp command" on page 179
- "mmlscomploc command" on page 187
- "mmlscompspec command" on page 190
- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196
- "mmsyncdisplayid command" on page 216
- Chapter 5, "Configuring components on the Elastic Storage Server," on page 47.

## Location

`/usr/lpp/mmfs/bin`

# mmlscomploc command

Displays storage component locations.

## Synopsis

```
mmlscomploc [--type CompType] [--part-number PartNumber]
            [--serial-number SerialNumber] [--name Name] [--comp-id CompId]
            [--container-id ContainerId] [--format Format]
            [--output-file OutputFile] [--sort SortKey]
```

## Availability

Available with the Elastic Storage Server.

## Description

The **mmlscomploc** command displays a list of the locations of storage components.

## Parameters

**--type** *CompType*
Specifies a component type, which may include wildcard characters. The resulting list includes those components whose component types (**rack**, **server**, or **storageEnclosure**) match the specified value.

**--part-number** *PartNumber*
Specifies a part number, which may include wildcard characters. The resulting list includes those components whose part numbers match the specified value.

**--serial-number** *SerialNumber*
Specifies a serial number, which may include wildcard characters. The resulting list includes those components whose serial numbers match the specified value.

**--name** *Name*
Specifies a name, which may include wildcard characters. The resulting list includes those components whose names match the specified value.

**--comp-id** *CompId*
Specifies a component ID, which may include wildcard characters. The resulting list includes those components whose component IDs match the specified value.

**--container-id** *ContainerId*
Specifies a container ID, which may include wildcard characters. The resulting list includes those components whose container IDs match the specified value.

**--format** *Format*
Specifies the format in which the component locations will be listed. Valid format values are:

- `bare`
- `colon`
- `csv`
- `none`
- `shell`
- `stanza`
- `table` (default)

**stanza**, and **table**. The default value is **table**.

**--output-file** *OutputFile*
Specifies the name of an output file in which the component locations will be listed.

**--sort** *SortKey*
Indicates that the command output is to be sorted by the attribute specified in *SortKey*.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmlscomploc** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

To list component locations, enter the following command:

```
mmlscomploc
```

The system displays output similar to this:

```
Component           Location
------------------  ------------------
1818-80E-SV12345007 Rack R01C01 U01-04
1818-80E-SV12345005 Rack R01C01 U05-08
1818-80E-SV12345003 Rack R01C01 U13-16
1818-80E-SV12345008 Rack R01C01 U17-20
1818-80E-SV12345001 Rack R01C01 U21-24
1818-80E-SV12345002 Rack R01C01 U25-28
1818-80E-SV12345006 Rack R01C01 U33-36
1818-80E-SV12345004 Rack R01C01 U37-40
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:


- "mmaddcomp command" on page 130
- "mmaddcompspec command" on page 133
- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144
- "mmchenclosure command" on page 146
- "mmchfirmware command" on page 149
- "mmdelcomp command" on page 167
- "mmdelcomploc command" on page 169
- "mmdelcompspec command" on page 171
- "mmdiscovercomp command" on page 179
- "mmlscomp command" on page 184
- "mmlscompspec command" on page 190
- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196
- "mmsyncdisplayid command" on page 216

- Chapter 5, “Configuring components on the Elastic Storage Server,” on page 47.

## Location

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

# mmlscompspec command

Displays storage component specifications.

## Synopsis

```
mmlscompspec [--type CompType] [--part-number PartNumber]
             [--format Format] [--output-file OutputFile]
```

## Availability

Available with the Elastic Storage Server.

## Description

The **mmlscompspec** command displays a list of the specifications of storage components with attributes that match the specified parameters.

## Parameters

**--type** *CompType*
: Specifies a component type, which may include wildcard characters. The resulting list includes those components whose component types (**rack**, **server**, or **storageEnclosure**) match the specified value.

**--part-number** *PartNumber*
: Specifies a part number, which may include wildcard characters. The resulting list includes those components whose part numbers match the specified value.

**--format** *Format*
: Specifies the format in which the component locations will be listed. Valid values are: **colon**, **column**, **stanza**, and **table**. The default value is **table**.

**--output-file** *OutputFile*
: Specifies the name of an output file in which the component specifications will be listed.

## Exit status

**0** Successful completion.

**nonzero**
: A failure has occurred.

## Security

You must have root authority to run the **mmlscompspec** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

To list existing component specifications, enter the following command:

```
mmlscompspec
```

The system displays output similar to this:

```
    Rack Specifications

Part Number  Height  Description
```

```
-----------  ------  --------------------------------------------------
1410HEA          42  42U 1200mm Deep Expansion Rack
1410HPA          42  42U 1200mm Deep Primary Rack

    Server Specifications

Part Number  Height  Description
-----------  ------  -------------------
824722L           2 IBM Power System S822L

    Storage Enclosure Specifications

Part Number  Height  Description                 Vendor ID  Product ID  Drive Slots  Has Display ID
-----------  ------  --------------------------  ---------  ----------  -----------  --------------
1818-80E          4  DCS3700 Expansion Enclosure  IBM        DCS3700              60               1
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddcomp command" on page 130
- "mmaddcompspec command" on page 133
- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144
- "mmchenclosure command" on page 146
- "mmchfirmware command" on page 149
- "mmdelcomp command" on page 167
- "mmdelcomploc command" on page 169
- "mmdelcompspec command" on page 171
- "mmdiscovercomp command" on page 179
- "mmlscomp command" on page 184
- "mmlscomploc command" on page 187
- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196
- "mmsyncdisplayid command" on page 216
- Chapter 5, "Configuring components on the Elastic Storage Server," on page 47.

## Location

`/usr/lpp/mmfs/bin`

# mmlsenclosure command

Displays the environmental status of IBM Spectrum Scale RAID disk enclosures.

## Synopsis

```
mmlsenclosure {all | SerialNumber}
                [ -N {all | Node,[Node...] | NodeFile | NodeClass} ]
                [-L] [--not-ok]
```

## Availability

Available with the Elastic Storage Server.

## Description

Use the **mmlsenclosure** command to display the environmental status of the IBM Spectrum Scale RAID disk enclosures in the cluster. The command reports data about enclosure fans, power supplies, and other FRUs so that the equipment can be monitored and serviced in the field. You can choose to display either a brief summary or a more detailed listing of information.

**Note:** This command displays SCSI Enclosure Services (SES) status only and does not necessarily report on all enclosures.

**Output values for mmlsenclosure** *SerialNumber* **-L**

Descriptions of the output values for the **mmlsenclosure** *SerialNumber* **-L** command follow, as displayed by row, from top to bottom and left to right.

**serial number**
Is the serial number of the enclosure.

**needs service**
Indicates whether any of the components is reporting a failure.

**Note:** If `enclosure` is the only component that is showing a failure, it is probably because there is an LED light on for least one of the drives. This is *not* an enclosure failure.

**nodes**
Are the hostnames of the nodes that have access to this enclosure. To see all of the nodes that could have access to the enclosures, you need to specify **-N all** or a list of all of the server nodes.

**component type**
Is the component type within the enclosure. The component types follow:

**currentSensor**
Is a current sensor.

**dcm**
Is a drawer control module.

**door**
Is a drawer latch.

**drawer**
Is a drawer.

**enclosure**
Is a storage enclosure.

**esm**
Is an environmental service module.

| **expander**
| Is a sub-expander.

**fan**
Is a cooling fan.

**powerSupply**
Is a power supply.

| **sideplane**
| Is an expander board.

**tempSensor**
Is a temperature sensor.

**voltageSensor**
Is a voltage sensor.

**component id**
Is the component ID for the component type.

**failed**
Indicates whether the component is reporting a failing state.

**value**
Is the value that is reported by the SES information, where applicable.

**unit**
Is the measurement unit for the value specified.

**properties**
Lists additional information, where applicable - about a possible failure, for example.

**Note:** In the case of failures, the field replaceable units are:

- Drives.
- Enclosure drawers. Any failure that is related to a drawer control module (DCM)'s component ID (component type: **dcm**) requires a drawer replacement.
- Environmental service modules (ESMs).
- Power supplies.
- Fan assemblies.

## Parameters

**all** | *SerialNumber*
Specifies the serial number of the enclosure for which status is to be displayed. **all** specifies that status for all enclosures on the specified nodes is to be displayed.

**-N {all** | *Node*,**[***Node***...]** | *NodeFile* | *NodeClass***}**
Specifies the nodes from which data is to be gathered. The default is to collect information from the node on which the command is issued.

**-L** Specifies that more detailed information is to be displayed. The default for this command is to display summary information only.

**--not-ok**
Specifies that you wish to display the status of only the components that are reporting an unusual condition.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

If you are a root user, the node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. To display the status of all enclosures on the current node, run:

   ```
   mmlsenclosure all
   ```

   The system displays information similar to this:

   ```
                      needs  nodes
    serial number     service
    -------------     ------- ------
    SV14507574        no      mgmt001st001
    SV14519824        no      mgmt001st001
    SV15103688        no      mgmt001st001
    SV15103765        yes     mgmt001st001
   ```

2. To display the status of all enclosures on all nodes, run:

   ```
   mmlsenclosure  all -N all
   ```

   The system displays information similar to this:

   ```
                      needs  nodes
    serial number     service
    -------------     ------- ------
    SV14507574        no      mgmt001st001,mgmt002st001
    SV14519824        no      mgmt001st001,mgmt002st001
    SV15103688        no      mgmt001st001,mgmt002st001
    SV15103765        yes     mgmt001st001,mgmt002st001
   ```

3. To display the status of all enclosures on all nodes that are reporting an unusual condition, run:

   ```
   mmlsenclosure all -N all --not-ok
   ```

   The system displays information similar to this:

   ```
                      needs  nodes
    serial number     service
    -------------     ------- ------
    SV15103765        yes     mgmt001st001,mgmt002st001
   ```

4. To display detailed status for the components of enclosure SV15103765, run:

   ```
   mmlsenclosure SV15103765 -L
   ```

   The system displays information similar to this:

   ```
                    needs  nodes
   serial number     service
   -------------     ------- ------
   SV15103765        yes     mgmt001st001

   component type  serial number     component id    failed value   unit   properties
   --------------  -------------     ------------    ------ -----   ----   ----------
   dcm             SV15103765        DCM_0A          no                    NOT_IDENTIFYING
   dcm             SV15103765        DCM_0B          no                    NOT_IDENTIFYING
   dcm             SV15103765        DCM_1A          no                    NOT_IDENTIFYING
   dcm             SV15103765        DCM_1B          no                    NOT_IDENTIFYING
   dcm             SV15103765        DCM_2A          no                    NOT_IDENTIFYING
   dcm             SV15103765        DCM_2B          no                    NOT_IDENTIFYING
   dcm             SV15103765        DCM_3A          no                    NOT_IDENTIFYING
   dcm             SV15103765        DCM_3B          no                    NOT_IDENTIFYING
   dcm             SV15103765        DCM_4A          no                    NOT_IDENTIFYING
   dcm             SV15103765        DCM_4B          yes                   FAILED,NOT_IDENTIFYING
   ```

```
component type  serial number     component id    failed value   unit   properties
--------------  -------------     ------------    ------ -----   ----   ----------
enclosure       SV15103765        ONLY            yes                   NOT_IDENTIFYING,FAILED

component type  serial number     component id    failed value   unit   properties
--------------  -------------     ------------    ------ -----   ----   ----------
esm             SV15103765        ESM_A           no                    REPORTER
esm             SV15103765        ESM_B           no                    NOT_REPORTER

component type  serial number     component id    failed value   unit   properties
--------------  -------------     ------------    ------ -----   ----   ----------
fan             SV15103765        0_TOP_LEFT      no     5080    RPM
fan             SV15103765        1_BOT_LEFT      no     5080    RPM
fan             SV15103765        2_BOT_RGHT      no     4990    RPM
fan             SV15103765        3_TOP_RGHT      no     5080    RPM

component type  serial number     component id    failed value   unit   properties
--------------  -------------     ------------    ------ -----   ----   ----------
powersupply     SV15103765        0_TOP           no
powersupply     SV15103765        1_BOT           no

component type  serial number     component id    failed value   unit   properties
--------------  -------------     ------------    ------ -----   ----   ----------
tempsensor      SV15103765        DCM_0A          no     41      C
tempsensor      SV15103765        DCM_0B          no     30      C
tempsensor      SV15103765        DCM_1A          no     41      C
tempsensor      SV15103765        DCM_1B          no     31      C
tempsensor      SV15103765        DCM_2A          no     42      C
tempsensor      SV15103765        DCM_2B          no     32      C
tempsensor      SV15103765        DCM_3A          no     39      C
tempsensor      SV15103765        DCM_3B          no     31      C
tempsensor      SV15103765        DCM_4A          no     39      C
tempsensor      SV15103765        DCM_4B          yes    3.45846e-323 C  BUS_FAILURE,
tempsensor      SV15103765        ESM_A           no     31      C
tempsensor      SV15103765        ESM_B           no     32      C
tempsensor      SV15103765        POWERSUPPLY_BOT no     29      C
tempsensor      SV15103765        POWERSUPPLY_TOP no     27      C

component type  serial number     component id    failed value   unit   properties
--------------  -------------     ------------    ------ -----   ----   ----------
voltagesensor   SV15103765        12v             no     11.93   V
voltagesensor   SV15103765        ESM_A_1_0v      no     0.98    V
voltagesensor   SV15103765        ESM_A_1_2v      no     1.19    V
voltagesensor   SV15103765        ESM_A_3_3v      no     3.3     V
voltagesensor   SV15103765        ESM_A_5v        no     5.04    V
voltagesensor   SV15103765        ESM_B_1_0v      no     0.98    V
voltagesensor   SV15103765        ESM_B_1_2v      no     1.18    V
voltagesensor   SV15103765        ESM_B_3_3v      no     3.33    V
voltagesensor   SV15103765        ESM_B_3_3v      no     5.07    V
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topic:

- "mmchenclosure command" on page 146.

## Location

`/usr/lpp/mmfs/bin`

# mmlsfirmware command

Displays the current firmware level of storage components.

## Synopsis

```
mmlsfirmware [ --type {storage-enclosure | drive | host-adapter} ]
             [ --serial-number SerialNumber ] [--not-latest]
             [ -N {Node[,Node...] | NodeFile | NodeClass} ]
```

## Availability

Available with the Elastic Storage Server.

## Description

Use the **mmlsfirmware** command to display the current firmware levels of storage components. By default, the **mmlsfirmware** command collects information from the node on which it is issued and displays the firmware levels for all component types.

An asterisk (*) prepended to the available firmware value indicates that newer firmware is available. In some cases, the available firmware level might have an asterisk even though it matches the current firmware level. This indicates that a subcomponent requires updating.

## Parameters

**--type { storage-enclosure | drive | host-adapter }**
Displays the firmware levels for a specific component type.

**--serial-number** *SerialNumber*
Displays the firmware levels for the storage enclosure with the specified serial number.

**--not-latest**
Displays the components for which there are available updates or for which there are no available updates.

**-N {***Node***[,***Node***...] |** *NodeFile* **|** *NodeClass***}**
Specifies the nodes from which to gather firmware data.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

If you are a root user, the node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. To display the firmware level of all drives, storage enclosures, and host adapters on the current node, issue this command:

   ```
   mmlsfirmware
   ```

   The system displays information similar to this:

```
                       enclosure      firmware   available
type       product id  serial number  level      firmware   location
----       ----------  -------------  --------   --------   --------
enclosure  DCS3700     0123456789AB   039A,039A  039A       Rack BB1RACK U01-04
enclosure  DCS3700     SV11812206     039A,039A  039A       Rack BB1RACK U13-16
enclosure  DCS3700     SV12616296     039A,039A  039A       Rack BB1RACK U05-08
enclosure  DCS3700     SV13306129     039A,039A  039A       Rack BB1RACK U17-20

                    enclosure      firmware  available
type   product id   serial number  level     firmware  location
----   ----------   -------------  --------  --------  --------
drive  ST2000NM0001 0123456789AB   BC4B      BC4B      Rack BB1RACK U01-04, Enclosure 1818-80E-0123456789AB
                                                       Drawer 1 Slot 10
drive  ST2000NM0001 0123456789AB   BC4B      BC4B      Rack BB1RACK U01-04, Enclosure 1818-80E-0123456789AB
                                                       Drawer 1 Slot 11
drive  ST2000NM0001 0123456789AB   BC4B      BC4B      Rack BB1RACK U01-04, Enclosure 1818-80E-0123456789AB
                                                       Drawer 1 Slot 12
.
.
.
         product   firmware      available     bios          available     UEFI         available
type     id        level         firmware      level         bios          level        UEFI          location
----     -------   --------      ---------     --------      ---------     --------     ---------     --------
adapter  0x3070    20.00.04.00   20.00.04.00   07.39.00.00   07.39.00.00   07.27.01.01  07.27.01.01   c55f04n03 0 00:52:00:00
adapter  0x3070    20.00.04.00   20.00.04.00   07.39.00.00   07.39.00.00   07.27.01.01  07.27.01.01   c55f04n03 1 00:54:00:00
adapter  0x3070    20.00.00.00   *20.00.04.00  07.35.00.00   *07.39.00.00  07.21.01.00  *07.27.01.01  c55f04n03 2 00:03:00:00
adapter  0x3070    20.00.04.00   20.00.04.00   07.39.00.00   07.39.00.00   07.21.01.00  *07.27.01.01  c55f04n03 3 00:05:00:00
adapter  0x3070    20.00.04.00   20.00.04.00   07.39.00.00   07.39.00.00   07.22.04.03  *07.27.01.01  c55f04n03 4 00:03:00:00
adapter  0x3070    20.00.04.00   20.00.04.00   07.39.00.00   07.39.00.00   07.22.04.03  *07.27.01.01  c55f04n03 5 00:05:00:00
```

**Note:** For adapters, the asterisk (*), which is prepended to the `available firmware`, `available bios`, and `available UEFI` values, indicates that newer firmware is available.

2. To display the components on the current node that are not at the latest available firmware levels, issue this command:

   ```
   mmlsfirmware --not-latest
   ```

   The system displays information similar to this:

```
       product        enclosure     firmware available
type   id             serial number level    firmware     location
----   ----------     ------------- -------- ---------    --------
drive  SDLKOEDM-200GL 0123456789AB  HD33     not_available Rack BB1RACK U01-04, Enclosure 1818-80E-0123456789AB Drawer 1 Slot 3
drive  SDLKOEDM-200GL 0123456789AB  HD33     not_available Rack BB1RACK U01-04, Enclosure 1818-80E-0123456789AB Drawer 5 Slot 12
.
.
.
         product   firmware     available     bios         available     UEFI         available
type     id        level        firmware      level        bios          level        UEFI          location
----     -------   --------     ---------     --------     ---------     --------     ---------     --------
adapter  0x3070    20.00.00.00  *20.00.04.00  07.35.00.00  *07.39.00.00  07.21.01.00  *07.27.01.01  c55f04n03 2 00:03:00:00
adapter  0x3070    20.00.04.00  20.00.04.00   07.39.00.00  07.39.00.00   07.21.01.00  *07.27.01.01  c55f04n03 3 00:05:00:00
adapter  0x3070    20.00.04.00  20.00.04.00   07.39.00.00  07.39.00.00   07.22.04.03  *07.27.01.01  c55f04n03 4 00:03:00:00
adapter  0x3070    20.00.04.00  20.00.04.00   07.39.00.00  07.39.00.00   07.22.04.03  *07.27.01.01  c55f04n03 5 00:05:00:00
```

   **Note:**

   a. For adapters, the asterisk (*), which is prepended to the `available firmware`, `available bios`, and `available UEFI` values, indicates that newer firmware is available.

   b. Firmware was not available for the **mmchfirmware** command to load for the two drives in this example. This would be an unexpected situation.

3. To display the firmware level of the storage enclosures on the current node, issue this command:

   ```
   mmlsfirmware --type storage-enclosure
   ```

   The system displays information similar to this:

```
                          enclosure          firmware     available
type       product id     serial number      level        firmware    location
----       ----------     -------------      --------      --------   --------
enclosure  DCS3700        SV11933001         039A,039A     039A       Rack BB1RACK U01-04
enclosure  DCS3700        SV11812206         039A,039A     039A       Rack BB1RACK U13-16
enclosure  DCS3700        SV12616296         039A,039A     039A       Rack BB1RACK U05-08
enclosure  DCS3700        SV13306129         039A,039A     039A       Rack BB1RACK U17-20
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:


- "mmaddcomp command" on page 130
- "mmaddcompspec command" on page 133

- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144
- "mmchfirmware command" on page 149
- "mmdelcomp command" on page 167
- "mmdelcomploc command" on page 169
- "mmdelcompspec command" on page 171
- "mmdiscovercomp command" on page 179
- "mmlscomploc command" on page 187
- "mmlscompspec command" on page 190
- "mmlsenclosure command" on page 192
- "mmsyncdisplayid command" on page 216.

## Location

`/usr/lpp/mmfs/bin`

# mmlspdisk command

Lists information for one or more IBM Spectrum Scale RAID pdisks.

## Synopsis

```
mmlspdisk {all | RecoveryGroupName [--declustered-array DeclusteredArrayName | --pdisk pdiskName]}
          [--not-in-use | --not-ok | --replace]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmlspdisk** command lists information for one or more pdisks, which can be specified in various ways.

## Parameters

**all | *RecoveryGroupName***
Specifies the recovery group for which the pdisk information is to be listed.

**all** specifies that pdisk information for all recovery groups is to be listed.

*RecoveryGroupName* specifies the name of a particular recovery group for which the pdisk information is to be listed.

**--declustered-array *DeclusteredArrayName***
Specifies the name of a declustered array for which the pdisk information is to be listed.

**--pdisk *pdiskName***
Specifies the name of a single pdisk for which the information is to be listed.

**--not-in-use**
Indicates that information is to be listed only for pdisks that are draining.

**--not-ok**
Indicates that information is to be listed only for pdisks that are not functioning correctly.

**--replace**
Indicates that information is to be listed only for pdisks in declustered arrays that are marked for replacement.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmlspdisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. The following command example shows how to display the details regarding pdisk c112d3 in recovery group 000DE37BOT:

   ```
   mmlspdisk 000DE37BOT --pdisk c112d3
   ```

   The system displays output similar to the following:

   ```
   pdisk:
      replacementPriority = 1000
      name = "e2s23"
      device = "/dev/sdbs,/dev/sdcq"
      recoveryGroup = "rg0"
      declusteredArray = "p30da_d"
      state = "ok/noData"
      capacity  = 299842404352
      freeSpace = 299573968896
      fru = "49Y1840"
      location = "SX31700361-23"
      WWN = "naa.5000C50067A91EC3"
      server = "perseus30ib.almaden.ibm.com"
      reads = 8
      writes = 5
      bytesReadInGiB = 0.002
      bytesWrittenInGiB = 0.001
      IOErrors = 0
      IOTimeouts = 0
      mediaErrors = 0
      checksumErrors = 0
      pathErrors = 0
      relativePerformance = 1.000
      dataBadness = 0.000
      rgIndex = 46
      userLocation = ""
      userCondition = "normal"
      hardware = "IBM-ESXS ST9300605SS B559 6XP5GQMP0000M338BUSD"
      hardwareType = Rotating 10000
      nPaths = 2 active (2 expected) 4 total (4 expected)
      nsdFormatVersion = NSD version 2
      paxosAreaOffset = 600122941440
      paxosAreaSize = 4194304
      logicalBlockSize = 4096
   ```

2. To show which pdisks in recovery group 000DE37BOT need replacing:

   ```
   mmlspdisk 000DE37BOT --replace
   ```

   The system displays output similar to the following:

   ```
   pdisk:
      replacementPriority = 0.98
      name = "c052d1"
      device = "/dev/rhdisk556,/dev/rhdisk460"
      recoveryGroup = "000DE37BOT"
      declusteredArray = "DA1"
      state = "dead/systemDrain/noRGD/noVCD/replace"
      .
      .
      .
   pdisk:
      replacementPriority = 0.98
      name = "c096d1"
      device = "/dev/rhdisk508,/dev/rhdisk412"
      recoveryGroup = "000DE37BOT"
      declusteredArray = "DA1"
      state = "dead/systemDrain/noRGD/noVCD/replace"
      .
      .
      .
   ```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmchpdisk command" on page 153
- "mmdelpdisk command" on page 173
- "mmlsrecoverygroup command" on page 202
- "mmlsrecoverygroupevents command" on page 211
- "mmlsvdisk command" on page 213.

## Location

`/usr/lpp/mmfs/bin`

# mmlsrecoverygroup command

Lists information about IBM Spectrum Scale RAID recovery groups.

## Synopsis

**mmlsrecoverygroup** [ *RecoveryGroupName* [**-L** [**--pdisk**] ] ]

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmlsrecoverygroup** command lists information about recovery groups. The command displays various levels of information, depending on the parameters specified.

**1. Output values for mmlsrecoverygroup**

Descriptions of the output values for the **mmlsrecoverygroup** command follow.

**recovery group**
Is the name of the recovery group.

**declustered arrays with vdisks**
Is the number of declustered arrays with vdisks in this recovery group.

**vdisks**
Is the number of vdisks in this recovery group.

**servers**
Is the server pair for the recovery group. The intended primary server is listed first, followed by the intended backup server.

**2. Output values for mmlsrecoverygroup** *RecoveryGroupName*

Descriptions of the output values for the **mmlsrecoverygroup** *RecoveryGroupName* command follow, as displayed by row, from top to bottom and left to right.

**recovery group**
Is the name of the recovery group.

**declustered arrays with vdisks**
Is the number of declustered arrays with vdisks in this recovery group.

**vdisks**
Is the number of vdisks in this recovery group.

**servers**
Is the server pair for the recovery group. The intended primary server is listed first, followed by the intended backup server.

**declustered array with vdisks**
Is the name of the declustered array.

**vdisks**
Is the number of vdisks in this declustered array.

**vdisk**
Is the name of the vdisk.

**RAID code**
Is the RAID code for this vdisk.

**declustered array**
Is the declustered array for this vdisk.

**remarks**
Indicates the special vdisk type. Only those vdisks with a dedicated function within the recovery group are indicated here: the log, log tip, log tip backup, and log reserved vdisks. This field is blank for file system NSD vdisks.

**3. Output values for mmlsrecoverygroup** *RecoveryGroupName* **-L**

Descriptions of the output values for the **mmlsrecoverygroup** *RecoveryGroupName* **-L** command follow, as displayed by row, from top to bottom and left to right.

**Recovery group section:**

**recovery group**
Is the name of the recovery group.

**declustered arrays**
Is the number of declustered arrays in this recovery group.

**vdisks**
Is the number of vdisks in this recovery group.

**pdisks**
Is the number of pdisks in this recovery group.

**format version**
Is the recovery group version.

**Declustered array section:**

**declustered array**
Is the name of the declustered array.

**needs service**
Indicates whether this declustered array needs service. A yes value means that disks need to be replaced.

**vdisks**
Is the number of vdisks in this declustered array.

**pdisks**
Is the number of pdisks in this declustered array.

**spares**
The first number of the pair is the amount of spare space that is reserved for rebuilding, expressed as an equivalent number of pdisks. This spare space is allocated equally among all of the pdisks in the declustered array.

The second number of the pair is the number of vdisk configuration data (VCD) replicas that are maintained across all of the pdisks of the declustered array. This is the internal IBM Spectrum Scale RAID metadata for the recovery group. The number of these VCD spares is set during recovery group creation and should not be changed.

**replace threshold**
Is the number of pdisks that must fail before the declustered array reports that it needs service and the pdisks are marked for required replacement.

**free space**
Is the amount of raw space in this declustered array that is unused and is available for creating vdisks. The pdisk spare space for rebuild has already been removed from this number. The size of the

vdisks that can be created using the raw free space depends on the redundancy requirements of the vdisk RAID code: a 4WayReplicated vdisk of size N uses 4N raw space; an 8+3P vdisk of size N uses 1.375N raw space.

**scrub duration**
Is the length of time in days over which the scrubbing of all vdisks in the declustered array will complete.

**background activity**

**task**
Is the task that is being performed on the declustered array.

**inactive**
Means that there are no vdisks defined or the declustered array is not currently available.

**scrub**
Means that vdisks are undergoing routine data integrity maintenance.

**rebuild-critical**
Means that vdisk tracks with no remaining redundancy are being rebuilt.

**rebuild-1r**
Means that vdisk tracks with one remaining redundancy are being rebuilt.

**rebuild-2r**
Means that vdisk tracks with two remaining redundancies are being rebuilt.

**progress**
Is the completion percentage of the current task.

**priority**
Is the priority given the current task. Critical rebuilds are given high priority; all other tasks have low priority.

**Vdisk section:**

**vdisk**
Is the name of the vdisk.

**RAID code**
Is the RAID code for this vdisk.

**declustered array**
Is the declustered array in which this vdisk is defined.

**vdisk size**
Is the usable size this vdisk.

**block size**
Is the block (track) size for this vdisk.

**checksum granularity**
Is the buffer granularity at which IBM Spectrum Scale RAID checksums are maintained for this vdisk. This value is set automatically at vdisk creation. For file system NSD vdisks, this value is 32 KiB. For log, log tip, log tip backup, and log reserved vdisks on Power Systems servers, this value is 4096.

**state**
Is the maintenance task that is being performed on this vdisk.

**ok** Means that the vdisk is being scrubbed or is waiting to be scrubbed. Only one vdisk in a DA is scrubbed at a time.

**1/3-deg**
Means that tracks with one fault from three redundancies are being rebuilt.

**2/3-deg**
Means that tracks with two faults from three redundancies are being rebuilt.

**1/2-deg**
Means that tracks with one fault from two redundancies are being rebuilt.

**critical**
Means that tracks with no remaining redundancy are being rebuilt.

**inactive**
Means that the declustered array that is associated with this vdisk is inactive.

**remarks**
Indicates the special vdisk type. Only those vdisks with a dedicated function within the recovery group are indicated here: the log, log tip, log tip backup, and log reserved vdisks. This field is blank for file system NSD vdisks.

**Fault tolerance section:**

**config data**
Is the type of internal IBM Spectrum Scale RAID recovery group metadata for which fault tolerance is being reported.

**rebuild space**
Indicates the space that is available for IBM Spectrum Scale RAID metadata relocation.

**declustered array**
Is the name of the declustered array.

**VCD spares**
Is the number of VCD spares that are defined for the declustered array.

**actual rebuild spare space**
Is the number of pdisks that are currently eligible to hold VCD spares.

**remarks**
Indicates the effect or limit the VCD spares have on fault tolerance.

**config data**
Is the type of internal IBM Spectrum Scale RAID recovery group metadata for which fault tolerance is being reported.

**rg descriptor**
Indicates the recovery group declustered array and pdisk definitions.

**system index**
Indicates the vdisk RAID stripe partition definitions.

**max disk group fault tolerance**
Shows the theoretical maximum fault tolerance for the `config data`.

**actual disk group fault tolerance**
Shows the current actual fault tolerance for the `config data`, given the current state of the recovery group, its pdisks, and its number, type, and size of vdisks.

**remarks**
Indicates whether the actual fault tolerance is limiting other fault tolerances or is limited by other fault tolerances.

**vdisk**
Is the name of the vdisk for which fault tolerance is being reported.

**max disk group fault tolerance**
Shows the theoretical maximum fault tolerance for the vdisk.

**actual disk group fault tolerance**
Shows the current actual fault tolerance for the vdisk, given the current state of the recovery group, its pdisks, and its number, type, and size of vdisks.

**remarks**
Indicates why the actual fault tolerance might be different from the maximum fault tolerance.

**Server section:**

**active recovery group server**
Is the currently-active recovery group server.

**servers**
Is the server pair for the recovery group. The intended primary server is listed first, followed by the intended backup server.

**4. Output values for mmlsrecoverygroup** *RecoveryGroupName* **-L --pdisk**

Descriptions of the output values for the **mmlsrecoverygroup** *RecoveryGroupName* **-L --pdisk** command follow, as displayed by row, from top to bottom and left to right.

**Recovery group section:**

**recovery group**
Is the name of the recovery group.

**declustered arrays**
Is the number of declustered arrays in this recovery group.

**vdisks**
Is the number of vdisks in this recovery group.

**pdisks**
Is the number of pdisks in this recovery group.

**format version**
Is the recovery group version.

**Declustered array section:**

**declustered array**
Is the name of the declustered array.

**needs service**
Indicates whether this declustered array needs service. A yes value means that disks need to be replaced.

**vdisks**
Is the number of vdisks in this declustered array.

**pdisks**
Is the number of pdisks in this declustered array.

**spares**
The first number of the pair is the amount of spare space that is reserved for rebuilding, expressed as an equivalent number of pdisks. This spare space is allocated equally among all of the pdisks in the declustered array.

The second number of the pair is the number of VCD replicas that are maintained across all of the pdisks of the declustered array. This is the internal IBM Spectrum Scale RAID metadata for the recovery group. The number of these VCD spares is set during recovery group creation and should not be changed.

**replace threshold**
Is the number of pdisks that must fail before the declustered array reports that it needs service and the pdisks are marked for required replacement.

**free space**
Is the amount of raw space in this declustered array that is unused and is available for creating vdisks. The pdisk spare space for rebuilding has already been removed from this number. The size of the vdisks that can be created using the raw free space depends on the redundancy requirements of the vdisk RAID code: a 4WayReplicated vdisk of size N uses 4N raw space; an 8+3P vdisk of size N uses 1.375N raw space.

**scrub duration**
Is the length of time in days over which the scrubbing of all vdisks in the declustered array will complete.

**background activity**

**task**
Is the task that is being performed on the declustered array.

**inactive**
Means that there are no vdisks defined or the declustered array is not currently available.

**scrub**
Means that vdisks are undergoing routine data integrity maintenance.

**rebuild-critical**
Means that vdisk tracks with no remaining redundancy are being rebuilt.

**rebuild-1r**
Means that vdisk tracks with only one remaining redundancy are being rebuilt.

**rebuild-2r**
Means that vdisk tracks with two remaining redundancies are being rebuilt.

**progress**
Is the completion percentage of the current task.

**priority**
Is the priority given the current task. Critical rebuilds are given high priority; all other tasks have low priority.

**Pdisk section:**

**pdisk**
Is the name of the pdisk.

**n. active, total paths**
Indicates the number of active (in-use) and total block device paths to the pdisk. The total paths include the paths that are available on the standby server.

**declustered array**
Is the name of the declustered array to which the pdisk belongs.

**free space**
Is the amount of raw free space that is available on the pdisk.

**Note:** A pdisk that has been taken out of service and completely drained of data will show its entire capacity as free space, even though that capacity is not available for use.

**user condition**
Is the condition of the pdisk from the system administrator's persepective. A "normal" condition requires no attention, while "replaceable" means that the pdisk might be, but is not necessarily required to be, physically replaced.

**state, remarks**
Is the state of the pdisk. For a description of pdisk states, see the topic "Pdisk states" on page 37 in *IBM Spectrum Scale RAID: Administration*.

**Server section:**

**active recovery group server**
Is the currently-active recovery group server.

**servers**
Is the server pair for the recovery group. The intended primary server is listed first, followed by the intended backup server.

## Parameters

*RecoveryGroupName*
Specifies the recovery group for which the information is being requested. If no other parameters are specified, the command displays only the information that can be found in the GPFS cluster configuration data.

**-L** Displays more detailed runtime information for the specified recovery group.

**--pdisk**
Indicates that pdisk information is to be listed for the specified recovery group.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmlsrecoverygroup** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. The following command example shows how to list all the recovery groups in the GPFS cluster:

   ```
   mmlsrecoverygroup
   ```

   The system displays output similar to the following:

   ```
                     declustered
                     arrays with
   recovery group       vdisks     vdisks  servers
   ------------------  -----------  ------  -------
   BB1RGL                      4       8  c45f01n01-ib0.gpfs.net,c45f01n02-ib0.gpfs.net
   BB1RGR                      3       7  c45f01n02-ib0.gpfs.net,c45f01n01-ib0.gpfs.net
   ```

2. The following command example shows how to list the basic non-runtime information for recovery group 000DE37BOT:

   ```
   mmlsrecoverygroup 000DE37BOT
   ```

   The system displays output similar to the following:

   ```
                     declustered
                     arrays with
   recovery group       vdisks     vdisks  servers
   ------------------  -----------  ------  -------
   ```

```
BB1RGL                       4       8  c45f01n01-ib0.gpfs.net,c45f01n02-ib0.gpfs.net

declustered array
    with vdisks     vdisks
------------------  ------
DA1                      3
DA2                      3
NVR                      1
SSD                      1

                                        declustered
vdisk               RAID code              array     remarks
------------------  ------------------  -----------  -------
BB1RGLDATA1         8+3p                DA1
BB1RGLDATA2         8+3p                DA2
BB1RGLMETA1         4WayReplication     DA1
BB1RGLMETA2         4WayReplication     DA2
lhome_BB1RGL        4WayReplication     DA1          log
ltbackup_BB1RGL     Unreplicated        SSD
ltip_BB1RGL         2WayReplication     NVR
reserved1_BB1RGL    4WayReplication     DA2
```

3. The following command example shows how to display the runtime status of recovery group BB1RGL:

   ```
   mmlsrecoverygroup BB1RGL -L
   ```

   The system displays output similar to the following:

```
                 declustered
recovery group      arrays     vdisks  pdisks  format version
-----------------  -----------  ------  ------  --------------
BB1RGL                     4       8     119  4.1.0.1

declustered   needs                            replace              scrub       background activity
   array     service  vdisks  pdisks  spares  threshold  free space  duration  task   progress  priority
-----------  -------  ------  ------  ------  ---------  ----------  --------  -------------------------
SSD          no            1       1     0,0          1     186 GiB   14 days  scrub        8%  low
NVR          no            1       2     0,0          1    3648 MiB   14 days  scrub        8%  low
DA1          no            3      58    2,31          2      50 TiB   14 days  scrub        7%  low
DA2          no            3      58    2,31          2      50 TiB   14 days  scrub        7%  low

                                        declustered                          checksum
vdisk               RAID code              array     vdisk size  block size  granularity  state remarks
------------------  ------------------  -----------  ----------  ----------  -----------  ----- -------
ltip_BB1RGL         2WayReplication     NVR             48 MiB      2 MiB       512      ok    logTip
ltbackup_BB1RGL     Unreplicated        SSD             48 MiB      2 MiB       512      ok    logTipBackup
lhome_BB1RGL        4WayReplication     DA1             20 GiB      2 MiB       512      ok    log
reserved1_BB1RGL    4WayReplication     DA2             20 GiB      2 MiB       512      ok    logReserved
BB1RGLMETA1         4WayReplication     DA1            750 GiB      1 MiB     32 KiB     ok
BB1RGLDATA1         8+3p                DA1             35 TiB     16 MiB     32 KiB     ok
BB1RGLMETA2         4WayReplication     DA2            750 GiB      1 MiB     32 KiB     ok
BB1RGLDATA2         8+3p                DA2             35 TiB     16 MiB     32 KiB     ok

config data         declustered array   VCD spares     actual rebuild spare space         remarks
------------------  ------------------  -------------  ---------------------------------  ----------------
rebuild space       DA1                 31             35 pdisk
rebuild space       DA2                 31             35 pdisk

config data         max disk group fault tolerance     actual disk group fault tolerance  remarks
------------------  ---------------------------------  ---------------------------------  ----------------
rg descriptor       1 enclosure + 1 drawer             1 enclosure + 1 drawer             limiting fault tolerance
system index        2 enclosure                        1 enclosure + 1 drawer             limited by rg descriptor

vdisk               max disk group fault tolerance     actual disk group fault tolerance  remarks
------------------  ---------------------------------  ---------------------------------  ----------------
ltip_BB1RGL         1 pdisk                            1 pdisk
ltbackup_BB1RGL     0 pdisk                            0 pdisk
lhome_BB1RGL        3 enclosure                        1 enclosure + 1 drawer             limited by rg descriptor
reserved1_BB1RGL    3 enclosure                        1 enclosure + 1 drawer             limited by rg descriptor
BB1RGLMETA1         3 enclosure                        1 enclosure + 1 drawer             limited by rg descriptor
BB1RGLDATA1         1 enclosure                        1 enclosure
BB1RGLMETA2         3 enclosure                        1 enclosure + 1 drawer             limited by rg descriptor
BB1RGLDATA2         1 enclosure                        1 enclosure

active recovery group server                    servers
----------------------------------------------  -------
c45f01n01-ib0.gpfs.net                          c45f01n01-ib0.gpfs.net,c45f01n02-ib0.gpfs.net
```

   For more information, see the following *IBM Spectrum Scale RAID: Administration* topic: "Determining pdisk-group fault-tolerance" on page 45.

4. The following example shows how to include pdisk information for BB1RGL:

   ```
   mmlsrecoverygroup BB1RGL -L --pdisk
   ```

The system displays output similar to the following:

```
                  declustered
recovery group      arrays     vdisks  pdisks  format version
-----------------  -----------  ------  ------  --------------
BB1RGL                      4       8     119  4.1.0.1

declustered   needs                            replace               scrub       background activity
   array     service  vdisks  pdisks  spares  threshold  free space  duration  task   progress  priority
-----------  -------  ------  ------  ------  ---------  ----------  --------  -------------------------
SSD          no             1       1     0,0          1     186 GiB   14 days  scrub        8%  low
NVR          no             1       2     0,0          1    3648 MiB   14 days  scrub        8%  low
DA1          no             3      58    2,31          2      50 TiB   14 days  scrub        7%  low
DA2          no             3      58    2,31          2      50 TiB   14 days  scrub        7%  low

                   n. active,   declustered                user     state,
pdisk              total paths     array     free space   condition  remarks
-----------------  -----------  -----------  ----------  ----------- -------
e1d1s01              2,  4      DA1             957 GiB  normal      ok
e1d1s02              2,  4      DA1             957 GiB  normal      ok
e1d1s03ssd           2,  4      SSD             186 GiB  normal      ok
e1d1s04              2,  4      DA2             955 GiB  normal      ok
...

active recovery group server                    servers
----------------------------------------------  -------
c45f01n01-ib0.gpfs.net                          c45f01n01-ib0.gpfs.net,c45f01n02-ib0.gpfs.net
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmchrecoverygroup command" on page 156
- "mmcrrecoverygroup command" on page 159
- "mmdelrecoverygroup command" on page 175
- "mmlspdisk command" on page 199
- "mmlsrecoverygroupevents command" on page 211
- "mmlsvdisk command" on page 213.

## Location

`/usr/lpp/mmfs/bin`

# mmlsrecoverygroupevents command

Displays the IBM Spectrum Scale RAID recovery group event log.

## Synopsis

```
mmlsrecoverygroupevents RecoveryGroupName [-T] [--days Days]
                        [--long-term Codes] [--short-term Codes]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmlsrecoverygroupevents** command displays the recovery group event log, internally divided into the following two logs:

**short-term log**
: Contains more detail than the long-term log, but due to space limitations may not extend far back in time

**long-term log**
: Contains only brief summaries of important events and therefore extends much further back in time

Both logs use the following severity codes:

**C** Commands (or configuration)

These messages record a history of commands that changed the specified recovery group.

**E** Errors

**W** Warnings

**I** Informational messages

**D** Details

By default, **mmlsrecoverygroupevents** displays both long-term and short-term logs merged together in order of message time stamp. Given the **--long-term** option, it displays only the requested severities from the long-term log. Given the **--short-term** option, it displays only the requested severities from the short-term log. Given both **--long-term** and **--short-term** options, it displays the requested severities from each log, merged by time stamp.

**Note:** The recovery group must be active to run this command.

## Parameters

*RecoveryGroupName*
: Specifies the name of the recovery group for which the event log is to be displayed.

**-T** Indicates that the time is to be shown in decimal format.

**--days** *Days*
: Specifies the number of days for which events are to be displayed.

For example, **--days 3** specifies that only the events of the last three days are to be displayed.

**--long-term** *Codes*
: Specifies that only the indicated severity or severities from the long-term log are to be displayed. You can specify any combination of the severity codes listed in "Description."

For example, **--long-term EW** specifies that only errors and warnings are to be displayed.

**--short-term** *Codes*
Specifies that only the indicated severity or severities from the short-term log are to be displayed. You can specify any combination of the severity codes listed in "Description" on page 211.

For example, **--short-term EW** specifies that only errors and warnings are to be displayed.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmlsrecoverygroupevents** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

The following command example shows how to print the event logs of recovery group `000DE37BOT`:

```
mmlsrecoverygroupevents 000DE37BOT
```

The system displays output similar to the following:

```
Mon May 23 12:17:36.916 2011 c08ap01 ST [I] Start scrubbing tracks of 000DE37BOTDA4META.
Mon May 23 12:17:36.914 2011 c08ap01 ST [I] Finish rebalance of DA DA4 in RG 000DE37BOT.
Mon May 23 12:13:00.033 2011 c08ap01 ST [D] Pdisk c109d4 of RG 000DE37BOT state changed from noRGD to ok.
Mon May 23 12:13:00.010 2011 c08ap01 ST [D] Pdisk c109d4 of RG 000DE37BOT state changed from noRGD/noVCD to noRGD.
Mon May 23 12:11:29.676 2011 c08ap01 ST [D] Pdisk c109d4 of RG 000DE37BOT state changed from noRGD/noVCD/noData to noRGD/noVCD.
Mon May 23 12:11:29.672 2011 c08ap01 ST [I] Start rebalance of DA DA4 in RG 000DE37BOT.
Mon May 23 12:11:29.469 2011 c08ap01 ST [I] Finished repairing metadata in RG 000DE37BOT.
Mon May 23 12:11:29.409 2011 c08ap01 ST [I] Start repairing metadata in RG 000DE37BOT.
Mon May 23 12:11:29.404 2011 c08ap01 ST [I] Abort scrubbing tracks of 000DE37BOTDA4META.
Mon May 23 12:11:29.404 2011 c08ap01 ST [D] Pdisk c109d4 of RG 000DE37BOT state changed from missing/systemDrain/noRGD/noVCD/noData/noPath
                                             to noRGD/noVCD/noData.
Mon May 23 12:11:29.401 2011 c08ap01 ST [D] Pdisk c109d4 of RG 000DE37BOT: path index 0 (/dev/rhdisk131): up.
Mon May 23 12:11:29.393 2011 c08ap01 ST [I] Path /dev/rhdisk131 of pdisk c109d4 reenabled.
Mon May 23 12:09:49.004 2011 c08ap01 ST [I] Start scrubbing tracks of 000DE37BOTDA4META.
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddpdisk command" on page 136
- "mmchrecoverygroup command" on page 156
- "mmchcarrier command" on page 138
- "mmdelpdisk command" on page 173
- "mmdelvdisk command" on page 177
- "mmlspdisk command" on page 199
- "mmlsrecoverygroup command" on page 202
- "mmlsvdisk command" on page 213.

## Location

`/usr/lpp/mmfs/bin`

# mmlsvdisk command

Lists information for one or more IBM Spectrum Scale RAID vdisks.

## Synopsis

**mmlsvdisk** [ **--vdisk** "*VdiskName*[;*VdiskName*...]" | **--non-nsd** ]

or

**mmlsvdisk --recovery-group** *RecoveryGroupName* [ **--declustered-array** *DeclusteredArrayName* ]

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmlsvdisk** command lists information for one or more vdisks, specified various ways. Unless the **--recovery-group** option is specified, the information comes from the GPFS cluster configuration data.

## Parameters

**--vdisk "***VdiskName***[;***VdiskName***...]"**
Specifies the name or names of the vdisk or vdisks for which the information is to be listed.

**--non-nsd**
Indicates that information is to be listed for the vdisks that are not associated with NSDs.

**--recovery-group** *RecoveryGroupName*
Specifies the name of the recovery group.

**Note:** The specified recovery group must be active to run this command.

**--declustered-array** *DeclusteredArrayName*
Specifies the name of the declustered array.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **mmlsvdisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. The following command example shows how to list all vdisks in the GPFS cluster:

   ```
   mmlsvdisk
   ```

   The system displays output similar to the following:

   ```
                                                      declustered  block size
    vdisk name          RAID code        recovery group         array       in KiB    remarks
    ------------------  ---------------  ------------------  -----------  ----------  -------
   ```

```
000DE37BOTDA1DATA   8+3p             000DE37BOT          DA1                8192
000DE37BOTDA1META   4WayReplication  000DE37BOT          DA1                1024
000DE37BOTDA2DATA   8+3p             000DE37BOT          DA2                8192
000DE37BOTDA2META   4WayReplication  000DE37BOT          DA2                1024
000DE37BOTDA3DATA   8+3p             000DE37BOT          DA3                8192
000DE37BOTDA3META   4WayReplication  000DE37BOT          DA3                1024
000DE37BOTDA4DATA   8+3p             000DE37BOT          DA4                8192
000DE37BOTDA4META   4WayReplication  000DE37BOT          DA4                1024
000DE37BOTLOG       3WayReplication  000DE37BOT          LOG                1024  log
000DE37TOPDA1DATA   8+3p             000DE37TOP          DA1                8192
000DE37TOPDA1META   4WayReplication  000DE37TOP          DA1                1024
000DE37TOPDA2DATA   8+3p             000DE37TOP          DA2                8192
000DE37TOPDA2META   4WayReplication  000DE37TOP          DA2                1024
000DE37TOPDA3DATA   8+3p             000DE37TOP          DA3                8192
000DE37TOPDA3META   4WayReplication  000DE37TOP          DA3                1024
000DE37TOPDA4DATA   8+3p             000DE37TOP          DA4                8192
000DE37TOPDA4META   4WayReplication  000DE37TOP          DA4                1024
000DE37TOPLOG       3WayReplication  000DE37TOP          LOG                1024  log
```

2. The following command example shows how to list only those vdisks in the cluster that do not have NSDs defined on them:

   ```
   # mmlsvdisk --non-nsd
   ```

   The system displays output similar to the following:

```
                                                        declustered  block size
vdisk name          RAID code        recovery group       array       in KiB    remarks
------------------  ---------------  ------------------  -----------  ----------  -------
000DE37BOTLOG       3WayReplication  000DE37BOT          LOG                1024  log
000DE37TOPLOG       3WayReplication  000DE37TOP          LOG                1024  log
```

3. The following command example shows how to see complete information about the vdisks in declustered array DA1 of recovery group 000DE37TOP:

   ```
   mmlsvdisk --recovery-group 000DE37TOP --declustered-array DA1
   ```

   The system displays output similar to the following:

```
vdisk:
  name = "000DE37TOPDA1META"
  raidCode = "4WayReplication"
  recoveryGroup = "000DE37TOP"
  declusteredArray = "DA1"
  blockSizeInKib = 1024
  size = "250 GiB"
  state = "ok"
  remarks = ""

vdisk:
  name = "000DE37TOPDA1DATA"
  raidCode = "8+3p"
  recoveryGroup = "000DE37TOP"
  declusteredArray = "DA1"
  blockSizeInKib = 16384
  size = "17 TiB"
  state = "ok"
  remarks = ""
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmcrvdisk command" on page 162
- "mmdelvdisk command" on page 177
- "mmlspdisk command" on page 199
- "mmlsrecoverygroup command" on page 202
- "mmlsrecoverygroupevents command" on page 211.

## Location

/usr/lpp/mmfs/bin

# mmsyncdisplayid command

Synchronizes enclosure display IDs with the GPFS cluster configuration.

## Synopsis

```
mmsyncdisplayid Enclosure | all
[ -N { nodeName[,nodeName...] | nodeFile | nodeClass } ]
[--dry-run] [ --log-level logLevel ]
```

## Availability

Available with the Elastic Storage Server.

## Description

The **mmsyncdisplayid** command synchronizes enclosure display IDs with the GPFS cluster configuration.

## Parameters

*Enclosure* | **all**
: Specifies the name, serial number, or component ID of the enclosure. Or, specify **all** for all enclosures.

**-N {** *nodeName*[**,***nodeName*...] | *nodeFile* | *nodeClass* **}**
: Specifies the nodes that apply to the command. If **-N** is not specified, the target is all nodes in the cluster that have a server license.

**--dry-run**
: Indicates that the changes resulting from the command are not to be recorded in the configuration file.

**--log-level** *logLevel*
: Specifies the log level. Valid values are: critical, error, warning, info, detail, debug, and none.

## Exit status

**0** Successful completion.

**nonzero**
: A failure has occurred.

## Security

You must have root authority to run the **mmsyncdisplayid** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

To set the display ID to match the location of an enclosure as defined by the component configuration, and to see the result, enter the following commands:

```
mmsyncdisplayid all
mmlscomp --type storageenclosure
```

The system displays output similar to this:

```
    Storage Enclosure Components

Comp ID  Part Number  Serial Number  Name                Display ID
-------  -----------  -------------  ------------------  ----------
      1  1818-80E     SV12345001     1818-80E-SV12345001  21
      2  1818-80E     SV12345007     1818-80E-SV12345007  01
      3  1818-80E     SV12345003     1818-80E-SV12345003  13
      4  1818-80E     SV12345005     1818-80E-SV12345005  05
      5  1818-80E     SV12345004     1818-80E-SV12345004  37
      6  1818-80E     SV12345002     1818-80E-SV12345002  25
      7  1818-80E     SV12345008     1818-80E-SV12345008  17
      8  1818-80E     SV12345006     1818-80E-SV12345006  33
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmaddcomp command" on page 130
- "mmaddcompspec command" on page 133
- "mmchcomp command" on page 141
- "mmchcomploc command" on page 144
- "mmchenclosure command" on page 146
- "mmchfirmware command" on page 149
- "mmdelcomp command" on page 167
- "mmdelcomploc command" on page 169
- "mmdelcompspec command" on page 171
- "mmdiscovercomp command" on page 179
- "mmlscomp command" on page 184
- "mmlscomploc command" on page 187
- "mmlscompspec command" on page 190
- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196.

## Location

`/usr/lpp/mmfs/bin`

# mmvdisk command

Manages IBM Spectrum Scale RAID node classes, servers, recovery groups, vdisk sets, file systems, pdisks, and vdisks.

## Synopsis

**mmvdisk nodeclass VERB** *[PARAMETERS]*

**mmvdisk server VERB** *[PARAMETERS]*

**mmvdisk recoverygroup VERB** *[PARAMETERS]*

**mmvdisk vdiskset VERB** *[PARAMETERS]*

**mmvdisk filesystem VERB** *[PARAMETERS]*

**mmvdisk pdisk VERB** *[PARAMETERS]*

**mmvdisk vdisk VERB** *[PARAMETERS]*

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mmvdisk** command is an integrated command suite that simplifies IBM Spectrum Scale RAID and enforces consistent best practices, for:

- Servers
- Recovery groups
- Vdisk NSDs
- File system configuration

Use the **mmvdisk** command to manage new IBM Spectrum Scale RAID installations from the start. Existing IBM Spectrum Scale RAID installations can be converted to enable **mmvdisk** command management. All existing recovery groups that are in a cluster but not part of **mmvdisk** command management must be converted to **mmvdisk**command management before you can use the **mmvdisk** command to create recovery groups in the cluster.

The **mmvdisk** command and the legacy IBM Spectrum Scale RAID administration commands are not compatible. For example, if you manage a recovery group by using the **mmvdisk** command, you must specify `mmvdisk pdisk replace` to replace a pdisk - you cannot specify **mmchcarrier** against a recovery group that is created by using the **mmvdisk** command. You cannot specify the **mmvdisk** to replace a pdisk against an existing recovery group that is not converted to **mmvdisk** command management (which means that the **mmchcarrier** command must be used for that unconverted recovery group).

The **mmvdisk** command structures IBM Spectrum Scale RAID around node classes, servers, recovery groups, vdisk sets, and file systems.

An **mmvdisk** node class contains the set of servers that is responsible for a recovery group pair. The IBM Spectrum Scale RAID configuration parameters for the servers are maintained under the node class, and the two recovery groups in the pair are also associated with the node class.

File system vdisk NSDs are managed collectively as members of vdisk sets. A vdisk set is a collection of identical vdisk NSDs from one or more recovery groups. Each recovery group included in a vdisk set contributes one member vdisk NSD. A vdisk set is managed as a unit, which means that member vdisk NSDs of a vdisk set must belong to the same **mmvdisk** file system.

An **mmvdisk** file system is an IBM Spectrum Scale file system that is constructed from one or more vdisk sets.

In addition to the management structure provided by node classes, servers, recovery groups, vdisk sets, and file systems, **mmvdisk** also provides an interface to list and replace recovery group pdisks and to list individual vdisks:

- The **mmvdisk nodeclass** command manages **mmvdisk** recovery group server node classes.
- The **mmvdisk server** command manages **mmvdisk** recovery group servers.
- The **mmvdisk recoverygroup** command manages **mmvdisk** recovery groups.
- The **mmvdisk vdiskset** command manages **mmvdisk** vdisk sets.
- The **mmvdisk filesystem** command manages **mmvdisk** file systems.
- The **mmvdisk pdisk** command manages **mmvdisk** recovery group pdisks.
- The **mmvdisk vdisk** command lists individual recovery group vdisks.

## Parameters

An **mmvdisk** command has the form:

```
mmvdisk Noun Verb [Parameters]
```

For example, if you specify `mmvdisk nodeclass list` without any parameters, all **mmvdisk** node classes in the cluster are displayed.

The nouns that **mmvdisk** recognizes are:

**nodeclass**
: The recovery group server node classes.

**server** The recovery group servers within node classes.

**recoverygroup**
: The recovery group.

**vdiskset**
: The vdisk sets across the recovery groups.

**filesystem**
: IBM Spectrum Scale file systems that are built from vdisk sets.

**pdisk** The recovery group pdisks that are for listing and maintenance.

**vdisk** The individual vdisks that are for listing only.

The verbs are functions to be performed specific to a noun, and the verb choices always include list. Other possible verbs include create, add, delete, and change.

After the noun and the verb, more parameters might be required or optional, might occur in any order, and are always introduced with option flags. Most of option flags are of the form **option-name** and might or might not take following arguments.

Some common traditional IBM Spectrum Scale RAID single-dash or single-character option flags are included:

**-N** The traditional node specification.

**-v** The traditional verification option to make sure that objects are not already in use.

**-p** The traditional permanently damaged option that deletes inaccessible objects.

**-L** The traditional long listing of information.

**-Y** Provides colon-delimited raw output.

For convenience, some parameters can be abbreviated. For example:

- You can use `mmvdisk nc` in place of `mmvdisk nodeclass`.
- You can use `mmvdisk rg` in place of `mmvdisk recoverygroup`.
- You can use `mmvdisk vs` in place of `mmvdisk vdiskset`.
- You can use `mmvdisk fs` in place of `mmvdisk filesystem`.

Some common option flags can also be abbreviated:

- You can use `--nc` in place of `--node-class`.
- You can use `--rg` in place of `--recovery-group`.
- You can use `--da` in place of `--declustered-array`.
- You can use `--vs` in place of `--vdisk-set`.
- You can use `--fs` in place of `--file-system`.
- You can use `--bs` in place of `--block-size`.
- You can use `--ss` in place of `--set-size`.
- You can use `--usage` in place of `--nsd-usage`.
- You can use `--pool` in place of `--storage-pool`.
- You can use `--ft` in place of `--fault-tolerance`.

Depending on the context, when it is possible to list multiple comma-separated objects as the argument to an option flag, you ca use the keyword **all** as shorthand for listing all objects of a type. For example, `--vs all` can be used to specify all vdisk sets. The individual command usage statements indicate this with `-vdisk set {all | VsName[,VsName...]}`.

If an **mmvdisk** command is going to delete (or undefine for vdisk sets) an object permanently, the command warns you by prompting you for a confirmation. In these cases, the prompt can be bypassed and answered in the affirmative by using the **--confirm** option.

The **mmvdisk** command also warns you and prompts you for confirmation before you perform potentially long-running commands. In this case with long-running commands, there is no option to bypass the prompt.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mmvdisk filesystem** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. To see all **mmvdisk**node classes in the cluster:

   ```
   mmvdisk nodeclass list
   ```

   The system displays output similar to the following:

```
node class  recovery groups
----------  --------------
ESS01       ESS01L, ESS01R
ESS02       ESS02L, ESS02R
```

2. To see all recovery group servers in the cluster:

   ```
   mmvdisk server list
   ```

   The system displays output similar to the following:

```
 node
number  server                          node class  recovery groups  remarks
------  ------------------------------  ----------  ---------------  -------
     1  ess01io1.gpfs.net               ESS01       ESS01L, ESS01R
     2  ess01io2.gpfs.net               ESS01       ESS01L, ESS01R
     3  ess02io1.gpfs.net               ESS02       ESS02L, ESS02R
     4  ess02io2.gpfs.net               ESS02       ESS02L, ESS02R
```

3. To see all recovery groups in the cluster:

   ```
   mmvdisk recoverygroup list
   ```

   The system displays output similar to the following:

```
                                                        needs    user
recovery group  active   current or master server      service  vdisks  remarks
--------------  -------  -------------------------------  -------  ------  -------
ESS01L          yes      ess01io1.gpfs.net                no            2
ESS01R          yes      ess01io2.gpfs.net                no            2
ESS02L          yes      ess02io1.gpfs.net                no            2
ESS02R          yes      ess02io2.gpfs.net                no            2
```

4. To see all vdisk sets in the cluster:

   ```
   mmvdisk vdiskset list
   ```

   The system displays output similar to the following:

```
vdisk set         created  file system  recovery groups
---------------  -------  -----------  ---------------
VS1               yes      FS1          ESS01L, ESS01R, ESS02L, ESS02R
VS2               yes      FS2          ESS01L, ESS01R, ESS02L, ESS02R
```

5. To see all file systems in the cluster:

   ```
   mmvdisk filesystem list
   ```

   The system displays output similar to the following:

```
file system  vdisk sets
-----------  ----------
FS1          VS1
FS2          VS2
```

## See also

- "mmvdisk nodeclass command" on page 222
- "mmvdisk server command" on page 225
- "mmvdisk recoverygroup command" on page 229
- "mmvdisk filesystem command" on page 238
- "mmvdisk vdiskset command" on page 243
- "mmvdisk vdisk command" on page 250
- "mmvdisk pdisk command" on page 253

## Location

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

# mmvdisk nodeclass command

Manages recovery group server node classes for IBM Spectrum Scale RAID.

## Synopsis

```
mmvdisk nodeclass create --node-class NcName
                         -N Node[,Node...]
```

or

```
mmvdisk nodeclass add --node-class NcName
                      -N  Node[,Node...]
```

or

```
mmvdisk nodeclass delete --node-class NcName
                         -N {Node[,Node...]
```

or

```
mmvdisk nodeclass delete --node-class NcName
                         [--confirm]
```

or

```
mmvdisk nodeclass list [--node-class {all | NcName[,NcName...]}]
                       [-Y]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

Use the **mmvdisk nodeclass** command to manage recovery group server node classes.

An **mmvdisk** node class is an IBM Spectrum Scale node class that can be created, modified, or deleted by using the **mmvdisk nodeclass** command. Each IBM Spectrum Scale RAID recovery group that is managed by using **mmvdisk** must be associated with a node class that contains only the servers for that recovery group.

You can establish an association between a node class and a recovery group in two ways:

1. Use the `mmvdisk nodeclass create` command to create a node class that contains the intended servers for the recovery group. Use the `mmvdisk recoverygroup create` command to create a recovery group that uses the node class.
2. Use the `mmvdisk recoverygroup convert` command to convert two paired recovery groups and their two servers to **mmvdisk** management. For example:
   - An existing node class that contains the two servers can be converted to the **mmvdisk** node class for the recovery group pair.
   - A new node class name can be specified that is created by using the two servers for the recovery group pair.

You can use the `mmvdisk nodeclass add` and `mmvdisk nodeclass delete` commands to modify or delete a node class that is not yet or is no longer associated with a recovery group. This corrects the member servers of a node class before a recovery group is created or after a recovery group is deleted.

While an **mmvdisk** node class is associated with a recovery group, changes to the node class can only be made using the `mmvdisk recoverygroup` command.

You can use the `mmvdisk nodeclass list` command to list all **mmvdisk** node classes with their associated recovery groups and to list the member servers in individual node classes.

## Parameters

**mmvdisk nodeclass create**
Creates a **mmvdisk** node class. The specified node class must not exist.

**mmvdisk nodeclass add**
Adds server nodes to an existing **mmvdisk** node class. The node class name cannot be associated with a recovery group. The server nodes cannot belong to an **mmvdisk** node class.

**mmvdisk nodeclass delete**
Deletes server nodes from an existing **mmvdisk** node class or deletes entire node classes. The node class name cannot be associated with a recovery group.

**mmvdisk nodeclass list**
Lists the **mmvdisk** node classes that are in a IBM Spectrum Scale cluster or lists the member servers that belong to individual **mmvdisk** node classes.

**--node-class** *NcName*
Specifies a single **mmvdisk** node class name to be created, deleted, added to, or deleted from.

**--node-class {***all* | *NcName[NcName...]***}**
Specifies the **mmvdisk** node class names for which the member servers are shown (when used with the `mmvdisk nodeclass list` command).

**-N** *Node[,Node...]*
Specifies the IBM Spectrum Scale cluster nodes with which to create a node class or the list of cluster nodes to be added to or deleted from an **mmvdisk** node class.

**--confirm**
Confirms deletion of an entire **mmvdisk** node class. This flag bypasses the interactive prompt that would otherwise be printed to confirm node class deletion.

**-Y** Specifies that the `mmvdisk nodeclass list` command produce colon-delimited raw output.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mmvdisk nodeclass** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages.

## Example

To create an **mmvdisk** node class that is named `ESS01` that contains nodes `server01` and `server02`:

```
mmvdisk nodeclass create --node-class ESS01 -N server01,server02
```

The system displays output similar to the following:

```
mmvdisk: Node class 'ESS01' created.
```

## Example

To list the **mmvdisk** node classes in a cluster:

```
mmvdisk nodeclass list
```

The system displays output similar to the following:

```
node class  recovery groups
----------  ---------------
ESS01       ESS01L, ESS01R
ESS02       ESS02L, ESS02R
```

## See also

- "mmvdisk command" on page 218
- "mmvdisk server command" on page 225
- "mmvdisk recoverygroup command" on page 229
- "mmvdisk filesystem command" on page 238
- "mmvdisk vdiskset command" on page 243
- "mmvdisk vdisk command" on page 250
- "mmvdisk pdisk command" on page 253

## Location

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

# mmvdisk server command

Manages **mmvdisk** recovery group servers for IBM Spectrum Scale RAID.

## Synopsis

```
mmvdisk server configure {--node-class NcName | -N Node[,Node...]}
                         [---recycle {none | one | all | Number}]
                         [--pagepool {nM | nG | n%} ]
                         [--maxblocksize {2M | 4M | 8M | 16M}]
```

or

```
mmvdisk server unconfigure {--node-class NcName | -N Node[,Node...]}
                           [---recycle {none | one | all | Number}]
```

or

```
mmvdisk server list [-Y]
```

or

```
mmvdisk server list {--node-class NcName |-N Node[,Node...] |--recovery-group RgName}
                    [[--version] | [--config] | [--disk-topology] [--fanout N]]
                    [-Y]
```

or

```
mmvdisk server list -N Node
                    --disk-topology
                    -L
                    [-Y]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

Use the `mmvdisk server` command to manage **mmvdisk** recovery group servers for IBM Spectrum Scale RAID.

Before you can create a IBM Spectrum Scale RAID recovery group:

- The **mmvdisk**node class for the intended recovery group must be configured to run IBM Spectrum Scale RAID.
- The IBM Spectrum Scale daemon must be restarted on each server in the node class.

Use the `mmvdisk server configure` command to:

- Examine the nodes in an **mmvdisk** node class.
- Verify that they have a supported IBM Spectrum Scale RAID disk topology.
- Set the appropriate configuration parameters.

You can also use the `mmvdisk server configure` command to recycle the IBM Spectrum Scale daemon on the node class to make sure that the new configuration is being used.

Use the `mmvdisk server configure` command to configure individual nodes for IBM Spectrum Scale RAID, which prepares for maintenance use cases in which servers are replaced or added to a recovery group and its associated node class.

Use the `mmvdisk server unconfigure` command to remove IBM Spectrum Scale RAID configuration from a node class from which a recovery group is either deleted or simply not created. You can also use the

mmvdisk server unconfigure command to unconfigure IBM Spectrum Scale RAID on individual nodes in maintenance use cases where servers a replaced or deleted from a recovery group and its node class.

Use the mmvdisk server list command without any options to list all the configured IBM Spectrum Scale RAID servers in the cluster. This parameter identifies all recovery group servers and all nodes that are configured to be recovery group servers, regardless of whether the servers are members of a node class. Use this parameter to help plan server maintenance and the conversion of recovery groups and servers to **mmvdisk** command management.

Use the mmvdisk server list command options to display server characteristics.

- Use node-class to display:
  - Whether the servers in a node class are active.
  - The recovery group that they serve.
- Use the -version parameter to display the IBM Spectrum Scale and operating system versions for servers.
- Use the -config parameter to display important characteristics of the IBM Spectrum Scale RAID configuration.

Use the mmvdisk server list --disk-topology command to display the supported IBM Spectrum Scale RAID disk topology on a node class or server. Use this parameter to identify disk problems that might prevent nodes from acting as recovery group servers.

## Parameters

**mmvdisk server configure**
Configures nodes or an **mmvdisk** node class to be IBM Spectrum Scale RAID servers. The nodes or node class must not already be configured or serving a recovery group.

**mmvdisk server unconfigure**
Unconfigures nodes or an **mmvdisk** node class to no longer be IBM Spectrum Scale RAID servers. The nodes or node class must no longer be associated with a recovery group.

**mmvdisk server list**
Lists IBM Spectrum Scale RAID servers and their characteristics.

**--node-class** *NcName*
Specifies a single node class name to be configured, unconfigured, or listed.

**-N** *Node[,Node...]*
Specifies individual nodes to be configured, unconfigured, or listed

**--recycle** *none* | *one* | *all* | *Number*
Specifies the number of nodes to simultaneously restart so IBM Spectrum Scale configuration changes can take effect. The default is*none*, which means that the administrator is responsible for using the **mmshutdown** and **mmstartup** commands to recycle the nodes or node class. You can specify *one* to recycle one node at a time, specify *all* to recycle all specified nodes simultaneously, or you can specify a *Number* of nodes to recycle simultaneously. To maintain quorum availability in the IBM Spectrum Scale cluster, exercise caution when you recycle nodes.

This parameter applies to the mmvdisk server configure and mmvdisk server unconfigure commands.

**--pagepool** *nM* | *nG* | *nT* | *n%*
Chooses a specific IBM Spectrum Scale page pool value.

**Attention:** Normally the mmvdisk server configure command chooses the most appropriate page pool size for the real memory that is available in the server node class, so use this option only when instructed by IBM.
This parameter applies to mmvdisk server configure.

**--maxblocksize** *2M* | *4M* | *8M* | *16M*
Sets the IBM Spectrum Scale cluster maximum block size. Even though maximum blocksize is not a server-specific configuration value, IBM Spectrum Scale RAID is often used with large file system block sizes. This parameter provides a convenient way to set the maximum file system block size. This would typically be done only when you configure the first recovery group in a cluster, and it requires that the administrator makes sure that the IBM Spectrum Scale daemon on all cluster nodes is recycled (as opposed to being recycled only on the recovery group server nodes). This parameter applies to `mmvdisk server configure`.

**--recovery-group** *RgName*
Lists the servers for the named recovery group (when used with the `mmvdisk server list` command).

**--version**
Lists the IBM Spectrum Scale RAID and operating system version for the servers (when used with the `mmvdisk server list` command).

**--config**
Lists the IBM Spectrum Scale RAID configuration for the servers (when used with the `mmvdisk server list` command).

**--disk-topology**
Lists the discovered IBM Spectrum Scale RAID disk topology (when used with the `mmvdisk server list` command).

**--fanout** *N*
Chooses how many servers to query simultaneously for disk topology information (when used with the `mmvdisk server list --disk-topology` command). The default is `32`. Disk fabric errors with twin-tailed servers make `--fanout 1` useful.

**-L** Provides a detailed listing of the discovered IBM Spectrum Scale RAID disk topology when used with the `mmvdisk server list --disk-topology` command and a single node. This parameter is useful if you suspect that there are disk topology errors.

**-Y** Specifies that the `mmvdisk server list` command produce colon-delimited raw output.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mmvdisk server** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Example

To display the IBM Spectrum Scale RAID disk topology of the servers in node class `ESS01`:

```
mmvdisk server list --node-class ESS01 --disk-topology
```

The system displays output similar to the following:

```
 node                                 needs    matching
number  server                       attention   metric   disk topology
------  ---------------------------- ---------  --------  -------------
     1  server01.gpfs.net            no          100/100  ESS GL6
     2  server02.gpfs.net            no          100/100  ESS GL6
```

## Example

To configure node class ESS01 to be a IBM Spectrum Scale RAID recovery group server and to restart the daemons one at a time to make the configuration take effect:

```
mmvdisk server configure --node-class ESS01 --recycle one
```

The system displays output similar to the following:

```
mmvdisk: Checking resources for specified nodes.
mmvdisk: Node 'server01.gpfs.net' has a paired recovery group disk topology.
mmvdisk: Node 'server02.gpfs.net' has a paired recovery group disk topology.
mmvdisk: Node class 'ESS01' has a paired recovery group disk topology.
mmvdisk: Setting configuration for node class 'ESS01'.
mmvdisk: Node class 'ESS01' is now configured to be recovery group servers.
mmvdisk: Restarting GPFS daemon on node 'server01.gpfs.net'.
mmvdisk: Restarting GPFS daemon on node 'server02.gpfs.net'.
```

To display the status of the newly-configured node class ESS01:

```
mmvdisk server list --node-class BB01 --config
```

The system displays output similar to the following:

```
 node
number  server                       active   remarks
------  ---------------------------- -------  -------
     1  server01.gpfs.net            yes      configured, idle
     2  server02.gpfs.net            yes      configured, idle
```

## See also

- “mmvdisk command” on page 218
- “mmvdisk nodeclass command” on page 222
- “mmvdisk recoverygroup command” on page 229
- “mmvdisk filesystem command” on page 238
- “mmvdisk vdiskset command” on page 243
- “mmvdisk vdisk command” on page 250
- “mmvdisk pdisk command” on page 253

## Location

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

# mmvdisk recoverygroup command

Manages **mmvdisk** recovery groups for IBM Spectrum Scale RAID.

## Synopsis

```
mmvdisk recoverygroup create  --recovery-group RgName1, RgName2
                              --node-class NcName
                              [--match  N]
                              [--multiple-da-legacy]
                              [-v yes | no]
                              [--fanout N]
                              [--defer-log-format]
```

or

```
mmvdisk recoverygroup create --complete_log_format NcName
```

or

```
mmvdisk recoverygroup convert --recovery-group RgName1[RgName2...]
                               --node-class NcName
                               [--pagepool {nM | nG | n% | retain}]
                               [--recycle {none | one | all | Number}]
                               [--dry-run]
```

or

```
mmvdisk recoverygroup change --recovery-group RgName
                             {--primary Node [backup Node]  [-v {yes | no}] |
                             --active Node |
                             --version {VersionString | LATEST} |
                             --declustered-array DaName  {[--spares {NumberOfSpares}]}
                             [--vcd-spares NumberOfSpares]
                             [--scrub-duration NumberOfDays]
                             [--replace-threshold NumberOfPdisks]}
```

or

```
mmvdisk recoverygroup list [-Y]
```

or

```
mmvdisk recoverygroup list --recovery-group RgName
                           [--version]
                           [--server]
                           [--declustered-array]
                           [--pdisk]
                           [--vdisk-set]
                           [--vdisk]
                           [--fault-tolerance]
                           [-Y]
```

or

```
mmvdisk recoverygroup list --recovery-group RgName
                           --all
                           [-Y]
```

or

```
mmvdisk recoverygroup list --recovery-group RgName
                           --events
                           [--days Days]
                           [--decimal]
                           [--long-term Codes]
                           [--short-term Codes]
```

or

```
mmvdisk recoverygroup delete --recovery-group RgName
                             [--confirm]
                             [-p]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

Use the `mmvdisk recoverygroup` command to manage recovery groups for IBM Spectrum Scale RAID.

The `mmvdisk recoverygroup` command:

- Creates new recovery groups.
- Converts existing recovery groups to run under **mmvdisk** management.
- Changes the attributes of recovery groups.
- Lists recovery group information.
- Deletes recovery groups.

Use the `mmvdisk recoverygroup create` command to create an IBM Spectrum Scale RAID recovery group pair. This requires a node class that contains the two associated servers. The node class must already be configured by using the `mmvdisk server configure` command. You must restart the servers must with the configuration in effect.

The two servers must have identical IBM Spectrum Scale RAID disk topologies. The names of the two paired recovery groups must be supplied by using the `--recovery-group RgName1,RgName2` command. The primary and backup servers for the two recovery groups are determined by a case-insensitive sorting of the two server names. The first supplied recovery group considers the server that is sorted first to be the primary server. The server that is sorted second is the backup. The second supplied recovery group is associated with the server that sorts the second as primary and the server that sorts first as backup. Use the `mmvdisk recoverygroup change` command to change these values.

The `mmvdisk recoverygroup create` command treats a recovery group and its log vdisks as an associated unit. When you create recovery groups, it also formats the appropriate log vdisks for the recovery groups as a serial part of the process. When you create large numbers of recovery groups in sequence, you might want to defer the format of the log vdisks until after you create all of the recovery groups. This can speed up the process because of the parallelization that is used to format the deferred log vdisks at once.

Use the `--defer-log-format` parameter to postpone log vdisk formatting. After you create all of the base recovery groups, use the `mmvdisk recoverygroup create --complete-log-format` command to format all deferred log vdisks. User vdisk sets cannot be defined in a recovery group that contains deferred log vdisks until the log format is completed.

Before new recovery groups can be created, all existing recovery groups in a cluster must be converted to **mmvdisk** management.

Existing recovery groups can be placed under **mmvdisk** management if you use the `mmvdisk recoverygroup convert` command. Apply this procedure once per recovery group pair. You must supply the name of one or both existing recovery groups in a pair. If you supply only one recovery group pair the other is automatically determined because they share two servers.

The node class name can be an existing node class that contains only the two servers that are associated with the recovery group pair. If this happens the existing node class name is converted to a **mmvdisk** node class. If the node class name is not in use, it is created as an **mmvdisk** node class that contains the two servers for the recovery group pair.

The `mmvdisk recoverygroup convert` command examines each vdisk in the two recovery groups, and matches them automatically into converted vdisk sets. Converted vdisk sets collect from all converted recovery groups the vdisks that have identical attributes and are in the same file system. Converted vdisk sets are given constructed names such as `VS002_fs2`, which might later be changed to something more meaningful when you use the `mmvdisk vdiskset rename` command. Recovery group conversion requires that you restart the IBM Spectrum Scale daemon on the converted servers. Use the **--recycle** parameter with the `mmvdisk recoverygroup convert` command to make this happen.

Use the `mmvdisk recoverygroup change` to change the:

- Server assignments of a recovery group
- Recovery group feature version
- Declustered array attributes

The normal use case for changing the server assignments is to swap the primary and backup servers of a paired recovery group by using the `--primary` *Node* `--backup` *Node* command. Regardless of the use case, the recovery group immediately switches over to being served by the new primary server.

The active server for a recovery group can be changed immediately without permanently changing the primary and backup assignments by using the `mmvdisk recoverygroup change --active` *Node* command. The new active server must be a member of the recovery group node class.

Use the `mmvdisk recoverygroup change --version` {*VersionString* | *LATEST*} command to change the recovery group feature version. The *LATEST* keyword refers to the latest feature version available with the currently installed IBM Spectrum Scale RAID software.

The spares, VCD spares, scrub duration, and replace threshold of each declustered array in a recovery group are set to the recommended values when the recovery group is created. Under the direction of your service team, you can change the values by using the `mmvdisk recoverygroup change --declustered-array` *DaName* command with the appropriate spare, VCD spare, scrub duration, and replacement threshold parameters.

Use the `mmvdisk recoverygroup list` command to list recovery group information that is displayed. Using the `mmvdisk recoverygroup list` command displays all recovery groups in the cluster whether they are active or not and the current server. Use this information to tell which recovery groups have and have not been converted to **mmvdisk** management. Use the `--recovery-group` *RgName* command to display information for a single recovery group. More detailed recovery group information is only available for recovery groups that are created by using the **mmvdisk** command. Use the following parameters to choose different output sections:

**--version**
: Shows recovery group version information

**--server**
: Shows recovery group server information.

**--declustered array**
: Shows recovery group declustered array information.

**--pdisk**
: Shows recovery group pdisk information.

**--vdisk-set**
: Shows recovery group vdisk set information.

**--vdisk**
: Shows recovery group vdisk information.

**--fault-tolerance**
: Shows recovery fault tolerance information.

Any combination of the detailed information flags can be used, or **--all** can be used to show all sections.

The recovery group event log is displayed independently of the detailed information by using the `mmvdisk recoverygroup list --events` command. By default, all available events in the short- and long-term event logs are displayed. The parameters **--days** *Days*, **--short-term** *Codes*, and **--long-term** *Codes* can be combined to select a subset of the event log to display. The event *Codes* argument is a string that contains one or more of the letters:

**C** Configuration events

**E** Error events

**W** Warning events

**I** Informational events

**D** Detail events

Use the `mmvdisk recoverygroup delete` command to delete a recovery group. Even though recovery groups might be created in pairs, they are deleted individually. The order in which the two recovery groups of a pair are deleted does not matter. A recovery group cannot be deleted until it is undefined from any associated vdisk sets. Deleting a recovery group also deletes all the log vdisks in the recovery group.

## Parameters

**mmvdisk recoverygroup create**
Creates **mmvdisk** recovery groups.

**mmvdisk recoverygroup convert**
Converts existing recovery groups to **mmvdisk** recovery groups.

**mmvdisk recoverygroup change**
Changes the characteristics of an **mmvdisk** recovery group.

**mmvdisk recoverygroup list**
Lists all recovery groups or lists detailed information for **mmvdisk** recovery groups.

**mmvdisk recoverygroup delete**
Deletes an **mmvdisk** recovery group.

**--recovery-group** *Rgname1,RgName2*
Creates or converts the two named recovery groups. This applies to the **mmvdisk recoverygroup create** or **mmvdisk recoverygroup convert** command.

**--recovery-group** *Rgname*
Specifies a recovery group for use with the **mmvdisk recoverygroup change**, **mmvdisk recoverygroup list**, or **mmvdisk recoverygroup delete** command.

**--node-class** *NcName*
Specifies the node class to be associated with the recovery groups. This applies to the **mmvdisk recoverygroup create** or **mmvdisk recoverygroup convert** command.

**--match** *N*
Specifies the minimum IBM Spectrum Scale RAID disk topology matching metric that to be accepted for recovery group creation. The default is `100` (out of 100). This means that you can use only perfectly matching disk topologies to create recovery groups. It is advised against creating recovery groups that have imperfectly matching disk topologies.

**--multiple-da-legacy**
Specifies that multiple declustered arrays be created within recovery groups on GL4 and GL6 IBM Spectrum Scale RAID disk topologies. This option is provided only for compatibility with existing

GL4 and GL6 installations that use multiple declustered arrays. Do not use this parameter if compatibility with existing multiple DA installations is not required.

**-v** *yes* | *no*
Use this parameter with the `mmvdisk recoverygroup create` command to specify whether pdisks should be verified to make sure that they do not contain IBM Spectrum Scale RAID data before they are initialized in a recovery group. The default is *yes*, which verifies that pdisks do not contain data and are empty.

Use this parameter with the `mmvdisk recoverygroup change --primary` *Node* `[--backup]` command to specify whether to verify that the recovery group pdisks are present on the new servers. The default is *yes*.

**--fanout** *N*
Specifies how many servers to query simultaneously for disk topology information. The default is `32`.

**--defer-log-format**
Specifies that the log vdisks should not be formatted at the time recovery groups are created..

**--complete-log-format**
Specifies that any deferred log vdisks in any recovery groups should be formatted.

**--pagepool** {*nM* | *nG* | *n%* | *retain*}
Chooses a specific IBM Spectrum Scale page pool value (when used with the `mmvdisk recoverygroup convert` command).

**Attention:** Normally, the `mmvdisk recoverygroup convert` command chooses the best appropriate pagepool size for the real memory available in the server node class. Use this parameter only under instruction by IBM.

**--recycle** {*none* | *one* | *all* | *Number*}
Specifies the number of nodes to be simultaneously restarted so that the converted IBM Spectrum Scale configuration changes can take effect (when used with the `mmvdisk recoverygroup convert` command). The default is *none*, which means that the administrator must use the **mmshutdown** command and the **mmstartup** command to recycle the nodes or node class. Specify the *one* variable to recycle one node at a time. Specify *all* to recycle all specified nodes simultaneously. Specify the *Number* of nodes to be chosen and to be recycled at the same time. Care must be taken when you recycle nodes if it is desired to maintain quorum availability in the IBM Spectrum Scale cluster.

**--dry-run**
Specifies that the process of converting an existing recovery group pair should be simulated without making any actual changes to the cluster configuration (when used with the `mmvdisk recoverygroup convert` command). This shows the conversion and vdisk set discovery process and to allow it to be examined prior to making the changes permanent.

**--primary** *Node*
Specifies the new primary and backup servers for a recovery group.

**Attention:** Exercise caution when you omit a backup server since it reduces recovery group availability.
See *IBM Spectrum Scale RAID: Administration* for more information.

**--version** *VersionString* | *Latest*
Specifies the recovery group feature version be updated (when used with the `mmvdisk recoverygroup change` command). Use *Latest* to update to the latest version available with the currently installed IBM Spectrum Scale RAID software.

**--declustered-array** *DaName*
Specifies the target declustered array for which attributes to change (when used with the `mmvdisk recoverygroup change` command).

**--spares** *NumberOfSpares*
Specifies the number of effective spares to reserve in the target declustered array (when used with the `mmvdisk recoverygroup change` command). The default depends on the recovery group IBM Spectrum Scale RAID disk topology.

**--vcd-spares** *NumberOfSpares*
Specifies the number of disks that can become unavailable while still maintaining of vdisk configuration data (VCD) replication (when used with the `mmvdisk recoverygroup change` command). The default depends on the recovery group IBM Spectrum Scale RAID disk topology.

**--scrub-duration** *NumberOfDays*
Specifies the background rate at which vdisks should be checked for errors (when used with the `mmvdisk recoverygroup change` command). The value is the number of days (from 1 to 365) that the process takes for the entire target declustered array. The default is 14.

**--replace-threshold** *NumberOfDisks*
Specifies the number of out-of-service pdisks in the target declustered array at which point the recovery group reports that it needs disk replacement service (when used with the `mmvdisk recoverygroup change` command). The values are 2 for DAs with more than 12 pdisks (default) or 1 for DAs with 12 or fewer pdisks.

**--version**
Specifies that the recovery group version information should be listed.

**--server**
Specifies that the recovery group server information should be listed.

**--declustered-array**
Specifies that recovery group declustered array information should be listed.

**--pdisk**
Specifies that the recovery group pdisk information should be listed.

**--vdisk-set**
Specifies that the recovery group vdisk set information should be listed.

**--vdisk**
Specifies that the recovery group vdisk information should be listed.

**--fault-tolerance**
Specifies that the recovery group fault tolerance information should be listed.

**--all**
Specifies that all sections of recovery group information should be listed.

**--events**
Specifies that the recovery group event log be listed (when used with the `mmvdisk recoverygroup list` command).

**--days** *Days*
Displays the number of past days for which recovery group events are to be displayed. The default is to display events as far back as recovery group creation.

**--decimal**
Specifies that recovery group event timestamps be displayed in decimal format.

**--long-term** *Codes*
Specifies that long-term recovery group events that correspond to the *Codes* string are displayed.

**--short-term** *Codes*
Specifies that short-term recovery group events that correspond to the *Codes* string are displayed. .

**--confirm**
Confirms the deletion of a recovery group. This bypasses the interactive prompt that would otherwise be printed to confirm recovery group deletion.

**-p** Indicates (when used with the `mmvdisk recoverygroup delete` command) that the recovery group is permanently damaged and must be deleted without accessing its pdisks. This option is not permitted if the recovery group is active.

**Attention:** This is an unusual maintenance case that should only be performed under the direction of the IBM Service team.
Before a recovery group can be deleted by using **-p**, any user vdisks and vdisk sets must be deleted from the recovery group by using the `mmvdisk vdiskset delete -p` command for each vdisk set, followed by undefining the recovery group from each vdisk set by using the `mmvdisk vdiskset undefine` command. Note that **-p** is not necessary to undefine a damaged recovery group from a vdisk set since the actual member vdisks in the recovery group are deleted.

**-Y** Specifies that the `mmvdisk recoverygroup list` command produce colon-delimited raw output.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mmchfirmware** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Example

To create recovery groups `ESS01L` and `ESS01R` using the node class `ESS01`:

```
mmvdisk recoverygroup create --node-class ESS01 --recovery-group ESS01L,ESS01R
```

The system displays output similar to the following:

```
mmvdisk: Checking node class configuration.
mmvdisk: Checking daemon status on node 'ess01io1.gpfs.net'.
mmvdisk: Checking daemon status on node 'ess01io2.gpfs.net'.
mmvdisk: Analyzing disk topology for node 'ess01io1.gpfs.net'.
mmvdisk: Analyzing disk topology for node 'ess01io2.gpfs.net'.
mmvdisk: Creating recovery group 'ESS01L'.
mmvdisk: Creating recovery group 'ESS01R'.
mmvdisk: Creating log vdisks for recovery groups.
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001LOGTIP
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001LOGTIPBACKUP
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001LOGHOME
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002LOGTIP
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002LOGTIPBACKUP
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002LOGHOME
mmvdisk: Created recovery groups 'ESS01L' and 'ESS01R'.
```

## Example

To list all recovery groups present in the cluster:

```
mmvdisk recoverygroup list
```

The system displays output similar to the following:

```
                                                          needs    user
recovery group  active   current or master server        service  vdisks  remarks
--------------  -------  -------------------------------  -------  ------  -------
ESS01L          yes      ess01io1.gpfs.net                no            2
ESS01R          yes      ess01io2.gpfs.net                no            2
```

## Example

To list the declustered array information for recovery group ESS01L:

```
mmvdisk recoverygroup list --recovery-group ESS01L --declustered-array
```

The system displays output similar to the following:

```
declustered   needs    vdisks      pdisks     replace        capacity        scrub
   array     service  user log  total spare  threshold  total raw free raw  duration  background task
-----------  -------  ---- ---  ----- -----  ---------  --------- --------  --------  ---------------
NVR          no         0   1      2     0          1          0        0   14 days  scrub (4%)
SSD          no         0   1      1     0          1          0        0   14 days  scrub (4%)
DA1          no         2   1    174     6          2    911 TiB  346 TiB   14 days  scrub (0%)

mmvdisk: Total capacity is the raw space before any vdisk set definitions.
mmvdisk: Free capacity is what remains for additional vdisk set definitions.
```

## Example

To convert existing recovery groups ESS01L and ESS01R with a new node class ESS01:

```
mmvdisk recoverygroup convert --recovery-group ESS01L,ESS01R --node-class ESS01
```

The system displays output similar to the following:

```
mmvdisk: This command will permanently change the GNR configuration
mmvdisk: attributes and disable the legacy GNR command set for the
mmvdisk: servers and recovery groups involved, and their subsequent
mmvdisk: administration must be performed with the mmvdisk command.

mmvdisk: Do you wish to continue (yes or no)? yes

mmvdisk: Converting recovery groups 'ESS01L' and 'ESS01R'.
mmvdisk: Creating node class 'ESS01'.
mmvdisk: Adding 'ess01io1' to node class 'ESS01'.
mmvdisk: Adding 'ess01io2' to node class 'ESS01'.
mmvdisk: Associating recovery group 'ESS01L' with node class 'ESS01'.
mmvdisk: Associating recovery group 'ESS01R' with node class 'ESS01'.
mmvdisk: Recording pre-conversion cluster configuration in /var/mmfs/tmp/mmvdisk.convert.ESS01L.ESS01R.before.m24
mmvdisk: Updating server configuration attributes.
mmvdisk: Checking resources for specified nodes.
mmvdisk: Setting configuration for node class 'ESS01'.
mmvdisk: Defining vdisk set 'VS001_fs3' with recovery group 'ESS01L' (vdisk 'ESS01Lv3fs3').
mmvdisk: Defining vdisk set 'VS002_fs4' with recovery group 'ESS01L' (vdisk 'ESS01Lv4fs4').
mmvdisk: Defining vdisk set 'VS003_fs1' with recovery group 'ESS01L' (vdisk 'ESS01Lv1fs1data').
mmvdisk: Defining vdisk set 'VS004_fs2' with recovery group 'ESS01L' (vdisk 'ESS01Lv2fs2').
mmvdisk: Defining vdisk set 'VS005_fs1' with recovery group 'ESS01L' (vdisk 'ESS01Lv1fs1meta').
mmvdisk: Adding recovery group 'ESS01R' to vdisk set 'VS003_fs1' (vdisk 'ESS01Rv1fs1data').
mmvdisk: Adding recovery group 'ESS01R' to vdisk set 'VS004_fs2' (vdisk 'ESS01Rv2fs2').
mmvdisk: Adding recovery group 'ESS01R' to vdisk set 'VS001_fs3' (vdisk 'ESS01Rv3fs3').
mmvdisk: Adding recovery group 'ESS01R' to vdisk set 'VS005_fs1' (vdisk 'ESS01Rv1fs1meta').
mmvdisk: Adding recovery group 'ESS01R' to vdisk set 'VS002_fs4' (vdisk 'ESS01Rv4fs4').
mmvdisk: Committing cluster configuration changes.
mmvdisk: Recording post-conversion cluster configuration in /var/mmfs/tmp/mmvdisk.convert.ESS01L.ESS01R.after.m24

mmvdisk: For configuration changes to take effect, GPFS should be restarted
mmvdisk: on node class 'ESS01'.
```

## See also

- "mmvdisk command" on page 218
- "mmvdisk nodeclass command" on page 222
- "mmvdisk server command" on page 225
- "mmvdisk filesystem command" on page 238
- "mmvdisk vdiskset command" on page 243
- "mmvdisk vdisk command" on page 250
- "mmvdisk pdisk command" on page 253

## Location

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

# mmvdisk filesystem command

Manages **mmvdisk** file systems for IBM Spectrum Scale RAID.

## Synopsis

```
mmvdisk filesystem create  --file-system FsName
                           --vdisk-set VdiskSet[,VdiskSet...]
                           [--mmcrfs mmcrfs-options]
```

or

```
mmvdisk filesystem add  --file-system FsName
                        --vdisk-set VdiskSet[,VdiskSet...]
```

or

```
mmvdisk filesystem delete  --file-system FsName
                           --vdisk-set VdiskSet[,VdiskSet...]
                           --recovery-group RgName
```

or

```
mmvdisk filesystem delete  --file-system FsName
                           [--confirm]
```

or

```
mmvdisk filesystem list  [--file-system FsName]
                         [-Y]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

Use the `mmvdisk filesystem` command to create **mmvdisk** file systems by using IBM Spectrum Scale RAID vdisk sets. The `mmvdisk filesystem` command can also be used to:

- Add or delete vdisk sets from an **mmvdisk** file system.
- Delete an **mmvdisk** file system in its entirety.
- List **mmvdisk** file systems and show the vdisk sets they contain.

An **mmvdisk** file system is an IBM Spectrum Scale file system where each vdisk NSD in the file system is from a vdisk set. The `mmvdisk filesystem` command applies IBM Spectrum Scale RAID best practices to the management of file systems that are constructed from vdisk sets. One important best practice is that the member vdisk NSDs of a given vdisk set might belong only to one file system, but a file system can contain multiple vdisk sets.

The `mmvdisk filesystem create` command takes a file system device name and one or more vdisk sets, and creates an IBM Spectrum Scale file system from the member vdisk NSDs of the vdisk sets. The file system device name might not be in use, and the member vdisk NSDs of the vdisk sets must be created. File system NSD usage, storage pools, and block size are set according to the vdisk set attributes. File system failure groups and maximum data and metadata replication factors are assigned, depending on the server structure of the recovery groups in the vdisk sets.

Extra IBM Spectrum Scale file system options that are supported by the **mmcrfs** command (for example, quota enablement or a custom mount point) can be specified by using the **-mmcrfs** parameter. All command line parameters after the **--mmcrfs** parameter are not interpreted by the **mmvdisk** command, but are instead passed (unchanged) to the **mmcrfs** command.

The `mmvdisk filesystem add` command is an interface between vdisk sets, **mmvdisk** file systems, and the IBM Spectrum Scale **mmadddisk** command. Use the `mmvdisk filesystem add` command to add new or newly extended vdisk sets to an existing **mmvdisk** file system. The member vdisk NSDs of the vdisk sets must be created. When a vdisk set is extended by adding recovery groups, the `mmvdisk filesystem add` command adds only those member vdisk NSDs that are not in the file system.

Use the `mmvdisk filesystem delete` command to delete vdisk sets or recovery groups from an **mmvdisk** file system, or to delete an **mmvdisk** file system in its entirety. When you delete vdisk sets or recovery groups, the `mmvdisk filesystem delete` command is an interface between vdisk sets, **mmvdisk** file systems, and the IBM Spectrum Scale **mmdeldisk** command.

When deleting an **mmvdisk** file system in its entirety, the `mmvdisk filesystem delete` command is an interface between vdisk sets, **mmvdisk** file systems, and the IBM Spectrum Scale **mmdelfs** command.

Deleting vdisk sets from a file system always prompts for confirmation because the underlying **mmdelvdisk** command can take a long time. This prompt cannot be bypassed.

Deleting an entire **mmvdisk** file system always prompts for confirmation because deleting an entire **mmvdisk** file system destroys all file system data. This prompt can be bypassed and answered in the affirmative by using the **--confirm** parameter.

Use the `mmvdisk filesystem list` command to list **mmvdisk** file systems and the file system NSD information for their vdisk sets.

The `mmvdisk filesystem` command manages only the vdisk NSD aspects of an **mmvdisk** file system. IBM Spectrum Scale file system attributes continue to be managed by other IBM Spectrum Scale file system commands - such as the **mmlsfs** and **mmchfs** commands.

## Parameters

**mmvdisk filesystem create**
: Creates **mmvdisk** file systems by using IBM Spectrum Scale RAID vdisk sets.

**mmvdisk filesystem add**
: Adds IBM Spectrum Scale RAID vdisk sets to **mmvdisk** file systems.

**mmvdisk filesystem delete**
: Deletes IBM Spectrum Scale RAID vdisk sets from **mmvdisk** file systems, or deletes an entire **mmvdisk** file system.

**mmvdisk filesystem list**
: Lists **mmvdisk** file systems or their component IBM Spectrum Scale RAID vdisk sets.

**--file-system***FsName*
: Specifies the name of the file system to be operated on.

**--vdisk-set***VdiskSet[,VdiskSet...]*
: Specifies the vdisk sets to be affected by a file system operation.

**--mmcrfs**
: Indicates, when used with the `mmvdisk recoverygroup delete` command, that all following command line parameters are not to be interpreted by **mmvdisk**, but are instead to be passed to the IBM Spectrum Scale **mmcrfs** command.

**--recovery-group***RgName*
: Restricts vdisk set deletion to specified recovery group.

  **Note:** This is required preparation for removing a recovery group from a vdisk set definition. This applies to the `mmvdisk filesystem delete` command.

**--confirm**
Confirms the deletion of an entire **mmvdisk** file system. This flag bypasses the interactive prompt that would otherwise be printed to confirm file system deletion.

**-Y** Specifies that the `mmvdisk filesystem list` command produce colon-delimited raw output.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mmvdisk filesystem** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Example

To create file system `udata` by using vdisk set `VS1`, and to pass the `-Q` *yes* and `-T /udata` parameters to **mmcrfs**:

```
mmvdisk filesystem create --file-system udata --vdisk-set VS1 --mmcrfs -Q yes -T /udata
```

The system displays output similar to the following:

```
mmvdisk: Creating file system 'udata'.
mmvdisk: The following disks of udata will be formatted on node ess01io1:
mmvdisk:     RG001VS001: size 346212352 MB
mmvdisk:     RG002VS001: size 346212352 MB
mmvdisk: Formatting file system ...
mmvdisk: Disks up to size 2.59 PB can be added to storage pool system.
mmvdisk: Creating Inode File
mmvdisk:    0 % complete on Tue Jun  5 11:36:34 2018
mmvdisk:   11 % complete on Tue Jun  5 11:36:39 2018
mmvdisk:   56 % complete on Tue Jun  5 11:37:05 2018
mmvdisk:   88 % complete on Tue Jun  5 11:37:15 2018
mmvdisk:  100 % complete on Tue Jun  5 11:37:17 2018
mmvdisk: Creating Allocation Maps
mmvdisk: Creating Log Files
mmvdisk:    6 % complete on Tue Jun  5 11:37:23 2018
mmvdisk:  100 % complete on Tue Jun  5 11:37:24 2018
mmvdisk: Clearing Inode Allocation Map
mmvdisk: Clearing Block Allocation Map
mmvdisk: Formatting Allocation Map for storage pool system
mmvdisk:   57 % complete on Tue Jun  5 11:37:36 2018
mmvdisk:  100 % complete on Tue Jun  5 11:37:39 2018
mmvdisk: Completed creation of file system /dev/udata.
```

## Example

To add vdisk set `VS2` to file system `udata`:

```
mmvdisk filesystem add --file-system udata --vdisk-set VS2
```

The system displays output similar to the following:

```
mmvdisk: The following disks of udata will be formatted on node ess01io1:
mmvdisk:     RG001VS002: size 346212352 MB
mmvdisk:     RG002VS002: size 346212352 MB
```

```
mmvdisk: Extending Allocation Map
mmvdisk: Checking Allocation Map for storage pool system
mmvdisk:   37 % complete on Tue Jun  5 11:43:06 2018
mmvdisk:   76 % complete on Tue Jun  5 11:43:16 2018
mmvdisk:  100 % complete on Tue Jun  5 11:43:22 2018
mmvdisk: Completed adding disks to file system udata.
```

## Example

To list file system udata and show its vdisk sets and file system NSD information:

```
mmvdisk filesystem list --file-system udata
```

The system displays output similar to the following:

```
                                vdisk     list of       holds    holds  storage
vdisk set       recovery group  count  failure groups  metadata  data    pool
--------------  --------------  -----  --------------  --------  -----  -------
VS1             ESS01L              1               1  yes       yes    system
VS1             ESS01R              1               2  yes       yes    system
VS2             ESS01L              1               1  yes       yes    system
VS2             ESS01R              1               2  yes       yes    system
```

## Example

To delete vdisk set VS2 from file system udata:

```
mmvdisk filesystem delete --file-system udata --vdisk-set VS2
```

The system displays output similar to the following:

```
mmvdisk: This will run the GPFS mmdeldisk command on file system 'udata'
mmvdisk: which may take hours to complete and should not be interrupted.

mmvdisk: Do you wish to continue (yes or no)? yes

mmvdisk: Removing vdisk set 'VS2' from file system 'udata'.
mmvdisk: Deleting disks ...
mmvdisk: Scanning file system metadata, phase 1 ...
mmvdisk: Scan completed successfully.
mmvdisk: Scanning file system metadata, phase 2 ...
mmvdisk: Scan completed successfully.
mmvdisk: Scanning file system metadata, phase 3 ...
mmvdisk: Scan completed successfully.
mmvdisk: Scanning file system metadata, phase 4 ...
mmvdisk: Scan completed successfully.
mmvdisk: Scanning user file metadata ...
mmvdisk:  100.00 % complete on Tue Jun  5 12:30:54 2018 (503808 inodes with total 13872 MB data processed)
mmvdisk: Scan completed successfully.
mmvdisk: Checking Allocation Map for storage pool system
mmvdisk:   11 % complete on Tue Jun  5 12:30:59 2018
mmvdisk:   41 % complete on Tue Jun  5 12:31:09 2018
mmvdisk:   77 % complete on Tue Jun  5 12:31:19 2018
mmvdisk:  100 % complete on Tue Jun  5 12:31:24 2018
mmvdisk: tsdeldisk completed.
```

## Example

To delete the entire file system udata and bypass the interactive confirmation prompt:

```
mmvdisk filesystem delete --file-system udata --confirm
```

The system displays output similar to the following:

```
mmvdisk: All data on the following disks of udata will be destroyed:
mmvdisk:     RG001VS001
mmvdisk:     RG002VS001
mmvdisk: Completed deletion of file system /dev/udata.
```

## See also

- "mmvdisk command" on page 218
- "mmvdisk server command" on page 225
- "mmvdisk recoverygroup command" on page 229
- "mmvdisk pdisk command" on page 253
- "mmvdisk vdiskset command" on page 243
- "mmvdisk vdisk command" on page 250
- "mmvdisk nodeclass command" on page 222

## Location

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

# mmvdisk vdiskset command

Manages **mmvdisk** vdisk sets for IBM Spectrum Scale RAID.

## Synopsis

```
mmvdisk vdiskset define  --vdisk-set VdiskSet
                         --recovery-group all | RgName[,RgName...]
                         --code RaidCode
                         --block-size BlockSize
                         --set-size SetSize
                         [--declustered-array DaName]
                         [--nsd-usage NsdUsage [--storage-pool Pool]]
```

or

```
mmvdisk vdiskset define  --vdisk-set VdiskSet
                         --recovery-group RgName[,RgName...]
                         [--force-incompatible]
```

or

```
mmvdisk vdiskset undefine  --vdisk-set VdiskSet
                         --recovery-group RgName[,RgName...]
```

or

```
mmvdisk vdiskset undefine  --vdisk-set VdiskSet
                         [--confirm]
```

or

```
mmvdisk vdiskset create  --vdisk-set {all | [VdiskSet[,VdiskSet]}
```

or

```
mmvdisk vdiskset delete --vdisk-set {all | [VsName[,VsName]}
                        [--recovery-group RgName[,RgName...]]
                        [-p]
```

or

```
mmvdisk vdiskset rename --vdisk-set VdiskSet
                                                --new-name NewName
```

or

```
mmvdisk vdiskset list --vdisk-set {all | VDiskSet[,VdiskSet...]}
                      [-Y]
```

or

```
mmvdisk vdiskset list --recovery-group {all | RgName[,RgName...]}
                      [-Y]
                      [--declustered-array DaName[,DaName...]]
```

or

```
mmvdisk vdiskset list --file-system {{all | FsName[,FsName...]}
                      [-Y]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

Use the `mmvdisk vdiskset` command to define, size, or create uniform vdisk NSDs across IBM Spectrum Scale RAID recovery groups. The vdisk sets that result can then be used as units in constructing IBM Spectrum Scale file systems.

With **`mmvdisk`** commands, user vdisks are managed collectively into vdisk sets that contain vdisk NSDs with identical attributes from one or more recovery groups. Each recovery group contributes one member vdisk NSD to the vdisk set. To create a file system that is balanced across several recovery groups, define a vdisk set that uses those recovery groups.

The vdisk set definition provides a preview of the sizing impact of the vdisk set on the recovery groups and the servers. If the sizing is acceptable, the vdisk set can be created. This means that the individual member vdisks and NSDs in each recovery group are all created. The vdisk set can be used to create an IBM Spectrum Scale RAID file system or added to an existing IBM Spectrum Scale RAID file system.All member vdisk NSDs that are part of a vdisk set must belong to the same file system. A file system can contain one or more vdisk sets.

Use the `mmvdisk vdiskset define` command to define vdisk sets or to add a recovery group to an existing vdisk set definition.

The definition of a vdisk set must include:

- You must have a vdisk set name.
- You must have one or more recovery groups.
- You must have the name of a single declustered array that is present in every recovery groups. The member vdisk NSDs are sized against and created in this declustered array. If each recovery group has only a single declustered array, `DA1`, you do not have to specify the name.
- You must have the IBM Spectrum Scale RAID code that is used by the member vdisk NSDs. Accepted IBM Spectrum Scale RAID codes are:
  - `3WayReplication`
  - `4WayReplication`
  - `4+2p`
  - `4+3p`
  - `8+2p`
  - `8+3p`
- You must have the block size that is used by the member vdisk NSDs. The block size is constrained by the IBM Spectrum Scale RAID code. Valid values for `3WayReplication` and `4WayReplication` are:
  - `256k`
  - `512k`
  - `1m`
  - `2m`

  You must have valid values for `4+2P` and `4+3P` are:
  - `512k`
  - `1m`
  - `2m`
  - `4m`
  - `8m`

  You must have valid values for `8+2P` and `8+3P` are:
  - `512k`

- 1m
- 2m
- 4m
- 8m
- 16m

- You must have the size of the vdisk set in one recovery group. The vdisk set size might be specified as a percentage or as a number of bytes. If the vdisk set size is given as a percentage, it specifies the raw size to use from the declustered array, includingIBM Spectrum Scale RAID code redundancy. If the vdisk set size is given as a number of bytes, it specifies the desired usable size of the vdisk set (without IBM Spectrum Scale RAID code redundancy). The vdisk set size is used to calculate the usable size of a single vdisk NSD member of the vdisk set in one recovery group. This calculated usable size becomes part of the vdisk set definition. This means that even if the size of a declustered array changes, the size of the individual member vdisk NSDs remains constant.
- You must have the NSD usage of the vdisk set. This is the IBM Spectrum Scale file system data usage for the vdisk NSDs. Valid values are:
  - `dataAndMetadata`
  - `metadataOnly`
  - `dataOnly`

  The default is `dataAndMetadata`.
- You must have the file system storage pool for the vdisk set. This is the IBM Spectrum Scale file system storage pool for the vdisk NSDs. If the NSD usage is `dataAndMetadata` or `metadataOnly`, the storage pool must be `system` and you do not have to specify it. If the NSD usage is `dataOnly`, the storage pool must be specified and cannot be `system`.

When multiple recovery groups are used in an initial vdisk set definition, the declustered arrays must have compatible characteristics (such as number of pdisks, number of spare pdisks, or pdisk hardware type). This ensures that the member vdisk NSDs in the vdisk set have the same characteristics.

To add recovery groups to an existing vdisk set definition, use `mmvdisk vdiskset define` with only the **`--vdisk-set`** and **`--recovery-group`** parameters. The declustered array in the added recovery groups must be compatible with the declustered array that was used in the initial vdisk set definition. In carefully considered circumstances, it might be possible to force a vdisk set definition into an incompatible declustered array by using the **`--force-incompatible`** parameter.

When a vdisk set is defined, **`mmvdisk`** shows the space that is used by the definition in both the declustered array and in the recovery group server memory. This permits the administrator to evaluate the definition in context before you commit to its creation, or deciding to undefine it and try a different definition.

A vdisk set can be undefined by using the `mmvdisk vdiskset undefine` command. Use this to remove individual recovery groups from a vdisk set, or to remove the entire vdisk set definition. Any member vdisk NSDs in the affected recovery groups must first be deleted by using the `mmvdisk vdiskset delete` command.

The member vdisk NSDs of a vdisk set definition are created by using the `mmvdisk vdiskset create` command. A vdisk set is created when all the actual member vdisk NSDs exist in each recovery group in the vdisk set definition. The `mmvdisk vdiskset create` command creates only those vdisk NSDs that do not exist. This permits a vdisk set to be defined and created, then to have another recovery group added and to create only the vdisk NSDs in the added recovery group. Only created vdisk sets can be used with the `mmvdisk filesystem` command to add to or create IBM Spectrum Scale RAID file systems.

Member vdisk NSDs can be deleted from a vdisk set by using the `mmvdisk vdiskset delete` command. The member vdisk NSDs to be deleted might be limited to a recovery group, or might include the entire vdisk set. In either case, it is not permitted to delete member vdisk NSDs that are in a file system.

Use the `mmvdisk vdiskset rename` command to change the name of a vdisk set. This is useful for giving meaningful names to the vdisk sets defined by the `mmvdisk recoverygroup convert` command.

Use the `mmvdisk vdiskset list` command to list vdisk sets and to show their attributes and sizing.

## Parameters

**mmvdisk vdiskset define**
: Define vdisk sets across IBM Spectrum Scale RAID recovery groups.

**mmvdisk vdiskset undefine**
: Removes vdisk set definitions from IBM Spectrum Scale RAID recovery groups.

**mmvdisk vdiskset create**
: Creates member vdisk NSDs for a vdisk set.

**mmvdisk vdiskset delete**
: Deletes member vdisk NSDs from a vdisk set. The member vdisk NSDs must not belong to a file system.

**mmvdisk vdiskset rename**
: Renames a vdisk set.

**mmvdisk vdiskset list**
: Lists vdisk sets and shows their attributes and sizing.

**--vdisk-set** *VsName*
: Specifies a single vdisk set to define, undefine, or rename.

**--vdisk-set {***all* | *VdiskSet[,VdiskSet...]***}**
: Specifies vdisk sets to create, delete, or list.

**--recovery-group** *all* | *RgName[,RgName...]*
: Specifies recovery groups in which vdisk sets are to be defined or listed.

**--recovery-group** *RgName[,RgName...]*
: Specifies recovery groups from which vdisk NSDs are to be deleted, or in which a vdisk set is to be defined or undefined.

**--code** *RaidCode*
: Specifies the IBM Spectrum Scale RAID code for a vdisk set definition.

**--block-size** *BlockSize*
: Specifies the block size of a vdisk set definition.

**--set-size** *SetSize*
: Specifies the set size (within a single recovery group) of a vdisk set definition.

**--declustered-array** *DaName*
: Specifies the declustered array of a vdisk set definition.

**--nsd-usage** *NsdUsage*
: Specifies the file system NSD usage of a vdisk set definition.

**--storage-pool** *Pool*
: Specifies the file system storage pool of a vdisk set definition.

**--force-incompatible**
: When extending a vdisk set definition to additional recovery groups, attempts to ignore incompatible declustered array characteristics.

**Attention:** This parameter should be used only under careful consideration.

**--new-name** *NewName*
Specifies the new name for a vdisk set (when used with the `mmvdisk vdiskset rename` command).

**--declustered-array** *DaName[,DaName...]*
Chooses the declustered arrays in which vdisk sets are to be listed (when used with the `mmvdisk vdiskset list` command).

**--file-system {***all* | *FsName[,FsName...]***}**
Chooses file systems from which vdisk sets are to be listed (when used with the `mmvdisk vdiskset list` command).

**--confirm**
Confirms the removal of a vdisk set definition. This flag bypasses the interactive prompt that would otherwise be printed to confirm undefining a vdisk set.

**-p** Indicates that the member vdisk NSDs are in a recovery group that is permanently damaged and must be deleted without access to the recovery group (when used with the `mmvdisk recoverygroup delete` command) . You cannot use this parameter if the recovery group is active.

**Attention:** This is an unusual maintenance case that should only be performed under the direction of the IBM Service team.
Before a vdisk set can be deleted using **-p**, the vdisk set must be deleted from any file system.

**-Y** Specifies that the `mmvdisk vdiskset list` command produce colon-delimited raw output.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mmvdisk vdiskset** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Example

To see the available space for defining vdisk sets in a newly created recovery group pair:

```
mmvdisk vdiskset list --recovery-group ESS01L,ESS01R
```

The system displays output similar to the following:

```
                declustered           capacity            all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
ESS01L          DA1            911 TiB   911 TiB   100%  -
ESS01R          DA1            911 TiB   911 TiB   100%  -

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
ESS01          29 GiB   387 MiB  -
```

This shows that:

- No vdisk sets are defined in either recovery group
- The total free raw disk capacity in each declustered array
- The available memory for vdisk set maps on the servers in the recovery group pair node class

## Example

To see the available space for defining vdisk sets in a newly created recovery group pair:

```
mmvdisk vdiskset list --recovery-group ESS01L,ESS01R
```

The system displays output similar to the following:

```
                declustered           capacity            all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
ESS01L          DA1            911 TiB   911 TiB   100%  -
ESS01R          DA1            911 TiB   911 TiB   100%  -

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
ESS01          29 GiB   387 MiB  -
```

This shows that:

- No vdisk sets are yet defined in either recovery group
- The total free raw disk capacity in each declustered array
- The available memory for vdisk set maps on the servers in the recovery group pair node class

## Example

To define vdisk set VS1 in recovery groups ESS01L and ESS01R using RAID code 8+3p, an 8 MiB block size, and 50% of the available raw space:

```
mmvdisk vdiskset define --vdisk-set VS1 --recovery-group ESS01L,ESS01R --code 8+3p --block-size 8m --set-size 50%
```

The system displays output similar to the following:

```
mmvdisk: Vdisk set 'VS1' has been defined.
mmvdisk: Recovery group 'ESS01L' has been defined in vdisk set 'VS1'.
mmvdisk: Recovery group 'ESS01R' has been defined in vdisk set 'VS1'.

                     member vdisks
vdisk set       count   size   raw size  created  file system and attributes
--------------  ----- -------- --------  -------  -------------------------
VS1                 2  330 TiB  455 TiB  no       -, DA1, 8+3p, 8 MiB, dataAndMetadata, system

                declustered           capacity            all vdisk sets defined
recovery group     array     total raw  free raw  free%  in the declustered array
--------------  -----------  ---------  --------  -----  ------------------------
ESS01L          DA1            911 TiB   455 TiB    50%  VS1
ESS01R          DA1            911 TiB   455 TiB    50%  VS1

                  vdisk set map memory per server
node class  available  required  required per vdisk set
----------  ---------  --------  ----------------------
ESS01          29 GiB  9227 MiB  VS1 (8839 MiB)
```

This output shows attributes of the newly defined vdisk set - that it is not yet created. And it also shows the space and memory sizing demands it places on the declustered arrays and the servers in the recovery group node class.

## Example

To create the member vdisk NSDs in vdisk set VS1:

```
mmvdisk vdiskset create --vdisk-set VS1
```

The system displays output similar to the following:

```
mmvdisk: 2 vdisks and 2 NSDs will be created in vdisk set 'VS1'.
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG001VS001
mmvdisk: (mmcrvdisk) [I] Processing vdisk RG002VS001
mmvdisk: Created all vdisks in vdisk set 'VS1'.
mmvdisk: (mmcrnsd) Processing disk RG001VS001
mmvdisk: (mmcrnsd) Processing disk RG002VS001
mmvdisk: Created all NSDs in vdisk set 'VS1'.
```

## See also

- "mmvdisk command" on page 218
- "mmvdisk server command" on page 225
- "mmvdisk recoverygroup command" on page 229
- "mmvdisk filesystem command" on page 238
- "mmvdisk pdisk command" on page 253
- "mmvdisk vdisk command" on page 250
- "mmvdisk nodeclass command" on page 222

## Location

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

# mmvdisk vdisk command

Lists individual IBM Spectrum Scale RAID vdisks.

## Synopsis

```
mmvdisk vdisk list  [--vdisk-set {all | VsName[,VsName...]}
                    [-Y]
```

or

```
mmvdisk vdisk list  --file-system FsName
                    [-Y]
```

or

```
mmvdisk vdisk list  --recovery-group {all | RgName[,RgName...]}
                    [--vdisk-set {all | VsName[,VsName...]}]
                    [--declustered-array DaName[,DaName...]]
                    [--log]
                    [-Y]
```

or

```
mmvdisk vdisk list  --vdisk VdiskName[,VdiskName...]
                    [-Y]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

Use the **mmvdisk vdisk** command to list individual IBM Spectrum Scale RAID vdisks.

Because **mmvdisk** administration of IBM Spectrum Scale RAID creates and deletes individual vdisks either as members of vdisk sets or as recovery group log vdisks, the **mmvdisk vdisk** command supports only the listing of individual vdisks.

Vdisks can be selected based on membership by vdisk set, by file system, or by recovery group and declustered array. Individual vdisks can also be listed by name. By default, only user vdisks are listed, which means vdisks that are in file systems or are eligible to be in file systems.

Log vdisks can be listed by specifying **--log** when a recovery group is specified.

## Parameters

**mmvdisk vdisk list**
: Lists individual vdisks.

**--vdisk-set** *all* | *VsName[,VsName...]*
: Specifies the vdisk sets from which member vdisks are listed.

**--file-system** *FsName*
: Specifies the file system from which vdisks are to be listed.

**--recovery-group** *all* | *RgName[,RgName...]*
: Specifies the recovery groups from which vdisks are to be listed.

**--declustered-array** *DaName[,DaName...]*
: Restricts the listing of recovery group vdisks to include only the specified declustered arrays. If you omit this parameter, vdisks are listed from all declustered arrays in a recovery group.

**--log**
Restricts the listing of recovery group vdisks to include only log vdisks. If you omit this parameter, only user vdisks are listed.

**-Y** Specifies that the `mmvdisk vdisk list` command produces colon-delimited raw output.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mmvdisk vdisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Example

To list the user vdisks that are in file system FS3, specify:

```
mmvdisk vdisk list --file-system FS3
```

The system displays output similar to the following:

```
                                                        declustered array, RAID code,
vdisk              vdisk set       file system  recovery group           block size           remarks
-----------------  --------------  -----------  --------------  ----------------------------  -------
RG001VS001         VS3             FS3          ESS01L          DA1, 8+3p, 8 MiB
RG002VS001         VS3             FS3          ESS01R          DA1, 8+3p, 8 MiB
RG003VS001         VS3             FS3          ESS02L          DA1, 8+3p, 8 MiB
RG004VS001         VS3             FS3          ESS02R          DA1, 8+3p, 8 MiB
```

## Example

To list the log vdisks in recovery group ESS01L, specify:

```
mmvdisk vdisk list --log --recovery-group ESS01L
```

The system displays output similar to the following:

```
                                                        declustered array, RAID code,
vdisk              vdisk set       file system  recovery group           block size           remarks
-----------------  --------------  -----------  --------------  ----------------------------  -------
RG001LOGHOME       -               -            ESS01L          DA1, 4WayReplication, 2 MiB   log home
RG001LLOGTIP       -               -            ESS01L          NVR, 2WayReplication, 2 MiB   log tip
RG001LLOGTIPBACKUP -               -            ESS01L          SSD, Unreplicated, 2 MiB      log tip backup
```

## See also

- "mmvdisk command" on page 218
- "mmvdisk server command" on page 225
- "mmvdisk recoverygroup command" on page 229
- "mmvdisk filesystem command" on page 238
- "mmvdisk vdiskset command" on page 243
- "mmvdisk nodeclass command" on page 222
- "mmvdisk pdisk command" on page 253

## Location

/usr/lpp/mmfs/bin

# mmvdisk pdisk command

Manages **mmvdisk** recovery group pdisks for IBM Spectrum Scale RAID.

## Synopsis

```
mmvdisk pdisk list --recovery-group {all | RgName[,RgName]}
                   [--declustered-array DaName | --pdisk PdiskName]
                   [--replace | --not-ok]
                   [-Y]
```

or

```
mmvdisk pdisk list  -L
                   --recovery-group {{all | RgNaame}  [--declustered-array DaName |  --pdisk PdiskName]
                   [--replace | --not-ok]
                   [-Y]
```

or

```
mmvdisk pdisk replace  --prepare
                     --recovery-group RgName
                     {--pdisk PdiskName | --location LocationCode}
                     [--force-rg]
                     [--force]
```

or

```
mmvdisk pdisk replace --recovery-group RgName
                     {--pdisk PdiskName | --location LocationCode}
                     [--force-rg]
                     [--force-fru]
                     [-v {yes | no}]
```

or

```
mmvdisk pdisk replace --cancel
                     --recovery-group RgName
                     {--pdisk PdiskName | --location LocationCode}
                     [--force-rg]
```

or

```
mmvdisk pdisk change --recovery-group RgName
                     --pdisk PdiskName
                     [--identify {on | off}] |
                     [--clear-error-counters]
                     [--diagnose |
                     --suspend |
                     --resume |
                     --revive |
                     --revive-failing |
                     --revive-slow |
                     --begin-service-drain |
                     --end-service-drain |
                     --simulate-dead |
                     --simulate-failing]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

Use the **mmvdisk pdisk** command to manage **mmvdisk** recovery group pdisks for IBM Spectrum Scale RAID.

Use the `mmvdisk pdisk list` command to show the pdisk information for selected pdisks. Pdisks can be selected for listing based on recovery group, declustered array, or pdisk maintenance state. The **--not-ok** parameter lists pdisks that are not in service, and the **--replace** parameter lists pdisks that are marked for replacement.

By default, the `mmvdisk pdisk list` command prints a one-line summary for each listed pdisk. If the **-L** parameter is used, pdisks are listed in stanza format, which shows detailed pdisk attributes and is voluminous.

Use the `mmvdisk pdisk replace` command to perform pdisk replacement when an IBM Spectrum Scale RAID recovery group is reported to need service.

Replacing a pdisk requires the following three steps:

1. Run the `mmvdisk pdisk replace --prepare` command to prepare the pdisk for physical removal.
2. Physically remove the disk and replace it with a new disk of the same type.
3. Run the `mmvdisk pdisk replace command` to complete the replacement.

**Note:** New disks take the name of the replaced pdisks.
If replaced pdisks have not completely drained, they are given a temporary name that consists of the old pdisk name with a suffix of the form `#nnnn`. The temporary pdisk has the `adminDrain` pdisk state flag set and is deleted once drained. For example, a pdisk named `e1s05` receives a temporary name similar to `e1s05#0010` when the `adminDrain` state flag is set. This allows the new disk that is replacing it to be named `e1s05` immediately rather than waiting for the old disk to be completely drained and deleted. Until the draining and deleting process completes, both the new pdisk `e1s05` and the old pdisk `e1s05#0010` appear in the listing of the `mmvdisk pdisk list` and `mmvdisk recoveryroup list --pdisk` commands.

You can use the `mmvdisk pdisk replace --cancel` command to cancel the replacement of a pdisk that is prepared for replacement. Canceling a prepared replacement does not make the required replacement unnecessary.

The `mmvdisk pdisk change` command changes the runtime state of a pdisk.

**Attention:** Incorrect use of this command can cause file system or recovery group data to become unavailable or even unrecoverable. It should be used only in extreme situations under the guidance of IBM Service
.

Recovery group pdisks are part of the declustered arrays within recovery groups. You can add, list, change, and replace recovery group pdisks.

## Parameters

**mmvdisk pdisk list**
Lists pdisks.

**mmvdisk pdisk replace**
Replaces failed pdisks.

**mmvdisk pdisk change**
Changes pdisk states.

**Attention:** This is a low-level command that should be used only in extreme situations under the guidance of the IBM service team.

**--recovery-group** *all | RgName[,RgName...]*
Specifies the recovery group from which to list pdisks (when used with the `mmvdisk pdisk list` command).

**--declustered-array** *DaName*
Restricts pdisk listing to the specified declustered array.

**--pdisk** *PdiskName*
Specifies a pdisk by name.

**--replace**
Specifies that only pdisks that are marked for replacement are to be listed.

**--not-ok**
Specifies that only pdisks that are not functioning correctly are to be listed.

**--recovery-group** *RgName*
Specifies the recovery group for the target pdisk (when used with the `mmvdisk pdisk replace` and `mmvdisk pdisk change` commands).

**--prepare**
Prepares a pdisk for removal and replacement (when used with the `mmvdisk pdisk replace` command).

**--cancel**
Cancels the preparation of a pdisk for removal and replacement (when used with the `mmvdisk pdisk replace` command).

**--location** *LocationCode*
Specifies a pdisk by location code (when used with the `mmvdisk pdisk replace` command).

Pdisk location codes can be found in the long form stanza output of the `mmvdisk pdisk list` command.

**--force**

Permits preparation of a pdisk that is not marked for replacement (when used with the `mmvdisk pdisk replace --prepare` command).

**--force-rg**
Indicates that the pdisk that is being replaced is in a multi-disk carrier that also contains pdisks that are not in a recovery group or in a different recovery group (when used with the `mmvdisk pdisk replace` command).

**--force-fru**
Permits the replacement disk to have a different FRU (or type) than the removed pdisk (when used with the `mmvdisk pdisk replace` command).

**--v** *yes | no*
Specifies whether the replacement disk should be verified to be empty of IBM Spectrum Scale RAID data before it is incorporated into the recovery group. The default is *yes*, which verifies that the pdisk is empty (no data). This applies to the `mmvdisk pdisk replace` command.

**--identify** *on | off*
Specifies that tie identify light for the pdisk be turned `on` or `off`.

**--clear-error-counters**
Specifies that the pdisk error counters should be cleared.

**--diagnose**
Runs basic tests on the pdisk. If no problems are found, the pdisk state automatically returns to `ok`.

**--suspend**
Suspends I/O to the pdisk until a subsequent **--resume** command is used. If a pdisk remains in the

suspended state for longer than a predefined timeout period, IBM Spectrum Scale RAID begins rebuilding the data from that pdisk into spare space.

**Attention:** Use this parameter with caution and only when instructed to perform disk maintenance manually, bypassing the automatic system provided by using the `mmvdisk pdisk replace` command. If a pdisk is removed by using this option, vdisks that store data on the removed pdisk becomes temporarily degraded, requiring data that was stored on the removed pdisk to be rebuilt from redundancy. Also, if you try to remove more pdisks than the redundancy level of the least redundant vdisk in that declustered array, data becomes inaccessible. Therefore, when you prepare to remove a pdisk, use the **--begin-service-drain** and **--end-service-drain** parameters instead of this parameter.

**--resume**
Cancels a previously given **--suspend** command and resumes use of the pdisk.

**Attention:** Use this option only when instructed to perform disk maintenance manually, bypassing the automatic system provided by the `mmvdisk pdisk replace` command.

**--revive**
Attempts to make an out-of-service pdisk usable again by removing dead, failing, and read-only pdisk state flags. Data allocated on the disk that has not been rebuilt onto spare space can become readable again. However, any data that has already been reported as lost cannot be recovered.

**--revive-failing**
Similar to the **--revive** parameter, except that additional diagnostics particular to the failing pdisk state flag are attempted.

**--revive-slow**
Similar to the **--revive** parameter, except that additional diagnostics particular to the slow pdisk state flag are attempted.

**--begin-service-drain**
Starts draining the pdisk so that it can be temporarily removed. After you issue the command with this option, wait until the pdisk is completely drained before you remove the pdisk.

**Note:** This process requires that there be sufficient spare space in the declustered array for the data that is to be drained.

**--end-service-drain**
Starts returning drained data to a pdisk after it is brought back online.

**--simulate-dead**
Forces the failure of a pdisk by setting the `simulatedDead` pdisk state flag.

**Attention:** Use this parameter with caution. If the total number of failures in a declustered array exceeds the fault tolerance of any vdisk in that array, permanent data loss might result.

**--simulate-failing**
Forces the failure of a pdisk by setting the `simulatedFailing` pdisk state flag.

**Attention:** Use this parameter with caution. If the total number of failures in a declustered array exceeds the fault tolerance of any vdisk in that array, permanent data loss might result.

**-L** Specifies that the `mmvdisk pdisk list` command produce long form stanza output.

**-Y** Specifies that the `mmvdisk pdisk list` command produce colon-delimited raw output.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mmvdisk pdisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Example

To list pdisks in recovery group ESS01L:

```
mmvdisk pdisk list --recovery-group ESS01L
```

The system displays output similar to the following:

```
                            declustered
recovery group  pdisk           array     paths  capacity  free space  FRU (type)       state
--------------  ------------  ----------  -----  --------  ----------  ---------------  -----
ESS01L          e1d1s01       DA1             2  5588 GiB     216 GiB  38L6721          ok
ESS01L          e1d1s02       DA1             2  5588 GiB     216 GiB  38L6721          ok
ESS01L          e1d1s04       DA1             2  5588 GiB     216 GiB  38L6721          ok
ESS01L          e1d1s05       DA1             2  5588 GiB     216 GiB  38L6721          ok
...
```

## Example

To show all pdisks that are marked for replacement in all recovery groups:

```
mmvdisk pdisk list --replace --recovery-group all
```

The system displays output similar to the following:

```
recovery group  pdisk         priority  FRU (type)       location
--------------  ------------  --------  ---------------  --------
ESS01L          e3d4s06           5.78  38L6721          Enclosure SV50919775 Drawer 4 Slot 6
ESS01L          e6d1s02           5.78  38L6721          Enclosure SV50918970 Drawer 1 Slot 2

mmvdisk: A lower priority value means a higher need for replacement.
```

## Example

To prepare pdisk e6d1s02 in recovery group ESS01L for replacement:

```
mmvdisk pdisk replace --prepare --recovery-group ESS01L --pdisk e6d1s02
```

The system displays output similar to the following:

```
mmvdisk: Suspending pdisk e6d1s02 of RG ESS01L in location SV50918970-1-2.
mmvdisk: Location SV50918970-1-2 is Enclosure SV50918970 Drawer 1 Slot 2.
mmvdisk: Carrier released.
mmvdisk:
mmvdisk:   - Remove carrier.
mmvdisk:   - Replace disk in location SV50918970-1-2 with type '38L6721'.
mmvdisk:   - Reinsert carrier.
mmvdisk:   - Issue the following command:
mmvdisk:
mmvdisk:   mmvdisk pdisk replace --recovery-group ESS01L --pdisk 'e6d1s02'
```

## Example

To replace pdisk e6d1s02 of recovery group ESS01L after a new disk has been inserted:

```
mmvdisk pdisk replace --recovery-group ESS01L --pdisk e6d1s02
```

The system displays output similar to the following:

```
mmvdisk:
mmvdisk: Preparing a new pdisk for use may take many minutes.
mmvdisk:
mmvdisk: The following pdisks will be formatted on node ess01io1:
mmvdisk:       /dev/sdrk
mmvdisk:
mmvdisk: Location SV50918970-1-2 is Enclosure SV50918970 Drawer 1 Slot 2.
mmvdisk: Pdisk e6d1s02 of RG ESS01L successfully replaced.
mmvdisk: Resuming pdisk e6d1s02#047 of RG ESS01L.
mmvdisk: Carrier resumed.
```

## Example

To replace pdisk `e6d1s02` of recovery group `ESS01L` after a new disk has been inserted:

```
mmvdisk pdisk replace --recovery-group ESS01L --pdisk e6d1s02
```

The system displays output similar to the following:

```
mmvdisk:
mmvdisk: Preparing a new pdisk for use may take many minutes.
mmvdisk:
mmvdisk: The following pdisks will be formatted on node ess01io1:
mmvdisk:       /dev/sdrk
mmvdisk:
mmvdisk: Location SV50918970-1-2 is Enclosure SV50918970 Drawer 1 Slot 2.
mmvdisk: Pdisk e6d1s02 of RG ESS01L successfully replaced.
mmvdisk: Resuming pdisk e6d1s02#047 of RG ESS01L.
mmvdisk: Carrier resumed.
```

## See also

- "mmvdisk command" on page 218
- "mmvdisk server command" on page 225
- "mmvdisk recoverygroup command" on page 229
- "mmvdisk filesystem command" on page 238
- "mmvdisk vdiskset command" on page 243
- "mmvdisk vdisk command" on page 250
- "mmvdisk nodeclass command" on page 222

## Location

`/usr/lpp/mmfs/bin`

# Appendix C. IBM Spectrum Scale RAID scripts

This section includes descriptions of the IBM Spectrum Scale RAID scripts.

Descriptions of these scripts follow:



For information about IBM Spectrum Scale RAID commands, see Appendix B, "IBM Spectrum Scale RAID commands," on page 129.

For information about other IBM Spectrum Scale commands, see the *IBM Spectrum Scale: Command and Programming Reference*.

# chdrawer script

A service aid for replacing an ESS enclosure drawer.

## Synopsis

```
chdrawer  EnclosureSerialNumber DrawerNumber
     {--release  | --replace } [--dry-run]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **chdrawer** script acts as a service aid for replacing an ESS enclosure drawer. For more information, see *Replacing a failed ESS storage drawer: a sample scenario* in *Elastic Storage Server: Problem Determination Guide*.

## Parameters

*EnclosureSerialNumber*
: Specifies the enclosure serial number, as displayed by **mmlsenclosure**.

*DrawerNumber*
: Specifies the drawer number to be replaced. Drawers are numbered from top to bottom in an enclosure.

**--release**
: Prepares the drawer for disk replacement by suspending all pdisks.

**--replace**
: Resumes all the pdisks once the drawer has been replaced (with all the former pdisks in their corresponding slots).

**--dry-run**
: Runs the command without actually changing the pdisk states; checks whether there is enough redundancy to safely replace the drawer on a live system.

## Exit status

**0** Successful completion.

**nonzero**
: A failure has occurred.

## Security

You must have root authority to run the **chdrawer** script.

The node on which the script is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topic:

- "mmchenclosure command" on page 146

## Location

/usr/lpp/mmfs/samples/vdisk

# gnrhealthcheck script

Checks the general health of an ESS configuration.

## Synopsis

```
gnrhealthcheck  [--topology] [--enclosure] [--rg] [--pdisk]
                [--ipr] [--perf-dd] [--local]
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **gnrhealthcheck** script checks the general health of an ESS configuration.

## Parameters

**--topology**
Checks the operating system topology. Runs **mmgetpdisktopology** and **topsummary** to look for cabling and path issues.

**--enclosure**
Checks enclosures. Runs **mmlsenclosure** to look for failures.

**--rg**
Checks recovery groups. Runs **mmlsrecoverygroup** to check whether all recovery groups are active and whether the active server is the primary server. Also checks for any recovery groups that need service.

**--pdisk**
Checks pdisks. Runs **mmlspdisk** to check that each pdisk has two paths.

**--ipr**
Checks IBM Power RAID Array status. Runs **iprconfig** to check if the local RAID adapter is running "Optimized" or "Degraded". The ESS NVR pdisks are created on a RAID 10 array on this adapter. If one of the drives has failed, it will affect performance and should be replaced.

**--perf-dd**
Checks basic performance of disks. Runs a dd read to each potential IBM Spectrum Scale RAID disk drive for a GB and reports basic performance statistics. Reads are done six disks at a time. These statistics will only be meaningful if run on an idle system. Available on Linux only.

**--local**
Runs tests only on the invoking node.

The default is to check everything *except* **--perf-dd** arguments on all NSD server nodes.

## Exit status

**0** No problems were found.

**1** Problems were found and information was displayed.

**Note:** The default is to display to standard output. There could be a large amount of data, so it is recommended that you pipe the output to a file.

## Security

You must have root authority to run the **gnrhealthcheck** script.

The node on which the script is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

1. In this example, all checks are successful.

   To run a health check on the local server nodes and place output in `/tmp/gnrhealthcheck.out`, issue the following command:

   ```
   gnrhealthcheck --local | tee /tmp/gnrhealthcheck.out
   ```

   The system displays information similar to this:

   ```
   ############################################################
   # Beginning topology checks.
   ############################################################
   Topology checks successful.

   ############################################################
   # Beginning enclosure checks.
   ############################################################
   Enclosure checks successful.

   ############################################################
   # Beginning recovery group checks.
   ############################################################
   Recovery group checks successful.

   ############################################################
   # Beginning pdisk checks.
   ############################################################
   Pdisk group checks successful.

   ##########################################################
   # Beginning IBM Power RAID checks.
   ##################################################################IBM Power RAID checks successful.
   ```

2. In this example, several issues need to be investigated.

   To run a health check on the local server nodes and place output in `/tmp/gnrhealthcheck.out`, issue the following command:

   ```
   gnrhealthcheck --local | tee /tmp/gnrhealthcheck.out
   ```

   The system displays information similar to this:

   ```
   ##########################################################
   # Beginning topology checks.
   ##########################################################
   Found topology problems on node c45f01n01-ib0.gpfs.net

   DCS3700 enclosures found: 0123456789AB SV11812206 SV12616296 SV13306129
   Enclosure 0123456789AB (number 1):
   Enclosure 0123456789AB ESM A sg244[0379][scsi8 port 4] ESM B sg4[0379][scsi7 port 4]
   Enclosure 0123456789AB Drawer 1 ESM sg244 12 disks diskset "19968" ESM sg4 12 disks diskset "19968"
   Enclosure 0123456789AB Drawer 2 ESM sg244 12 disks diskset "11294" ESM sg4 12 disks diskset "11294"
   Enclosure 0123456789AB Drawer 3 ESM sg244 12 disks diskset "60155" ESM sg4 12 disks diskset "60155"
   Enclosure 0123456789AB Drawer 4 ESM sg244 12 disks diskset "03345" ESM sg4 12 disks diskset "03345"
   Enclosure 0123456789AB Drawer 5 ESM sg244 11 disks diskset "33625" ESM sg4 11 disks diskset "33625"
   Enclosure 0123456789AB sees 59 disks

   Enclosure SV12616296 (number 2):
   Enclosure SV12616296 ESM A sg63[0379][scsi7 port 3] ESM B sg3[0379][scsi9 port 4]
   Enclosure SV12616296 Drawer 1 ESM sg63 11 disks diskset "51519" ESM sg3 11 disks diskset "51519"
   Enclosure SV12616296 Drawer 2 ESM sg63 12 disks diskset "36246" ESM sg3 12 disks diskset "36246"
   Enclosure SV12616296 Drawer 3 ESM sg63 12 disks diskset "53750" ESM sg3 12 disks diskset "53750"
   Enclosure SV12616296 Drawer 4 ESM sg63 12 disks diskset "07471" ESM sg3 12 disks diskset "07471"
   Enclosure SV12616296 Drawer 5 ESM sg63 11 disks diskset "16033" ESM sg3 11 disks diskset "16033"
   Enclosure SV12616296 sees 58 disks
   ```

```
Enclosure SV11812206 (number 3):
Enclosure SV11812206 ESM A sg66[0379][scsi9 port 3] ESM B sg6[0379][scsi8 port 3]
Enclosure SV11812206 Drawer 1 ESM sg66 11 disks diskset "23334" ESM sg6 11 disks diskset "23334"
Enclosure SV11812206 Drawer 2 ESM sg66 12 disks diskset "16332" ESM sg6 12 disks diskset "16332"
Enclosure SV11812206 Drawer 3 ESM sg66 12 disks diskset "52806" ESM sg6 12 disks diskset "52806"
Enclosure SV11812206 Drawer 4 ESM sg66 12 disks diskset "28492" ESM sg6 12 disks diskset "28492"
Enclosure SV11812206 Drawer 5 ESM sg66 11 disks diskset "24964" ESM sg6 11 disks diskset "24964"
Enclosure SV11812206 sees 58 disks

Enclosure SV13306129 (number 4):
Enclosure SV13306129 ESM A sg64[0379][scsi8 port 2] ESM B sg353[0379][scsi7 port 2]
Enclosure SV13306129 Drawer 1 ESM sg64 11 disks diskset "47887" ESM sg353 11 disks diskset "47887"
Enclosure SV13306129 Drawer 2 ESM sg64 12 disks diskset "53906" ESM sg353 12 disks diskset "53906"
Enclosure SV13306129 Drawer 3 ESM sg64 12 disks diskset "35322" ESM sg353 12 disks diskset "35322"
Enclosure SV13306129 Drawer 4 ESM sg64 12 disks diskset "37055" ESM sg353 12 disks diskset "37055"
Enclosure SV13306129 Drawer 5 ESM sg64 11 disks diskset "16025" ESM sg353 11 disks diskset "16025"
Enclosure SV13306129 sees 58 disks

DCS3700 configuration: 4 enclosures, 1 SSD, 7 empty slots, 233 disks total
Location 0123456789AB-5-12 appears empty but should have an SSD
Location SV12616296-1-3 appears empty but should have an SSD
Location SV12616296-5-12 appears empty but should have an SSD
Location SV11812206-1-3 appears empty but should have an SSD
Location SV11812206-5-12 appears empty but should have an SSD

scsi7[07.00.00.00] 0000:11:00.0 [P2 SV13306129 ESM B (sg353)] [P3 SV12616296 ESM A (sg63)] [P4 0123456789AB ESM B (sg4)]
scsi8[07.00.00.00] 0000:8b:00.0 [P2 SV13306129 ESM A (sg64)] [P3 SV11812206 ESM B (sg6)] [P4 0123456789AB ESM A (sg244)]
scsi9[07.00.00.00] 0000:90:00.0 [P3 SV11812206 ESM A (sg66)] [P4 SV12616296 ESM B (sg3)]

############################################################
# Beginning enclosure checks.
############################################################
Enclosure checks successful.

############################################################
# Beginning recovery group checks.
############################################################
Found recovery group BB1RGR, primary server is not the active server.

############################################################
# Beginning pdisk checks.
############################################################
Found recovery group BB1RGL pdisk e4d5s06 has 0 paths.

##########################################################
# Beginning IBM Power RAID checks.
##########################################################
IBM Power RAID Array is running in degraded mode.

Name   PCI/SCSI Location          Description               Status
------ -------------------------  ------------------------- -----------------
       0007:90:00.0/0:            PCI-E SAS RAID Adapter    Operational
       0007:90:00.0/0:0:1:0       Advanced Function Disk    Failed
       0007:90:00.0/0:0:2:0       Advanced Function Disk    Active sda
       0007:90:00.0/0:2:0:0       RAID 10  Disk Array       Degraded
       0007:90:00.0/0:0:0:0       RAID 10  Array Member     Active
       0007:90:00.0/0:0:3:0       RAID 10  Array Member     Failed
       0007:90:00.0/0:0:4:0       Enclosure                 Active
       0007:90:00.0/0:0:6:0       Enclosure                 Active
       0007:90:00.0/0:0:7:0       Enclosure                 Active
```

## See also

See also the following *Elastic Storage Server: Problem Determination Guide* topic:

- *Checking the health of an ESS configuration: a sample scenario*

## Location

`/usr/lpp/mmfs/samples/vdisk`

# mkrginput script

Generates stanza files for recovery group creation from IBM Spectrum Scale RAID topology files.

## Synopsis

**mkrginput** *TopologyFile1 TopologyFile2* [**-s** | **-m** ] [ **--match percent** ]

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **mkrginput** script generates the stanza files needed for the creation of IBM Spectrum Scale RAID recovery groups by the **mmcrrecoverygroup** command. It takes as input the topology files generated by the **mmgetpdisktopology** command on the two servers that will share the recovery groups, and generates as output two stanza files for use by the **mmcrrecoverygroup** command.

The **mkrginput** script tries to verify that the two topology files are from different servers that share the same set of disk enclosures, and will exit with an error if this is not the case.

Despite this checking, it is imperative that the topology files first be examined with the **topsummary** command to ensure that both of them reflect the intended IBM Spectrum Scale RAID disk topology. If there are any disk or cabling errors, they should be corrected, and the topology files should be reacquired with **mmgetpdisktopology** and verified with **topsummary** before they are used with **mkrginput** to generate the stanza files.

After the topologies have been verified to be correct, they can be supplied as command arguments to **mkrginput**. The **mkrginput** command will then generate a left and right stanza file, each named with the serial number of the first disk enclosure. The left and right recovery groups can then be created by supplying the respective stanza files and servers in two separate **mmcrrecoverygroup** commands.

To create recovery group stanza files, follow these steps:

1. On the first server, `server1`, run:
   ```
   mmgetpdisktopology > server1.top
   ```
2. On the second server, `server2`, run:
   ```
   mmgetpdisktopology > server2.top
   ```
3. To verify the correctness of `server1`'s topology, run:
   ```
   topsummary server1.top
   ```
4. To verify the correctness of `server2`'s topology, run:
   ```
   topsummary server2.top
   ```
5. Repeat steps 1 - 4, correcting any disk subsystem problems until both topologies reflect the intended configuration.
6. Run:
   ```
   mkrginput server1.top server2.top
   GNR server disk topology: ESS GS6 HDD
   GNR recovery group layout: GS GSS/ESS RG
   Number of recovery groups: 2
   Created stanza file: G46503NL.stanza
   Created stanza file: G46503NR.stanza
   Stanza file contains '#%vdisk:' lines with log vdisk attributes.
   ```
   The **mkrginput** command identifies the server topology that was detected in the server topologies and the recovery group layout that is used for the detected topology. It prescribes that two recovery groups be created, and gives the names of the stanza files that have been created for them. The stanza

files will have commented out %vdisk stanzas for use after the recovery groups have been created, with suggested values for all the required log vdisks.

At this point look in the current directory for two files, one named *serialnumber*L.stanza and one named *serialnumber*R.stanza. Suppose the first enclosure in the topologies has the serial number G46503N. The **mkrginput** command will have created two files: G46503NL.stanza and G46503NR.stanza. Suppose the left recovery group will be named RGL and the right recovery group will be named RGR.

7. To create the left recovery group, RGL, run:

```
mmcrrecoverygroup RGL -F G46503NL.stanza --servers server1,server2
```

8. To create the right recovery group, RGR, run:

```
mmcrrecoverygroup RGR -F G46503NR.stanza --servers server2,server1
```

It is imperative that the disk topology is not changed between the capture of the topology files with **mmgetpdisktopology** and the creation of the two recovery groups with **mmcrrecoverygroup**. No server reboots or disk enclosure changes should be performed after the topologies have been verified as correct with **topsummary** until after recovery group creation.

After the recovery groups have been created, the commented-out %vdisk stanzas can be extracted from the recovery group input file and edited to include the recovery group names:

```
grep %vdisk G46503NL.stanza
#%vdisk: vdiskName={rg}LOGTIP rg={rg} da=NVR diskUsage=vdiskLogTip raidCode=2WayReplication blocksize=2m size=48m
#%vdisk: vdiskName={rg}LOGTIPBACKUP rg={rg} da=SSD diskUsage=vdiskLogTipBackup raidCode=Unreplicated blocksize=2m size=48m
#%vdisk: vdiskName={rg}LOGHOME rg={rg} da=DA1 diskUsage=vdiskLog raidCode=4WayReplication blocksize=2m size=60g
                  longTermEventLogSize=4m shortTermEventLogSize=4m fastWriteLogPct=90
```

The string {rg} should be replaced with the name of the recovery group that was created using this stanza file. The comment symbol # should be removed, and the resulting vdisk stanza file may be supplied to the **mmcrvdisk** command to create the required log vdisks for the recovery group. The process should be repeated for the second recovery group. See the "mmcrvdisk command" on page 162 command for more information.

## Parameters

***TopologyFile1***
Specifies the name of the topology file from the first server.

***TopologyFile2***
Specifies the name of the topology file from the second server.

**-s** Creates a single large declustered array. It is the default.

**-m** This is a deprecated legacy option to create multiple smaller declustered arrays. It should only be used to maintain compatibility with certain GSS and ESS server topologies where there are preexisting recovery groups with multiple declustered arrays. The default is **-s**.

**--match percent**
Specifies the minimum percentage match of the number of disks and their locations in the server topologies, as reported by the **topsummary** command. The default is 100, since it is not recommended to create recovery groups from imperfect server topologies. Valid values are 75 - 100, inclusive.

## Exit status

**0** Successful completion.

**nonzero**
A failure occurred.

## Security

You must have root authority to run the **mkrginput** script.

The node on which the script is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmcrrecoverygroup command" on page 159
- "mmcrvdisk command" on page 162
- "mmgetpdisktopology command" on page 181
- "topsummary script" on page 271

## Location

```
/usr/lpp/mmfs/samples/vdisk
```

# topselect script

Selects enclosures and disks from an IBM Spectrum Scale RAID topology file.

## Synopsis

```
topselect { -l | [ -d DiskEnclosure[,DiskEnclosure] ]
[ -a Adapter[,Adapter] ]  [ -p EnclosurePort[,EnclosurePort] ]
[-n] } TopologyFile
```

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **topselect** script provides a simple interface for examining the contents of a topology file produced by the **mmgetpdisktopology** command. It produces two types of output. One output type lists the supported enclosures that are found in the topology file and the other output type lists selected subsets of the disk block devices in the topology.

Together with the **topsummary** script, this script can be used to read information out of a IBM Spectrum Scale RAID topology file. It is sometimes useful to know which disk devices are connected to what hardware: for example, when looking to see if a pattern of disk faults might be isolated to a certain adapter.

You can list the enclosures within a topology file using the **-l** option:

```
topselect -l server1.top
Enclosure GSS SV34607290 found
Adapter scsi5 (naa.500605B00687D3A0) connects to SV34607290 ESM(s) B
Adapter scsi7 (naa.500605B006940730) connects to SV34607290 ESM(s) A
Enclosure GSS SV34607449 found
Adapter scsi3 (naa.500605B006BB9AD0) connects to SV34607449 ESM(s) B
Adapter scsi5 (naa.500605B00687D3A0) connects to SV34607449 ESM(s) A
```

Two disk enclosures are represented in the topology file `server1.top`. It shows that there are two connections to the enclosure with serial number `SV34607290`: one over adapter `scsi5` to ESM B and the other over adapter `scsi7` to ESM A. In parentheses after each adapter is the unique WWID for that adapter; this information can be useful for diagnostic purposes, together with the physical location codes and firmware levels displayed by the **topsummary** script.

The default behavior of the **topselect** script is to list disk block devices within the topology file. When used with no options, all the disk block devices contained within all disk enclosures are listed. The available IBM Spectrum Scale RAID log tip NVRAM devices can be included with the **-n** option. The listing of disk block devices can be narrowed down by using a combination of the options **-d** for disk enclosures, **-a** for adapters, and **-p** for enclosure ports (ESMs).

The information listed for disk block devices follows:

```
Enclosure:  Enclosure serial number
ESM port:   A or B
Adapter:    SCSI hostbus of the adapter
SES device: /dev name of the ESM expander controlling the disk
Device:     /dev name of the disk block device
Type:       SSD, HDD, or NVR
WWID:       Unique WWID of the disk
Location:   Disk location within the enclosure
```

For example, the `/dev/sdic` disk block device might appear in the output of **topselect** as:

```
SX33100383 B scsi4 sg249 sdic HDD 5000C5006C2C5837 21
```

## Parameters

**-l** Lists the enclosure connections present within the topology.

**-d *DiskEnclosure*[,*DiskEnclosure*]**
Lists the disk block devices in the specified disk enclosures.

**-a *Adapter*[,*Adapter*]**
Lists the disk block devices accessed over the specified adapters.

**-p *EnclosurePort*[,*EnclosurePort*]**
Lists the disk block devices accessed over the specified ESMs.

**-n** Includes the available NVRAM block devices present in the topology.

***TopologyFile***
Specifies the name of the topology file from which to select.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **topselect** script.

The node on which the script is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Examples

In the following examples, the underscores in the FC5887 disk location codes are used by the **topselect** script in place of literal space characters in the actual location code.

1. To select the disk block devices on ESM B of enclosure G46600Y in the server1.top topology file, run:

```
topselect -d G46600Y -p B server1.top
```

The system displays output similar to this:

```
> topselect -d G46600Y -p B server1.top

G46600Y B scsi6 sg232 sdgp HDD 5000C50067BC9C63 _P1-D5_____2SS6
G46600Y B scsi6 sg232 sdhd HDD 5000C50067BC9CA7 _P1-D19____2SS6
G46600Y B scsi6 sg232 sdgo HDD 5000C50067E9D6CB _P1-D4_____2SS6
G46600Y B scsi6 sg232 sdgq HDD 5000C50067E9D7A3 _P1-D6_____2SS6
G46600Y B scsi6 sg232 sdgt HDD 5000C50067E9DA17 _P1-D9_____2SS6
G46600Y B scsi6 sg232 sdhf HDD 5000C500686881FF _P1-D21____2SS6
G46600Y B scsi6 sg232 sdgw HDD 5000C50068688213 _P1-D12____2SS6
G46600Y B scsi6 sg232 sdgv HDD 5000C5006868827F _P1-D11____2SS6
G46600Y B scsi6 sg232 sdgu HDD 5000C500686884A3 _P1-D10____2SS6
G46600Y B scsi6 sg232 sdhb HDD 5000C50068688673 _P1-D17____2SS6
G46600Y B scsi6 sg232 sdgm HDD 5000C50068688793 _P1-D2_____2SS6
G46600Y B scsi6 sg232 sdgz HDD 5000C500686889E7 _P1-D15____2SS6
G46600Y B scsi6 sg232 sdhi HDD 5000C5006868947F _P1-D24____2SS6
G46600Y B scsi6 sg232 sdgx HDD 5000C5006868AA37 _P1-D13____2SS6
G46600Y B scsi6 sg232 sdgl HDD 5000C500686B84F7 _P1-D1_____2SS6
G46600Y B scsi6 sg232 sdgn HDD 5000C500686BAB57 _P1-D3_____2SS6
G46600Y B scsi6 sg232 sdha HDD 5000C500686BB0F3 _P1-D16____2SS6
G46600Y B scsi6 sg232 sdhc HDD 5000C500686BD21B _P1-D18____2SS6
G46600Y B scsi6 sg232 sdgy HDD 5000C500686BD333 _P1-D14____2SS6
```

```
G46600Y B scsi6 sg232 sdhg HDD 5000C500686BD387 _P1-D22_____2SS6
G46600Y B scsi6 sg232 sdhe HDD 5000C500686BF457 _P1-D20_____2SS6
G46600Y B scsi6 sg232 sdgs HDD 5000C500686BFF37 _P1-D8______2SS6
G46600Y B scsi6 sg232 sdhh HDD 5000C5006B9F0D97 _P1-D23_____2SS6
G46600Y B scsi6 sg232 sdgr HDD 5000C5006BA184EB _P1-D7______2SS6
```

2. To list all the disk block devices in the **server1.top** topology file and find only the disk with WWID 5000c500686bd333, use **topselect** in a pipeline with **grep**, run:

```
topselect server1.top | grep -i 5000c500686bd333
```

The system displays output similar to this:

```
> topselect server1.top | grep -i 5000c500686bd333

G46600Y B scsi6 sg232 sdgy HDD 5000C500686BD333 _P1-D14_____2SS6
G46600Y A scsi2 sg32 sdo HDD 5000C500686BD333 _P1-D14_____2SS6
```

This shows the two disk block device paths for the disk in slot 14 (D14).

3. To list the disk device paths found over HBA scsi9 and ESM A for the attached enclosure, run:

```
topselect -a scsi9 -p A server1.top
```

The system displays output similar to this:

```
> topselect -a scsi9 -p A server1.top

SV21313978 A scsi9 sg365 sdpb SSD 500051610005F8A8 5-12
SV21313978 A scsi9 sg365 sdmw SSD 500051610005F8EC 1-3
SV21313978 A scsi9 sg365 sdod HDD 5000C50034239FC7 3-12
SV21313978 A scsi9 sg365 sdnz HDD 5000CCA01B119598 3-8
.
.
.
SV21313978 A scsi9 sg365 sdni HDD 5000CCA01C515230 2-3
SV21313978 A scsi9 sg365 sdne HDD 5000CCA01C515294 1-11
.
.
.
```

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mmgetpdisktopology command" on page 181
- "mmlspdisk command" on page 199
- "topsummary script" on page 271.

## Location

```
/usr/lpp/mmfs/samples/vdisk
```

# topsummary script

Summarizes the contents of an IBM Spectrum Scale RAID disk subsystem topology file.

## Synopsis

**topsummary** *TopologyFile*

## Availability

Available on all IBM Spectrum Scale editions.

## Description

The **topsummary** script examines the topology file produced by the **mmgetpdisktopology** command and prints a concise summary of the disk enclosures and their cabling. Any discrepancies from what is expected will be noted. This is useful in verifying that IBM Spectrum Scale RAID servers are cabled correctly to the disk enclosures they manage. Discrepancies should be corrected and verified before creating IBM Spectrum Scale RAID recovery groups.

The typical scenario for using the **topsummary** script will be in the preparation and verification of a IBM Spectrum Scale RAID server building block. The servers are cabled to the disk enclosures according to specification. The **mmgetpdisktopology** command will be run to capture a topology file for each server. Then the **topsummary** script will be used on each topology file to verify that the servers are correctly cabled and that the expected disks are present. If no discrepancies are noted in either of the server disk topologies, the next step is to proceed to recovery group creation using **mkrginput** and **mmcrrecoverygroup**.

If discrepancies are found, **topsummary** describes the problem with information that should be useful in correcting it. Among the problems that **topsummary** detects are:

- Incorrect cabling
- Missing enclosures
- Missing disks or paths to disks.

If no errors are indicated, the topology matches a supported ESS / IBM Spectrum Scale RAID configuration. It is remotely possible, though, that this is still an unintended topology. For example, if enclosures 3 and 4 in an intended four-enclosure topology are missing, the remaining two enclosures could look exactly like a two-enclosure topology. It is imperative to know the intended structure of the IBM Spectrum Scale RAID configuration.

Enclosures are identified by cabling order and serial number. Disks are identified in sets based on the enclosure or drawer in which they reside. The "diskset" is indicated by a 5-digit checksum on the WWIDs of the disks in the set. Adapters are identified by their bus number and address or slot location. The firmware levels of adapters and enclosure ESMs are also provided for easy reference. The total number of enclosures and disks is indicated. The source of the adapter firmware levels found in the topology file is indicated by "[cli]" if the topology was acquired using the adapter CLI, or by "[sys]" if only sysfs information was available.

The total number of enclosures and disks is indicated. The name of the matching topology is provided with a "match" metric that indicates the degree to which the disk locations and contents match the named topology. A match of "100/100" means that all the expected disks were found in the expected locations.

## Parameters

***TopologyFile***
Specifies the name of the topology file to summarize.

## Exit status

**0** Successful completion.

**nonzero**
A failure has occurred.

## Security

You must have root authority to run the **topsummary** script.

The node on which the script is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For additional details, see the following *IBM Spectrum Scale RAID: Administration* topic: "Requirements for administering IBM Spectrum Scale RAID" on page 11.

## Example

The following command example shows how to acquire and summarize the topology file on an IBM Spectrum Scale RAID server:

```
mmgetpdisktopology > server1.top

topsummary server1.top
GNR server: name server1.ibm.net arch ppc64 model 8284-22A serial 022168C6V
GNR enclosures found: G46503N G465013 G46502A G46600J G46600G G46600Y
Enclosure G46503N (IBM 5887, number 1):
Enclosure G46503N ESM A sg307[60RG][scsi7 port 4] ESM B sg207[60RG][scsi5 port 4]
Enclosure G46503N ESM sg307 24 disks diskset "02977" ESM sg207 24 disks diskset "02977"
Enclosure G46503N sees 24 disks (2 SSDs, 22 HDDs)

Enclosure G465013 (IBM 5887, number undetermined):
Enclosure G465013 ESM B sg107[60RG][scsi3 port 4] ESM A not found
Enclosure G465013 ESM sg107 24 disks diskset "28956"
Enclosure G465013 sees 24 disks (0 SSDs, 24 HDDs)

Enclosure G46502A (IBM 5887, number 3):
Enclosure G46502A ESM A sg82[60RG][scsi3 port 3] ESM B sg282[60QG][scsi7 port 3]
Enclosure G46502A ESM sg82 24 disks diskset "41537" ESM sg282 24 disks diskset "41537"
Enclosure G46502A sees 24 disks (0 SSDs, 24 HDDs)

Enclosure G46600J (IBM 5887, number 4):
Enclosure G46600J ESM A sg257[60RG][scsi6 port 2] ESM B sg157[60RG][scsi4 port 2]
Enclosure G46600J ESM sg257 24 disks diskset "33442" ESM sg157 24 disks diskset "33442"
Enclosure G46600J sees 24 disks (0 SSDs, 24 HDDs)

Enclosure G46600G (IBM 5887, number 5):
Enclosure G46600G ESM A sg132[60RG][scsi4 port 1] ESM B sg57[60RG][scsi0 port 2]
Enclosure G46600G ESM sg132 24 disks diskset "20820" ESM sg57 24 disks diskset "20820"
Enclosure G46600G sees 24 disks (0 SSDs, 24 HDDs)

Enclosure G46600Y (IBM 5887, number 6):
Enclosure G46600Y ESM A sg32[60RG][scsi0 port 1] ESM B sg232[60RG][scsi6 port 1]
Enclosure G46600Y ESM sg32 24 disks diskset "34159" ESM sg232 24 disks diskset "34159"
Enclosure G46600Y sees 24 disks (0 SSDs, 24 HDDs)

GNR server disk topology: ESS GS6 HDD (match: 100/100)
GNR configuration: 6 enclosures, 2 SSDs, 0 empty slots, 144 disks total, 6 NVRAM partitions
Unable to determine enclosure order. Check the HBA to enclosure cabling.
Location G465013-1 appears only on the sg107 path
Location G465013-2 appears only on the sg107 path
```

```
Location G465013-3 appears only on the sg107 path
Location G465013-4 appears only on the sg107 path
Location G465013-5 appears only on the sg107 path
Location G465013-6 appears only on the sg107 path
Location G465013-7 appears only on the sg107 path
Location G465013-8 appears only on the sg107 path
Location G465013-9 appears only on the sg107 path
Location G465013-10 appears only on the sg107 path
Location G465013-11 appears only on the sg107 path
Location G465013-12 appears only on the sg107 path
Location G465013-13 appears only on the sg107 path
Location G465013-14 appears only on the sg107 path
Location G465013-15 appears only on the sg107 path
Location G465013-16 appears only on the sg107 path
Location G465013-17 appears only on the sg107 path
Location G465013-18 appears only on the sg107 path
Location G465013-19 appears only on the sg107 path
Location G465013-20 appears only on the sg107 path
Location G465013-21 appears only on the sg107 path
Location G465013-22 appears only on the sg107 path
Location G465013-23 appears only on the sg107 path
Location G465013-24 appears only on the sg107 path

Slot C2 HBA model LSISAS2308 firmware[cli] 20.00.02.00 bios[cli] 07.37.00.00 uefi[cli] 07.26.01.00
Slot C2 HBA scsi4 U78CB.001.WZS01V4-P1-C2-T1 [P1 G46600G ESM A (sg132)] [P2 G46600J ESM B (sg157)]
Slot C2 HBA scsi5 U78CB.001.WZS01V4-P1-C2-T2 [P4 G46503N ESM B (sg207)]
Slot C3 HBA model LSISAS2308 firmware[cli] 20.00.02.00 bios[cli] 07.37.00.00 uefi[cli] 07.26.01.00
Slot C3 HBA scsi6 U78CB.001.WZS01V4-P1-C3-T1 [P1 G46600Y ESM B (sg232)] [P2 G46600J ESM A (sg257)]
Slot C3 HBA scsi7 U78CB.001.WZS01V4-P1-C3-T2 [P3 G46502A ESM B (sg282)] [P4 G46503N ESM A (sg307)]
Slot C11 HBA model LSISAS2308 firmware[cli] 20.00.02.00 bios[cli] 07.37.00.00 uefi[cli] 07.26.01.00
Slot C11 HBA scsi0 U78CB.001.WZS01V4-P1-C11-T1 [P1 G46600Y ESM A (sg32)] [P2 G46600G ESM B (sg57)]
Slot C11 HBA scsi3 U78CB.001.WZS01V4-P1-C11-T2 [P3 G46502A ESM A (sg82)] [P4 G465013 ESM B (sg107)]
```

This shows an ESS GS6 HDD topology. This is a six-enclosure topology with 2 SSDs and 142 HDDs. For illustration purposes, this topology contains a single cabling error. The error is indicated by the undetermined enclosure order, the message that enclosure `G465013` ESM A is not found, and by 24 disks that can only be found over one path. The cabling error can also be seen in the missing P3 connection on the adapter in slot C2. The problem here is the missing ESM A connection from enclosure `G465013` to port 3 of the adapter in slot C2. (When an enclosure's ESM A is connected to port 3 of slot C2 and its ESM B is connected to port 4 of slot C11, only then will **topsummary** conclude that it is enclosure number 2 in an ESS GS6 HDD topology.)

A close look at the firmware levels shows all of the adapter firmware levels to have been acquired using the adapter CLI, with base firmware at level `20.00.02.00`, BIOS firmware at level `07.37.00.00`, and UEFI firmware at level `07.26.01.00`. But one of the ESMs, enclosure `G46502A` ESM B, is at a lower level (`60QG`) than the others (`60RG`). This is not necessarily an error, but is worth taking note of, and ESS provides other tools for managing firmware levels.

## See also

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- "mkrginput script" on page 265
- "mmcrrecoverygroup command" on page 159
- "mmgetpdisktopology command" on page 181
- "mmlsenclosure command" on page 192
- "mmlsfirmware command" on page 196
- "topselect script" on page 268.

## Location

/usr/lpp/mmfs/samples/vdisk

# Appendix D. Setting up GNR on the Power 775 Disk Enclosure

The IBM Power 775 Disk Enclosure includes version 3.5 of GPFS Native RAID (GNR). This topic includes sample scenarios for setting up GNR and replacing disks on the Power 775 Disk Enclosure.

## Disk replacement recording and reporting

The disk hospital keeps track of disks that require replacement according to the disk replacement policy of the declustered array, and it can be configured to report the need for replacement in a variety of ways. It records and reports the FRU number and physical hardware location of failed disks to help guide service personnel to the correct location with replacement disks.

If the storage JBOD supports multiple disks that are mounted on a removable carrier, such as the Power 775, disk replacement requires the hospital to suspend other disks in the same carrier temporarily. On the Power 775 storage JBOD, the disk carriers are also not removable until GNR actuates a solenoid-controlled latch, in order to guard against human error.

In response to administrative commands, the hospital quiesces the appropriate disk (or multiple disks on a carrier), releases the carrier latch solenoid (if necessary), and turns on identify lights to guide replacement. After one or more disks are replaced and the disk or carrier is re-inserted, the hospital, in response to administrative commands, verifies that the repair has taken place and adds any new disks to the declustered array automatically, which causes GPFS Native RAID to rebalance the tracks and spare space across all of the disks of the declustered array. If service personnel fail to re-insert the disk or carrier within a reasonable period, the hospital declares the disks missing and starts rebuilding the affected data.

## Configuring GNR recovery groups: a sample scenario

This topic provides a detailed example of configuring GNR using the JBOD SAS disks on the Power 775 Disk Enclosure.

The example considers one fully populated Power 775 Disk Enclosure cabled to two recovery group servers, and shows how the architecture of the Power 775 Disk Enclosure determines the structure of the recovery groups. Throughout this topic, it may be helpful to have Power 775 Disk Enclosure documentation at hand.

## Preparing recovery group servers

### Disk enclosure and HBA cabling

The Power 775 Disk Enclosure should be cabled to the intended recovery group servers according to the Power 775 Disk Enclosure hardware installation instructions. The fully populated Power 775 Disk Enclosure consists of 8 STORs of 48 disks, for a total of 384 JBOD disks. Each STOR provides redundant left and right port cards for host server HBA connections (STOR is short for *physical storage group*, meaning the part of the disk enclosure controlled by a pair of port cards). To ensure proper multi-pathing and redundancy, each recovery group server must be connected to each port card using different HBAs. For example, STOR 1 has port cards P1-C4 and P1-C5. Server 1 may be connected to P1-C4 using HBA `hba1` and to P1-C5 using HBA `hba2`; similarly for server 2 and its respective HBAs `hba1` and `hba2`.

GNR provides system administration tools for verifying the correct connectivity of the Power 775 Disk Enclosure, which will be seen later during the operating system preparation.

When the port cards of the Power 775 Disk Enclosure have been cabled to the appropriate HBAs of the two recovery group servers, the Power 775 Disk Enclosure should be powered on and the recovery group servers should be rebooted.

## Initial operating system verification

Preparation then continues with the operating system, which must be either AIX 7.1 or Red Hat Enterprise Linux 6.1, and which must be the same on both recovery group servers. It is not necessary to do a complete verification of the Power 775 Disk Enclosure connectivity at this point. Logging in to the servers to perform a quick check that at least some disks have been detected and configured by the operating system will suffice. The operating system device configuration should be examined for the Power 775 Disk Enclosure VPD enclosure type, which is 78AD.001.

One way to quickly verify that AIX has configured devices with enclosure type 78AD.001 for the Power 775 Disk Enclosure is:

```
# lsdev -t ses -F 'name physloc parent' | grep 78AD.001
```

The output should include lines resembling the following:

```
ses12 U78AD.001.000DE37-P1-C4 sas3
```

This is the SAS expander device on port card P1-C4 of the Power 775 Disk Enclosure with serial number 000DE37, together with the SAS protocol device driver `sas3` under which it has been configured. To see what disks have been detected by the SAS protocol driver, use:

```
# lsdev -p sas3
```

The output should include all the disks and port card expanders that successfully configured under the `sas3` SAS protocol driver (which corresponds to the HBA device `mpt2sas3`).

If AIX has not configured any port card expanders of enclosure type 78AD.001, the hardware installation of the server HBAs and the Power 775 Disk Enclosure must be reexamined and corrected.

One way to quickly verify that Red Hat Enterprise Linux has configured devices with enclosure type 78AD.001 for the Power 775 Disk Enclosure is:

```
# grep 78AD.001 /proc/scsi/scsi
```

The output should include lines resembling the following:

```
Vendor: IBM      Model: 78AD-001         Rev: 0150

Further examination of the /proc/scsi/scsi file should reveal contents similar to:
Host: scsi7 Channel: 00 Id: 394 Lun: 00
  Vendor: IBM      Model: ST9600204SS      Rev: 631C
  Type:   Direct-Access                    ANSI  SCSI revision: 06
Host: scsi7 Channel: 00 Id: 395 Lun: 00
  Vendor: IBM      Model: 78AD-001         Rev: 0150
  Type:   Enclosure                        ANSI  SCSI revision: 04
```

The above indicates that a Power 775 Disk Enclosure port card SAS expander and a disk drive have been configured on SCSI host bus 7 (the HBA corresponding to `scsi7`).

As with AIX, if Linux has not configured any port card expanders of enclosure type 78AD.001, the hardware installation of the server HBAs and the Power 775 Disk Enclosure must be reexamined and corrected.

## Disabling operating system multi-pathing

When it has been verified that at least some of the Power 775 Disk Enclosure has been configured by the operating system, the next step is to disable any operating system multi-pathing. Because GNR performs its own disk multi-pathing, AIX MPIO (Multiple Path I/O) and Linux DMM (Device Mapper Multipath) must be disabled as appropriate.

To disable AIX MPIO for SAS disks, use:

```
# manage_disk_drivers -d SAS_SCSD -o AIX_non_MPIO
```

To disable Linux DMM, use:

```
# chkconfig --del multipathd
```

**Note:** This blanket disabling of operating system multi-pathing is appropriate because a Power 775 Disk Enclosure installation provides the only available disk devices to the recovery group servers. When operating system multi-pathing has been disabled, the recovery group servers should be rebooted.

## Operating system device attributes

For best performance, the operating system disk device driver should be configured to allow GNR I/O operations to be made with one disk access, rather than being fragmented. Under AIX this is controlled by the **max_transfer** attribute of the HBAs and disk devices. Under Red Hat Enterprise Linux, this is controlled by the disk block device **max_sectors_kb** attribute.

The disk I/O size performed by GNR depends on the strip size of the RAID code of the vdisk NSD. This in turn is related to the vdisk track size and its corresponding GPFS file system block size. The operating system I/O size should be equal to or greater than the largest strip size of the planned vdisk NSDs.

Because GNR stores checksums with each strip, strips have an additional 4 KiB or 8 KiB than might be expected just from the user data (strips containing 2 MiB of user data have an additional 8 KiB; all smaller strips have an additional 4 KiB). The strip size for a replicated vdisk RAID code is equal to the vdisk track size plus the size of the checksum. The strip size for a Reed-Solomon vdisk RAID code is equal to one-eighth of the vdisk track size plus the size of the checksum.

The default **max_transfer** value of 1 MiB under AIX is suitable for GNR vdisk strip sizes under 1 MiB.

The default **max_sectors_kb** value of 512 KiB sectors under Red Hat Enterprise Linux is suitable for GNR vdisk strip sizes under 512 KiB.

However, for vdisk strip sizes greater than 1 MiB under AIX or greater than 512 KiB under Linux, the operating system disk device driver I/O size should be increased for best performance.

The following table indicates the relationship between file system NSD block size, vdisk track size, vdisk RAID code, vdisk strip size, and the non-default operating system I/O size for all permitted GNR vdisks. The AIX **max_transfer** attribute is specified in hexadecimal, and the only allowable values greater than the 1 MiB default are 0x200000 (2 MiB) and 0x400000 (4 MiB). Linux **max_sectors_kb** is specified as the desired I/O size in KiB.

*Table 22. NSD block size, vdisk track size, vdisk RAID code, vdisk strip size, and non-default operating system I/O size for permitted GNR vdisks*

| NSD block size | vdisk track size | vdisk RAID code | RAID code strip size | AIX max_transfer | Linux max_sectors_kb |
|---|---|---|---|---|---|
| 256 KiB | 256 KiB | 3- or 4-way replication | 260 KiB | default | 260 |

*Table 22. NSD block size, vdisk track size, vdisk RAID code, vdisk strip size, and non-default operating system I/O size for permitted GNR vdisks (continued)*

| NSD block size | vdisk track size | vdisk RAID code | RAID code strip size | AIX max_transfer | Linux max_sectors_kb |
|---|---|---|---|---|---|
| 512 KiB | 512 KiB | 3- or 4-way replication | 516 KiB | default | 516 |
| 1 MiB | 1 MiB | 3- or 4-way replication | 1028 KiB | 0x200000 | 1028 |
| 2 MiB | 2 MiB | 3- or 4-way replication | 2056 KiB | 0x400000 | 2056 |
| 512 KiB | 512 KiB | 8 + 2p or 8 + 3p | 68 KiB | default | default |
| 1 MiB | 1 MiB | 8 + 2p or 8 + 3p | 132 KiB | default | default |
| 2 MiB | 2 MiB | 8 + 2p or 8 + 3p | 260 KiB | default | 260 |
| 4 MiB | 4 MiB | 8 + 2p or 8 + 3p | 516 KiB | default | 516 |
| 8 MiB | 8 MiB | 8 + 2p or 8 + 3p | 1028 KiB | 0x200000 | 1028 |
| 16 MiB | 16 MiB | 8 + 2p or 8 + 3p | 2056 KiB | 0x400000 | 2056 |

If the largest strip size of all the vdisk NSDs planned for a GNR installation exceeds the operating system default I/O size, the operating system I/O size should be changed.

**AIX** Under AIX, this involves changing the HBA and disk device **max_transfer** size. For an HBA to accommodate an increased **max_transfer** size, the **max_commands** for the HBA will also need to be decreased. With a 0x400000 **max_transfer** size, the four-port HBA requires a **max_commands** value of 124 and the two-port HBA requires a **max_commands** value of 248.

To change the **max_transfer** attribute to 4 MiB for the HBA `mpt2sas0` under AIX, use the following command:

```
# chdev -P -l mpt2sas0 -a max_transfer=0x400000
```

To change the **max_commands** value to 124 for the four-port HBA `mpt2sas0` (AIX device type 001072001410f60), use the following command:

```
# chdev -P -l mpt2sas0 -a max_commands=124
```

To change the **max_commands** value to 248 for the two-port HBA `mpt2sas0` (AIX device type 001072001410ea0), use the following command:

```
# chdev -P -l mpt2sas0 -a max_commands=248
```

Repeat the previous commands for each HBA.

Changing the hdisk **max_transfer** attribute requires changing its default value for AIX device type nonmpioscsd disks. It is not sufficient to change the **max_transfer** for individual hdisks, because performing disk replacement deletes and recreates hdisk objects.

To change the default hdisk **max_transfer** attribute for type nonmpioscsd hdisks, use the following command:

```
# chdef -a max_transfer=0x400000 -c disk -s sas -t nonmpioscsd
```

The new **max_transfer** and **max_commands** values will not take effect until AIX reconfigures the HBAs and hdisks. This can be done either by rebooting the recovery group server, or by deconfiguring (but not removing) and then reconfiguring the affected HBAs. The **max_transfer** and **max_commands** attribute are recorded in the CuAt ODM class and will persist across reboots.

**Linux** Under Linux, the **max_sectors_kb** I/O size is reset to the default each time a disk block device is configured. This means that upon reboot and upon adding or replacing disks, a desired

nondefault I/O size must be explicitly set. Because **max_sectors_kb** is dynamically reconfigurable, GPFS Native RAID manages this setting under Linux.

The value of **max_sectors_kb** under Linux is set on all the disks in a recovery group when GPFS Native RAID begins serving the recovery group, and on disks that are configured as part of GPFS Native RAID disk replacement.

The default **max_sectors_kb** set by GPFS Native RAID is 4096, which is large enough for any of the strip sizes listed in Table 22 on page 277. It can be changed to exactly match the largest strip size in use by GPFS Native RAID by setting the GPFS **nsdRAIDBlockDeviceMaxSectorsKB** configuration parameter.

To set the **max_sectors_kb** value used by GPFS Native RAID to 2056, use the following command:

```
# mmchconfig nsdRAIDBlockDeviceMaxSectorsKB=2056
```

The new value will take effect the next time GPFS is started. To have the new value take effect immediately, append the **-i** option to the above command.

For optimal performance, additional device attributes may need to be changed (for example, the HBA and block device command queue depths); consult the operating system documentation for the device attributes.

## Verifying that a Power 775 Disk Enclosure is configured correctly

When a superficial inspection indicates that the Power 775 Disk Enclosure has been configured on the recovery group servers, and especially after operating system multi-pathing has been disabled, it is necessary to perform a thorough discovery of the disk topology on each server.

To proceed, GPFS must be installed on the recovery group servers, and they should be members of the same GPFS cluster. Consult the *IBM Spectrum Scale: Administration Guide* for instructions for creating a GPFS cluster.

GNR provides tools in `/usr/lpp/mmfs/samples/vdisk` for collecting and collating information on any attached Power 775 Disk Enclosure and for verifying that the detected topology is correct. The **mmgetpdisktopology** command examines the operating system's list of connected devices and produces a colon-delimited database with a line for each discovered Power 775 Disk Enclosure physical disk, port card expander device, and HBA. **mmgetpdisktopology** should be run on each of the two intended recovery group server nodes, and the results examined to verify that the disk enclosure hardware and software configuration is as expected. An additional tool called **topsummary** concisely summarizes the output of the **mmgetpdisktopology** command.

Create a directory in which to work, and then capture the output of the **mmgetpdisktopology** command from each of the two intended recovery group server nodes:

```
# mkdir p7ihde
# cd p7ihde
# ssh server1 /usr/lpp/mmfs/bin/mmgetpdisktopology > server1.top
# ssh server2 /usr/lpp/mmfs/bin/mmgetpdisktopology > server2.top
```

Then view the summary for each of the nodes (`server1` example shown):

```
# /usr/lpp/mmfs/samples/vdisk/topsummary server1.top
P7IH-DE enclosures found: DE00022
Enclosure DE00022:
Enclosure DE00022 STOR P1-C4/P1-C5 sees both portcards: P1-C4 P1-C5
Portcard P1-C4: ses0[0150]/mpt2sas0/24 diskset "37993" ses1[0150]/mpt2sas0/24 diskset "18793"
Portcard P1-C5: ses4[0150]/mpt2sas1/24 diskset "37993" ses5[0150]/mpt2sas1/24 diskset "18793"
Enclosure DE00022 STOR P1-C4/P1-C5 sees 48 disks
Enclosure DE00022 STOR P1-C12/P1-C13 sees both portcards: P1-C12 P1-C13
Portcard P1-C12: ses8[0150]/mpt2sas2/24 diskset "40657" ses9[0150]/mpt2sas2/24 diskset "44382"
```

```
Portcard P1-C13: ses12[0150]/mpt2sas3/24 diskset "40657" ses13[0150]/mpt2sas3/24 diskset "44382"
Enclosure DE00022 STOR P1-C12/P1-C13 sees 48 disks
Enclosure DE00022 STOR P1-C20/P1-C21 sees both portcards: P1-C20 P1-C21
Portcard P1-C20: ses16[0150]/mpt2sas4/24 diskset "04091" ses17[0150]/mpt2sas4/24 diskset "31579"
Portcard P1-C21: ses20[0150]/mpt2sas5/24 diskset "04091" ses21[0150]/mpt2sas5/24 diskset "31579"
Enclosure DE00022 STOR P1-C20/P1-C21 sees 48 disks
Enclosure DE00022 STOR P1-C28/P1-C29 sees both portcards: P1-C28 P1-C29
Portcard P1-C28: ses24[0150]/mpt2sas6/24 diskset "64504" ses25[0150]/mpt2sas6/24 diskset "62361"
Portcard P1-C29: ses28[0150]/mpt2sas7/24 diskset "64504" ses29[0150]/mpt2sas7/24 diskset "62361"
Enclosure DE00022 STOR P1-C28/P1-C29 sees 48 disks
Enclosure DE00022 STOR P1-C60/P1-C61 sees both portcards: P1-C60 P1-C61
Portcard P1-C60: ses30[0150]/mpt2sas7/24 diskset "10913" ses31[0150]/mpt2sas7/24 diskset "52799"
Portcard P1-C61: ses26[0150]/mpt2sas6/24 diskset "10913" ses27[0150]/mpt2sas6/24 diskset "52799"
Enclosure DE00022 STOR P1-C60/P1-C61 sees 48 disks
Enclosure DE00022 STOR P1-C68/P1-C69 sees both portcards: P1-C68 P1-C69
Portcard P1-C68: ses22[0150]/mpt2sas5/24 diskset "50112" ses23[0150]/mpt2sas5/24 diskset "63400"
Portcard P1-C69: ses18[0150]/mpt2sas4/24 diskset "50112" ses19[0150]/mpt2sas4/24 diskset "63400"
Enclosure DE00022 STOR P1-C68/P1-C69 sees 48 disks
Enclosure DE00022 STOR P1-C76/P1-C77 sees both portcards: P1-C76 P1-C77
Portcard P1-C76: ses14[0150]/mpt2sas3/23 diskset "45948" ses15[0150]/mpt2sas3/24 diskset "50856"
Portcard P1-C77: ses10[0150]/mpt2sas2/24 diskset "37258" ses11[0150]/mpt2sas2/24 diskset "50856"
Enclosure DE00022 STOR P1-C76/P1-C77 sees 48 disks
Enclosure DE00022 STOR P1-C84/P1-C85 sees both portcards: P1-C84 P1-C85
Portcard P1-C84: ses6[0150]/mpt2sas1/24 diskset "13325" ses7[0150]/mpt2sas1/24 diskset "10443"
Portcard P1-C85: ses2[0150]/mpt2sas0/24 diskset "13325" ses3[0150]/mpt2sas0/24 diskset "10443"
Enclosure DE00022 STOR P1-C84/P1-C85 sees 48 disks
Carrier location P1-C79-D4 appears only on the portcard P1-C77 path
Enclosure DE00022 sees 384 disks

mpt2sas7[1005470001] U78A9.001.9998884-P1-C1 DE00022 STOR 4 P1-C29 (ses28 ses29) STOR 5 P1-C60 (ses30 ses31)
mpt2sas6[1005470001] U78A9.001.9998884-P1-C3 DE00022 STOR 4 P1-C28 (ses24 ses25) STOR 5 P1-C61 (ses26 ses27)
mpt2sas5[1005470001] U78A9.001.9998884-P1-C5 DE00022 STOR 3 P1-C21 (ses20 ses22) STOR 6 P1-C68 (ses21 ses23)
mpt2sas4[1005470001] U78A9.001.9998884-P1-C7 DE00022 STOR 3 P1-C20 (ses16 ses17) STOR 6 P1-C69 (ses18 ses19)
mpt2sas3[1005470001] U78A9.001.9998884-P1-C9 DE00022 STOR 2 P1-C13 (ses12 ses13) STOR 7 P1-C76 (ses14 ses15)
mpt2sas2[1005470001] U78A9.001.9998884-P1-C11 DE00022 STOR 2 P1-C12 (ses8 ses9) STOR 7 P1-C77 (ses10 ses11)
mpt2sas1[1005470001] U78A9.001.9998884-P1-C13 DE00022 STOR 1 P1-C5 (ses4 ses5) STOR 8 P1-C84 (ses6 ses7)
mpt2sas0[1005470001] U78A9.001.9998884-P1-C15 DE00022 STOR 1 P1-C4 (ses0 ses1) STOR 8 P1-C85 (ses2 ses3)
```

In the preceding output, the Power 775 Disk Enclosure with serial number DE00022 is discovered, together with its eight individual STORs and the component port cards, port card expanders (with their firmware levels in brackets), and physical disks. One minor discrepancy is noted: The physical disk in location P1-C79-D4 is only seen over one of the two expected HBA paths. This can also be seen in the output for the STOR with port cards P1-C76 and P1-C77:

```
Enclosure DE00022 STOR P1-C76/P1-C77 sees both portcards: P1-C76 P1-C77
Portcard P1-C76: ses14[0150]/mpt2sas3/23 diskset "45948" ses15[0150]/mpt2sas3/24 diskset "50856"
Portcard P1-C77: ses10[0150]/mpt2sas2/24 diskset "37258" ses11[0150]/mpt2sas2/24 diskset "50856"
Enclosure DE00022 STOR P1-C76/P1-C77 sees 48 disks
```

Here the connection through port card P1-C76 sees just 23 disks on the expander `ses14` and all 24 disks on the expander `ses15`, while the connection through port card P1-C77 sees all 24 disks on each of the expanders `ses10` and `ses11`. The “disksets” that are reached over the expanders are identified by a checksum of the unique SCSI WWNs of the physical disks that are present; equal disksets represent the same collection of physical disks.

The preceding discrepancy can either be corrected or ignored, as it is probably due to a poorly seated or defective port on the physical disk. The disk is still present on the other port.

If other discrepancies are noted (for example, physical disks that are expected but do not show up at all, or SSDs or HDDs in the wrong locations), they should be corrected before proceeding.

The HBAs (firmware levels in brackets) are also listed with their slot location codes to show the cabling pattern. Each HBA sees two STORs, and each STOR is seen by two different HBAs, which provides the multiple paths and redundancy required by a correct Power 775 Disk Enclosure installation.

This output can be compared to the hardware cabling specification to verify that the disk enclosure is connected correctly.

The `server2.top` topology database should also be examined with the **topsummary** sample script and verified to be correct.

When the Power 775 Disk Enclosure topologies are verified to be correct on both intended recovery group server nodes, the recommended recovery group configuration can be created using GNR commands.

# Creating recovery groups on a Power 775 Disk Enclosure

## Configuring GPFS nodes to be recovery group servers

Before a GPFS node can create and serve recovery groups, it must be configured with a vdisk track cache. This is accomplished by setting the **nsdRAIDTracks** configuration parameter.

**nsdRAIDTracks** is the GPFS configuration parameter essential to define a GPFS cluster node as a recovery group server. It specifies the number of vdisk tracks of which the attributes will be held in memory by the GPFS daemon on the recovery group server.

The actual contents of the vdisk tracks, the user data and the checksums, are stored in the standard GPFS page pool. Therefore, the size of the GPFS page pool configured on a recovery group server should be considerable, on the order of tens of gigabytes. The amount of page pool dedicated to hold vdisk track data is governed by the **nsdRAIDBufferPoolSizePct** parameter, which defaults to 50%. In practice, a recovery group server will not need to use the GPFS page pool for any significant amount of standard file caching, and the **nsdRAIDBufferPoolSizePct** value can be increased to 80%. Also applicable, since a recovery group server is by definition an NSD server, is the **nsdBufSpace** parameter, which defaults to 30% of page pool. Since the vdisk buffer pool doubles as the NSD buffer spool, the **nsdBufSpace** parameter should be decreased to its minimum of 10%. Together these values leave only 10% of the page pool for application program file cache, but this should not be a problem as a recovery group server should not be running application programs.

In this example, the recovery group servers will be configured to cache the information on 16384 vdisk tracks and to have 64 GiB of page pool, of which 80% will be used for vdisk data. Once the configuration changes are made, the servers will need to be restarted.

```
# mmchconfig nsdRAIDTracks=16384,nsdRAIDBufferPoolSizePct=80,nsdBufSpace=10,pagepool=64G -N server1,server2
# mmshutdown -N server1,server2
# mmstartup -N server1,server2
```

## Defining the recovery group layout

The definition of recovery groups on a Power 775 Disk Enclosure is dictated by the architecture and cabling of the disk enclosure. Two servers sharing a Power 775 Disk Enclosure implies two recovery groups; one is served by one node and one by the other, and each server acts as the other's backup. Half the disks in each STOR should belong to one recovery group, and half to the other. One recovery group will therefore be defined on the disks and carriers in the top halves of the eight STORs, and one on the bottom halves. Since the disks in a STOR are placed four to a removable carrier, thereby having a common point of failure, each disk in a carrier should belong to one of four different declustered arrays. Should a carrier fail or be removed, then each declustered array will only suffer the loss of one disk. There are four SSDs distributed among the top set of carriers, and four in the bottom set of carriers. These groups of four SSDs will make up the vdisk log declustered arrays in their respective halves.

GNR provides a tool that understands the layout of the Power 775 Disk Enclosure and will automatically generate the **mmcrrecoverygroup** stanza files for creating the top and bottom recovery groups. `/usr/lpp/mmfs/samples/vdisk/mkp7rginput`, when supplied with output of the **mmgetpdisktopology** command, will create recovery group stanza files for the top and bottom halves of each Power 775 Disk Enclosure found in the topology.

Each recovery group server, though it may see the same functional disk enclosure topology, will almost certainly differ in the particulars of which disk device names (e.g., `/dev/rhdisk77` on AIX or `/dev/sdax` on Linux) refer to which physical disks in what disk enclosure location.

There are two possibilities then for creating the recovery group stanza files and the recovery groups themselves:

**Alternative 1:**
Generate the recovery group stanza files and create the recovery groups from the perspective of just one of the servers as if that server were to be primary for both recovery groups, and then use the **mmchrecoverygroup** command to swap the primary and backup servers for one of the recovery groups

**Alternative 2:**
Generate the recovery group stanza files for each server's primary recovery group using the primary server's topology file.

This example will show both alternatives.

**Creating the recovery groups, alternative 1**

To create the recovery group input stanza files from the perspective of `server1`, run:

```
# /usr/lpp/mmfs/samples/vdisk/mkp7rginput server1.top
```

This will create two files for each disk enclosure present in the `server1` topology; in this case, `DE00022TOP.server1` for the top half of disk enclosure DE00022 and `DE00022BOT.server2` for the bottom half. (An extra file, `DEXXXXXbad`, may be created if any discrepancies are present in the topology; if such a file is created by **mkp7rginput**, it should be examined and the discrepancies corrected.)

The recovery group stanza files will follow the recommended best practice for the Power 775 Disk Enclosure of defining in each half of the disk enclosure a separate declustered array of 4 SSDs for recovery group transaction logging, and four file system data declustered arrays using the regular HDDs according to which of the four disk enclosure carrier slots each HDD resides in.

The defaults are accepted for other recovery group declustered array parameters such as scrub duration, spare space, and disk replacement policy.

The stanza file will look something like this:

```
# head DE00022TOP.server1
%pdisk: pdiskName=c081d1
        device=/dev/hdisk10
        da=DA1
        nPathActive=2
        nPathTotal=4
%pdisk: pdiskName=c065d1
        device=/dev/hdisk211
        da=DA1
        nPathActive=2
        nPathTotal=4
%pdisk: pdiskName=c066d1
        device=/dev/hdisk259
        da=DA1
        nPathActive=2
        nPathTotal=4
```

All the pdisk stanzas for declustered array DA1 will be listed first, followed by those for DA2, DA3, DA4, and the LOG declustered array. The pdisk names will indicate the carrier and disk location in which the physical disk resides. Notice that only one block device path to the disk is given; the second path will be discovered automatically soon after the recovery group is created.

Now that the DE00022TOP.server1 and DE00022BOT.server1 stanza files have been created from the perspective of recovery group server node server1, these two recovery groups can be created using two separate invocations of the **mmcrrecoverygroup** command:

```
# mmcrrecoverygroup DE00022TOP -F DE00022TOP.server1 --servers server1,server2
mmcrrecoverygroup: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.

# mmcrrecoverygroup DE00022BOT -F DE00022BOT.server1 --servers server1,server2
mmcrrecoverygroup: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

Note that both recovery groups were created with server1 as primary and server2 as backup. It is now necessary to swap the primary and backup servers for DE00022BOT using the **mmchrecoverygroup** command:

```
# mmchrecoverygroup DE00022BOT --servers server2,server1
mmchrecoverygroup: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

GNR will automatically discover the appropriate disk devices on server2.

**Creating the recovery groups, alternative 2**

To create the recovery groups from the start with the intended primary and backup servers, the stanza files from both server topologies will need to be created.

To create the server1 recovery group input stanza files, run:

```
# /usr/lpp/mmfs/samples/vdisk/mkp7rginput server1.top
```

To create the server2 recovery group input stanza files, run:

```
# /usr/lpp/mmfs/samples/vdisk/mkp7rginput server2.top
```

These two commands will result in four stanza files: DE00022TOP.server1, DE00022BOT.server1, DE00022TOP.server2, and DE00022BOT.server2. (As in alternative 1, if any files named DEXXXXXbad are created, they should be examined and the errors within should be corrected.)

The DE00022TOP recovery group must then be created using server1 as the primary and the DE00022TOP.server1 stanza file. The DE00022BOT recovery group must be created using server2 as the primary and the DE00022BOT.server2 stanza file.

```
# mmcrrecoverygroup DE00022TOP -F DE00022TOP.server1 --servers server1,server2
mmcrrecoverygroup: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.

# mmcrrecoverygroup DE00022BOT -F DE00022BOT.server2 --servers server2,server1
mmcrrecoverygroup: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

Because each recovery group was created using the intended primary server and the stanza file for that server, it is not necessary to swap the primary and backup servers.

## Verifying recovery group creation

Use the **mmlsrecoverygroup** command to verify that each recovery group was created:

```
# mmlsrecoverygroup DE00022TOP -L

                   declustered
 recovery group       arrays     vdisks  pdisks
 -----------------  -----------  ------  ------
 DE00022TOP                   5       0     192
```

```
 declustered   needs                            replace              scrub       background activity
    array     service  vdisks  pdisks  spares  threshold  free space  duration  task   progress  priority
 -----------  -------  ------  ------  ------  ---------  ----------  --------  -------------------------
 DA1          no            0      47       2          2      24 TiB   14 days  inactive     0%  low
 DA2          no            0      47       2          2      24 TiB   14 days  inactive     0%  low
 DA3          no            0      47       2          2      24 TiB   14 days  inactive     0%  low
 DA4          no            0      47       2          2      24 TiB   14 days  inactive     0%  low
 LOG          no            0       4       1          1     558 GiB   14 days  inactive     0%  low

                                      declustered
 vdisk               RAID code           array     vdisk size  remarks
 ------------------  ------------------  -----------  ----------  -------

 active recovery group server                    servers
 ----------------------------------------------  -------
 server1                                         server1,server2

# mmlsrecoverygroup DE00022BOT -L

                   declustered
 recovery group       arrays     vdisks  pdisks
 -----------------  -----------  ------  ------
 DE00022BOT                 5       0     192

 declustered   needs                            replace              scrub       background activity
    array     service  vdisks  pdisks  spares  threshold  free space  duration  task   progress  priority
 -----------  -------  ------  ------  ------  ---------  ----------  --------  -------------------------
 DA1          no            0      47       2          2      24 TiB   14 days  inactive     0%  low
 DA2          no            0      47       2          2      24 TiB   14 days  inactive     0%  low
 DA3          no            0      47       2          2      24 TiB   14 days  inactive     0%  low
 DA4          no            0      47       2          2      24 TiB   14 days  inactive     0%  low
 LOG          no            0       4       1          1     558 GiB   14 days  inactive     0%  low

                                      declustered
 vdisk               RAID code           array     vdisk size  remarks
 ------------------  ------------------  -----------  ----------  -------

 active recovery group server                    servers
 ----------------------------------------------  -------
 server1                                         server2,server1
```

Notice that the vdisk sections of the newly created recovery groups are empty; the next step is to create the vdisks.

## Defining and creating the vdisks

Once the recovery groups are created and being served by their respective servers, it is time to create the vdisks using the **mmcrvdisk** command.

Each recovery group requires a single log vdisk for recording RAID updates and diagnostic information. This is internal to the recovery group, cannot be used for user data, and should be the only vdisk in the LOG declustered array. The log vdisks in this example use 3-way replication in order to fit in the LOG declustered array, which contains 4 SSDs and spare space equivalent to one disk.

Data vdisks are required to be defined in the four data declustered arrays for use as file system NSDs. In this example, each of the declustered arrays for file system data is divided into two vdisks with different characteristics: one using 4-way replication and a 1 MiB block size and a total vdisk size of 250 GiB suitable for file system metadata, and one using Reed-Solomon 8 + 3p encoding and a 16 MiB block size suitable for file system data. The vdisk size is omitted for the Reed-Solomon vdisks, meaning that they will default to use the remaining non-spare space in the declustered array (for this to work, any vdisks with specified total sizes must be defined first).

The possibilities for the vdisk creation stanza file are quite great, depending on the number and type of vdisk NSDs required for the number and type of file systems desired, so the vdisk stanza file will need to be created by hand, possibly following a template.

In this example, a single stanza file, `mmcrvdisk.DE00022ALL`, is used. The single file contains the specifications for all the vdisks in both the `DE00022TOP` and `DE00022BOT` recovery groups. Here is what the example stanza file for use with **mmcrvdisk** should look like:

```
# cat mmcrvdisk.DE00022ALL
%vdisk: vdiskName=DE00022TOPLOG
        rg=DE00022TOP
        da=LOG
        blocksize=1m
        size=4g
        raidCode=3WayReplication
        diskUsage=vdiskLog
%vdisk: vdiskName=DE00022BOTLOG
        rg=DE00022BOT
        da=LOG
        blocksize=1m
        size=4g
        raidCode=3WayReplication
        diskUsage=vdiskLog
%vdisk: vdiskName=DE00022TOPDA1META
        rg=DE00022TOP
        da=DA1
        blocksize=1m
        size=250g
        raidCode=4WayReplication
        diskUsage=metadataOnly
        failureGroup=22
        pool=system
%vdisk: vdiskName=DE00022TOPDA1DATA
        rg=DE00022TOP
        da=DA1
        blocksize=16m
        raidCode=8+3p
        diskUsage=dataOnly
        failureGroup=22
        pool=data
%vdisk: vdiskName=DE00022BOTDA1META
        rg=DE00022BOT
        da=DA1
        blocksize=1m
        size=250g
        raidCode=4WayReplication
        diskUsage=metadataOnly
        failureGroup=22
        pool=system
%vdisk: vdiskName=DE00022BOTDA1DATA
        rg=DE00022BOT
        da=DA1
        blocksize=16m
        raidCode=8+3p
        diskUsage=dataOnly
        failureGroup=22
        pool=data
[DA2, DA3, DA4 vdisks omitted.]
```

Notice how the file system metadata vdisks are flagged for eventual file system usage as **metadataOnly** and for placement in the `system` storage pool, and the file system data vdisks are flagged for eventual **dataOnly** usage in the `data` storage pool. (After the file system is created, a policy will be required to allocate file system data to the correct storage pools; see “Creating the GPFS file system” on page 287.)

Importantly, also notice that block sizes for the file system metadata and file system data vdisks must be specified at this time, may not later be changed, and must match the block sizes supplied to the eventual **mmcrfs** command.

Notice also that the eventual `failureGroup=22` value for the NSDs on the file system vdisks is the same for vdisks in both the `DE00022TOP` and `DE00022BOT` recovery groups. This is because the recovery groups, although they have different servers, still share a common point of failure in the disk enclosure `DE00022`, and GPFS should be informed of this through a distinct failure group designation for each disk enclosure. It is up to the GPFS system administrator to decide upon the failure group numbers for each Power 775 Disk Enclosure in the GPFS cluster.

To create the vdisks specified in the `mmcrvdisk.DE00022ALL` file, use the following **mmcrvdisk** command:

```
# mmcrvdisk -F mmcrvdisk.DE00022ALL
mmcrvdisk: [I] Processing vdisk DE00022TOPLOG
mmcrvdisk: [I] Processing vdisk DE00022BOTLOG
mmcrvdisk: [I] Processing vdisk DE00022TOPDA1META
mmcrvdisk: [I] Processing vdisk DE00022TOPDA1DATA
mmcrvdisk: [I] Processing vdisk DE00022TOPDA2META
mmcrvdisk: [I] Processing vdisk DE00022TOPDA2DATA
mmcrvdisk: [I] Processing vdisk DE00022TOPDA3META
mmcrvdisk: [I] Processing vdisk DE00022TOPDA3DATA
mmcrvdisk: [I] Processing vdisk DE00022TOPDA4META
mmcrvdisk: [I] Processing vdisk DE00022TOPDA4DATA
mmcrvdisk: [I] Processing vdisk DE00022BOTDA1META
mmcrvdisk: [I] Processing vdisk DE00022BOTDA1DATA
mmcrvdisk: [I] Processing vdisk DE00022BOTDA2META
mmcrvdisk: [I] Processing vdisk DE00022BOTDA2DATA
mmcrvdisk: [I] Processing vdisk DE00022BOTDA3META
mmcrvdisk: [I] Processing vdisk DE00022BOTDA3DATA
mmcrvdisk: [I] Processing vdisk DE00022BOTDA4META
mmcrvdisk: [I] Processing vdisk DE00022BOTDA4DATA
mmcrvdisk: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

Creation of the vdisks can be verified through the **mmlsvdisk** command (the **mmlsrecoverygroup** command can also be used):

```
# mmlsvdisk
                                                  declustered  block size
 vdisk name          RAID code        recovery group         array       in KiB    remarks
 ------------------  ---------------  ------------------  -----------  ----------  -------
 DE00022BOTDA1DATA   8+3p             DE00022BOT          DA1               16384
 DE00022BOTDA1META   4WayReplication  DE00022BOT          DA1                1024
 DE00022BOTDA2DATA   8+3p             DE00022BOT          DA2               16384
 DE00022BOTDA2META   4WayReplication  DE00022BOT          DA2                1024
 DE00022BOTDA3DATA   8+3p             DE00022BOT          DA3               16384
 DE00022BOTDA3META   4WayReplication  DE00022BOT          DA3                1024
 DE00022BOTDA4DATA   8+3p             DE00022BOT          DA4               16384
 DE00022BOTDA4META   4WayReplication  DE00022BOT          DA4                1024
 DE00022BOTLOG       3WayReplication  DE00022BOT          LOG                1024  log
 DE00022TOPDA1DATA   8+3p             DE00022TOP          DA1               16384
 DE00022TOPDA1META   4WayReplication  DE00022TOP          DA1                1024
 DE00022TOPDA2DATA   8+3p             DE00022TOP          DA2               16384
 DE00022TOPDA2META   4WayReplication  DE00022TOP          DA2                1024
 DE00022TOPDA3DATA   8+3p             DE00022TOP          DA3               16384
 DE00022TOPDA3META   4WayReplication  DE00022TOP          DA3                1024
 DE00022TOPDA4DATA   8+3p             DE00022TOP          DA4               16384
 DE00022TOPDA4META   4WayReplication  DE00022TOP          DA4                1024
 DE00022TOPLOG       3WayReplication  DE00022TOP          LOG                1024  log
```

## Creating NSDs from vdisks

The **mmcrvdisk** command rewrites the input file so that it is ready to be passed to the **mmcrnsd** command that creates the NSDs from which GPFS builds file systems. To create the vdisk NSDs, run the **mmcrnsd** command on the rewritten **mmcrvdisk** stanza file:

```
# mmcrnsd -F mmcrvdisk.DE00022ALL
mmcrnsd: Processing disk DE00022TOPDA1META
mmcrnsd: Processing disk DE00022TOPDA1DATA
mmcrnsd: Processing disk DE00022TOPDA2META
mmcrnsd: Processing disk DE00022TOPDA2DATA
mmcrnsd: Processing disk DE00022TOPDA3META
mmcrnsd: Processing disk DE00022TOPDA3DATA
mmcrnsd: Processing disk DE00022TOPDA4META
mmcrnsd: Processing disk DE00022TOPDA4DATA
mmcrnsd: Processing disk DE00022BOTDA1META
mmcrnsd: Processing disk DE00022BOTDA1DATA
mmcrnsd: Processing disk DE00022BOTDA2META
mmcrnsd: Processing disk DE00022BOTDA2DATA
mmcrnsd: Processing disk DE00022BOTDA3META
mmcrnsd: Processing disk DE00022BOTDA3DATA
mmcrnsd: Processing disk DE00022BOTDA4META
mmcrnsd: Processing disk DE00022BOTDA4DATA
mmcrnsd: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

Notice how the recovery group log vdisks are omitted from NSD processing.

The **mmcrnsd** command then once again rewrites the stanza file in preparation for use as input to the **mmcrfs** command.

## Creating the GPFS file system

Run the **mmcrfs** command to create the file system:

```
# mmcrfs gpfs -F mmcrvdisk.DE00022ALL -B 16m --metadata-block-size 1m -T /gpfs -A no

The following disks of gpfs will be formatted on node c250f09c01ap05.ppd.pok.ibm.com:
    DE00022TOPDA1META: size 262163456 KB
    DE00022TOPDA1DATA: size 8395522048 KB
    DE00022TOPDA2META: size 262163456 KB
    DE00022TOPDA2DATA: size 8395522048 KB
    DE00022TOPDA3META: size 262163456 KB
    DE00022TOPDA3DATA: size 8395522048 KB
    DE00022TOPDA4META: size 262163456 KB
    DE00022TOPDA4DATA: size 8395522048 KB
    DE00022BOTDA1META: size 262163456 KB
    DE00022BOTDA1DATA: size 8395522048 KB
    DE00022BOTDA2META: size 262163456 KB
    DE00022BOTDA2DATA: size 8395522048 KB
    DE00022BOTDA3META: size 262163456 KB
    DE00022BOTDA3DATA: size 8395522048 KB
    DE00022BOTDA4META: size 262163456 KB
    DE00022BOTDA4DATA: size 8395522048 KB
Formatting file system ...
Disks up to size 2.5 TB can be added to storage pool 'system'.
Disks up to size 79 TB can be added to storage pool 'data'.
Creating Inode File
Creating Allocation Maps
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool 'system'
Formatting Allocation Map for storage pool 'data'
Completed creation of file system /dev/gpfs.
mmcrfs: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

Notice how the 16 MiB data block size is specified with the traditional **-B** parameter and the 1 MiB metadata block size is specified with the **--metadata-block-size** parameter. Since a file system with different metadata and data block sizes requires the use of multiple GPFS storage pools, a file system placement policy is needed to direct user file data to the data storage pool. In this example, the file placement policy is simple:

```
# cat policy
rule 'default' set pool 'data'
```

The policy must then be installed in the file system by using the **mmchpolicy** command:

```
# mmchpolicy gpfs policy -I yes
Validated policy `policy': parsed 1 Placement Rules, 0 Restore Rules, 0 Migrate/Delete/Exclude Rules,
        0 List Rules, 0 External Pool/List Rules
Policy `policy' installed and broadcast to all nodes.
```

If a policy is not placed in a file system with multiple storage pools, attempts to place data into files will return ENOSPC as if the file system were full.

This file system, which is built on a Power 775 Disk Enclosure by using two recovery groups, two recovery group servers, eight file system metadata vdisk NSDs and eight file system data vdisk NSDs, can now be mounted and placed into service:

```
# mmmount gpfs -a
```

# Replacing failed disks in a Power 775 Disk Enclosure recovery group: a sample scenario

The scenario presented here shows how to detect and replace failed disks in a recovery group built on a Power 775 Disk Enclosure.

## Detecting failed disks in your enclosure

Assume a fully-populated Power 775 Disk Enclosure (serial number 000DE37) on which the following two recovery groups are defined:

- 000DE37TOP containing the disks in the top set of carriers
- 000DE37BOT containing the disks in the bottom set of carriers

Each recovery group contains the following:

- one log declustered array (LOG)
- four data declustered arrays (DA1, DA2, DA3, DA4)

The data declustered arrays are defined according to Power 775 Disk Enclosure best practice as follows:

- 47 pdisks per data declustered array
- each member pdisk from the same carrier slot
- default disk replacement threshold value set to 2

The replacement threshold of 2 means that GNR will only require disk replacement when two or more disks have failed in the declustered array; otherwise, rebuilding onto spare space or reconstruction from redundancy will be used to supply affected data.

This configuration can be seen in the output of **mmlsrecoverygroup** for the recovery groups, shown here for 000DE37TOP:

```
# mmlsrecoverygroup 000DE37TOP -L

                    declustered
 recovery group       arrays     vdisks  pdisks
 -----------------  -----------  ------  ------
```

```
000DE37TOP                   5       9     192

declustered   needs                            replace                scrub       background activity
   array     service  vdisks  pdisks  spares  threshold  free space  duration  task   progress  priority
-----------  -------  ------  ------  ------  ---------  ----------  --------  -------------------------
DA1          no            2      47       2          2   3072 MiB   14 days  scrub       63%  low
DA2          no            2      47       2          2   3072 MiB   14 days  scrub       19%  low
DA3          yes           2      47       2          2      0   B   14 days  rebuild-2r  48%  low
DA4          no            2      47       2          2   3072 MiB   14 days  scrub       33%  low
LOG          no            1       4       1          1    546 GiB   14 days  scrub       87%  low

                                    declustered
vdisk               RAID code          array     vdisk size  remarks
------------------  -----------------  -----------  ----------  -------
000DE37TOPLOG       3WayReplication    LOG            4144 MiB  log
000DE37TOPDA1META   4WayReplication    DA1             250 GiB
000DE37TOPDA1DATA   8+3p               DA1              17 TiB
000DE37TOPDA2META   4WayReplication    DA2             250 GiB
000DE37TOPDA2DATA   8+3p               DA2              17 TiB
000DE37TOPDA3META   4WayReplication    DA3             250 GiB
000DE37TOPDA3DATA   8+3p               DA3              17 TiB
000DE37TOPDA4META   4WayReplication    DA4             250 GiB
000DE37TOPDA4DATA   8+3p               DA4              17 TiB

active recovery group server                    servers
----------------------------------------------  -------
server1                                         server1,server2
```

The indication that disk replacement is called for in this recovery group is the value of `yes` in the `needs service` column for declustered array `DA3`.

The fact that `DA3` (the declustered array on the disks in carrier slot 3) is undergoing rebuild of its RAID tracks that can tolerate two strip failures is by itself not an indication that disk replacement is required; it merely indicates that data from a failed disk is being rebuilt onto spare space. Only if the replacement threshold has been met will disks be marked for replacement and the declustered array marked as needing service.

GNR provides several indications that disk replacement is required:

- entries in the AIX error report or the Linux syslog
- the **pdReplacePdisk** callback, which can be configured to run an administrator-supplied script at the moment a pdisk is marked for replacement
- the POWER7® cluster event notification TEAL agent, which can be configured to send disk replacement notices when they occur to the POWER7 cluster EMS
- the output from the following commands, which may be performed from the command line on any GPFS cluster node (see the examples that follow):
  1. **mmlsrecoverygroup** with the **-L** flag shows `yes` in the `needs service` column
  2. **mmlsrecoverygroup** with the **-L** and **--pdisk** flags; this shows the states of all pdisks, which may be examined for the `replace` pdisk state
  3. **mmlspdisk** with the **--replace** flag, which lists only those pdisks that are marked for replacement

**Note:** Because the output of **mmlsrecoverygroup -L --pdisk** for a fully-populated disk enclosure is very long, this example shows only some of the pdisks (but includes those marked for replacement).

```
# mmlsrecoverygroup 000DE37TOP -L --pdisk

                  declustered
recovery group      arrays     vdisks  pdisks
----------------  -----------  ------  ------
000DE37TOP                 5       9     192

declustered   needs                            replace                scrub       background activity
   array     service  vdisks  pdisks  spares  threshold  free space  duration  task   progress  priority
```

```
 -----------  -------  ------  ------  ------  ---------  ----------  --------  -------------------------
 DA1          no            2      47       2          2    3072 MiB   14 days  scrub       63%  low
 DA2          no            2      47       2          2    3072 MiB   14 days  scrub       19%  low
 DA3          yes           2      47       2          2       0   B   14 days  rebuild-2r  68%  low
 DA4          no            2      47       2          2    3072 MiB   14 days  scrub       34%  low
 LOG          no            1       4       1          1     546 GiB   14 days  scrub       87%  low

                    n. active,   declustered                user     state,
pdisk               total paths     array     free space   condition  remarks
------------------  -----------  -----------  ----------  ----------- -------
[...]
c014d1                2,  4      DA1             62 GiB  normal      ok
c014d2                2,  4      DA2            279 GiB  normal      ok
c014d3                0,  0      DA3            279 GiB  replaceable dead/systemDrain/noRGD/noVCD/replace
c014d4                2,  4      DA4             12 GiB  normal      ok
[...]
c018d1                2,  4      DA1             24 GiB  normal      ok
c018d2                2,  4      DA2             24 GiB  normal      ok
c018d3                2,  4      DA3            558 GiB  replaceable dead/systemDrain/noRGD/noVCD/noData/replace
c018d4                2,  4      DA4             12 GiB  normal      ok
[...]
```

The preceding output shows that the following pdisks are marked for replacement:

- c014d3 in DA3
- c018d3 in DA3

The naming convention used during recovery group creation indicates that these are the disks in slot 3 of carriers 14 and 18. To confirm the physical locations of the failed disks, use the **mmlspdisk** command to list information about those pdisks in declustered array DA3 of recovery group 000DE37TOP that are marked for replacement:

```
# mmlspdisk 000DE37TOP --declustered-array DA3 --replace
pdisk:
   replacementPriority = 1.00
   name = "c014d3"
   device = "/dev/rhdisk158,/dev/rhdisk62"
   recoveryGroup = "000DE37TOP"
   declusteredArray = "DA3"
   state = "dead/systemDrain/noRGD/noVCD/replace"
   .
   .
   .
pdisk:
   replacementPriority = 1.00
   name = "c018d3"
   device = "/dev/rhdisk630,/dev/rhdisk726"
   recoveryGroup = "000DE37TOP"
   declusteredArray = "DA3"
   state = "dead/systemDrain/noRGD/noVCD/noData/replace"
   .
   .
   .
```

The preceding location code attributes confirm the pdisk naming convention:

| Disk | Location code | Interpretation |
|---|---|---|
| pdisk c014d3 | 78AD.001.000DE37-C14-D3 | Disk 3 in carrier 14 in the disk enclosure identified by enclosure type 78AD.001 and serial number 000DE37 |
| pdisk c018d3 | 78AD.001.000DE37-C18-D3 | Disk 3 in carrier 18 in the disk enclosure identified by enclosure type 78AD.001 and serial number 000DE37 |

## Replacing the failed disks in a Power 775 Disk Enclosure recovery group

**Note:** In this example, it is assumed that two new disks with the appropriate Field Replaceable Unit (FRU) code, as indicated by the `fru` attribute (`74Y4936` in this case), have been obtained as replacements for the failed pdisks `c014d3` and `c018d3`.

Replacing each disk is a three-step process:

1. Using the **mmchcarrier** command with the **--release** flag to suspend use of the other disks in the carrier and to release the carrier.
2. Removing the carrier and replacing the failed disk within with a new one.
3. Using the **mmchcarrier** command with the **--replace** flag to resume use of the suspended disks and to begin use of the new disk.

GNR assigns a priority to pdisk replacement. Disks with smaller values for the **replacementPriority** attribute should be replaced first. In this example, the only failed disks are in `DA3` and both have the same **replacementPriority**.

Disk `c014d3` is chosen to be replaced first.

1. To release carrier 14 in disk enclosure `000DE37`:

```
# mmchcarrier 000DE37TOP --release --pdisk c014d3
  [I] Suspending pdisk c014d1 of RG 000DE37TOP in location 78AD.001.000DE37-C14-D1.
  [I] Suspending pdisk c014d2 of RG 000DE37TOP in location 78AD.001.000DE37-C14-D2.
  [I] Suspending pdisk c014d3 of RG 000DE37TOP in location 78AD.001.000DE37-C14-D3.
  [I] Suspending pdisk c014d4 of RG 000DE37TOP in location 78AD.001.000DE37-C14-D4.
  [I] Carrier released.

    - Remove carrier.
    - Replace disk in location 78AD.001.000DE37-C14-D3 with FRU 74Y4936.
    - Reinsert carrier.
    - Issue the following command:

        mmchcarrier 000DE37TOP --replace --pdisk 'c014d3'

      Repair timer is running.  Perform the above within 5 minutes
      to avoid pdisks being reported as missing.
```

   GNR issues instructions as to the physical actions that must be taken. Note that disks may be suspended only so long before they are declared missing; therefore the mechanical process of physically performing disk replacement must be accomplished promptly.

   Use of the other three disks in carrier 14 has been suspended, and carrier 14 is unlocked. The identify lights for carrier 14 and for disk 3 are on.

2. Carrier 14 should be unlatched and removed. The failed disk 3, as indicated by the internal identify light, should be removed, and the new disk with FRU `74Y4936` should be inserted in its place. Carrier 14 should then be reinserted and the latch closed.
3. To finish the replacement of pdisk `c014d3`:

```
# mmchcarrier 000DE37TOP --replace --pdisk c014d3
[I] The following pdisks will be formatted on node server1:
    /dev/rhdisk354
[I] Pdisk c014d3 of RG 000DE37TOP successfully replaced.
[I] Resuming pdisk c014d1 of RG 000DE37TOP.
[I] Resuming pdisk c014d2 of RG 000DE37TOP.
[I] Resuming pdisk c014d3#162 of RG 000DE37TOP.
[I] Resuming pdisk c014d4 of RG 000DE37TOP.
[I] Carrier resumed.
```

When the **mmchcarrier --replace** command returns successfully, GNR has resumed use of the other 3 disks. The failed pdisk may remain in a temporary form (indicated here by the name `c014d3#162`) until all data from it has been rebuilt, at which point it is finally deleted. The new replacement disk, which has

assumed the name c014d3, will have RAID tracks rebuilt and rebalanced onto it. Notice that only one block device name is mentioned as being formatted as a pdisk; the second path will be discovered in the background.

This can be confirmed with **mmlsrecoverygroup -L --pdisk**:

```
# mmlsrecoverygroup 000DE37TOP -L --pdisk

                   declustered
 recovery group       arrays     vdisks  pdisks
 -----------------  -----------  ------  ------
 000DE37TOP                   5       9     193

 declustered   needs                             replace               scrub       background activity
    array     service  vdisks  pdisks  spares  threshold  free space  duration  task   progress  priority
 -----------  -------  ------  ------  ------  ---------  ----------  --------  -------------------------
 DA1          no            2      47       2          2    3072 MiB   14 days  scrub       63%  low
 DA2          no            2      47       2          2    3072 MiB   14 days  scrub       19%  low
 DA3          yes           2      48       2          2       0   B   14 days  rebuild-2r  89%  low
 DA4          no            2      47       2          2    3072 MiB   14 days  scrub       34%  low
 LOG          no            1       4       1          1     546 GiB   14 days  scrub       87%  low

                   n. active,   declustered               user     state,
pdisk              total paths     array     free space   condition  remarks
-----------------  -----------  -----------  ----------  ----------- -------
[...]
c014d1               2,  4      DA1             23 GiB  normal      ok
c014d2               2,  4      DA2             23 GiB  normal      ok
c014d3               2,  4      DA3            550 GiB  normal      ok
c014d3#162           0,  0      DA3            543 GiB  replaceable dead/adminDrain/noRGD/noVCD/noPath
c014d4               2,  4      DA4             23 GiB  normal      ok
[...]
c018d1               2,  4      DA1             24 GiB  normal      ok
c018d2               2,  4      DA2             24 GiB  normal      ok
c018d3               0,  0      DA3            558 GiB  replaceable dead/systemDrain/noRGD/noVCD/noData/replace
c018d4               2,  4      DA4             23 GiB  normal      ok
[...]
```

Notice that the temporary pdisk c014d3#162 is counted in the total number of pdisks in declustered array DA3 and in the recovery group, until it is finally drained and deleted.

Notice also that pdisk c018d3 is still marked for replacement, and that DA3 still needs service. This is because GNR replacement policy expects all failed disks in the declustered array to be replaced once the replacement threshold is reached. The replace state on a pdisk is not removed when the total number of failed disks goes under the threshold.

Pdisk c018d3 is replaced following the same process.

1. Release carrier 18 in disk enclosure 000DE37:

```
# mmchcarrier 000DE37TOP --release --pdisk c018d3
  [I] Suspending pdisk c018d1 of RG 000DE37TOP in location 78AD.001.000DE37-C18-D1.
  [I] Suspending pdisk c018d2 of RG 000DE37TOP in location 78AD.001.000DE37-C18-D2.
  [I] Suspending pdisk c018d3 of RG 000DE37TOP in location 78AD.001.000DE37-C18-D3.
  [I] Suspending pdisk c018d4 of RG 000DE37TOP in location 78AD.001.000DE37-C18-D4.
  [I] Carrier released.

    - Remove carrier.
    - Replace disk in location 78AD.001.000DE37-C18-D3 with FRU 74Y4936.
    - Reinsert carrier.
    - Issue the following command:

        mmchcarrier 000DE37TOP --replace --pdisk 'c018d3'

      Repair timer is running.  Perform the above within 5 minutes
      to avoid pdisks being reported as missing.
```

2. Unlatch and remove carrier 18, remove and replace failed disk 3, reinsert carrier 18, and close the latch.
3. To finish the replacement of pdisk c018d3:

```
# mmchcarrier 000DE37TOP --replace --pdisk c018d3

  [I] The following pdisks will be formatted on node server1:
      /dev/rhdisk674
  [I] Pdisk c018d3 of RG 000DE37TOP successfully replaced.
  [I] Resuming pdisk c018d1 of RG 000DE37TOP.
  [I] Resuming pdisk c018d2 of RG 000DE37TOP.
  [I] Resuming pdisk c018d3#166 of RG 000DE37TOP.
  [I] Resuming pdisk c018d4 of RG 000DE37TOP.
  [I] Carrier resumed.
```

Running **mmlsrecoverygroup** again will confirm the second replacement:

```
# mmlsrecoverygroup 000DE37TOP -L --pdisk

                   declustered
 recovery group       arrays     vdisks  pdisks
 -----------------  -----------  ------  ------
 000DE37TOP                   5       9     192

 declustered   needs                            replace                 scrub       background activity
    array     service  vdisks  pdisks  spares  threshold  free space  duration  task   progress  priority
 -----------  -------  ------  ------  ------  ---------  ----------  --------  -------------------------
 DA1          no            2      47       2          2    3072 MiB   14 days  scrub       64%  low
 DA2          no            2      47       2          2    3072 MiB   14 days  scrub       22%  low
 DA3          no            2      47       2          2    2048 MiB   14 days  rebalance   12%  low
 DA4          no            2      47       2          2    3072 MiB   14 days  scrub       36%  low
 LOG          no            1       4       1          1     546 GiB   14 days  scrub       89%  low

                   n. active,   declustered               user     state,
pdisk              total paths     array     free space   condition  remarks
-----------------  -----------  -----------  ----------  ----------- -------
 [...]
 c014d1              2,  4       DA1             23 GiB  normal      ok
 c014d2              2,  4       DA2             23 GiB  normal      ok
 c014d3              2,  4       DA3            271 GiB  normal      ok
 c014d4              2,  4       DA4             23 GiB  normal      ok
 [...]
 c018d1              2,  4       DA1             24 GiB  normal      ok
 c018d2              2,  4       DA2             24 GiB  normal      ok
 c018d3              2,  4       DA3            542 GiB  normal      ok
 c018d4              2,  4       DA4             23 GiB  normal      ok
 [...]
```

Notice that both temporary pdisks have been deleted. This is because `c014d3#162` has finished draining, and because pdisk `c018d3#166` had, before it was replaced, already been completely drained (as evidenced by the `noData` flag). Declustered array `DA3` no longer needs service and once again contains 47 pdisks, and the recovery group once again contains 192 pdisks.

# Accessibility features for IBM Spectrum Scale RAID

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 RAID:

- 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 U.S.A.

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 Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing Legal and Intellectual Property Law IBM Japan Ltd. 19-21,

Nihonbashi-Hakozakicho, Chuo-ku Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Dept. 30ZA/Building 707
Mail Station P300

2455 South Road,
Poughkeepsie, NY 12601-5400
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment or a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

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, Windows, and Windows NT 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.

# Glossary

This glossary provides terms and definitions for the ESS solution.

The following cross-references are used in this glossary:

- *See* refers you from a non-preferred 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 (opens in new window):

http://www.ibm.com/software/globalization/terminology

## B

**building block**
A pair of servers with shared disk enclosures attached.

**BOOTP**
See *Bootstrap Protocol (BOOTP)*.

**Bootstrap Protocol (BOOTP)**
A computer networking protocol thst is used in IP networks to automatically assign an IP address to network devices from a configuration server.

## C

**CEC** See *central processor complex (CPC)*.

**central electronic complex (CEC)**
See *central processor complex (CPC)*.

**central processor complex (CPC)**
A physical collection of hardware that consists of channels, timers, main storage, and one or more central processors.

**cluster**
A loosely-coupled collection of independent systems, or *nodes*, organized into a network for the purpose of sharing resources and communicating with each other. See also *GPFS cluster*.

**cluster manager**
The node that monitors node status using disk leases, detects failures, drives recovery, and selects file system managers. The cluster manager is the node with the lowest node number among the quorum nodes that are operating at a particular time.

**compute node**
A node with a mounted GPFS file system that is used specifically to run a customer job. ESS disks are not directly visible from and are not managed by this type of node.

**CPC** See *central processor complex (CPC)*.

## D

**DA** See *declustered array (DA)*.

**datagram**
A basic transfer unit associated with a packet-switched network.

**DCM** See *drawer control module (DCM)*.

**declustered array (DA)**
A disjoint subset of the pdisks in a recovery group.

**dependent fileset**
A fileset that shares the inode space of an existing independent fileset.

**DFM** See *direct FSP management (DFM)*.

**DHCP** See *Dynamic Host Configuration Protocol (DHCP)*.

**direct FSP management (DFM)**
The ability of the xCAT software to communicate directly with the Power Systems server's service processor without the use of the HMC for management.

**drawer control module (DCM)**
Essentially, a SAS expander on a storage enclosure drawer.

**Dynamic Host Configuration Protocol (DHCP)**
A standardized network protocol that is used on IP networks to dynamically distribute such network configuration parameters as IP addresses for interfaces and services.

## E

**Elastic Storage Server (ESS)**
A high-performance, GPFS NSD solution

made up of one or more building blocks that runs on IBM Power Systems servers. The ESS software runs on ESS nodes - management server nodes and I/O server nodes.

**ESS Management Server (EMS)**
An xCAT server is required to discover the I/O server nodes (working with the HMC), provision the operating system (OS) on the I/O server nodes, and deploy the ESS software on the management node and I/O server nodes. One management server is required for each ESS system composed of one or more building blocks.

**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 (FEK)*, *master encryption key (MEK)*.

**ESS** See *Elastic Storage Server (ESS)*.

**environmental service module (ESM)**
Essentially, a SAS expander that attaches to the storage enclosure drives. In the case of multiple drawers in a storage enclosure, the ESM attaches to drawer control modules.

**ESM** See *environmental service module (ESM)*.

**Extreme Cluster/Cloud Administration Toolkit (xCAT)**
Scalable, open-source cluster management software. The management infrastructure of ESS is deployed by xCAT.

## 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 (FEK)*.

**file encryption key (FEK)**
A key used to encrypt sectors of an individual file. See also *encryption key*.

**file system**
The methods and data structures used to control how data is stored and retrieved.

**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.

**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.

**flexible service processor (FSP)**
Firmware that provices diagnosis, initialization, configuration, runtime error detection, and correction. Connects to the HMC.

**FQDN**
See *fully-qualified domain name (FQDN)*.

**FSP** See *flexible service processor (FSP)*.

**fully-qualified domain name (FQDN)**
The complete domain name for a specific computer, or host, on the Internet. The FQDN consists of two parts: the hostname and the domain name.

## G

**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 Storage Server (GSS)**
A high-performance, GPFS NSD solution made up of one or more building blocks that runs on System x servers.

**GSS** See *GPFS Storage Server (GSS)*.

## H

**Hardware Management Console (HMC)**
Standard interface for configuring and operating partitioned (LPAR) and SMP systems.

**HMC** See *Hardware Management Console (HMC)*.

## I

**IBM Security Key Lifecycle Manager (ISKLM)**
For GPFS encryption, the ISKLM is used as an RKM server to store MEKs.

**independent fileset**
A fileset that has its own inode space.

**indirect block**
A block that contains 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.

**Internet Protocol (IP)**
The primary communication protocol for relaying datagrams across network boundaries. Its routing function enables internetworking and essentially establishes the Internet.

**I/O server node**
An ESS node that is attached to the ESS storage enclosures. It is the NSD server for the GPFS cluster.

**IP** See *Internet Protocol (IP)*.

**IP over InfiniBand (IPoIB)**
Provides an IP network emulation layer on top of InfiniBand RDMA networks, which allows existing applications to run over InfiniBand networks unmodified.

**IPoIB** See *IP over InfiniBand (IPoIB)*.

**ISKLM**
See *IBM Security Key Lifecycle Manager (ISKLM)*.

## J

**JBOD array**
The total collection of disks and enclosures over which a recovery group pair is defined.

## 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.

## L

**LACP** See *Link Aggregation Control Protocol (LACP)*.

**Link Aggregation Control Protocol (LACP)**
Provides a way to control the bundling of several physical ports together to form a single logical channel.

**logical partition (LPAR)**
A subset of a server's hardware resources virtualized as a separate computer, each with its own operating system. See also *node*.

**LPAR** See *logical partition (LPAR)*.

## M

**management network**
A network that is primarily responsible for booting and installing the designated server and compute nodes from the management server.

**management server (MS)**
An ESS node that hosts the ESS GUI and xCAT and is not connected to storage. It can be part of a GPFS cluster. From a system management perspective, it is the

central coordinator of the cluster. It also serves as a client node in an ESS building block.

**master encryption key (MEK)**
A key that is used to encrypt other keys. See also *encryption key*.

**maximum transmission unit (MTU)**
The largest packet or frame, specified in octets (eight-bit bytes), that can be sent in a packet- or frame-based network, such as the Internet. The TCP uses the MTU to determine the maximum size of each packet in any transmission.

**MEK** See *master encryption key (MEK)*.

**metadata**
A data structure that contains access information about file data. Such structures include inodes, indirect blocks, and directories. These data structures are not accessible to user applications.

**MS** See *management server (MS)*.

**MTU** See *maximum transmission unit (MTU)*.

## N

**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 hexadecimal 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 can contain one or more nodes. In a Power Systems environment, synonymous with *logical partition*.

**node descriptor**
A definition that indicates how IBM Spectrum Scale uses a node. Possible functions include: manager node, client node, quorum node, and non-quorum node.

**node number**
A number that is generated and maintained by IBM Spectrum Scale 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 IBM Spectrum Scale 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.

## O

**OFED** See *OpenFabrics Enterprise Distribution (OFED)*.

**OpenFabrics Enterprise Distribution (OFED)**
An open-source software stack includes software drivers, core kernel code, middleware, and user-level interfaces.

## P

**pdisk** A physical disk.

**PortFast**
A Cisco network function that can be configured to resolve any problems that could be caused by the amount of time STP takes to transition ports to the Forwarding state.

## R

**RAID** See *redundant array of independent disks (RAID)*.

**RDMA**
See *remote direct memory access (RDMA)*.

**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.

**recovery group (RG)**
A collection of disks that is set up by IBM Spectrum Scale RAID, in which each disk is connected physically to two servers: a primary server and a backup server.

**remote direct memory access (RDMA)**
A direct memory access from the memory of one computer into that of another without involving either one's operating system. This permits high-throughput, low-latency networking, which is especially useful in massively-parallel computer clusters.

**RGD** See *recovery group data (RGD)*.

**remote key management server (RKM server)**
A server that is used to store master encryption keys.

**RG** See *recovery group (RG)*.

**recovery group data (RGD)**
Data that is associated with a recovery group.

**RKM server**
See *remote key management server (RKM server)*.

## S

**SAS** See *Serial Attached SCSI (SAS)*.

**secure shell (SSH)**
A cryptographic (encrypted) network protocol for initiating text-based shell sessions securely on remote computers.

**Serial Attached SCSI (SAS)**
A point-to-point serial protocol that moves data to and from such computer storage devices as hard drives and tape drives.

**service network**
A private network that is dedicated to managing POWER8 servers. Provides Ethernet-based connectivity among the FSP, CPC, HMC, and management server.

**SMP** See *symmetric multiprocessing (SMP)*.

**Spanning Tree Protocol (STP)**
A network protocol that ensures a loop-free topology for any bridged Ethernet local-area network. The basic function of STP is to prevent bridge loops and the broadcast radiation that results from them.

**SSH** See *secure shell (SSH)*.

**STP** See *Spanning Tree Protocol (STP)*.

**symmetric multiprocessing (SMP)**
A computer architecture that provides fast performance by making multiple processors available to complete individual processes simultaneously.

## T

**TCP** See *Transmission Control Protocol (TCP)*.

**Transmission Control Protocol (TCP)**
A core protocol of the Internet Protocol Suite that provides reliable, ordered, and error-checked delivery of a stream of octets between applications running on hosts communicating over an IP network.

## V

**VCD** See *vdisk configuration data (VCD)*.

**vdisk** A virtual disk.

**vdisk configuration data (VCD)**
Configuration data that is associated with a virtual disk.

## X

**xCAT** See *Extreme Cluster/Cloud Administration Toolkit*.

# Index

## A



## B



## C

# D



# E

# F



# G



# H



# I

# L



# M



# N



# O



# P

# R



# S

# T



# U



# V



# W

Product Number: 5641-GRS
5725-N21

Printed in USA

SC27-9271-00