General Parallel File System for AIX

Installation and Administration

SA22-7278-01

General Parallel File System for AIX

Installation and Administration

SA22-7278-01

Before using this information and the product it supports, be sure to read the general information under “Notices” on page ix.

First Edition (April 1998)

This edition applies to Version 1 Release 1 of IBM General Parallel File System for AIX (5765–B95) and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

IBM welcomes your comments. A form for your comments may be provided at the back of this publication, or you may address your comments to the following:

International Business Machines Corporation
Department 55JA, Mail Station P384
522 South Road
Poughkeepsie, NY 12601–5400
USA

FAX (United States and Canada): 1+914+432–9405

FAX (Other Countries): Your International Access Code +1+914+432+9405

IBMLink (United States customers only): KGNVMC (MHVRCFS)

IBM Mail Exchange: USIB6TC9 at IBMMAIL

Internet e–mail: mhvrcfs@vnet.ibm.com

World Wide Web: http://www.rs6000.ibm.com

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this book
- Page number or topic related to your comment

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

© **Copyright International Business Machines Corporation 1998. All rights reserved.**
Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Notices ix
Trademarks and Service Marks ix

About This Book xi
Who Should Use This Book xi
What You Need to Know First xi
How This Book is Organized xi
Typography and Terminology xii
Related Information xii
Obtaining Documentation xiii

Chapter 1. IntroducingGeneral Parallel File System for AIX 1
GPFS Improves System Performance 1
GPFS Assures File Consistency 2
GPFS Increases Data Availability 2
GPFS Enhances System Flexibility 2
GPFS Simplifies Administration 3

Chapter 2. Planning for GPFS 5
Configuration Considerations 5
Planning Nodes 5
Cache 6
Performance 7
File System Creation Considerations 7
File System Sizing 8
IBM Virtual Shared Disks 12
Recoverability 14
Additional File System Considerations 18
Planning a Sample GPFS Configuration 18
Planning a Sample GPFS File System 19
Sizing Sample File Systems 19
Planning Sample IBM Virtual Shared Disks 20

Chapter 3. Installing GPFS 23
Installation Requirements 23
Hardware Requirements 23
Software Requirements 23
Step 1. Installation Procedure 24
Create GPFS Directory 24
Create Installation Image 24
Install GPFS on Network 25
Install Control Workstation Image 25
Step 2. Verify GPFS Installation 25
Step 3. Tune Switch 26
Step 4. Configure sysctl 26
Step 5. Tune IBM Virtual Shared Disk 26

Chapter 4. Performing GPFS Administration Tasks 29
Configuring GPFS 30
Configure Using SMIT 31

© Copyright IBM Corp. 1998

Configure Using mmconfig Command . . . 31
Starting and Stopping GPFS . . . 32
Before starting GPFS . . . 32
Starting GPFS . . . 32
Stopping GPFS . . . 33
Changing GPFS Configuration . . . 33
Before You Change the GPFS Configuration . . . 33
Changing Configuration Using SMIT . . . 33
Changing Configuration Using mmchconfig . . . 34
Managing File Systems . . . 34
Creating a File System . . . 35
Mounting a File System . . . 41
Unmounting a File System . . . 42
Deleting a File System . . . 43
Checking and Repairing a File System . . . 43
Listing File System Attributes . . . 45
Modifying File System Attributes . . . 46
Querying and Changing File Replication Attributes . . . 47
Restriping a GPFS File System . . . 48
Querying File System Space . . . 49
Managing Disks . . . 49
Adding Disks to a File System . . . 50
Deleting Disks from a File System . . . 51
Replacing Disks in a GPFS File System . . . 52
Displaying GPFS Disk States . . . 53
Changing GPFS Disk States and Parameters . . . 55
Managing Nodes . . . 57
Adding Nodes to the GPFS Configuration . . . 57
Deleting Nodes from the GPFS Configuration . . . 58
Managing GPFS Quotas . . . 59
Installing and Uninstalling GPFS Quotas . . . 60
Establishing and Changing Quotas . . . 61
Checking Quotas . . . 62
Listing Quotas . . . 62
Activating Quota Enforcement . . . 63
Deactivating Quota Enforcement . . . 64
Creating File System Quota Reports . . . 64
Recovering from Over-Quota Conditions . . . 65
Managing Access Control Lists . . . 65
Setting Access Control Lists . . . 66
Displaying Access Control Lists . . . 67
Changing Access Control Lists . . . 67
Deleting Access Control Lists . . . 68

Chapter 5. Tuning GPFS Performance . . . 69
GPFS Tuning Parameters . . . 69
File System Block Size . . . 69
GPFS Memory Parameters . . . 69
Additional Considerations . . . 70
Priority . . . 70
Token Management . . . 70
Switch Parameters and GPFS Performance . . . 71
IBM Virtual Shared Disk Parameters and GPFS Performance . . . 71

Chapter 6. GPFS Commands 73
mmadddisk 74
mmaddnode 78
mmchattr 81
mmchconfig 83
mmchdisk 86
mmcheckquota 90
mmchfs 92
mmconfig 94
mmcrfs 97
mmdelacl 104
mmdeldisk 105
mmdelfs 108
mmdelnode 110
mmdf 112
mmeditacl 114
mmedquota 116
mmfsck 118
mmgetacl 122
mmlsattr 123
mmlsdisk 125
mmlsfs 127
mmlsquota 129
mmputacl 131
mmquotaoff 133
mmquotaon 135
mmrepquota 137
mmrestripefs 139
mmrpldisk 141

Appendix A. Considerations for GPFS Applications 145
Exceptions to X/Open Standards 145
Application Support 145

Glossary 147

Index 149

Figures

1. Files Can Span Multiple Disks on Multiple SP Nodes . . . 1
2. GPFS Files Have Traditional UNIX Structure . . . 8
3. Primary Node Serving RAID Device . . . 15
4. Backup Node Serving RAID Device . . . 15
5. Basic Failure Groups with Servers and Disks . . . 16
6. Failure Groups with Twin-Tailed Disks . . . 16
7. Planning Disks for Sample File Systems . . . 20
8. SMIT GPFS Administration Menu . . . 30
9. File System Startup Configuration Menu . . . 31
10. Select Attribute Dialog . . . 34
11. Change File System Startup Configuration Dialog . . . 34
12. Create/Change a Disk Descriptor Dialog . . . 39
13. Create File System Dialog . . . 40
14. Mount File System Dialog . . . 41
15. Unmount File System Dialog . . . 42
16. Delete File System Dialog . . . 43
17. Modify File System Dialog . . . 47
18. Add Disk(s) Dialog . . . 50
19. Delete Disk(s) Dialog . . . 52
20. Replace Disk Dialog . . . 53
21. Display Status of Disk(s) Dialog . . . 55
22. Add New Node(s) Dialog . . . 58
23. Delete Node(s) Dialog . . . 59

© Copyright IBM Corp. 1998

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
USA

Licensees of these programs 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
Mail Station P300
522 South Road
Poughkeepsie, NY 12601–5400
USA
Attention: Information Request

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment or a fee.

Trademarks and Service Marks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

AIX

IBM

RS/6000

RS/6000 SP

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

© Copyright IBM Corp. 1998

About This Book

This book describes how to install and manage the IBM General Parallel File System for AIX (GPFS) licensed program product. It includes information to help you plan and configure your GPFS system, and explains how to use commands unique to GPFS.

Throughout this publication you will see various command and component names beginning with the prefix **mmfs**. This is not an error. GPFS shares many components with the IBM Multi-Media LAN Server, a related product.

Who Should Use This Book

Use this book if you are considering installing the General Parallel File System for AIX licensed program product.

Use this book for installing and administering a GPFS file system.

What You Need to Know First

Before using this book, you should be familiar with, and have documentation for, the following:

- RS/6000 SP system administration, as described in *IBM Parallel Systems Support Programs: Administration Guide*, GC23–3897.
- The AIX operating system, including the Logical Volume Manager subsystem as described in *IBM AIX Version 4.2 System Management Guide: Operating Systems and Devices*, SC23–2525.
- The IBM Virtual Shared Disk information in *IBM Parallel Systems Support Programs: Managing Shared Disks*, SA22–7279.
- The IBM Virtual Shared Disk and High Availability information in *IBM Parallel Systems Support Programs: Administration Guide*, GC23–3897.
- The IBM Virtual Shared Disk commands in *IBM Parallel Systems Support Programs: Command and Technical Reference*, GC23–3900.
- Configuration and tuning information in *IBM RS/6000 SP Management, Easy, Lean, and Mean*, GG24–2563.

How This Book is Organized

This book consists of the following chapters:

- Chapter 1 provides an overview of GPFS and describes its features.
- Chapter 2 contains planning information to help you prepare for GPFS installation, configuration, and file systems creation.
- Chapter 3 contains the procedures for GPFS installation and configuration.
- Chapter 4 deals with administration tasks and contains procedures for file system creation and maintenance.

© Copyright IBM Corp. 1998

- Chapter 5 discusses tuning issues and how you can improve GPFS performance.
- Chapter 6 contains the command reference pages.
- Appendix A describes some differences in how GPFS calls standard UNIX routines, and how these calls might affect certain applications.

Typography and Terminology

This book uses the following typographical conventions:

Convention	Usage
Bold	**Bold** words or characters represent system elements that you must use literally, such as commands, subcommands, flags, path names, directories, file names, values, and selected menu options.
Bold Underlined	**Bold Underlined** keywords are defaults. These take effect if you fail to specify a different keyword.
Italic	• *Italic* words or characters represent variable values that you must supply • *Italics* are used for book titles • *Italics* are used for general emphasis
`Monospace`	All of the following are displayed in `monospace` type: • Displayed information • Message text • Example text • Specified text typed by the user • Field names as displayed on the screen • Prompts from the system • References to example text
[ ]	Brackets enclose optional items in format and syntax descriptions.
{ }	Braces enclose a list from which you must choose an item in format and syntax descriptions.
< >	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.
...	An ellipsis indicates that you can repeat the preceding item one or more times.
<**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**>.

Related Information

The *General Parallel File System for AIX: Problem Determination Guide*, a softcopy document shipped on the product media and installed at **/usr/lpp/mmfs/gpfsdocs/docs/gpfspd.html**, contains procedures for correcting error conditions, as well as additional information that can help you optimize GPFS performance under unusual operating conditions. For the latest updates, visit the RS/6000 SP Product Documentation Library site on the World Wide Web, **http://www.rs6000.ibm.com/resource/aix_resource/sp_books**.

Obtaining Documentation

This book is shipped on the product media in Postscript format and installed at **/usr/lpp/mmfs/gpfsdocs/docs/gpfsadmin.ps**

You can also view or download this book from the IBM RS/6000 web site at **http://www.rs6000.ibm.com**

Chapter 1. IntroducingGeneral Parallel File System for AIX

IBM General Parallel File System for AIX (GPFS) provides file system services to parallel and serial applications running on the RS/6000 SP. GPFS allows users shared access to files that may span multiple disk drives on multiple SP nodes.

GPFS is designed for seamless integration in your UNIX** environment. Almost all applications run exactly the same under GPFS as they do with other file systems. AIX file system utilities are also supported by GPFS. This means that users can continue to use the UNIX commands they have always used for ordinary file operations. The only unique commands are those for administering the GPFS file system. (See Appendix A, “Considerations for GPFS Applications” on page 145 for more information.)

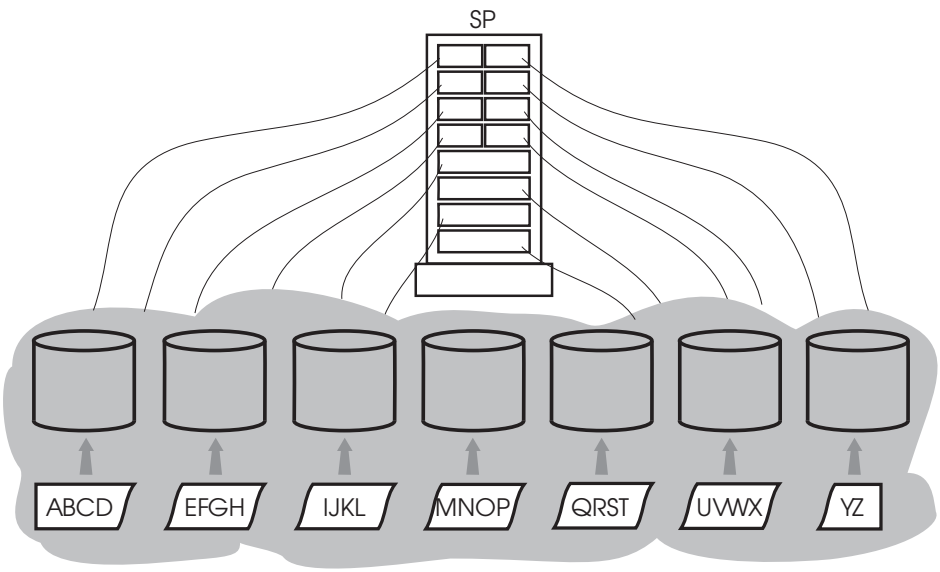


Figure 1. Files Can Span Multiple Disks on Multiple SP Nodes

GPFS allows parallel applications simultaneous access to the same files, or different files, from any node in the configuration while managing a high level of control over all file system operations. It offers extremely high recoverability while maximizing data accessibility.

GPFS Improves System Performance

Using GPFS to store and retrieve your files can improve system performance by:

- Allowing multiple processes or applications on all nodes of the SP system simultaneous access to the same file using standard file system calls.
- Increasing aggregate bandwidth of your file system by spreading reads and writes across multiple disks.
- Balancing the load evenly across all disks to maximize their combined throughput. One disk is no more active than another.
- Supporting large amounts of data and allowing you to have bigger file systems.

© Copyright IBM Corp. 1998

- Allowing concurrent reads and concurrent writes to files in the file system — very important in parallel processing.

GPFS builds on the shared disk concept at the heart of the IBM Virtual Shared Disk component of the Parallel System Support Programs for AIX by taking advantage of the speed of the SP Switch and using it to accelerate parallel file operations that would overload serial file system management.

GPFS Assures File Consistency

GPFS uses a sophisticated token management system to guarantee data consistency while providing multiple, independent paths to the same file—by the same name—from anywhere in the SP system. Even when nodes are down or hardware resource demands are high, GPFS can find an available path to file system data.

GPFS Increases Data Availability

GPFS is a logging file system that creates separate logs for each SP node. These logs record the allocation of metadata and aid fast recovery and consistency of data in the event of node failure, even when a node fails while modifying file data.

GPFS failover support allows you to organize your hardware to minimize single points of failure. You can guard against loss of file system availability by configuring backup servers and mirroring data.

The *replication* feature of GPFS allows you to determine how many copies of a file to maintain. File system replication assures that the latest updates to critical data are preserved in the event of disk failure. During configuration, you assign a replication factor to indicate the total number of copies you wish to store. Replication allows you to set different levels of protection for each file or one level for an entire file system. Since replication uses additional disk space and requires extra write time, you might want to consider replicating only files that are frequently read but seldom written.

Once your file system is created, you can have it automatically mounted whenever the GPFS daemons are started. The automount feature assures that whenever the SP system and disks are up, the file system will be available.

GPFS Enhances System Flexibility

With GPFS, your system resources are not frozen. You can add or delete disks while the file system is mounted. When the time is right and system demand is low, you can rebalance the file system across all currently configured disks.

You can configure one or more nodes as dedicated disk servers, or allow applications to share server nodes as well. Later, you can reconfigure GPFS to enlarge file size capacity or to increase throughput depending on your applications, hardware, and workload. Set up your GPFS environment for today's applications and users, secure in the knowledge that you can expand in the future without jeopardizing your data. GPFS capacity can grow as your hardware expands.

GPFS Simplifies Administration

All GPFS administration tasks, from installation to maintenance and reconfiguration, can be performed from any node in the SP configuration. GPFS administration commands are similar in name and function to UNIX file system commands, with one important difference: the GPFS commands operate on multiple nodes. A single GPFS multi-node command can perform a file system function across the entire SP system. GPFS administration tasks can also be executed from SMIT menus.

Chapter 2. Planning for GPFS

Although you can modify your GPFS configuration after it has been set, a little consideration before installation and initial setup will reward you with a more efficient and immediately useful file system.

GPFS configuration requires you to specify several operational parameters that reflect your hardware resources and operating environment. Later, during file system creation, you have the opportunity to specify additional parameters based on the expected size of the files. These parameters define the disks for the file system and how data will be written to them.

This chapter, and the ones that follow, contain a series of examples that step through the planning process leading to the configuration of a typical GPFS system. The creation and subsequent modification of two GPFS files systems are also illustrated.

Configuration Considerations

Configuration involves defining the nodes to be included in the GPFS subsystem and specifying how they operate. You can provide a list of nodes as input to configuration, or allow GPFS to configure all the nodes in your SP system. You also have the option of installing a sample configuration file with preset values that you can accept as defaults or modify to suit your needs, or you can create and install a configuration file of your own.

Planning Nodes

When you configure GPFS for your SP system, you have the opportunity to specify a list of all the nodes that will mount GPFS file systems. Later, when you create a file system, you have the opportunity to specify an estimate of how many nodes the file system will be mounted on. These two specifications serve different purposes.

Estimating Node Count

When creating a GPFS file system, overestimate the number of nodes that will mount the file system. This input is used in the creation of GPFS data structures that are essential for achieving the maximum degree of parallelism in file system operations. Although a larger estimate consumes a bit more memory, insufficient allocation of these data structures can limit node ability to process certain parallel requests efficiently, such as the allotment of disk space to a file. If you cannot anticipate the number of nodes, allow the default value of 32 to be applied. Specify a larger number if you expect to add nodes, but avoid wildly overestimating, which can affect buffer operations. This value cannot be changed later.

Creating a Node List

Before you configure GPFS on your SP, prepare a file that lists all the nodes in the system. The list must contain only one entry per line. During configuration, GPFS copies this list to the **/etc/cluster.nodes** file and uses it for setup.

Although the list can include any of the following formats, be sure to identify nodes by their switch hostnames. The use of Ethernet hostnames results in much slower I/O performance.

© Copyright IBM Corp. 1998

Format	Example
Short Hostname	k145s02
Long Hostname	k145s02.dpd.ibm.com
IP Address	9.119.19.102

If you do not provide this list when configuring GPFS, the switch hostnames of all nodes in the System Data Repository (SDR) are added to the **/etc/cluster.nodes** file.

Note: You can configure only one group of GPFS nodes. This means that the **/etc/cluster.nodes** file must be the same on all nodes. Any attempt to create a second GPFS group, either by changing the **/etc/cluster.nodes** file on some nodes, or by rerunning—rather than changing—the GPFS configuration, will fail. Only one GPFS configuration can exist on an SP partition.

Quorums

GPFS also uses the list of nodes to establish a quorum requirement. A quorum is the minimum number of nodes that must be running in order for the GPFS daemon to start. GPFS uses the quorum to maintain data consistency. By default, a quorum is half the GPFS nodes plus one, as defined in the **/etc/cluster.nodes** file. Refer to *GPFS Problem Determination Guide* for more information.

Starting GPFS Automatically on Nodes

You can configure GPFS to start automatically on all nodes whenever they come up, by specifying the autoload (**–A**) option. This eliminates the need to start GPFS manually when the nodes have been rebooted.

Cache

GPFS creates a number of cache segments on each node in the SP system. This pinned memory is used to increase performance through *read-ahead* and *write-behind* operations, as well as for reuse of cached data. On a dedicated IBM Virtual Shared Disk server, cache can be the minimal size provided by the default, but on nodes where GPFS supports application usage, larger cache sizes usually improve GPFS performance.

During configuration, you can specify two parameters that control how much cache is dedicated to GPFS. These values can be changed later, so experiment with larger values to find the optimum cache size that improves GPFS performance without affecting other applications. In order for changed values to take effect, you must restart GPFS.

Note: The sum of these two parameters must not exceed 80% of real memory.

pagepool This is the size of the cache on each node. It can range from a minimum of 4MB to a maximum of 512MB per node. This value must be specified with the character **M**, for example `80M`. The default is `20M`.

mallocsize
This area is used exclusively for GPFS control structures. It can range from a minimum of 2MB to a maximum of 128MB per node. This value must be specified with the character **M**, for example `8M`. The default is `4M`.

maxFilesToCache

This is the number of i–nodes to cache for recently used files that have been closed. Storing a file's i–node in cache permits faster reaccess to the file. The default is 200 but increasing this number may improve throughput for workloads with high file reuse. Increasing it where file reuse is not great will waste kernel heap storage. Do not increase this parameter if you are in doubt.

Each cached file requires space in the mallocpool for an i–node plus approximately 800 bytes of metadata. GPFS will not use more than 50% of the mallocpool for this purpose. If you increase **maxFilesToCache**, you should also increase the **mallocsize** parameter to a value equal to 2 x (maximum i–node size for the file system + 800) x **maxFilesToCache**. For example, to set **maxFilesToCache** to 1000 for a file system with a maximum i–node size of 4096, **mallocsize** should be approximately 10MB.

Performance

The only configuration parameter that directly addresses performance is **priority**. Sometimes called scheduling priority, this parameter sets the UNIX priority for the GPFS daemon. The default is 40. Priority increases with lower values. Refer to AIX Documentation for details.

Refer to Chapter 5, “Tuning GPFS Performance” on page 69 for information about other UNIX tunables and additional methods of adjusting GPFS performance.

File System Creation Considerations

Each GPFS file system can be mounted separately. Within each file system, files are written to disk as traditional UNIX file systems, using i-nodes, indirect blocks, and data blocks. Each file has an i-node containing information such as file size and time of last modification. The i-nodes of small files also contain the addresses of all disk blocks that comprise the file data. A large file can use too many data blocks for an i-node to address. In such a case, the i-node points instead to indirect blocks that are large enough to hold data block addresses.

I-nodes and indirect blocks are considered *metadata*, as distinguished from data, or actual file content. You can control which disks GPFS uses for storing metadata when you create a file system.

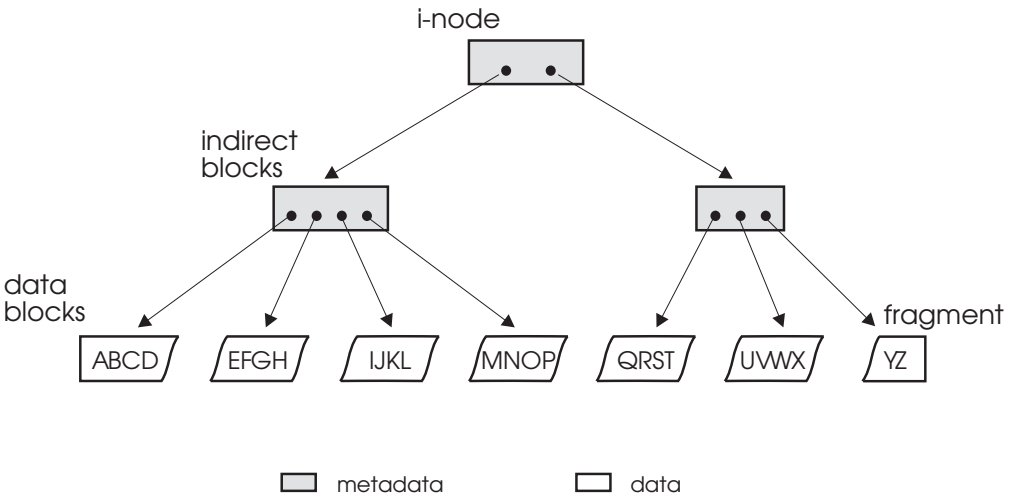


Figure 2. GPFS Files Have Traditional UNIX Structure

File System Sizing

Before creating a file system, consider how much data will be stored and how large the files are likely to grow. You should also try to anticipate how great will be the demand for the files in the system. Will several users want to read a file at the same time? How many users will be writing to it? Each of these factors can help you to determine how much disk resource to devote to the file system, where to store data and metadata, and how many replicas to maintain.

Block Size

GPFS offers a choice of three block sizes at file system creation time: 16KB, 64KB, or 256KB. Familiarity with the applications running on your system will help you determine which block size to use. Many technical applications handle large amounts of data in a single read/write operation. These are better served by a larger block size. A smaller block size may result in more efficient use of disk space for smaller files.

The default block size is 256KB.

Note: If you plan to use RAID devices (Redundant Arrays of Independent Disks) in your file system, a larger block size may be more effective and help you to avoid the penalties involved in small block write operations to RAID devices.

Fragments and Subblocks

GPFS divides each block into 32 *subblocks*. If the block size is the largest contiguous amount of disk space allocated to a file (and therefore, the largest amount of data that can be accessed in a single I/O operation), then the subblock is the smallest unit of disk space that can be allocated. For a block size of 256KB, GPFS reads as much as 256KB of data in a single I/O operation and small files can occupy as little as 8KB of disk space. With a block size of 16KB, small files occupy as little as 512 bytes of disk space (not counting the i-node), but GPFS is unable to read more than 16KB in a single I/O operation.

Files smaller than one block size are stored in *fragments*, which are made up of one or more subblocks. Large files are stored in a number of full blocks plus one or more fragments to hold the data at the end of the file.

I-Nodes

The i-node, or file index, is the internal structure that describes an individual file to AIX. The size of a file's i-node defaults to 512 bytes, the standard disk sector size. You can specify a larger i-node size at file system creation time if you anticipate larger files. The maximum i-node size is 4KB.

"Maximum Number of Files" on page 10 describes how to estimate the number of i-nodes for your file system.

Indirect Blocks

Another parameter that affects file capacity is indirect block size. This is the unit of disk space GPFS allocates for data block addressing of large files. Indirect block size can be as small as a single sub–block or as large as one full block, provided it does not exceed 32KB. The only additional requirement is that it be a multiple of the sub–block size (1/32 of BlockSize).

Applications that read and write large amounts of data in a single I/O operation work better with larger indirect blocks. If you run such applications, specify a larger indirect block size rather than a larger i-node size. Larger i-nodes may waste disk space when applied to small files, while large indirect blocks incur a disk space penalty only for large files. The recommended value for the indirect block is *block size* / *n*, where *n* is a power of 2. The default indirect block size is *block size*/16, which is equal to 2 subblocks.

Two additional parameters affect maximum file size and control replication. *MaxMetaDataReplicas* sets the maximum number of copies of metadata GPFS maintains, and *MaxDataReplicas* sets the maximum number of data copies. Although replication is a recoverability consideration (see "Recoverability" on page 14 for a more detailed discussion), remember that it has a dramatic effect on potential file size. The more data and metadata replicas GPFS maintains, the more limited your potential file size. The more metadata replicas GPFS maintains, the more limited your number of files, because a single file with replicated metadata uses the same number of i-nodes as two files. A quick look at Table 1 on page 11 gives you an idea how this relationship works.

Putting it All Together

This leads to the final and perhaps most important characteristic of any file system, capacity. How big can your files grow, and how much can the file system hold?

Maximum File Size: Block size, indirect size, and i-node size, combined with the maximum replication factors for data and metadata, determine the maximum possible file size in a GPFS file system. The following formula illustrates these relationships.

$$\max\_fsize = \left(\frac{i - 104}{6M}\right) \times \left(\frac{I - 44}{6R}\right) \times B$$

where all units are bytes:

B = Blocksize

i = I-Node Size

I = Indirect Block Size

M = Maximum Metadata Replication

R = Maximum Data Replication

For example, the maximum file size for an unreplicated file system with a block size of 64KB, an indirect size of 16KB, and an i-node size of 1KB could be calculated as follows:

$$\text{max\_fsize} = \left(\frac{1024 - 104}{6}\right) \times \left(\frac{16384 - 44}{6}\right) \times 65536 = \frac{2736637724 4.44}{1024^3}\text{GB} = 26.1\text{GB}$$

If the same file system were modified to allow replication, the formula shows how the maximum file size is substantially reduced:

$$\text{max\_fsize} = \left(\frac{1024 - 104}{12}\right) \times \left(\frac{16384 - 44}{12}\right) \times 65536 = \frac{6841594311.11}{1024^3}\text{ GB} = 6.4\text{GB}$$

Maximum File System Size: Once you know the maximum file size (max_fsize), you can determine the maximum size of a GPFS file system using the following formula:

$$\text{max\_fs\_size} = \text{blks\_per\_fs} \times B$$

where

blks_per_fs = max_fsize / 24

B = Blocksize

We could calculate a maximum file system size using the maximum file size value arrived at in the example above:

$$\text{max\_fs\_size} = \left(\frac{2736637724 4.44}{24}\right) \times 65536 = \frac{74728454128829}{1024^3}\text{ GB} = 69596.3\text{GB}$$

(Please note that 1 terabyte is the maximum supported file system size for release 1 of GPFS. File systems greater than 1 terabyte in size are not supported by IBM service.)

Maximum Number of Files: Since one i-node is required for each file, the number of files a GPFS file system can hold depends on the number of i-nodes specified when you create the file system, but there are two ways to get a more realistic estimate:

1. Divide the maximum file system size (max_fs_size) by the average file size
2. Divide the maximum file system size (max_fs_size) by the i-node size

The actual number of i-nodes your file system supports will usually be closer to the second estimate. If you are in doubt, plan for a maximum number of files according to the smaller of the two estimates.

When creating a file system, keep in mind that the number of i-nodes cannot be changed, so you should allow for plenty of file system growth. If you do not specify this parameter, it defaults to the size of the file system at creation, divided by 1MB.

Table 1 on page 11 illustrates how block size, indirect size, i–node size, and replication affect both maximum file size and maximum file system size by showing selected parameter values with the capacities they yield. For purposes of

simplification these calculations use the same replication values for both data and metadata.

Table 1. Sample GPFS Block Sizes and Related Parameters

(*B*) Block Size	(*I*) Indirect Size	(*i*) I-Node Size	(*M, R*) Maximum Replication	Approx. Maximum File Size	Approx. Maximum File System Size
16KB	1KB	512	1	173.5MB	115.7GB
			2	43.4MB	28.9GB
16KB	4KB	512	1	717.5MB	478.4GB
			2	179.4MB	119.6GB
16KB	16KB	512	1	2.9GB	*1TB
			2	723.4MB	482.3GB
64KB	4KB	512	1	2.8GB	*1TB
			2	717.5MB	*1TB
64KB	4KB	1KB	1	6.4GB	*1TB
			2	1.6GB	*1TB
64KB	4KB	2KB	1	13.7GB	*1TB
			2	3.4GB	*1TB
64KB	16KB	512	1	11.5GB	*1TB
			2	2.9GB	*1TB
64KB	16KB	1KB	1	26.1GB	*1TB
			2	6.5GB	*1TB
64KB	16KB	2KB	1	57.8GB	*1TB
			2	13.8GB	*1TB
64KB	32KB	512	1	23.1GB	*1TB
			2	5.8GB	*1TB
64KB	32KB	1KB	1	52.2GB	*1TB
			2	13.0GB	*1TB
64KB	32KB	2KB	1	110.4GB	*1TB
			2	27.6GB	*1TB
256KB	16KB	512	1	46.3GB	*1TB
			2	11.5GB	*1TB
256KB	16KB	4KB	1	452.9GB	*1TB
			2	113.2GB	*1TB
256KB	32KB	512	1	90.2GB	*1TB
			2	231.8GB	*1TB
256KB	32KB	4KB	1	901.1GB	*1TB
			2	226.8GB	*1TB

*

IBM Virtual Shared Disks

GPFS accesses data using the facilities of the IBM Recoverable Virtual Shared Disk Licensed Program, and the IBM Virtual Shared Disk component of the Parallel System Support Programs for AIX, which allows application programs executing on different SP nodes to access a logical volume as if it were local at each node. IBM Recoverable Virtual Shared Disk, which allows a secondary or backup server to be defined for such a logical volume, is required even when there are no twin-tailed disks that can be physically accessed from more than one node because it provides fencing capabilities that preserve data integrity in the event of certain failures. *Managing Shared Disks*, SC23–3849, contains installation, management, and usage information for both IBM Virtual Shared Disk and IBM Recoverable Virtual Shared Disk.

Proper planning for GPFS installation provides:

- Sufficient processing capability for IBM Virtual Shared Disk servers to deliver data to clients.
- Sufficient disks to meet the expected I/O load.
- Sufficient connectivity (adapters and buses) between disks and IBM Virtual Shared Disk servers.
- Sufficient free disk space for the correct operation of the IBM Recoverable Virtual Shared Disk and PSSP components used by GPFS

IBM Virtual Shared Disk Servers

Will your IBM Virtual Shared Disk servers be dedicated or will you also be using them to run applications? In order to answer this question, estimate whether the overhead of serving your I/O will exploit the processing potential of dedicated IBM Virtual Shared Disk servers, or whether these servers have the additional potential to run applications.

The actual processing capability required for IBM Virtual Shared Disk service is a function of the application I/O access patterns, the node model, the type of disk, and the disk connection. You can run **iostat** on the server to determine how much of a load your access pattern will place on an IBM Virtual Shared Disk server.

Note: If you plan to run time-critical applications on an IBM Virtual Shared Disk server, remember that servicing disk requests from other nodes might conflict with the demands of these applications.

Assure that you have sufficient resources to run the IBM Virtual Shared Disk program efficiently. This includes enough buddy buffers of sufficient size to match your file system block size, as well as sufficient *rpoolsize* and *spoolsize* blocks in the communications subsystem.

Buddy Buffers: The IBM Virtual Shared Disk server node uses buddy buffers to temporarily store data for I/O operations originating at a non-server node. Buddy buffers are used only when a shortage in the switch buffer pool occurs, or on

* 1 terabyte is the maximum supported file system size. File systems greater than 1 terabyte in size are not supported by IBM Service.

certain networks. In contrast to the data in the cache buffer, the data in a buddy buffer is purged immediately after the I/O operation completes.

Buddy buffer space is allocated in powers of two. If an I/O request size is not a power of two, the smallest power of two that is larger than the request is allocated. For example, for a request size of 24KB, 32KB is allocated on the server.

If you do not plan to create any GPFS file system with a block size larger than 16KB, two or three buddy buffers of 16KB each on IBM Virtual Shared Disk server nodes should suffice. Create one buddy buffer of 16KB on each non-server node.

If you do plan to create a GPFS file system with a block size larger than 16KB, one buddy buffer equal to the largest block size used will do for non-server nodes, but the number of buddy buffers on IBM Virtual Shared Disk server nodes should be sufficient to handle the maximum number of disk I/Os expected at any given time. Start with two buffers per physical disk attached to a server. If you monitor the output of the **statvsd** command for queued buddy buffer requests, you can determine whether this setting is correct.

For more information on buddy buffers, see *Managing Shared Disks*.

Disks

Plan how to distribute your disks among the IBM Virtual Shared Disk servers. Two considerations should guide your decision. One involves providing sufficient disks and adapters on the system to yield the required I/O bandwidth. The other involves knowing approximately how much storage capacity you will need for your data. Dedicated IBM Virtual Shared Disk servers should have sufficient disks and adapters to drive the I/O load you expect them to handle. The current maximum number of disks that can be defined for a file system is 512.

Connectivity

If your disks are capable of connection to multiple nodes (twin-tailing) and you wish to exploit this capability, you must select an alternate node as the backup IBM Virtual Shared Disk server. See the *Managing Shared Disks* publication for help in selecting these nodes.

Free Disk Space

Sufficient free disk space needs to be kept in **/var** for the correct operation of the IBM Recoverable Virtual Shared Disk and PSSP components used by GPFS. While GPFS does not use major amounts of **/var**, PSSP components require several MB to correctly support GPFS recovery. Failure to supply this space on all nodes may appear as a hang in your GPFS file system when recovering failed nodes.

Defining IBM Virtual Shared Disks

GPFS uses IBM Virtual Shared Disks to let application programs executing at different nodes access raw logical volumes as if they were local at each of the nodes. Although *Managing Shared Disks*, the IBM Virtual Shared Disk publication, is the definitive source for instructions on how to create IBM Virtual Shared Disks, you can have GPFS create them automatically when you create your file systems.

In order to simplify file system creation, GPFS creates one IBM Virtual Shared Disk for each physical disk specified for the file system, and assigns an optimal partition size based on the disk's capacity. An IBM Virtual Shared Disk name is also automatically generated. If you want to take advantage of the flexibility available in

creating IBM Virtual Shared Disks, follow the instructions in Managing Shared Disks and then pass the existing IBM Virtual Shared Disk to the GPFS file system by specifying the *VSD name*.

In order to define IBM Virtual Shared Disks for your GPFS file system, you will need to know the names of the disks and their servers, and then decide whether to store only data, only metadata, or both on the disks. You may also wish to assign IBM Virtual Shared Disks to specific failure groups in order to avoid single point of failure situations. "Building Disk Descriptors" on page 35 describes the procedure.

Recoverability

GPFS provides several layers of protection against failures of various types:

1. Node failure
2. Server failure
3. Disk failure
4. Connectivity failure

Node Failure

This basic layer of protection covers the failure of file system nodes, and requires that Group Services (the high availability subsystem of Parallel System Support Programs for AIX) and IBM Recoverable Virtual Shared Disk be set up on your system. When an inoperative node is detected by Group Services, GPFS uses the facilities of IBM Recoverable Virtual Shared Disk to fence it off and prevent it from performing write operations that might interfere with recovery. Recovery involves rebuilding metadata structures, which may have been under modification at the time of the failure, and takes place automatically with no administrative action required.

File system recovery from node failure should not be noticeable to applications running on other nodes, except for delays in accessing objects being modified on the failing node. If the failing node is the coordinator of locks for a file system, the delay will be longer and proportional to the activity on the file system at the time of failure, but no administrative intervention will be needed.

Server and Disk Failure

The two most common reasons why data becomes unavailable are disk failure and IBM Virtual Shared Disk server failure. In the event of either type of failure, GPFS discontinues use of the disk and awaits its return to an available state. You can guard against loss of data availability from such failures using one or more of the following methods to maintain additional copies of files.

One means of protection is the use of a Redundant Array of Independent Disks (RAID) controller, which masks disk failures with parity disks. An ideal configuration is shown in Figure 3 on page 15, where a RAID device is twin-tailed to two nodes. This protects against server failure as well.

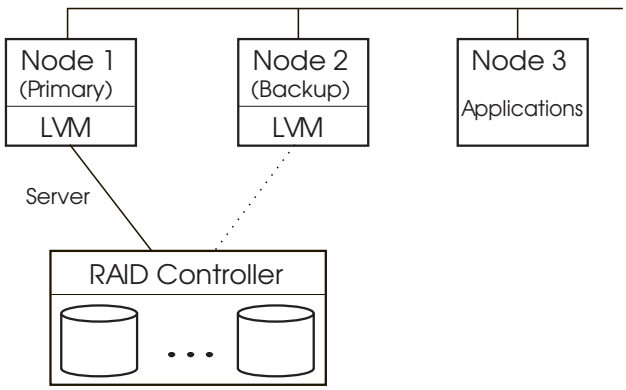


Figure 3. Primary Node Serving RAID Device

If node 1, the primary server , fails, its responsibilities are assumed by node 2, the backup server , as shown in Figure 4.

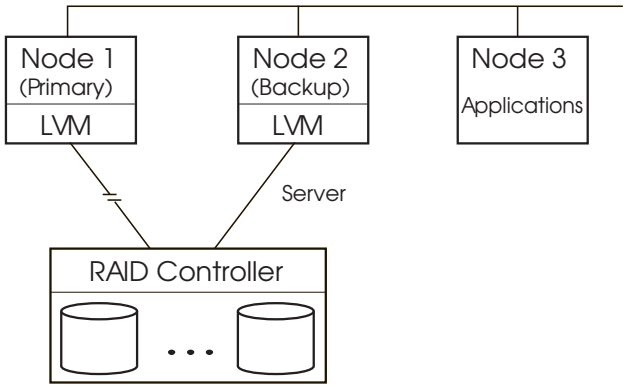


Figure 4. Backup Node Serving RAID Device

You can also protect your file system against disk failure by *mirroring data* at the logical volume manager (LVM), writing the data twice to two different disks. The addition of twin-tailed disks to such a configuration adds protection against server failure by allowing the IBM Recoverable Virtual Shared Disk program to route requests through a backup server.

GPFS offers yet another method of protection, *replication*, which overcomes both disk failure and server failure at the expense of additional disk space. GPFS allows replication of both file data and metadata. This means that two or more instances of data, or metadata, or both, can be automatically created and maintained for any file in a GPFS file system. If one instance becomes unavailable due to disk failure, another instance is used instead. You can set different replication specifications for each file, or apply default settings specified at file system creation.

Metadata and data replication are specified independently. Each has a default replication factor and a maximum replication factor. Although replication of metadata is less costly in terms of disk space than replication of file data, excessive replication of metadata also affects GPFS efficiency because all metadata replicas must be written. In general, more replication uses more space.

Each method of protection has its cost, whether it be the installation of additional hardware or the consumption of large amounts of disk space. If your primary concern is loss of data, and you can tolerate temporary loss of data availability

while an IBM Virtual Shared Disk server reboots, a RAID device may be the most inexpensive protection for you. If, on the other hand, you cannot tolerate loss of data access if a server fails, and you are willing to restore data from a backup tape, a twin-tailed disk, with or without LVM mirroring, might be your best protection. Should the installation of a RAID controller and a twin-tailed disk be equally unappealing, and you want protection from both server and disk failure, replication is the choice for you.

Connectivity Failure

Plan to organize your GPFS disks into a number of *failure groups*. A failure group is a set of disks that share a common point of failure that could cause them all to become simultaneously unavailable. In order to assure file availability, GPFS maintains each instance of replicated data on disks in different failure groups. Even if you do not specify replication when creating a file system, GPFS automatically replicates recovery logs in separate failure groups.

At minimum, organize failure groups that protect against server failure. All disks that share an IBM Virtual Shared Disk server (but do not share any attachment hardware, such as an SSA loop) should belong to a unique failure group, as shown in Figure 5. GPFS builds such discrete failure groups by default if you do not specify *Failure Group* parameters when creating your file system.

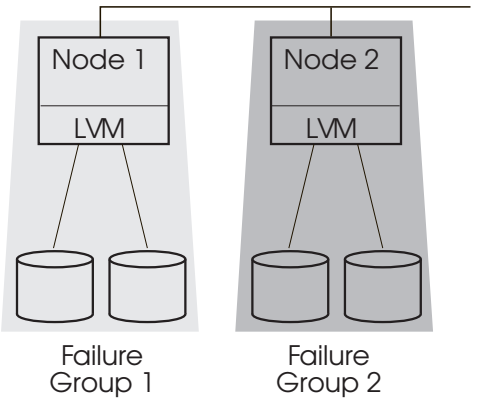


Figure 5. Basic Failure Groups with Servers and Disks

If you plan to use both twin-tailed disks and replication, assign disks to the failure groups with their primary servers, as shown in Figure 6. This arrangement would assure availability of replicated data if either server failed.

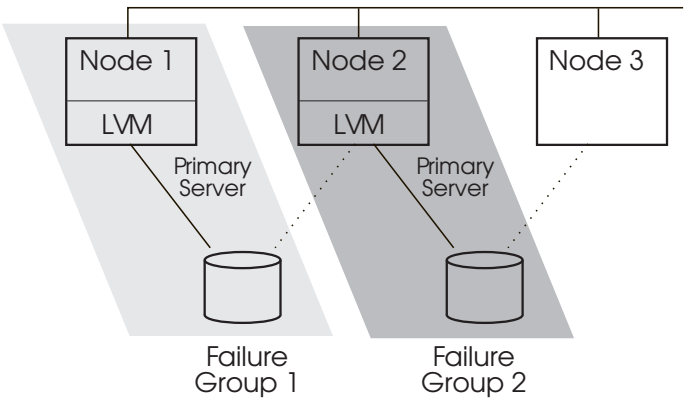


Figure 6. Failure Groups with Twin-Tailed Disks

Recoverability Parameters

Sound file system planning includes consideration of replication, as well as structuring your data so that information is not vulnerable to a single point of failure. It also includes distributing data across a large number of disks. The following parameters enable you to create a highly available file system with fast recoverability from node or disk failure.

At the File System Level: The following recoverability parameters are set at the file system level and apply to all files. They can be changed for an existing file system but modifications only apply to files subsequently created.

Default Metadata Replicas
: This is the default replication factor for metadata. This value must be equal to or less than *MaxMetadataReplicas*, and cannot exceed the number of failure groups with disks that can store metadata. The allowable values are 1 or 2, with a default of 1 (no replication). This parameter also controls the number of copies of i-nodes, directory entries, and indirect blocks for all files in the file system.

Maximum Metadata Replicas
: This is the maximum replication factor for metadata. This parameter also controls the maximum number of copies of i-nodes, directory entries, and indirect blocks for all files in the file system. The allowable values are 1 or 2, with a default of 1 (no replication).

Default Data Replicas
: This is the default replication factor for data blocks. This value must be equal to or less than *MaxDataReplicas*, and the value cannot exceed the number of failure groups with disks that can store data. The allowable values are 1 or 2, with a default of 1 (no replication).

Maximum Data Replicas
: This is the maximum replication factor for data blocks. The allowable values are 1 or 2, with a default of 1 (no replication).

At the Disk Level: You can include these positional parameters in any disk description when creating a GPFS file system or adding or replacing disks in your GPFS file system.

Disk Usage
: What is to be stored on the disk. **metadataOnly** specifies that this disk may only be used for metadata, not for data. **dataOnly** specifies that only data, and not metadata, is allowed on this disk. You can limit vulnerability to disk failure by confining metadata to a small number of conventional mirrored or replicated disks. The default, **dataAndMetadata**, allows both on the disk.

  Note: RAID devices are not well-suited to storing metadata. Their inefficient handling of I/O in small block sizes may affect GPFS performance.

Failure Group
: A number identifying the failure group to which this disk belongs. All disks that have a common point of failure, such as disks that are attached to the same IBM Virtual Shared Disk server node, should be placed in the same failure group. GPFS uses this information during data and metadata placement to assure that no two replicas of the same block will become unavailable due to a single failure. You can specify

any value from –1 (where –1 indicates that the disk has no point of failure in common with any other disk) to 4000. If you specify no failure group, the value defaults to the server node number plus 4000, assuring that replicated data remains available in the event of server failure.

Additional File System Considerations

Two more parameters can be set when you create a file system, and can be changed at a later time:

–A yes | **no**
: Automatic Mount option. Indicates whether the GPFS file system should automatically mount when system reboots and the GPFS daemon starts, provided that enough nodes are up to meet quorum requirements. The default is **yes**.

–s *StripeMethod*
: Set the stripe method for this file system. Possible values are:

  roundRobin
  : Places one block on all of the disks in the file system before writing to the first disk again, repeating the same sequence of disks as the file grows. This is the default.

  random
  : Places the next block randomly, subject only to the constraint that no two replicas of a block be in the same failure group.

  balancedRandom
  : Randomly places one block on all of the disks in the file system before repeating one.

Planning a Sample GPFS Configuration

The first step in planning a GPFS configuration is to evaluate the hardware. Let us say that we have one SP system with two RAID devices and eight disks. Although our SP currently has only one frame with eight nodes, we anticipate adding a second frame of eight nodes next year. Therefore, we estimate the **node count** to be 16.

Next, we create a node list, which we name **gpfs1.nodes**, using the switch hostnames as follows:

```
sp1sw00.ibm.com
sp1sw01.ibm.com
sp1sw02.ibm.com
sp1sw03.ibm.com
sp1sw04.ibm.com
sp1sw05.ibm.com
sp1sw06.ibm.com
sp1sw07.ibm.com
```

Although we could use the Ethernet hostnames, such as sp1en01.ibm.com, in our list, we specify the switch hostnames because we want GPFS to take advantage of the faster switch communication.

We will allow the default values to be applied for **pagepool** and **mallocsize**, as well as **priority**. If we feel performance can be improved, we will reconfigure GPFS with new values for these parameters later.

We want GPFS to start automatically whenever the nodes have been shut down and brought back up, so we will specify the **autoload** option.

This planning exercise provides the information needed to follow the procedure in "Configuring GPFS" on page 30. We have more information to collect, however, before we can actually create a GPFS file system. "Planning a Sample GPFS File System" describes a process leading to this information.

Planning a Sample GPFS File System

File system planning is divided into two phases, one for file system sizing, and one for IBM Virtual Shared Disks.

Sizing Sample File Systems

We want to store two different kinds of files using GPFS: small files that users keep in their home directories, and larger files characteristic of technical applications. Small files do not require high bandwidth for acceptable I/O operations, but large files do. Therefore, the best approach is to create two separate file systems, which we will call **/gpfs/fs1** and **/gpfs/fs2**. This allows each file system to have its own block size and indirect block size. The **gpfs** directory will be mounted in the root (/) directory.

Since **fs1**, where user home directories will reside, will have few files greater than 256KB in length, a **block size** of **16KB** will allow the most efficient use of disk space. The default **i-node** size of **512 bytes** will allow inexpensive storage of small files. The formula below shows that if we replicate metadata but do not replicate data, the largest file size we can expect to have with a **block size** of **16KB**, an **i-node** size of **512 bytes,** and an **indirect block size** of **1KB** is **86.7MB**. Since this is more than we expect to need, these are the values we choose for **fs1**.

$$\text{max\_fsize} = \left(\frac{i - 104}{6M}\right) \times \left(\frac{I - 44}{6R}\right) \times B$$

$$\text{max\_fsize} = \left(\frac{512 - 104}{12}\right) \times \left(\frac{1024 - 44}{6}\right) \times 16384 = \frac{9098581333}{1024^2}\text{MB} = 86.7\text{MB}$$

The majority of files in **fs2**, where our large, technical application files will be stored, will be greater than 128MB in length, so we will define a **block size** of **256KB** to provide faster read/write operations. Since there will be some smaller files, we can accept the default **i-node** value of **512 bytes** to avoid wasting disk space. We want to allow files in **fs2** to grow as large as **20GB**, and the formula tells us that an **indirect block size** of **16KB**, used with a **block size** of **256KB**, and an **i-node** size of **512 bytes**, will allow that capacity if we replicate metadata but not data.

$$\text{max\_fsize} = \left(\frac{512 - 104}{12}\right) \times \left(\frac{16384 - 44}{6}\right) \times 262144 = \frac{245583489706}{1024^3}\ \text{GB} = 228.7\text{GB}$$

Planning Sample IBM Virtual Shared Disks

Now that we know the characteristics of the two GPFS file systems we want to create, we can plan how to store them on our two RAID devices and eight disks.

The relatively small size of files in the **fs1** file system makes the RAID devices ill-suited for their storage, since RAID operates more efficiently with larger block sizes. Therefore, we decide to store **fs1** only on the non-RAID disks, which have a capacity of 4GB each. We will dedicate the RAID devices, each of which holds 50GB, to **fs2**, as Figure 7 shows.

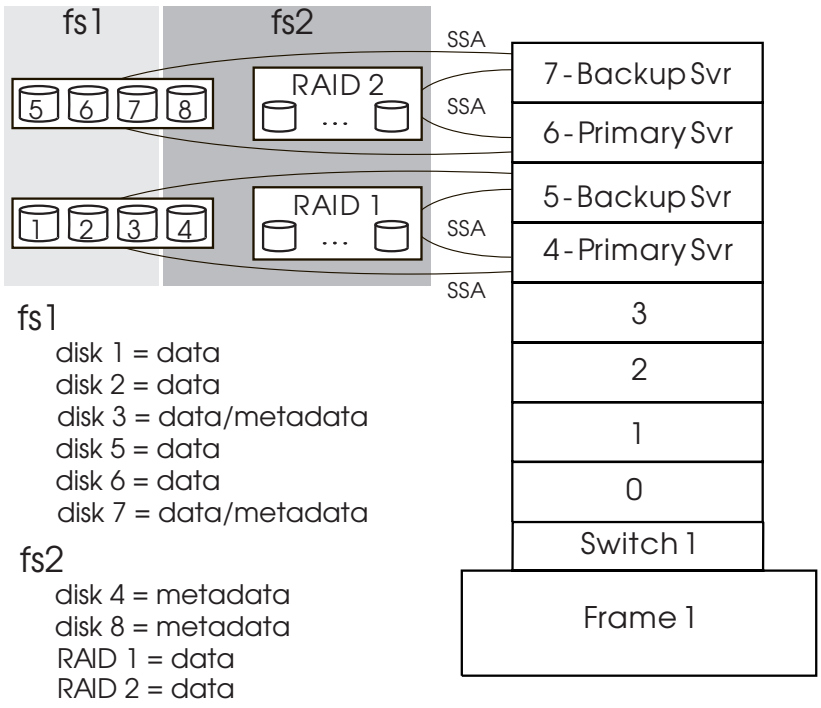


Figure 7. Planning Disks for Sample File Systems

We have decided not to replicate **fs1** data because it will be backed up regularly, and a brief period of unavailability will not be intolerable if it must be restored. We will not replicate **fs2** data either, because the RAID controller will handle recovery. We will, however, replicate metadata for both file systems.

We want to replicate **fs1** metadata in order to limit the impact of a disk failure. Metadata replication will ensure that files with no data blocks on a failed disk will still be accessible, and we will be able to create new files even when the disk is down. Without metadata replication, a large fraction of files would become inaccessible and we might be unable to write to the file system.

We also want to replicate **fs2** metadata to ensure availability, but we do not want to place metadata on RAID disks because of its smaller indirect block size.

According to our plan, **fs1** data will be striped across disks 1, 2, 5, and 6, and part of disks 3 and 7. The **fs1** metadata will be written on the other part of disk 3 and

replicated on part of disk 7. (We could also distribute metadata across all disks in **fs1**, which might be more efficient under some usage conditions.) This allows 20GB of disk space (4 X 4GB + 2 X 2GB) for **fs1** data. Since 20GB does not seem sufficient for anticipated growth, we plan to double this capacity in the coming year. Therefore, we can estimate the number of i-nodes we will need for **fs1** by dividing the maximum projected file size of 89MB by the maximum i-node size, 512, giving us **182,272 i-nodes**.

Data in **fs2** will be striped across the RAID 1 and RAID 2 arrays. Metadata will be written to disk 4 because the relatively small indirect block size of 32KB is inappropriate for RAID devices. The **fs2** metadata will be replicated on disk 8. Although our two RAID devices hold a total of 100GB, we can foresee the need to add six more RAID devices. Therefore, we can divide a future total capacity of 400GB by an average file size of 64MB for an estimated **i-nodes count of 6000** for **fs2**.

Since both our RAID devices would consume too much I/O bandwidth for one server to support, we've decided to create two servers and two backup servers. As Figure 7 on page 20 shows, switch node 4 is the server for RAID 1 and disks 1 through 4, while switch node 6 serves RAID 2 and disks 5 through 8. Switch node 5 is the backup server for node 4 and node 7 backs up node 6. The disk devices are all twin-tailed and are connected through SSA loops, but SCSI adapters could also be used.

When we create the file systems, we will have GPFS create 10 IBM Virtual Shared Disks. These will be automatically partitioned for optimal storage based on capacity.

We will have GPFS create two failure groups organized by servers. **Failure group 1** will contain the **fs1** disks served by node 4. **Failure group 2** will contain **fs1** disks served by node 6. Likewise, **failure group 3** contains **fs2** disks served by node 4, **failure group 4** contains **fs2** disks served by node 6.

These specifications are input to disk descriptor files we will create in “Building Disk Descriptors” on page 35.

Chapter 3. Installing GPFS

Although GPFS installation is straightforward, you will benefit from reading Chapter 2, “Planning for GPFS” on page 5 before continuing.

Do not attempt to install GPFS if you do not have the hardware and software prerequisites listed in “Installation Requirements.”

Assure that the path on your control workstation and on each SP node includes the following:

- **/usr/lpp/csd/bin**

The path on each SP node must include:

- **/usr/lpp/mmfs/bin**

Throughout this procedure you will see various component names beginning with the prefix **mmfs**. This is not an error. GPFS shares many components with the IBM Multi-Media LAN Server, a related product.

Installation Requirements

GPFS is designed for optimum performance in a specific environment. Before attempting installation, be sure that your environment meets the following requirements.

Hardware Requirements

1. RS/6000 SP
2. High Performance Switch or SP Switch
3. Enough Disks to contain file system

Software Requirements

1. AIX Version 4 Release 2.1 (5765–655 or 5765–C34) or later modifications, or AIX Version 4 Release 3.1 (5765–C34) or later
2. Parallel System Support Programs for AIX, Version 2 Release 4 or later (5765–529) with the following options installed:
   - SP System Support Package (ssp.basic)
   - SP Communication Subsystem Package (ssp.css)
   - Group Services High Availability Subsystem (ssp.ha)
   - Sysctl (ssp.sysctl)
   - SP Centralized Management Interface (ssp.csd.cmi)
   - IBM Virtual Shared Disk (ssp.csd. vsd)
   - SP VSD Usability Package (ssp.csd. sysctl)
3. IBM Recoverable Virtual Shared Disk Version 2 Release 1.1 (5765–646) or later with the following options installed:
   - SP High Availability RVSD (rcsd.rvsd)

© Copyright IBM Corp. 1998

- SP RVSD Base Package (rcsd.vsd)

Step 1. Installation Procedure

Follow these steps to install the GPFS software using the **installp** command. This procedure installs GPFS on all SP nodes at once. The use of SMIT requires you to install GPFS on each SP node individually, and is not recommended. Although you can perform the installation procedure from the control workstation, you must log on to an SP node to execute all parts of "Install GPFS on Network" on page 25, with the single exception of "Install Control Workstation Image" on page 25.

Create GPFS Directory

On the control workstation, create a subdirectory in **/spdata/sys1/install/pssplpp/PSSP-2.4** with the following command:

```
mkdir /spdata/sys1/install/pssplpp/PSSP-2.4/mmfs
```

Then copy the installation images from the tape to the new directory, using the **bffcreate** command:

```
bffcreate -qvX -t /spdata/sys1/install/pssplpp/PSSP-2.4/mmfs -d /dev/rmt0.1 all
```

Note: If your tape drive is other than /dev/rmt0, be sure to substitute its no–rewind–on–close device name in place of /dev/rmt0.1.

This will place the following GPFS images in the image directory:

1. mmfs.base.usr.3.0.0.0
2. mmfs.gpfs.usr.1.1.0.0
3. mmfs.util.usr.3.0.0.0
4. mmfs.msg.en_US.usr.3.0.0.0
5. mmfs.man.en_US.shr.3.0.0.0
6. mmfs.gpfsdocs.shr.3.0.0.0

You can also copy files to this directory from other LPPs that you want to install on the same nodes, such as IBM Recoverable Virtual Shared Disk.

Create Installation Image

1. Make the new image directory the current directory:

   ```
   cd /spdata/sys1/install/pssplpp/PSSP-2.4/mmfs
   ```

2. Use the **inutoc** command to create a **.toc** file. The **.toc** file is used by the **installp** command.

   ```
   inutoc .
   ```

3. To view the product **README** after creating the installation images:

   ```
   installp -i -d . mmfs | more
   ```

Install GPFS on Network

Install GPFS according to one of the following procedures, depending on whether or not your network has a shared file system.

To Install on Shared File System Network...

1. Assure that image directory is NFS-exported to the SP nodes. For example, check the output of the **showmount –e sys1cws** command, where *sys1cws* is the hostname of the control workstation. You must specify a control workstation name that is known to the nodes on the SP Ethernet. Your control workstation IP address may also be used.

   Look for the following return:

   ```
   export list for sys1cws.pok.ibm.com:
   /spdata/sys1/install/pssplpp/PSSP-2.4
   ```

2. NFS mount the image directory from every SP node:

   ```
   dsh mount sys1cws:/spdata/sys1/install/pssplpp/PSSP-2.4 /mnt
   ```

3. Run the **installp** command on all nodes from the GPFS installation directory:

   ```
   dsh –f4 installp –agXd /mnt/mmfs all
   ```

   Note: Fanning by 4 nodes is suggested to avoid overloading the SP Ethernet.

To Install on Non-Shared File System Network...

If the GPFS installation directory is not in a shared network file system, copy the images to each node as follows:

```
dsh mkdir /tmp/mmfs
cd /spdata/sys1/install/pssplpp/PSSP-2.4/mmfs
hostlist | pcp –w – mmfs.base.usr.3.0.0.0 /tmp/mmfs
hostlist | pcp –w – mmfs.gpfs.usr.1.1.0.0 /tmp/mmfs
hostlist | pcp –w – mmfs.util.usr.3.0.0.0 /tmp/mmfs
hostlist | pcp –w – mmfs.msg.en_US.usr.3.0.0.0 /tmp/mmfs
hostlist | pcp –w – mmfs.man.en_US.shr.3.0.0.0 /tmp/mmfs
hostlist | pcp –w – mmfs.gpfsdocs.shr.3.0.0.0 /tmp/mmfs
hostlist | pcp –w – .toc /tmp/mmfs
```

Then, install on each node from its local GPFS installation directory:

```
dsh installp –agXd /tmp/mmfs all
```

Install Control Workstation Image

Install the **mmfs.gpfs** image on the control workstation.

```
installp –agXd /spdata/sys1/install/pssplpp/PSSP-2.4/mmfs mmfs.gpfs
```

Step 2. Verify GPFS Installation

Use the **lslpp** command to verify the installation of GPFS filesets on each SP node:

```
lslpp –l mmfs.*
```

Output similar to the following should be returned:

```
Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  mmfs.base.cmds             3.0.0.0  COMMITTED  GPFS File Manager Commands
  mmfs.base.rte              3.0.0.0  COMMITTED  GPFS File Manager
  mmfs.gpfs.rte              1.1.0.0  COMMITTED  GPFS File Manager
  mmfs.msg.en_US             3.0.0.0  COMMITTED  GPFS Server Messages -
                                                 U.S. English
  mmfs.util.cmds             3.0.0.0  COMMITTED  GPFS Server Utilities
  mmfs.util.smit             3.0.0.0  COMMITTED  GPFS Server SMIT Panels

Path: /etc/objrepos
  mmfs.base.rte              3.0.0.0  COMMITTED  GPFS File Manager
  mmfs.gpfs.rte              1.1.0.0  COMMITTED  GPFS File Manager

Path: /usr/share/lib/objrepos
  mmfs.man.en_US.data        3.0.0.0  COMMITTED  GPFS Server Man Pages -
                                                 U.S. English
```

Step 3. Tune Switch

Use the **dsh** command to propagate the following **rpoolsize** and **spoolsize** values to all nodes:

Note: Do not set these values on the control workstation.

```
dsh /usr/lpp/ssp/css/chgcss -l css0 -a rpoolsize=16777216 -a spoolsize=16777216
```

Ignore returned message saying `...poolsize != default`.

Step 4. Configure sysctl

Make the following change on all GPFS nodes, including the control workstation:

- Edit and uncomment the file located at /**etc**/**sysctl.mmcmd.acl** to change the Kerberos realm to the correct name for your SP system.

Refer to *IBM Parallel System Support Programs: Administration Guide* or *IBM RS/6000 SP Management, Easy, Lean, and Mean*, for more information about Kerberos and **sysctl**.

Step 5. Tune IBM Virtual Shared Disk

GPFS requires each of its nodes to enter IBM Virtual Shared Disk parameters in the System Data Repository (SDR). These IBM Virtual Shared Disk parameters must be set to sufficient values in order for GPFS to operate efficiently:

- IP Packet Size
- Cache Buffers
- Request Count
- Buddy Buffers

The purpose of these parameters is outside the scope of this document. Consult the *Managing Shared Disks* manual for detailed descriptions, however, the following

is an example of a **vsdnode** command that could be used to initially setup IBM Virtual Shared Disk for GPFS on an eight-node SP:

```
vsdnode 1 2 3 4 5 6 7 8 css0 64 256 256 48 4096 262144 33 61440
```

On a system where IBM Virtual Shared Disk is already set up, these parameters could be changed using the **updatevsdnode** command instead.

To verify these values on the SP nodes, enter

```
vsdatalst -n
```

Output similar to the following is returned:

```
VSD Node Information
                                          Initial Maximum     VSD      rw      Buddy Buffer
   node                 VSD      IP packet   cache   cache request request minimum maximum size: #
 number host_name       adapter    size    buffers buffers   count   count    size    size maxbufs
 ------ --------------- -------- --------- ------- ------- ------- ------- ------- ------- -------
      1 k13n01.ppd.pok. css0         61440      64     256     256      48    4096  262144      33
      2 k13n02.ppd.pok. css0         61440      64     256     256      48    4096  262144      33
      3 k13n03.ppd.pok. css0         61440      64     256     256      48    4096  262144      33
      4 k13n04.ppd.pok. css0         61440      64     256     256      48    4096  262144      33
      5 k13n05.ppd.pok. css0         61440      64     256     256      48    4096  262144      33
      6 k13n06.ppd.pok. css0         61440      64     256     256      48    4096  262144      33
      7 k13n07.ppd.pok. css0         61440      64     256     256      48    4096  262144      33
      8 k13n08.ppd.pok. css0         61440      64     256     256      48    4096  262144      33
```

"IBM Virtual Shared Disk Parameters and GPFS Performance" on page 71 contains more information on buddy buffers.

Chapter 4. Performing GPFS Administration Tasks

The administration and maintenance of GPFS is organized in six categories:

1. “Configuring GPFS” on page 30
2. “Managing File Systems” on page 34
3. “Managing Disks” on page 49
4. “Managing Nodes” on page 57
5. “Managing GPFS Quotas” on page 59
6. “Managing Access Control Lists” on page 65

Notes:

1. Root authority is required to perform all GPFS administration tasks except those with a function limited to listing GPFS operating characteristics.
2. Kerberos authentication is required to perform most GPFS administration tasks. If you do not have Kerberos authentication, issue the **kinit** command.
3. GPFS administration tasks **cannot be performed on the control workstation**. Perform these tasks from one of the GPFS nodes.

Most GPFS administration tasks can be performed using either SMIT menus or by entering command strings. Whenever you want to perform one of these tasks using SMIT, you can begin by invoking SMIT and selecting Applications→GPFS Administration, or by entering:

```
smit gpfs
```

The SMIT GPFS Administration Menu (Figure 8 on page 30) is displayed:

© Copyright IBM Corp. 1998

Figure 8. SMIT GPFS Administration Menu

You can display help information from the SMIT GPFS Administration Menu by selecting `Help` from the menu bar at the top. All GPFS SMIT dialogs display help information at the task level when you click on the question mark (**?**) button.

Configuring GPFS

Configuration sets up GPFS to run on a partition of SP nodes. It creates the **/etc/cluster.nodes** file and identifies GPFS nodes in the System Data Repository. It also defines settings for GPFS use of memory and control of parallel operations. No GPFS commands can be run until GPFS is configured.

See “Configuration Considerations” on page 5 before performing this procedure.

Note: Configuration can only be performed once. You cannot have two GPFS configurations at the same time, but you can change a configuration. Refer to “Changing GPFS Configuration” on page 33.

Use SMIT or the **mmconfig** command to configure GPFS on the SP system.

Configure Using SMIT

1. On the SMIT GPFS Administration Menu (Figure 8 on page 30) and select `Create Startup Configuration`, or enter the command,

   ```
   smit mmconfig
   ```

   The following SMIT panel is displayed:

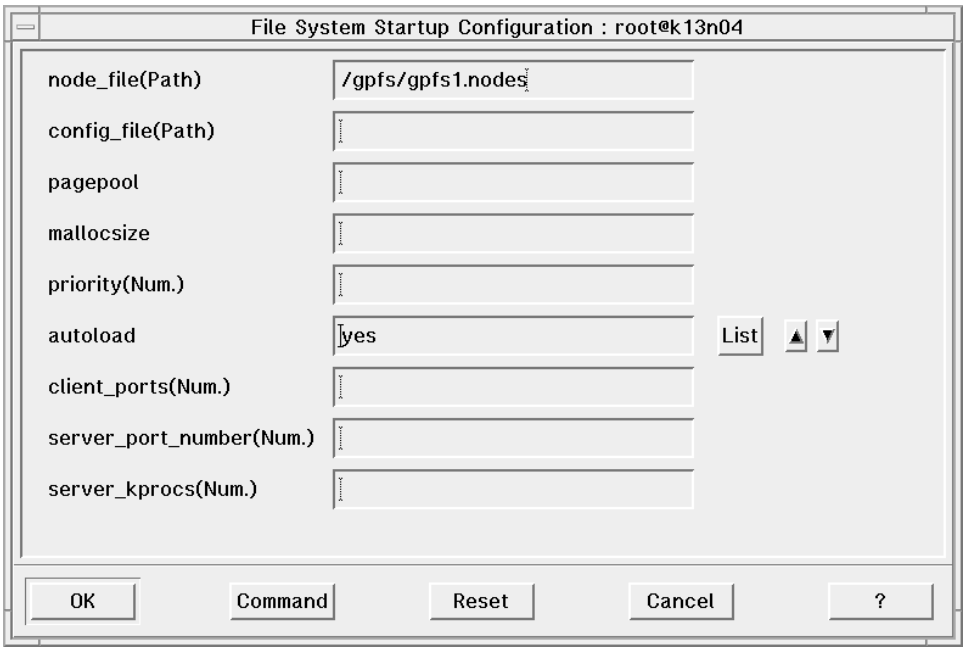


Figure 9. File System Startup Configuration Menu

All parameters on this menu are optional.

If *node_file* is not specified, all nodes in the System Data Repository are added to the configuration.

If *config_file* is not specified, any other parameters left blank default to the values contained in the sample configuration file, **/usr/lpp/mmfs/samples/mmfs.cfg.sample**.

For example, Figure 9 shows the path name of the node file we created in "Planning a Sample GPFS Configuration" on page 18. **Yes** in the space provided next to autoload specifies that we want the GPFS daemons to start automatically whenever the SP nodes boot up. All other parameters are left to default.

Configure Using mmconfig Command

You can use the **mmconfig** command to configure GPFS. For example, to configure all SP nodes and start GPFS automatically when they boot up, enter:

```
mmconfig -A
```

To configure all the nodes in our sample configuration file, **gpfs.nodes** (see "Planning a Sample GPFS Configuration" on page 18), and have GPFS automatically started when the nodes boot up, we would enter:

```
mmconfig -n /gpfs/gpfs1.nodes -A
```

See "mmconfig" on page 94 for a command description and usage information.

Starting and Stopping GPFS

Before starting GPFS

1. **You must have created at least one IBM Virtual Shared Disk (not necessarily for use by GPFS) before starting the GPFS daemons.**

   This is a requirement of IBM Recoverable Virtual Shared Disk, which GPFS uses. If no IBM Virtual Shared Disk exists in your system, GPFS will not start.

2. **DO NOT** start GPFS until it is configured. The first time you start GPFS, you must do so manually using the **startsrc** command. If you configure GPFS to start automatically when the nodes boot up, you will not have to start GPFS manually the next time you reboot your nodes.

   If you have not configured GPFS, using either SMIT or the **mmconfig** command, starting GPFS will load dummy kernel extensions and you will be unable to create a file system. If this occurs, configure GPFS and reboot the nodes.

3. Verify that the /**etc**/**sysctl.mmcmd.acl** file on each node contains the correct name for the Kerberos realm. (See “Step 4. Configure sysctl” on page 26.)

Starting GPFS

1. Set the WCOLL variable to target all nodes in the GPFS configuration for the **dsh** command. *IBM Parallel Systems Support Programs: Administration Guide*, *IBM Parallel Systems Support Programs: Command and Technical Reference*, and *IBM RS/6000 SP Management, Easy, Lean, and Mean* all contain information on the WCOLL variable.

2. Assure that the required availability services groups are established by issuing the following command:

   ```
   lssrc -l -s hags
   ```

   Check for the following two groups in the output:

   - `cssMembership`
   - `ha.vsd`

     If `ha.vsd` does not appear in the list even though IBM Recoverable Virtual Shared Disk is installed, it may be that the node has not yet been rebooted. If you believe this to be the case, check the /**etc**/**inittab** file for the following command entry:

     ```
     rvsd:2:once:/usr/lpp/csd/bin/ha_vsd > /dev/console 2>&1
     ```

     If the entry exists, you can run the command as follows:

     ```
     /usr/lpp/csd/bin/ha_vsd  /dev/console 2>&1
     ```

     If you rerun the **lssrc** command now, both groups should appear in the output.

3. Start the daemons by entering:

   ```
   dsh startsrc -s mmfs
   ```

   The GPFS daemons will start on all nodes that the configuration process recorded in the /**etc**/**cluster.nodes** file.

Check the messages recorded in **/var/adm/ras/mmfs.log*** on one node for the following:

```
mmfsd initializing ...
mmfsd ready for sessions.
```

This indicates successful startup of a quorum of nodes.

If these messages are not written to the log, your system may not have met the requirements for a quorum. See “Quorums” on page 6 for more information.

Stopping GPFS

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

```
dsh stopsrc -c -s mmfs
```

Changing GPFS Configuration

Once GPFS has been configured, your file system operations should function efficiently. You may, however, be able to tune your configuration for better performance by reconfiguring one or more attributes. Refer to Chapter 5, “Tuning GPFS Performance” on page 69 for a detailed discussion.

You can use SMIT or the **mmchconfig** command to change the following configuration attributes after the initial configuration has been set.

1. pagepool
2. mallocsize
3. priority
4. autoload
5. client_ports
6. server_port_number[1]
7. server_kprocs

Items 1 through 3 take effect the next time GPFS is started. Items 4 through 7 require that the nodes be rebooted before new values take effect.

Before You Change the GPFS Configuration

Consider how the changes will affect GPFS operation. Refer to Chapter 5, “Tuning GPFS Performance” on page 69.

Changing Configuration Using SMIT

1. On the SMIT GPFS Administration Menu (Figure 8 on page 30), select `Change Startup Configuration` or enter the command,

   ```
   smit mmchconfig
   ```

   The SMIT dialog shown in Figure 10 on page 34 is displayed.

[1] server_port_number must be changed on all GPFS nodes. You cannot change it on one node at a time.

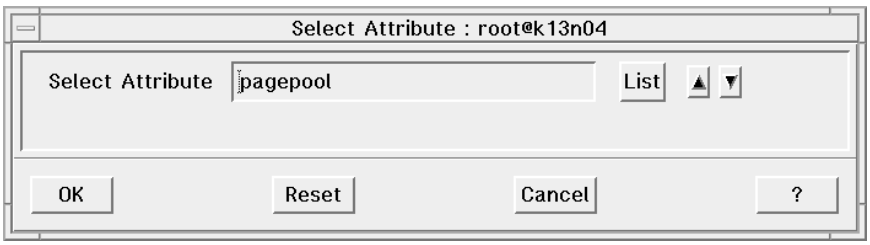


Figure 10. Select Attribute Dialog

2. Click on `List` to display the parameters that you can change. Select the target parameter and click on `OK`. A SMIT dialog similar to Figure 11 is displayed.

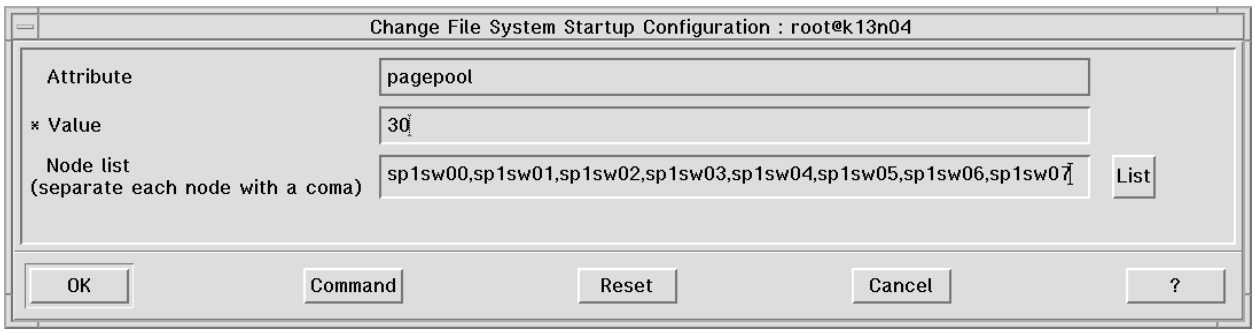


Figure 11. Change File System Startup Configuration Dialog

3. Enter the new value for the parameter and the hostnames of the target nodes for the configuration change. Figure 11 shows the nodes from "Planning a Sample GPFS Configuration" on page 18. Click on **OK**.

Changing Configuration Using mmchconfig

Specify the nodes you wish to target for change, and the attributes with their new values, in the **mmchconfig** command. For example, to change the priority of the GPFS daemon to 2 on all the nodes in the node list we created in "Planning a Sample GPFS Configuration" on page 18, we would enter:

```
mmchconfig priority=2
```

allowing the node list to default to all configured nodes.

Refer to "mmchconfig" on page 83 for syntax and usage information.

Managing File Systems

GPFS file system management tasks can be performed using either SMIT menus or by entering command strings. These tasks include:

1. "Creating a File System" on page 35
2. "Building Disk Descriptors" on page 35
3. "Deleting a File System" on page 43
4. "Modifying File System Attributes" on page 46

.

Creating a File System

You can use SMIT or the **mmcrfs** command to create a GPFS file system.

Before You Begin...

Some characteristics of a file system are not easily changed. Before you proceed, be sure you have read "File System Creation Considerations" on page 7.

Before you create your file system, you need to:

1. Decide which disks to use for the file system and how to use them.

   Have ready a list of all disks you want GPFS to use, as well as any existing IBM Virtual Shared Disks you plan to use in the file system. Note too, whether each disk will store data, metadata, or both (the default). See "IBM Virtual Shared Disks" on page 12.

2. Decide how to structure the file system.

   Write down the block size, i-node size, and indirect size you plan to use. See "File System Sizing" on page 8.

3. Decide how to replicate your files.

   Write down the replication factors for data and metadata for the file system. The default is no replication.

4. Structure Failure Groups.

   Record a failure group number for each disk you plan to use. Default failure groups are organized by server. See "Recoverability" on page 14.

5. Be sure that GPFS has been started. See "Starting and Stopping GPFS" on page 32.

Building Disk Descriptors

Using the information gathered in "Before You Begin...," create one disk descriptor for each IBM Virtual Shared Disk in your file system. You can use a SMIT dialog to enter and save your specifications in a disk descriptor file, or use any AIX editor to create such a file.

You must provide a descriptor for each GPFS IBM Virtual Shared Disk to be created or passed to the GPFS file system. Each descriptor contains the following positional parameters for its IBM Virtual Shared Disk.

Disk Name

The device name, for example, `hdisk1`, of any disk you wish to use to create an IBM Virtual Shared Disk, or the name of an existing IBM Virtual Shared Disk.

GPFS performance and recovery processes function best with one disk per IBM Virtual Shared Disk. If you want to create IBM Virtual Shared Disks with more than one disk, refer to Managing Shared Disks. Then pass the existing IBM Virtual Shared Disks to GPFS using the VSD names.

Server Name

The name of the IBM Virtual Shared Disk server node. This can be in any recognizable form and is unnecessary when specifying a predefined IBM Virtual Shared Disk.

Backup Server Name
The backup server name. This can be in any recognizable form and is unnecessary when specifying a predefined IBM Virtual Shared Disk.

Disk Usage
Specify one of the following or allow to default:

- **dataAndMetadata** (default)
- **dataOnly**
- **metadataOnly**

.

Failure Group
A number identifying the failure group to which this disk belongs. All disks that have a common point of failure, such as all disks that are attached to the same VSD server node, should be placed in the same failure group. GPFS uses this information during data and metadata placement to assure that no two replicas of the same block are written in such a way as to become unavailable due to a single failure. You can specify any value from -1 (where -1 indicates that the disk has no point of failure in common with any other disk) to 4000. If you do not specify a failure group, the value defaults to the server node number plus 4000.

Notes:

1. To use an existing IBM Virtual Shared Disk volume group in the file system, only the *Disk Name* (the IBM Virtual Shared Disk name) need be specified in the disk descriptor. The *Disk Usage* and *Failure Group* parameters are optional and may be allowed to default.
2. To create an IBM Virtual Shared Disk with one disk and use it in a file system, you must specify the *Disk Name* and *Server Name* parameters in the descriptor. *Backup Server*, *Failure Group*, and *Disk Usage* are optional.

Examples: The following examples show disk descriptors defined in several different ways.

1. Start with a basic disk descriptor that takes advantage of GPFS defaults:

   ```
   hdisk5:sp1sw06:sp1sw07::
   ```

   This descriptor translates as follows:

   Disk Name hdisk5

   Server Name Switch node 6

   Backup Server Switch node 7

   Disk Usage Defaults to **dataAndMetadata**, allowing both

   Failure Group Defaults to 4006 (server node number + 4000)

   This descriptor would create an IBM Virtual Shared Disk with the name **gpfs1n6** on hdisk5.

2. If we add another descriptor to the file,

   ```
   hdisk6:sp1sw04::dataOnly:28
   ```

   which translates as follows:

Disk Name hdisk6

Server Name Switch node 4

Backup Server None

Disk Usage Data, not metadata

Failure Group 28

GPFS would also create an IBM Virtual Shared Disk with the name **gpfs2n4** on hdisk6.

3. This descriptor,

   ```
   hdisk9:sp1sw04::metadataOnly:-1
   ```

   translates to:

   Disk Name hdisk9

   Server Name Switch node 4

   Backup Server none

   Disk Usage Metadata, not data

   Failure Group Creates a unique failure group containing only this device

   This descriptor tells GPFS to also create an IBM Virtual Shared Disk with the name **gpfs3n4** on hdisk9.

4. One additional descriptor,

   ```
   vsdx3n12::::
   ```

   which could also be entered simply as

   ```
   vsdx3n12
   ```

   translates to:

   Disk Name vsdx3n12, the name of an existing IBM Virtual Shared Disk

   Server Name Predefined

   Backup Server Predefined

   Disk Usage Defaults to **dataAndMetadata**, allowing both

   Failure Group Defaults to server node number + 4000

   This descriptor specifies that an existing IBM Virtual Shared Disk with the name **vsdx3n12** be added to the GPFS file system.

Building a Disk Descriptor File Manually: If you create a disk descriptor file using an AIX editor, follow these rules:

1. Enter one descriptor to a line
2. Separate parameters within a descriptor with a colon (:)
3. Indicate an omitted parameter by skipping its position and entering two consecutive colons (::). For example, if you had no backup server and wanted to use your disk for both data and metadata (the default), you could create a disk descriptor that looked like this:

   ```
   hdisk1:sp1sw05:::5
   ```

   If you also wanted to allow the last parameter, Failure Group, to default, you could omit the last three delimiting colons:

```
hdisk1:sp1sw05
```

or leave them in if you prefer:

```
hdisk1:sp1sw05:::
```

Trailing delimiters are acceptable.

Sample Disk Descriptor Files: Using the information we gathered in "Planning Sample IBM Virtual Shared Disks" on page 20, we can create a disk descriptor file for each file system we planned.

fs1desc contains the following:

```
hdisk1:sp1sw04:sp1sw05:dataOnly:1
hdisk2:sp1sw05:sp1sw05:dataOnly:1
hdisk3:sp1sw04:sp1sw05:dataAndMetadata:1
hdisk5:sp1sw06:sp1sw07:dataOnly:2
hdisk6:sp1sw06:sp1sw07:dataOnly:2
hdisk7:sp1sw06:sp1sw07:dataAndMetadata:2
```

fs2desc contains the following:

```
pdisk1:sp1sw04:sp1sw05:dataOnly:3
pdisk2:sp1sw06:sp1sw07:dataOnly:4
hdisk4:sp1sw04:sp1sw05:metadataOnly:3
hdisk8:sp1sw06:sp1sw07:metadataOnly:4
```

Building Disk Descriptors Using SMIT: If you are creating a number of disk descriptors, use an editor such as **vi**. The SMIT dialog is better suited to creating or modifying a single descriptor.

1. On the SMIT GPFS Administration Menu (Figure 8 on page 30), select `Prepare Disk Descriptor File`. A dialog box asks if you are creating a descriptor file. Click on **OK** if you are starting a new file. The **List** button allows you to choose **Append**, if you are adding descriptors to an existing file, or **Modify**, to change an existing descriptor in a file. Enter a pathname for your disk descriptor file and click on **OK**. The Create/Change a Disk Descriptor dialog appears.

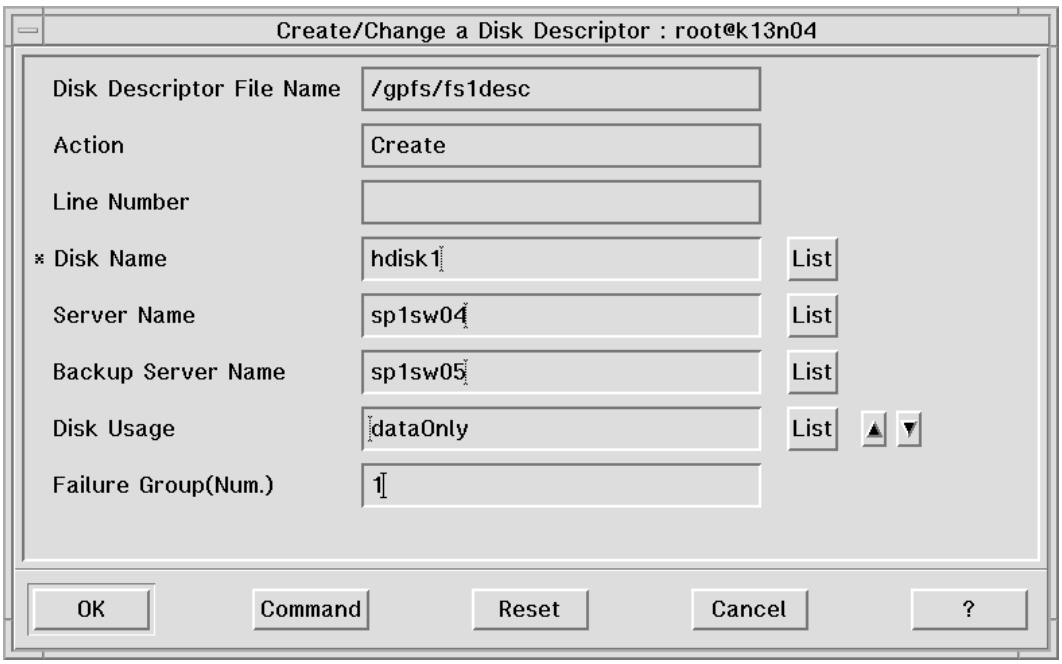


Figure 12. Create/Change a Disk Descriptor Dialog

Figure 12 shows the values we would enter for one of the disk descriptors we planned in "Planning Sample IBM Virtual Shared Disks" on page 20.

Create File System Using SMIT

Note: The following options cannot be specified when creating a file system using SMIT:

- Maximum data replication
- Default data replication
- Maximum metadata replication
- Default metadata replication

To specify replication factors (the default is no replication), see "Create File System Using mmcrfs" on page 40.

1. On the SMIT GPFS Administration Menu (Figure 8 on page 30), select `Create File System`, or enter the command,

   ```
   smit mmcrfs
   ```

   A dialog appears in which you choose `Descriptor File` or `Disk Descriptor List` as the disk descriptor source. To create **fs1** from "Planning a Sample GPFS File System" on page 19, we would select `Descriptor File`. Click on **OK**.

   The SMIT dialog shown in Figure 13 on page 40 is displayed.

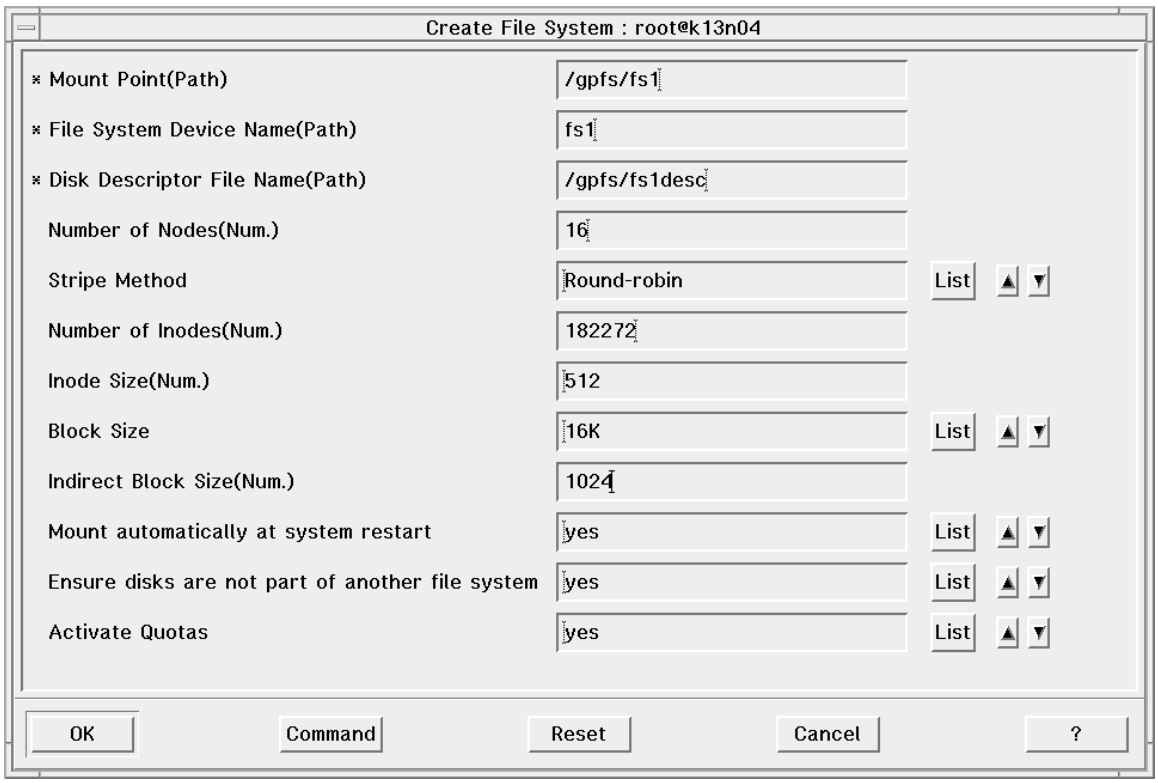


Figure 13. Create File System Dialog

2. In the Create File System dialog, required parameters are indicated with an asterisk (*). Enter the file system name in the space provided for device name, and the mount point for the file system. Since we selected `Descriptor File` in the previous dialog, the third required parameter is the path name for our disk descriptor file. Had we specified `Disk Descriptor List` in the previous dialog, we would be prompted for our list of disk descriptors in this space. All other parameters in this dialog can be allowed to default. "File System Creation Considerations" on page 7 describes these parameters and explains their defaults.

   To create **fs1** from "Planning a Sample GPFS File System" on page 19, we have entered the path name of the disk descriptor file we created in "Sample Disk Descriptor Files" on page 38. We then entered the remaining values from "Sizing Sample File Systems" on page 19 and clicked on **OK**.

Create File System Using mmcrfs

Specify the file system name, mount point, and disk descriptor parameters with the **mmcrfs** command. For example, to create file system **fs1** from "Planning a Sample GPFS File System" on page 19 using the disk descriptor file named /**gpfs**/**fs1desc**, and have it automatically mounted at /**gpfs**/**fs1** when the system reboots, we could enter:

```
mmcrfs /gpfs/fs1 fs1 -F /gpfs/fs1desc -A yes -B 16K -i 512 -I 1K -M 2 -m 2 -n 16 -N 2600000 -Q yes -v yes
```

Likewise, the second file system we designed in "Planning a Sample GPFS File System" on page 19 could be created by entering:

```
mmcrfs /gpfs/fs2 fs2 -F /gpfs/fs2desc -A yes -B 256K -i 512 -I 32K -M 2 -m 2 -n 16 -N 6000 -Q yes -v yes
```

In “Planning a Sample GPFS File System” on page 19 we decided against replicating data for either **fs1** or **fs2** so we allowed the **-r** options to default to 1 when issuing the **mmcrfs** commands. We have, however, specified values of 2 for metadata replication and maximum metadata replication for both file systems. These options are not available when using SMIT.

See “mmcrfs” on page 97 for a detailed command description and usage information.

Mounting a File System

You must explicitly mount a GPFS file system if one or both of the following are true:

1. This is the first time the file system is being mounted after its creation
2. You did not specify the automatic mount option (**-A yes**) when you created the file system

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

You can mount a GPFS file system using SMIT or the AIX **mount** command.

Mount a GPFS File System Using SMIT

1. Enter `smit fs` and select `Mount File System`. The AIX Mount a File System dialog is displayed.

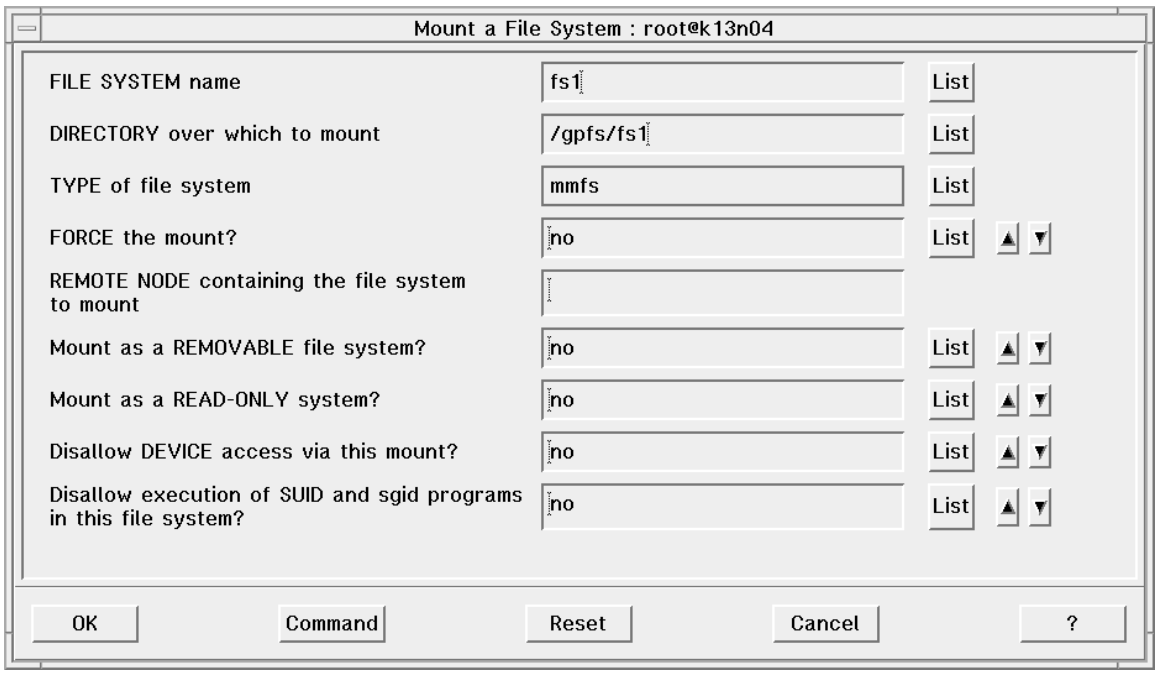


Figure 14. Mount File System Dialog

2. Type the file system name and mount point in the prompt space. The file system type is **MMFS**. Enter additional values as you choose. Figure 14 shows the values we would enter to mount **fs1** from “Planning a Sample GPFS File System” on page 19. Click on **OK**.

Mount a GPFS File System Using AIX mount Command

To mount a GPFS file system using the mount command, enter:

```
mount directory
```

where *directory* is the file system mount point. For example, to mount the sample file systems from “Planning a Sample GPFS File System” on page 19, we would enter:

```
mount /gpfs/fs1
mount /gpfs/fs2
```

Unmounting a File System

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

Unmount a GPFS File System Using SMIT

1. Enter `smit fs` and select `Unmount File System`. The Unmount a File System dialog is displayed.

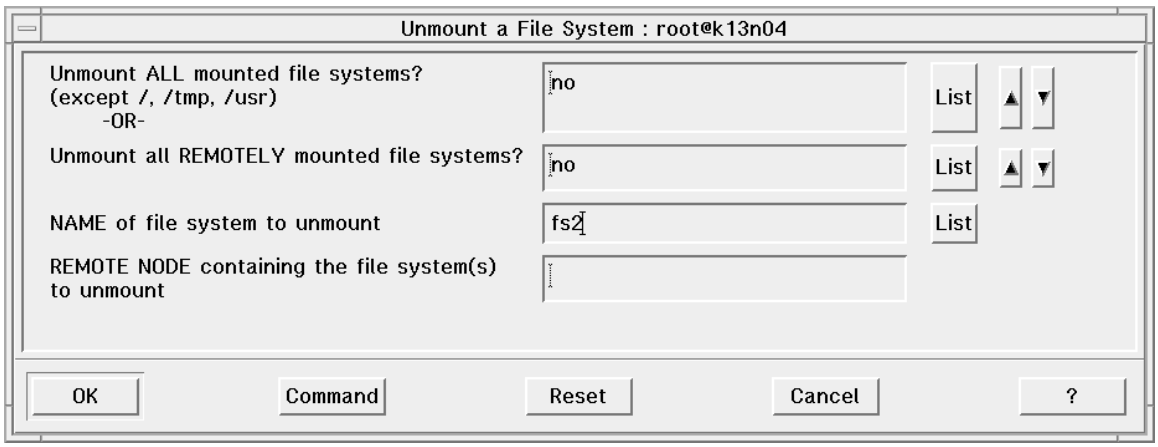


Figure 15. Unmount File System Dialog

2. Type the name the file system in the prompt space. Figure 15 shows the entry we would make to unmount **fs2** . Click on **OK**.

Unmount a GPFS File System Using the AIX unmount Command

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

```
unmount directory
```

where *directory* is the file system mount point. For example, to unmount the sample file systems from “Planning a Sample GPFS File System” on page 19, we would enter:

```
unmount /gpfs/fs1
unmount /gpfs/fs2
```

Deleting a File System

You can use SMIT or the **mmdelfs** command to delete a GPFS file system.

Before Deleting a File System

Unmount the file system on all nodes before deleting it. See “Unmounting a File System” on page 42.

Delete a File System Using SMIT

1. On the SMIT GPFS Administration Menu (Figure 8 on page 30), select `Delete File System` or enter the command:

   ```
   smit mmdelfs
   ```

   The Delete File System dialog is displayed:

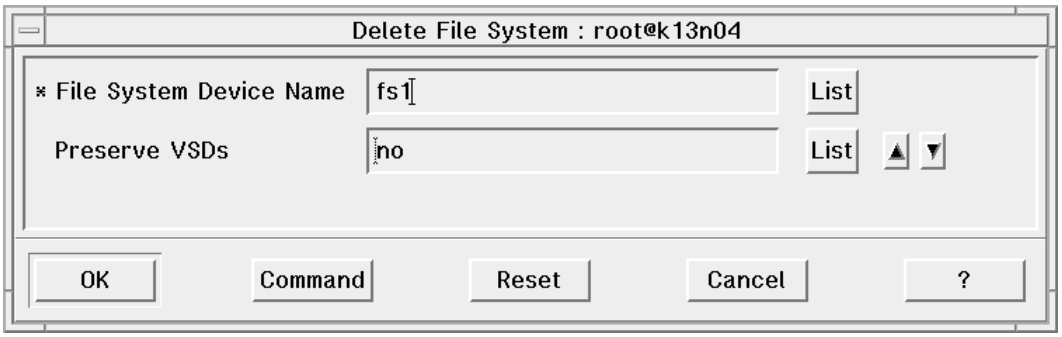


Figure 16. Delete File System Dialog

2. Type the file system name where prompted. Specify whether or not GPFS should preserve the IBM Virtual Shared Disks it created with the file system. Click on **OK**.

Delete a File System Using mmdelfs

Specify the file system you wish to remove with the **mmdelfs** command. For example, to delete the file systems from “Planning a Sample GPFS File System” on page 19, enter:

```
mmdelfs fs1
mmdelfs fs2
```

Refer to “mmdelfs” on page 108 for usage information.

Checking and Repairing a File System

The **mmfsck** command is useful for finding and repairing conditions that can cause problems in your file system.

If a file system will not mount or you receive messages to the effect that a file cannot be read, you can have GPFS check and repair the unmounted file system.

If a GPFS file operation fails due to an out of space condition, the cause may be disk blocks that have become unavailable after repeated node failures. You can check the mounted file system for unallocated blocks using the **mmfsck** command.

Use the **mmfsck** command to have GPFS check for the following file inconsistencies that might cause problems:

- Blocks marked allocated that do not belong to any file. These can be repaired while the file system is mounted. The corrective action is to mark the block free in the allocation map.
- Files for which an i-node is allocated and no directory entry exists (orphaned files). The corrective action is to create directory entries for these files in a **lost+found** subdirectory at the root of this file system. The index number of the i-node is assigned as the name. If you do not allow **mmfsck** to reattach an orphaned file, it asks for permission to destroy the file.
- Directory entries pointing to an i-node that is not allocated. The corrective action is to remove the directory entry.
- Ill-formed directory entries. The corrective action is to remove the directory entry.
- Incorrect link counts on files and directories. The corrective action is to update them with accurate counts.
- Cycles in the directory structure. The corrective action is to break any detected cycles. If the cycle was a disconnected cycle, the new top level directory is moved to the **lost+found** directory.

Before Repairing a File System

You must unmount the file system if you want repairs made. The only exception is the freeing of allocated blocks that do not belong to any file. Plan to make file system repairs when system demand is low. This is I/O intensive activity and it can affect system performance.

Checking and Repairing a File System Using mmfsck

Specify the file system you want to check and a repair option with the **mmfsck** command. For example, to check our sample **fs1** file system from "Planning a Sample GPFS File System" on page 19 and save the report in a file called /**mmfs/fs1_status** without making any changes to the file system, we would enter:

```
mmfsck fs1 -n > /mmfs/fs1_status
```

Output similar to the following is written to the **fs1_status** file:

```
Checking "fs1"
Checking inodes
Checking directories and files
Checking log files
Processing inodes

        7680 inodes
         907   allocated
           0   repairable
           0   repaired
           0   damaged
           0   deallocated
           0   orphaned
           0   attached

      823296 subblocks
       38211   allocated
           0   unreferenced
           0   deletable
           0   deallocated


File system is clean
```

For usage information and complete syntax, see “mmfsck” on page 118.

Listing File System Attributes

GPFS allows you to display the current values for many file system attributes.

Listing File System Attributes Using mmlsfs

Specify the file system and attributes to list with the **mmlsfs** command. Attributes are specified with the following flags:

-a Display the estimated average file size, in bytes

-B Display the block size, in bytes

-d Display the names of all disks in the file system

-f Display the fragment size, in bytes

-i Display i-node size, in bytes

-I Display the indirect block size, in bytes

-m Display the default number of replicas for metadata

-M Display the maximum number of replicas for metadata

-n Display the estimated number of nodes that will mount the file system

-Q Display which quotas are currently enforced on the file system

-r Display the default number of replicas for data

-R Display the maximum number of replicas for data

-s Display the stripe method for this file system

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

For example, to list the expected average file size and current stripe method for our sample **fs1** file system from “Planning a Sample GPFS File System” on page 19, we would enter:

```
mmlsfs fs1 -a -s
```

or, to list all attributes for **fs0**, simply enter:

```
mmlsfs fs0
```

For output examples and more information, see “mmlsfs” on page 127.

Modifying File System Attributes

There are five file system attributes that you can change for an existing file system:

1. Automatic Mount

   Change the specification that determines whether or not your file systems are automatically mounted when the GPFS daemon starts.

2. Default Metadata Replication

   If your file system availability requirements change, you may want to begin or cease replicating metadata for new files.

3. Default Data Replication

   If your file system availability requirements change, you may want to begin or cease replicating data for new files.

4. Stripe Method

   You may want to determine whether your file system efficiency improves with another stripe method.

5. Quota Enforcement

   Activate quota enforcement whenever the GPFS daemon starts.

Note: Changes to the Default Data and Metadata Replication attributes, as well as to the Stripe Method attribute, apply only to new files created after the attributes change. If you want to change the replication of existing files, you must use the **mmchattr** command. See “Querying and Changing File Replication Attributes” on page 47. To change the stripe method of existing files, you must restripe the file system using the **mmrestripefs** command. See “Restriping a GPFS File System” on page 48.

Specify the file system name and the target attribute with its new value using SMIT or the **mmchfs** command.

Change File System Attributes Using SMIT

1. Select `Modify File System` from the SMIT GPFS Administration Menu, Figure 8 on page 30, or enter:

   ```
   smit mmchfs
   ```

   The Modify File Systems dialog is displayed:

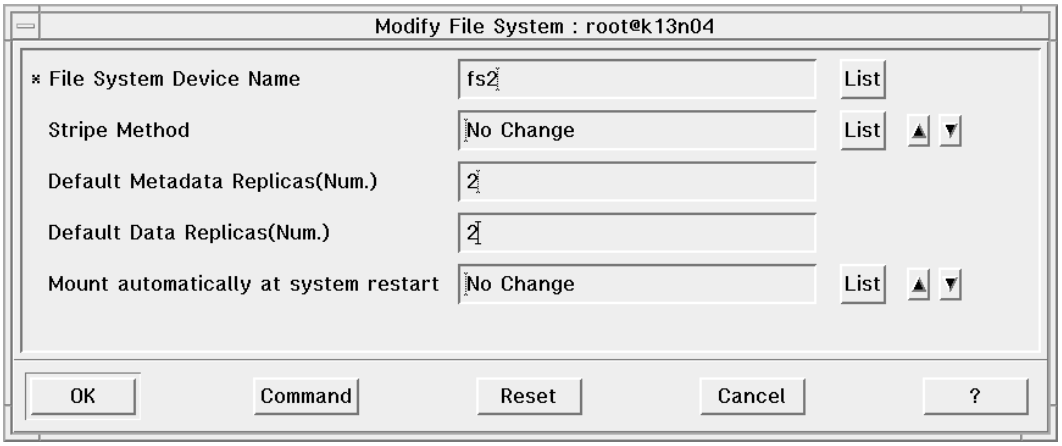


Figure 17. Modify File System Dialog

2. Enter the new value for the target attribute(s). The Stripe Method and Automatic Mount attribute choices can be displayed for selection by clicking on **List**. Figure 17 shows how we would change the default data replication factor to 2 for sample file system **fs2** from "Planning a Sample GPFS File System" on page 19. Click on **OK**.

Change File System Attributes Using mmchfs

1. Specify the file system and target attribute with its new value, using the **mmchfs** command with one of the following options:

   -A Automatic Mount

   -m Default Metadata Replication

   -Q Activate quota enforcement

   -r Default Data Replication

   -s Stripe Method

   For example, to change the default data replication factor to 2 for sample file system **fs1** from "Planning a Sample GPFS File System" on page 19, we would enter:

   ```
   mmchfs fs1 -r 2
   ```

See "mmchfs" on page 92 for complete syntax and usage information.

Querying and Changing File Replication Attributes

If you become concerned that your file system is using too much disk space, or if your availability requirements change, you can have GPFS display the current replication factors for one or more files by using the **mmlsattr** command. You might then decide to change replication for one or more files with the **mmchattr** command.

Querying File Replication Using mmlsattr

Specify one or more filenames with the **mmlsattr** command. For example, to display the replication factors for two files named **project4.sched** and **project4.resource** in sample file system **fs1** from "Planning a Sample GPFS File System" on page 19, we would enter:

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

Refer to "mmlsattr" on page 123 for details about the information returned by this command.

Change File Replication Attributes Using mmchattr

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

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

Specify the filename, attribute, and new value with the **mmchattr** command. For example, to change the default metadata replication factor to 2 and the default data replication factor to 2 for the file named **project7.resource** in sample file system **fs1** from "Planning a Sample GPFS File System" on page 19, we would enter:

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

Refer to "mmchattr" on page 81 for syntax and usage information.

Restriping a GPFS File System

If you have added disks to a GPFS file system that is seldom updated, you can restripe the file system to distribute data across all disks using the **mmrestripefs** command.

Note: Consider this operation carefully. Restriping involves a large amount of insert and delete activity and may affect system performance. A file system as large as a terabyte may take a long time to restripe.

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

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

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

Before Restriping a File System

Consider the necessity of restriping and the current demands on the system. Files that are written with any regularity are automatically restriped. Restriping a large file system requires extensive insert and delete operations and may result in processing delays. Plan to perform this task when system demand is low.

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

1. Use the **mmchdisk** command to suspend any disks to which you DO NOT want the file system restriped. You may want to exclude disks from file system restriping because they are failing. (See "Changing GPFS Disk States and Parameters" on page 55.)
2. Use the **mmlsdisk** command to assure that all disk devices to which you DO WANT the file system restriped are in the up/normal state. (See "Displaying GPFS Disk States" on page 53.)

Restriping File System Using mmrestripefs

Specify the target file system with the **mmrestripefs** command. For example, to rebalance (**-b** option) sample file system **fs2** after adding an additional RAID device, we would enter:

```
mmrestripefs fs2 -b
```

Refer to "mmrestripefs" on page 139 for syntax and usage information.

Querying File System Space

Although you can use the AIX **df** command to summarize the amount of free space on all GPFS disks, the **mmdf** command is useful for determining how well-balanced the file system is across your disks. You can also use **mmdf** to diagnose space problems that might result from fragmentation.

Query Space Using mmdf

Specify the file system you want to query with the **mmdf** command. For example, to query available space on all disks in the sample file system **fs2** from "Planning a Sample GPFS File System" on page 19, we would enter:

```
mmdf fs2
```

Refer to "mmdf" on page 112 for usage and syntax information.

Managing Disks

There are five disk-related tasks you can perform on a GPFS file system:

1. "Adding Disks to a File System" on page 50
2. "Deleting Disks from a File System" on page 51
3. "Displaying GPFS Disk States" on page 53

4. “Changing GPFS Disk States and Parameters” on page 55
5. “Replacing Disks in a GPFS File System” on page 52

Most can be done through SMIT dialogues as well as through GPFS commands.

Adding Disks to a File System

Many files grow rapidly and you may decide that more disk space is required for your file system soon after creation. You can add disks to a GPFS file system using either SMIT or the **mmadddisk** command.

Add Disks Using SMIT

1. Select `Add Disk(s) to a File System` from the SMIT GPFS Administration Menu (Figure 8 on page 30) . A dialog window appears in which you choose `Descriptor File` or `Disk Descriptor List` as the disk descriptor source. To add a new 9GB disk to **fs1** from “Planning a Sample GPFS File System” on page 19, we would select `Disk Descriptor List`. Click on **OK**. The following SMIT dialog is displayed:

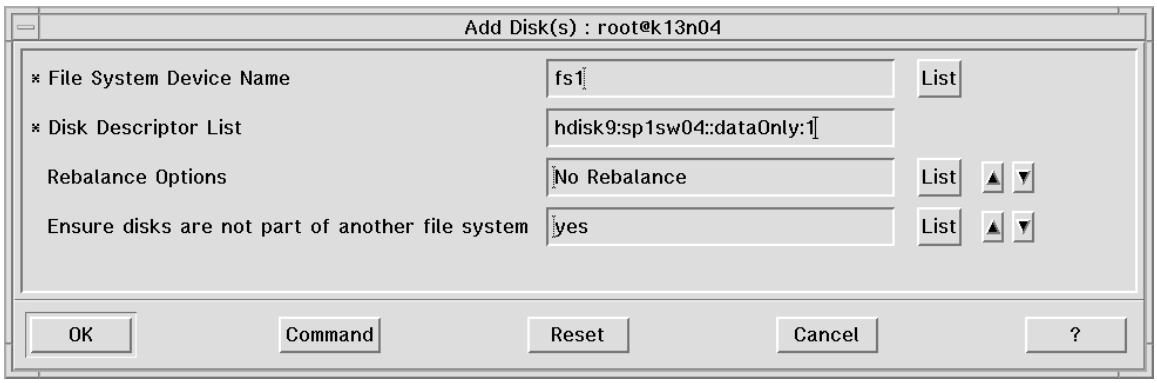


Figure 18. Add Disk(s) Dialog

2. Enter all information at the prompts, or click on **List** to display eligible parameters. In the disk descriptor list in Figure 18 we have specified that a new IBM Virtual Shared Disk be created in failure group 1 of our **fs1** from “Planning Sample IBM Virtual Shared Disks” on page 20. No metadata is to be written to hdisk9. Since this is not a twin-tailed disk, we have not named a backup server. We specified that GPFS verify hdisk9 does not belong to any existing file system before adding it to **fs1**.

Add Disks Using mmadddisk

Specify the file system and descriptor data for the disks or IBM Virtual Shared Disks you want to add with the **mmadddisk** command. For example, to create a IBM Virtual Shared Disk for new disk hdisk9, and add it to sample failure group 1 in file system **fs1** from “Planning a Sample GPFS File System” on page 19 by entering disk descriptor parameters on the command line, we would enter:

```
mmadddisk fs1 "hdisk9:sp1sw04::dataOnly:1"
```

Wherever two consecutive colons appear, a positional parameter is either omitted or allowed to default. In this example, the two consecutive colons indicate that no backup server is specified, since hdisk9 is not a twin-tailed disk.

Refer to “mmadddisk” on page 74 for usage syntax and information. “IBM Virtual Shared Disks” on page 12 contains a more detailed explanation of the parameters required to define disks for GPFS.

Deleting Disks from a File System

If you detect a faulty disk, or decide to move disk resources out of the GPFS file system, you can delete a disk from a previously defined file system using either SMIT or the **mmdeldisk** command.

Before Deleting a Disk

Use the **mmdf** command to determine whether there is enough free space on the remaining disks to store the file system (see “Querying File System Space” on page 49). Consider how fragmentation may increase your storage requirements, especially when the file system contains a large number of small files. A margin of 20 percent of the file system size should be sufficient to allow for fragmentation when small files predominate.

If you replicate your file system data you should suspend the disk you intend to delete and then rereplicate the file system using the **mmrestripefs -r** command. This ensures that all data will still exist with correct replication after the disk is deleted. The **mmdeldisk** command only migrates data that would otherwise be lost, not data that will be left in a single copy.

You cannot delete a disk in a stopped state. Start any stopped disk before attempting to delete it from the file system.

In order to delete gpfs6n6 , a 4GB disk, from our sample file system, **fs1**, which contains user home directories with small files, we would first have to assure that the other disks in **fs1** contained a total of 5GB of free space.

Delete Disks Using SMIT

1. Select `Delete Disk(s) from a File System` from the SMIT GPFS Administration Menu (Figure 8 on page 30), or enter the command:

   ```
   smit mmdeldisk
   ```

   The Select File System dialog appears in which you enter the file system device name, or click on **List**, and select the device from the names displayed. Click on **OK**. The Delete Disk(s) dialog is displayed.

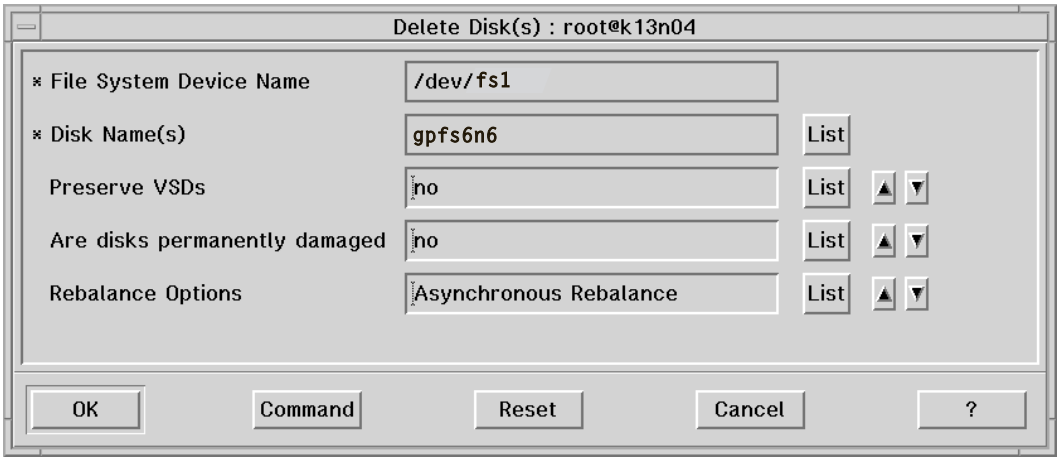


Figure 19. Delete Disk(s) Dialog

To delete the IBM Virtual Shared Disk created from hdisk6 in our sample **fs1** file system, we would select `gpfs6n6` and specify the rebalance option.

Delete Disks Using mmdeldisk

Specify the file system and the names of one or more IBM Virtual Shared Disks to delete, along with any instructions about how to handle the disk data, with the **mmdeldisk** command. For example, to delete the IBM Virtual Shared Disk named gpfs6n6 from file system **fs1** and rebalance the files across the remaining disks, we would enter:

```
mmdeldisk fs1 gpfs6n6 -r
```

Refer to “mmdeldisk” on page 105 for syntax and usage information.

Replacing Disks in a GPFS File System

You can replace an existing disk in a GPFS file system with a new one using either SMIT or the **mmrpldisk** command. This is the same as performing a delete disk operation followed by an add disk, but this operation eliminates the need to restripe the file system following two separate delete disk and add disk operations.

Replacing Disks Using SMIT

1. Select `Replace Disk in a File System` from the SMIT GPFS Administration Menu (Figure 8 on page 30) , or enter the command:

   ```
   smit mmrpldisk
   ```

   The Select File System dialog appears in which you enter the file system device name, or click on **List**, and select the file system from the device names displayed.

   To replace hdisk6 from our sample **fs1** file system, we would select `/dev/fs1` and click on **OK**. The Replace Disk dialog appears next.

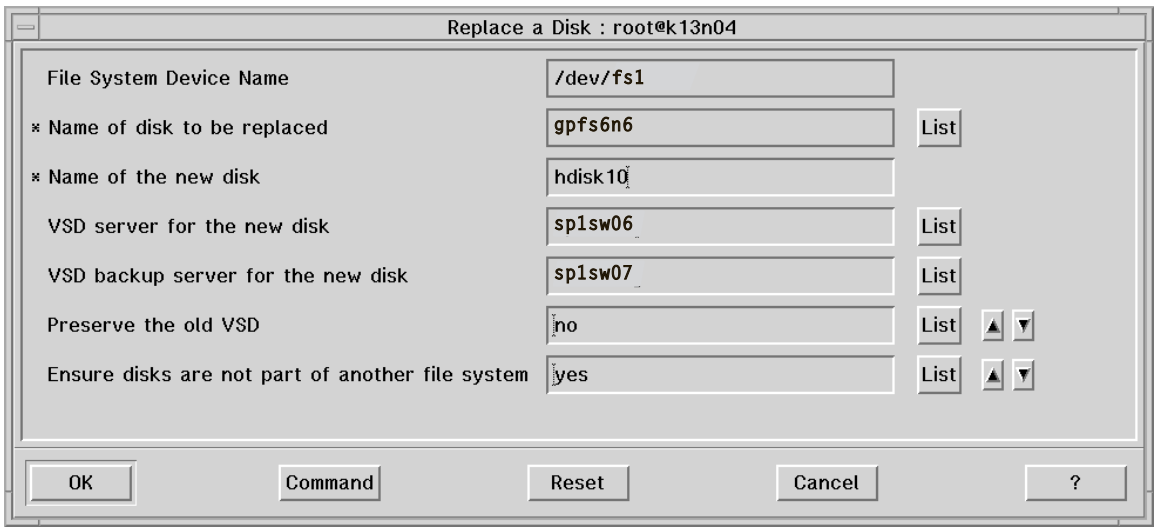


Figure 20. Replace Disk Dialog

2. Click on **List** and select the disk to be replaced. Enter the name of the new disk and the names of the primary and backup VSD servers. Specify whether the old disk should be preserved and the new disk should be verified as not yet belonging to a file system. Click on **OK**.

   To replace hdisk6 in sample file system **fs1**, we would click on **List** and select `gpfs6n6` from the IBM Virtual Shared Disk names displayed. Then we would enter the name of the replacement disk, hdisk10, along with the rest of the information in Figure 20. A new IBM Virtual Shared Disk would be created with sp1sw06 (switch node 6) as server and sp1sw07 (switch node 7) as backup.

Replacing Disks Using mmrpldisk

Specify the file system and old disk name, along with a set of disk description information for the new disk, with the **mmrpldisk** command. For example, to replace the **gpfs6n6** disk in file system **fs1** with a new IBM Virtual Shared Disk created from hdisk10, and specify that it have switch node 6 as the server and switch node 7 as the backup server, we would enter:

```
mmrpldisk fs1 gpfs6n6 hdisk10:sp1sw06:sp1sw07
```

Refer to “mmrpldisk” on page 141 for complete syntax and disk descriptor usage information.

Displaying GPFS Disk States

You can display the current state of one or more disks in your file system using either SMIT or the **mmlsdisk** command. This information includes parameters that were specified when the file system was created (see “Creating a File System” on page 35 for descriptions) and the current availability and status of the disks. Regardless of the method used, disk status will be display in the following format:

```
disk         driver     sector     failure      holds        holds       status     availability
name         type       size       group        metadata     data
hdisk1       disk       512        -1           yes          yes         ready      up
```

Availability

A disk's availability determines whether GPFS is able to read and write the disk. There are three possible values for availability:

up Disk is available to GPFS for normal read and write operations

down No read and write operations can be performed on this disk

recovering
An intermediate state for disks coming up, during which GPFS verifies and corrects data. Write operations can be performed while a disk is in this state but read operations cannot.

unrecovered
The disks was not successfully brought up.

Disk availability is automatically changed from `up` to `down` when GPFS detects repeated I/O errors. You can also make this change by issuing an **mmchdisk stop** command.

You can change a disk's availability from `down` to `recovering` by issuing the **mmchdisk start** command. When the **mmchdisk start** operation completes successfully on all disks, the disk availability automatically changes from `recovering` to `up`.

When disk availability remains `recovering` after **mmchdisk** has been issued, only partial recovery was achieved. It may be that another disk has an availability of `down` or an I/O error has occurred. Repair all disks and paths to disks before reissuing **mmchdisk** and specifying all `down` disks. If this is unsuccessful and some disks remain in `recovering`, run **mmfsck** to repair the data structure on those disks.

Status

Disk status controls data placement and migration. Status changes as a result of a pending delete operation, or when the **mmchdisk** is issued to allow file rebalancing or rereplicating prior to disk replacement or deletion.

Disk status has five possible values, but three are transitional:

ready Normal status

suspended Indicates that data is to be migrated off this disk

being emptied Transitional status in effect while a disk deletion is pending

replacing Transitional status in effect for old disk while replacement is pending

replacement Transitional status in effect for new disk while replacement is pending

GPFS allocates space only on disks with a status of `ready` or `replacement`. No space is allocated on disks being deleted or replaced, or on disks with status of `suspended`.

GPFS moves data off (migrates) disks that are being deleted, replaced, or disks with status of `suspended`, onto disks with a status of `ready` or `replacement`. During disk deletion or replacement, data is automatically migrated as part of the operation. The **mmrestripefs** command must be issued to initiate data migration from a suspended disk.

See “Deleting Disks from a File System” on page 51, “Replacing Disks in a GPFS File System” on page 52, and “Restriping a GPFS File System” on page 48.

Display Disk States Using SMIT

1. Select the `Display Status of Disks in a File System` dialog from the SMIT GPFS Administration Menu (Figure 8 on page 30), or enter the command:

   ```
   smit mmlsdisk
   ```

   The following SMIT dialog is displayed:

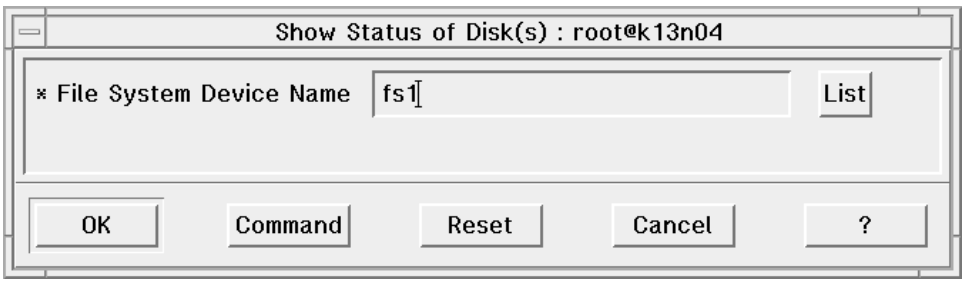


Figure 21. Display Status of Disk(s) Dialog

   Enter the name of the files system to which the disks belong, or click on **List** and select the file system. Figure 21 shows what we would enter to display status of all disks in **fs1** from “Planning Sample IBM Virtual Shared Disks” on page 20. Click on **OK**.

Display Disk States Using mmlsdisk

Specify the file system and name of one or more disks with the **mmlsdisk** command. For example, to display the current status of IBM Virtual Shared Disks gpfs5n6 and gpfs6n6 in our **fs1** file system from “Planning Sample IBM Virtual Shared Disks” on page 20, enter:

```
mmlsdisk fs1 -d "gpfs5n6;gpfs6n6"
```

Refer to “mmlsdisk” on page 125 for syntax and usage information.

Changing GPFS Disk States and Parameters

You might find it necessary to change a disk's state if there is some indication of disk failure, or if you need to restripe the file system. Refer to “Displaying GPFS Disk States” on page 53 for a detailed description of disk states. You can change both the availability and status of a disk using the **mmchdisk** command:

- Change disk availability using the **mmchdisk** command and the **stop** and **start** options
- Change disk status using the **mmchdisk** command and the **suspend** and **resume** options. Suspend a disk to allow data to be migrated off prior to disk deletion or replacement. Resume a disk only when you've suspended it and decided not to delete or replace it.

Issue **mmchdisk** with one of the following four options to change disk state:

suspend Instructs GPFS to stop allocating space on the specified disk. Place a disk in this state when you are preparing to restripe the file system off this disk because of faulty performance. This is a user-initiated state that GPFS will never use without an explicit command to change disk state.

Note: A disk remains suspended until it is explicitly resumed. Restarting GPFS or rebooting nodes does not restore normal access to a suspended disk.

resume Informs GPFS that a disk previously suspended is now available for allocating new space. If the disk is currently in a stopped state, it remains stopped until you specify the **start** option. Otherwise, normal read and write access to the disk resumes.

stop Instructs GPFS to stop any attempts to access the specified disk. Use this option to inform GPFS that a disk has failed or is currently inaccessible because of maintenance. A disk remains stopped until it is explicitly started with the **start** option.

start Informs GPFS that a disk previously stopped is now accessible. GPFS does this by first changing the disk availability from `down` to `recovering`. The file system metadata is then scanned and any missing updates (replicated data that was changed while the disk was down) are repaired. If this operation is successful, the availability is then changed to `up`. If the metadata scan fails, availability is set to `unrecovered`. This could occur if other disks are down. The metadata scan can be reinitiated at a later time by issuing the **mmchdisk start** command again.

You can also use **mmchdisk** with the **change** option to change the *Disk Usage* and *Failure Group* parameters for one or more disks in a GPFS file system. This can be useful in situations where, for example, a file system that contains only RAID disks is being upgraded to add conventional disks that are better suited to storing metadata. After adding the disks using **mmadddisk**, the metadata currently stored on the RAID disks would have to be moved to the new disks to achieve the desired performance improvement. To accomplish this, first the **mmchdisk** change command would be issued to change the *Disk Usage* parameter for the RAID disks to **dataOnly**. Then the **mmrpldisk** command would be used to restripe the metadata off the RAID device and onto the conventional disks.

Change Disks Using mmchdisk

Specify the file system, the new state (option), and the disk name, using the **mmchdisk** command. For example, to suspend the gpfs1n4 IBM Virtual Shared Disk in our sample **fs1** file system (see “Planning Sample IBM Virtual Shared Disks” on page 20), we would enter this command:

```
mmchdisk fs1 suspend -d gpfs1n4
```

To change a disk's parameters, specify the **change** option with a disk descriptor containing the new parameter value. For example, to specify that metadata should no longer be stored on disk gpfs3n4 , enter:

```
mmchdisk /dev/fs0 change &minus.d "gpfs3n4:::dataOnly"
```

For complete usage and syntax information, refer to “mmchdisk” on page 86

Managing Nodes

If your processor requirements change, you can add or delete nodes in an existing GPFS configuration. These tasks can be performed using SMIT dialogues or GPFS commands.

Adding Nodes to the GPFS Configuration

You can add nodes to an existing GPFS configuration using either SMIT or the **mmaddnode** command.

Whenever you add nodes to the GPFS configuration, you must consider how this change affects the quorum. The number of original GPFS nodes that are *down* (not operational), added to the number of nodes being added, should not be equal or greater than the new quorum requirement. The safest way to add nodes is in small increments. The following examples illustrate the consequences of adding nodes to an existing configuration.

Adding Nodes - Example 1 (Successful)

GPFS is currently configured on eight nodes, all of which are up and running. Five nodes are required for a quorum. If two nodes are added for a total of ten nodes, the quorum changes to six. Since eight nodes are already running, the quorum requirement is met. All that remains to be done is to stop and restart GPFS on all existing nodes in order for the change to be propagated across the GPFS configuration. This can be done gradually, as the workload allows.

Adding Nodes - Example 2 (Successful)

GPFS is currently configured on eight nodes, five of which are up and running. Five nodes are required for a quorum. If two nodes are added for a total of ten nodes, the quorum changes to six. With only five nodes currently running, the quorum requirement is not met. The five running nodes automatically stop and restart the GPFS daemons to load the new node information. Then they enter and remain in a wait state until GPFS is started on another node, whether old or new, in order to meet quorum. The first time GPFS is started on the new nodes, it must be done manually. Until quorum is met, no file system operations can be performed.

Adding Nodes - Example 3 (Unsuccessful)

GPFS is currently configured on eight nodes, five of which are up and running. Five nodes are required for a quorum. If four nodes are added, the new total is twelve nodes and the quorum changes to seven. A problem can occur if the network partitions before the quorum changes, and if the original five nodes where GPFS is running are in one partition and the remaining seven nodes are in the other partition. You will then be able to start GPFS on the seven nodes, but the two partitions will not be able to communicate and file system corruption can result.

You can avoid this situation by assuring that the total of new nodes, when added to existing nodes where GPFS is not running, does not satisfy the new quorum. Alternatively, simply stop GPFS on all configured nodes before adding nodes.

Note: When you add nodes, the original nodes in the configuration recognize new ones only after a GPFS restart. Until then, they do not allow the new nodes to become part of the configuration.

Adding Nodes Using SMIT

1. Select `Add Node(s) to a File System` from the SMIT GPFS Administration Menu (Figure 8 on page 30), or enter the command:

   ```
   smit mmaddnode
   ```

   The Add New Node(s) dialog appears, in which you can enter the node name, or click on **List**, and select the node from the names displayed. Then click on **OK**.

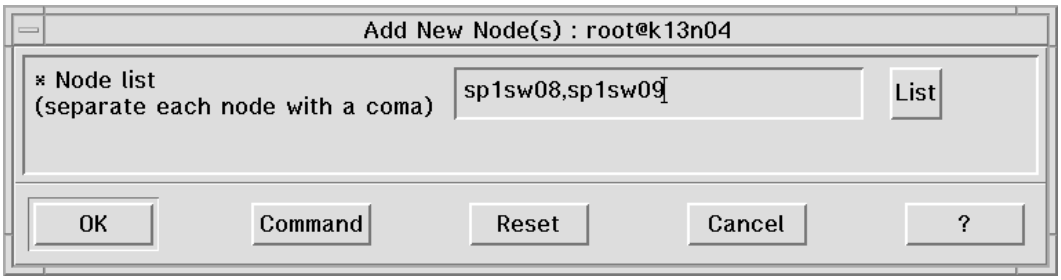


Figure 22. Add New Node(s) Dialog

Figure 22 shows the switch hostnames we would enter to add node8 and node9 to our sample **fs1** file system from "Planning a Sample GPFS Configuration" on page 18. Since there are eight nodes in the original configuration, we can add both new nodes at once without encountering a quorum problem. If, however, we added another eight nodes all at once, we would not be able to mount the file system because the quorum would immediately require that nine nodes be up and running in PSSP Group Services. In such a case, we would add the new nodes four at a time, provided at least seven of the original nodes were up and running GPFS.

Adding Nodes Using mmaddnode

Specify one or more nodes to add to your configuration with the **mmaddnode** command. For example, to add switch nodes 8 and 9 to the GPFS **fs1** file system, we would specify the following switch hostnames:

```
mmaddnode sp1sw08,sp1sw09
```

Refer to "mmaddnode" on page 78 for usage information.

Deleting Nodes from the GPFS Configuration

You can delete nodes from an existing GPFS configuration using either SMIT or the **mmdelnode** command.

Notes:

1. Before you can delete a node, you must unmount the file system and stop GPFS on the node to be deleted.
2. Exercise caution when deleting nodes from the GPFS configuration. If the number of remaining nodes falls below the requirement for a quorum, you will be unable to perform file system operations. See "Quorums" on page 6 for more information.
3. When a node is deleted from the GPFS configuration, its entry is not automatically deleted from the **/etc/cluster.nodes** file. Instead, a deleted hostname is replaced with `localhost` in the **/etc/cluster.nodes** file. This

`localhost` entry is not a factor in calculating quorum. If you want to compress the /**etc**/**cluster.nodes** file by removing the entry altogether, specify the **-c** option with the **mmdelnode** command.

Deleting Nodes Using SMIT

1. Select `Delete Node(s) from a File System` from the SMIT GPFS Administration Menu (Figure 8 on page 30). The Delete Node(s) dialog appears, in which you can enter the node name, or click on **List**, and select the node from the names displayed. Then click on **OK**.

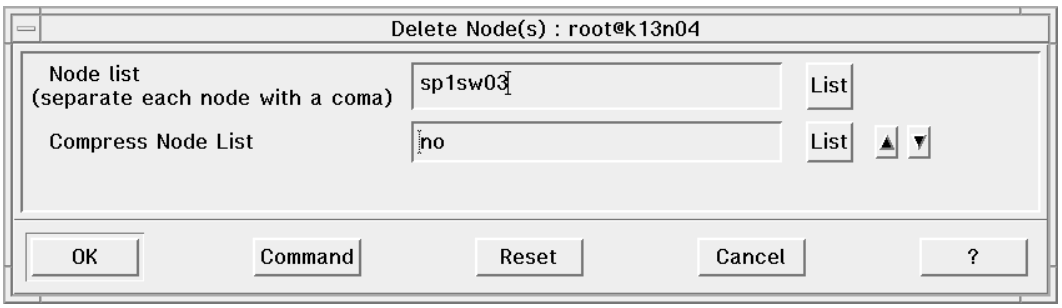


Figure 23. Delete Node(s) Dialog

Figure 23 shows the switch hostname we would enter to delete switch node 3 from our sample **fs1** file system from “Planning a Sample GPFS Configuration” on page 18.

Deleting Nodes Using mmdelnode

Specify one or more nodes to be deleted from the configuration with the **mmdelnode** command.

For example, to delete node03, we would specify the hostname used during configuration:

```
mmdelnode -c sp1sw03
```

The **-c** option tells GPFS to compress the /**etc**/**cluster.nodes** file by removing any `localhost` entries.

Refer to “mmdelnode” on page 110 for usage information.

Managing GPFS Quotas

The GPFS quota system helps you to control the allocation of files and data blocks in a file system. GPFS quotas can be defined for individual users or groups of users. Quota related tasks include:

1. “Installing and Uninstalling GPFS Quotas” on page 60
2. “Establishing and Changing Quotas” on page 61
3. “Checking Quotas” on page 62
4. “Listing Quotas” on page 62
5. “Activating Quota Enforcement” on page 63
6. “Deactivating Quota Enforcement” on page 64
7. “Creating File System Quota Reports” on page 64

8. “Recovering from Over-Quota Conditions” on page 65

In a quota-enabled configuration, one node is automatically nominated as the quota manager whenever GPFS is started. The quota manager allocates disk blocks to the other nodes writing to the file system, and compares the allocated space to quota limits at regular intervals. In order to reduce the need for frequent space requests from nodes writing to the file system, the quota manager allocates more disk blocks than requested.

Like JFS quotas, GPFS quotas operate with three parameters that you can set using the **mmedquota** command:

1. Soft limit
2. Hard limit
3. Grace period

The soft limits define levels of disk space and files below which the user or group can safely operate. The hard limits define the maximum disk space and files the user or group can accumulate. Specify hard and soft limits for disk space in multiples of the file system sub-block size (see “Fragments and Subblocks” on page 8.)

The grace period allows the user or group to exceed the soft limit for a specified period of time (the default period is one week). If usage is not reduced to a level below the soft limit during that time, the quota manager interprets the soft limit as the hard limit and no further allocation is allowed. The user or group can reset this condition by reducing usage enough to fall below the soft limit.

The quota system maintains usage and limits data in the **quota.user** and **quota.group** files that reside in the root directories of GPFS file systems. These files are built with information input to the **mmedquota** command, and gathered during execution of the **mmcheckquota** command. **quota.user** and **quota.group** are readable by the **mmlsquota** and **mmrepquota** commands. The **mmquotaon** and **mmquotaoff** commands control activation of quota enforcement.

Quotas are typically used for file systems containing user home directories. Consider installing quotas on a GPFS file system if:

- Your system has limited disk space
- You need more file system security
- Your disk usage levels are high

If these conditions do not apply to your environment there may be no need to limit disk usage with quotas.

Installing and Uninstalling GPFS Quotas

To install quotas on your GPFS file system, follow this procedure:

1. Specify the **-Q yes** option when creating a new file system. This option automatically activates quota enforcement whenever the file system is mounted, and is also available when changing an existing file system with the **mmchfs** command (92).
2. Mount (or remount) the file system.

3. Set quota values. Refer to “Establishing and Changing Quotas” on page 61.
4. Compile i-node and disk block statistics using the **mmcheckquota** command (see “Checking Quotas” on page 62). Do this after creating a new file system, and whenever numerous node failures have occurred. This command reconciles quota information with actual disk allocation.
5. Activate quota enforcement. You must do this manually using the **mmquotaon** command if the file system has not yet been remounted, even when the **-Q yes** option (step 1 on page 60) has been specified.

During normal operation, there is no need to deactivate quota enforcement (“Deactivating Quota Enforcement” on page 64) and doing so can affect quota accuracy. The only reason you might have to deactivate quota enforcement is when users are denied allocation that their quotas should allow, due to loss of quota information during node failure. If this occurs, use the **mmcheckquota** command after reactivating quotas to reconcile allocation data.

You may find it helpful to maintain a *quota prototype*, a set of limits that you can apply, by name, to any user or group, without entering the values manually. This makes it easy to set the same limits for all. The **mmedquota** command includes an option for naming a prototypical user or group upon which limits are to be based.

To uninstall quotas, edit the **/etc/filesystems** file to delete the following from the options line in the appropriate file system stanza on every node in the GPFS configuration:

```
quota=userquota;groupquota
```

Establishing and Changing Quotas

Use the **mmedquota** command to create file system quotas for individual users or groups of users. You can also change a quota once it has been created.

Establish/Change Quotas Using mmedquota

The **mmedquota** command allows you to establish or change quotas for one or more users or user groups. It opens a session using your default editor, and prompts you for soft and hard limits for i-nodes and blocks.

For example, to set user quotas for user `jesmith`, enter:

```
mmedquota -u jesmith
```

A prompt similar to the following appears in your default editor:

```
MMFS Edit quota limits for USR jesmith:
sg2: blocks in use (in KB): 24576, limits (soft = 0, hard = 0)
inodes in use: 3, limits (soft = 0, hard = 0)
```

Current (in use) i-node and block usage is for display only; it cannot be changed. When establishing a new quota, zeros appear as limits. Replace the zeros, or old values if you are changing existing limits, with values based on the user's needs and the resources available. When you close the editor, GPFS checks the values and applies them. If an erroneous value was specified, GPFS generates an error message. If this occurs, re-enter the **mmedquota** command.

To set group quotas, for example, for all users in a group named `blueteam`, to the prototypical values established for `prototeam`, specify the prototype option as follows:

```
mmedquota -g -p prototeam blueteam
```

When using the **-p** option, only a group prototype can be specified for group quotas, and only a user prototype can be specified for user quotas.

Refer to "mmedquota" on page 116 for syntax and usage information.

Checking Quotas

The **mmcheckquota** command counts i-node and space usage for a file system and writes the collected data into quota files.

Before activating quotas for the first time, you must use the **mmcheckquota** command to reconcile quota limits with current allocations. This must also be done in the event that quota information is lost due to node failure leaving users unable to open files or denying them disk space that their quotas should allow.

Check Quotas Using mmcheckquota

Specify the file system name with the **mmcheckquota** command. For example, to check quotas for file system **fs1** and report differences between calculated and recorded disk quotas, enter:

```
mmcheckquota -v fs1
```

Output similar to the following is displayed:

```
fs1: quota-check found following differences:
USR 0: 96 subblocks counted (was 0); 3 inodes counted (was 0)
```

This return says that the quota information for user 0 was corrected. Due to a system failure, this information was lost at the server, which recorded 0 sub-blocks and 0 files. **mmcheckquota** counted the current usage data, 96 subblocks and 3 files, and used it to update the quota.

Refer to "mmcheckquota" on page 90 for usage information.

Listing Quotas

Display file system quota limits and current usage information using the **mmlsquota** command.

List Quotas Using mmlsquota

Specify one user or user group with the **mmlsquota** command. For example, to display quota information about user `paul`, enter:

```
mmlsquota -u paul
```

Information similar to the following is returned:

```
                  Block Limits                                File Limits
Name type    KB       quota    limit in_doubt    grace    files   quota    limit in_doubt    grace
paul USR     16384    17920    19968 16384       none     2       45       50    3           none
```

This output shows the quotas for user `paul` set to a soft limit of 17920KB and a hard limit of 19968KB. 16384KB is currently allocated to him. 16384KB is also *in doubt*, meaning that the quota manager allocated this space to other nodes writing the file system, but has not yet received updated information accounting for it. The quota manager, therefore, cannot yet report whether the space has been used by

the nodes or is still available to them. This value may also represent a file that was open for a write operation when the quota manager was last updated.

If you issue the **mmlsquota** command with the **-e** option, the quota system collects updated information from all nodes before returning output. If the node to which *in-doubt* space was allocated should fail before updating the quota system about its actual usage, this space might be lost. Should the amount of space in doubt approach a significant percentage of the quota, run the **mmcheckquota** command to account for the lost space.

The soft limit for files (i-nodes) is set at 45 and the hard limit is 50. Two files are currently allocated to this user, and the three in doubt have been allocated to the nodes but the quota manager has not yet been updated as to whether they have been used or are still available.

No grace period appears because the user has not exceeded his quotas. Had he exceeded 17920KB or 45 files, the grace period, either explicitly set or allowed to default to one week, would be in effect and the user would have that time period to bring his usage below the soft limits. If he failed to do so, he would not be allocated any more space.

To collect and display updated quota information about a group named blueteam, specify the **-g** and **-e** options:

```
mmlsquota -g blueteam -e
```

Information is returned for groups and users in the same format.

Refer to “mmlsquota” on page 129 for syntax and usage information.

Activating Quota Enforcement

You can activate quota limit checking for users, groups, or both, using the **mmquotaon** command.

Note: Before activating quotas for the first time, you must run the **mmcheckquota** command to check file system quota consistency.

You can have quotas activated automatically whenever the file system is mounted by specifying the quota option when creating or changing a GPFS file system.

Activate Quotas Using mmquotaon

Specify the file system name, and whether user quotas or group quotas are to be activated, with the **mmquotaon** command. If you want both user and group quotas activated specify only the file system name.

For example, to activate user quotas on file system **fs1,** enter:

```
mmquotaon -u /gpfs/fs1
```

To activate group quotas on all GPFS file systems, enter:

```
mmquotaon -g -a
```

To activate both user and group quotas on **fs2**, enter:

```
mmquotaon fs2
```

Refer to “mmquotaon” on page 135 for syntax and usage information.

Deactivating Quota Enforcement

You can deactivate quota limit checking for users, groups, or both, using the **mmquotaoff** command. There is no need to deactivate quotas under normal conditions. Disk space and file allocations are made without regard to limits when quota enforcement is deactivated.

Deactivate Quotas Using mmquotaoff

Specify the file system name, and whether user or group quotas are to be deactivated, with the **mmquotaoff** command. If you want both types of quotas deactivated, specify only the file system name.

For example, to deactivate user quotas on file system **fs1,** enter:

```
mmquotaoff -u /gpfs/fs1
```

To deactivate group quotas on all GPFS file systems, enter:

```
mmquotaoff -g -a
```

To deactivate all quotas on **fs2,** enter:

```
mmquotaoff fs2
```

Refer to “mmquotaoff” on page 133 for syntax and usage information.

Creating File System Quota Reports

You can have GPFS prepare a quota report for a file system using the **mmrepquota** command. This lists:

1. Number of existing files
2. Amount of disk space used
3. Current quota limits
4. In doubt quotas (disk space allocated but currently unaccounted for)

for all users or all groups. If no options are specified, the report lists all users and all groups.

Create Quota Reports Using mmrepquota

Specify whether you want to list only user quota information or group quota information, along with the file system name, in the **mmrepquota** command. The default is to summarize both user quotas and group quotas.

To report on the group quotas for all file systems, enter:

```
mmrepquota -g -a
```

To report on user quotas for file system **fs1,** enter:

```
mmrepquota -u fs1
```

Refer to “mmrepquota” on page 137 for syntax and usage information.

Recovering from Over-Quota Conditions

Several methods of reducing file system usage are available to a user or group that has exceeded quota limits:

1. Abort the current process that caused the file system to reach its limit. Then remove surplus files to bring the limit below quota, and retry the failed program.
2. If an editor such as **vi** is in use, the shell escape key sequence can be used to check file space, remove surplus files, and return to the editor without losing the open file. The C and Korn shells allow the editor to be suspended with the **Ctrl-Z** key sequence so the file system commands can be issued. Return to the editor using the **fg** (foreground) command.
3. Temporarily write the file to a file system where limits have not been exceeded. Delete surplus files, then return the file to the correct file system.

Managing Access Control Lists

Access control protects directories and files by providing a means of specifying who should be granted access. GPFS access control lists (ACLs) give you added control over GPFS file access by allowing you to set permissions that extend the base AIX permissions assigned to individual users and groups.

The following administrative tasks are associated with GPFS ACLs:

1. “Setting Access Control Lists” on page 66
2. “Displaying Access Control Lists” on page 67
3. “Changing Access Control Lists” on page 67
4. “Deleting Access Control Lists” on page 68

GPFS ACLs extend the base AIX permissions, or traditional file access modes, of read (r), write (w), and execute (x) beyond the three categories of file owner, file group, and other users, to allow definition of additional users and user groups. In this way, an ACL can be created that might look like this:

```
#owner:jesmith
#group:team_A
user::rwx
group::rwx
other::--x
mask::rwx
user:alpha:r-x
group:audit:r-x
group:system:rwx
```

In the ACL above,

- The first two lines are comments showing the file's owner, jesmith, and group name, team_A
- The next three lines contain the base AIX permissions for the file:
  1. The permissions set for the file owner (**user**), jesmith
  2. The permissions set for the owner's **group**, team_A

3. The permissions set for **other** groups or users outside the owner's group and not belonging to any named entry

These three entries are the minimum necessary for a GPFS ACL.

- The next line, with an entry type of **mask**, contains the maximum permissions allowed for any entries other than the owner (the **user** entry) and those covered by **other** in the ACL.
- The last 3 lines contain additional entries for specific users and groups. These permissions are limited by those specified in the mask entry, but you may specify any number of additional entries up to a memory page (approximately 4KB) in size.

GPFS ACL s are fully compatible with the AIX permission set. Any change to the AIX permissions, using the **chmod** command, for example, modifies the corresponding ACL as well. Similarly, any change to the GPFS ACL is reflected in the output of AIX commands such as **ls -l**.

Each GPFS file or directory has an *access ACL* that determines its access privileges. These ACLs control who is allowed to read or write at the file or directory level.

In addition to an *access ACL*, a directory may also have a *default ACL*, which is assigned as an access ACL to every file created in that directory. This allows a user to protect all files in a directory without explicitly setting an ACL for each one. When a new subdirectory is created, both its access ACL and its default ACL are set to the default ACL of its parent directory.

Setting Access Control Lists

Use the **mmputacl** or **mmeditacl** command to set the access ACL of a file or subdirectory, or the default ACL of a directory. For example, to set the ACL for a file named **project2.history**, we might create a file named **project2.acl** that contains the following:

```
user::rwx
group::rwx
other::--x
mask::rwx
user:alpha:r-x
group:audit:rw-
group:system:rwx
```

In the **project2.acl** file above,

- The first three lines are the required ACL entries setting permissions for the file's owner, the owner's group, and for processes that are not covered by any other ACL entry.
- The last 3 lines contain named entries for specific users and groups.
- Because the ACL contains named entries for specific users and groups, it must also contain a mask entry, which is applied to all named entries (entries other than the **user** and **other**).

Once you are satisfied that the correct permissions are set in the ACL file, you can apply them to the target file with the **mmputacl** command:

```
mmputacl -i project2.acl project2.history
```

Although you can issue the **mmputacl** without using the **-i** option to specify an ACL input file, and make ACL entries through standard input instead, you will probably find the **-i** option more useful for avoiding errors when creating a new ACL.

See “mmputacl” on page 131 for syntax and usage information.

Displaying Access Control Lists

Use the **mmgetacl** command to display the access ACL of a file or subdirectory, or the default ACL of a directory. For example, to display the ACL created in the previous example for a file named **project2.history** , enter

```
mmgetacl project2.history
```

A display similar to the following is sent to standard output:

```
#owner:guest
#group:usr
user::rwx
group::rwx #effective:rw-
other::--x
mask::rw-
user:alpha:rwx #effective:rw-
group:audit:rwx #effective:rw-
group:system:-w-
```

The first two lines are comments displayed by the **mmgetacl** command, showing the owner and owning group. All entries containing permissions that are not allowed (because they are not set in the mask entry) display with a comment showing their effective permissions.

You can use the **-o** option to specify a file to which the ACL is written, for example,

```
mmgetacl -o old.acl project2.history
```

Then, if you want to assign the same permissions to another file, **project.notes**, for example, you could specify,

```
mmputacl -i old.acl project.notes
```

See “mmgetacl” on page 122 and “mmputacl” on page 131 for syntax and usage information.

Changing Access Control Lists

Use the **mmeditacl** command to change or create the access ACL of a file or directory, or the default ACL of a directory. For example, to edit the ACL for a file named **project2.history** interactively, enter

```
mmeditacl project2.history
```

The current ACL entries are displayed using the default editor, provided that the EDITOR environment variable specifies a complete path name.

See “mmeditacl” on page 114 for syntax and usage information.

Deleting Access Control Lists

Use the **mmdelacl** command to delete the extended entries in an access ACL of a file or directory, or the default ACL of a directory. For example, to delete the default ACL for a directory named **project2**, enter

```
mmdelacl -d project2
```

To delete the ACL for a file named **project2.history**, enter

```
mmdelacl project2.history
```

You cannot delete the base AIX permissions. These remain in effect after this command is executed. See “mmdelacl” on page 104 for syntax and usage information.

Chapter 5. Tuning GPFS Performance

GPFS performance depends on the correct specification of its parameters as well as the correct tuning of the functions that it uses. The following information describes the parameters that affect performance.

GPFS Tuning Parameters

For most work loads, the parameters that have the greatest impact on performance are the file system block size and the amount of memory allocated to GPFS. There are other parameters that affect specific work loads.

File System Block Size

GPFS offers three block sizes for file systems: 16KB, 64KB, and 256KB. Once you have set this parameter at file system creation, you cannot change it without recreating the file system.

The 256KB block size is the best choice for file systems that contain large files accessed in large reads and writes. This block size allows the most effective I/O operations.

The 16KB block size optimizes use of disk storage at the expense of large data transfers and should be used for other types of applications.

The 64KB block size offers a compromise. It makes more efficient use of disk space than 256KB while allowing faster I/O operations than 16KB.

You should choose the block size based on the application set that you plan to support.

GPFS Memory Parameters

GPFS pins an area of memory for its exclusive use. The memory allocated to GPFS is separated into two pools: the page pool and malloc pool. The size of these areas is controlled by GPFS configuration parameters.

Page Pool

The page pool is used primarily for storage of user data and indirect blocks. Increasing the size of the page pool allows GPFS to cache more data for access by applications. This provides two benefits. It reduces I/O operations by reusing previously fetched data, and it also gives GPFS the ability to overlap I/O and application execution through prefetch and deferred writes. If the applications you run have operational characteristics that would benefit from these features, an increase in page pool size might help performance. Examples of such characteristics are:

- Heavy use of writes that can be overlapped with application execution
- Heavy reuse of files and sequential reads of a size such that prefetch will benefit the application

© Copyright IBM Corp. 1998

Malloc Pool

The malloc pool is used for control structures. Applications that reuse files will benefit from increased malloc size because metadata can be cached.

Caching i–nodes

Storing a recently used file's i–node in the malloc pool permits faster access to the file the next time it must be opened. The **maxFilesToCache** parameter controls how many i–nodes are stored in cache. The default is 200 but increasing this number may improve throughput for workloads with high file reuse. Increasing it where file reuse is not great will waste kernel heap storage. Do not increase this parameter if you are in doubt.

Each cached file requires space in the mallocpool for an i–node plus approximately 800 bytes of metadata. GPFS will not use more than 50% of the mallocpool for this purpose. If you increase **maxFilesToCache**, you should also increase the **mallocsize** parameter to a value equal to 2 x (maximum i–node size for the file system + 800) x **maxFilesToCache**. For example, to set **maxFilesToCache** to 1000 for a file system with a maximum i–node size of 4096, **mallocsize** should be approximately 10MB.

Additional Considerations

There are several other parameters that are set during GPFS configuration. In most environments, the default values result in efficient performance. Some environments, however, can benefit from adjustment.

Priority

Much of the GPFS code runs as a multi-threaded daemon. By default, the priority of the daemon is 4. You may wish to increase the priority of the daemon if your users execute a large number of time–critical I/O operations.

Token Management

The token management parameters control simultaneous file access. Do not change these settings unless you are familiar with AIX kernel and thread concepts.

The number of client ports and server processes control the number of token operations that can be performed concurrently. The default settings should be adequate for workloads that are not very demanding on locks. If your applications are doing large amounts of very fine-grained *write sharing* of files, you might consider increasing these values.

client_ports

This is the number of communication ports that are reserved for requesting file access tokens.

server_kprocs

This is the number of kernel processes that are dedicated to processing file access token requests. These processes await token requests, and their number determines how efficiently multiple parallel processes can access a single file.

server_port_number

Two consecutive UDP ports are defined starting at this port number. The token manager receives token requests through these ports. The value of this parameter must be the same for all nodes in the GPFS configuration.

The number of these processes begins with the value specified as **server_port_number** and continues for the value specified as **server_kprocs**. **server_port_number** must be the same for all nodes in the GPFS configuration.

Switch Parameters and GPFS Performance

Two communication subsystem parameters affect GPFS performance: **rpoolsize** and **spoolsize**. Insufficient memory allocation to these switch pools can result in major performance degradation. Unless your nodes are very short of memory, we recommend that both of these pools be set at 16MB (the maximum allowed). See *Managing Shared Disks* for more information.

IBM Virtual Shared Disk Parameters and GPFS Performance

GPFS performance and recovery processes function best with one disk per IBM Virtual Shared Disk. If you want to create IBM Virtual Shared Disks with more than one disk, refer to *Managing Shared Disks*.

The IBM Virtual Shared Disk uses *buddy buffers* when it cannot hold sufficient switch buffers for the duration of an I/O operation. The number and size of these buffers is a function of the block sizes of the file systems and the number of disks. The optimal number of buddy buffers is different for IBM Virtual Shared Disk server nodes and IBM Virtual Shared Disk client nodes.

For your initial configuration, define buddy buffers at the largest block size of any GPFS file system you will create. If the block size of the largest file system is 16KB, define only 1 buddy buffer on IBM Virtual Shared Disk client nodes, but allow three or four on IBM Virtual Shared Disk servers because the IBM Virtual Shared Disk can use small buffers borrowed from the communication subsystem sendpool for blocks of this size.

You may find that you need more buddy buffers if you have a large number of disks attached to one IBM Virtual Shared Disk server. You can determine whether this is so by running the **statvsd** command and monitoring retries.

If the block size of any GPFS file system will be greater than 16KB, buddy buffers will be used for all I/O operations on the IBM Virtual Shared Disk servers. Only one buddy buffer may be needed on client nodes, but the number of buddy buffers required on a IBM Virtual Shared Disk server will be a function of the expected number of disk I/Os outstanding at any time. Start with an initial configuration of two buffers per spindle attached to a server. Again, monitoring the output of **statvsd** for buddy buffer retries will give you an indication of whether this initial value is too large or too small.

See *Managing Shared Disks* for information on buddy buffers.

Chapter 6. GPFS Commands

© Copyright IBM Corp. 1998

mmadddisk

Purpose

Makes specified disk(s) available for GPFS use.

Syntax

mmadddisk *Device* {*DiskDescList* | **-F** *DescFile*} [**-a**] [**-r**] [**-v** **yes** | **no**]

Operands

Device
: The name of the file system device where the disk(s) are to be added. File system names need not be fully qualified. **fs0** and /**dev**/**fs0**, for example, are both acceptable.

-F *DescFile*
: Points to a list of disk descriptors (see "Disk Descriptors") contained, one per line, in the specified file. This file eliminates the need for command line entry of these descriptors using the *DiskDescList* parameter. A sample file can be found in /**usr**/**lpp**/**mmfs**/**samples**/**diskdesc**.

DiskDescList
: A descriptor for each IBM Virtual Shared Disk to be added. Each descriptor (see "Disk Descriptors") is separated by a semicolon (;). The entire *DiskDescList* must be enclosed in quotation marks (' or ").

Disk Descriptors

Whether you use a *DescFile* or *DiskDescList*, the positional parameters of the disk descriptor are the same. The only difference is how the disk descriptors are delimited. When entering them on the command line using a *DiskDescList*, each descriptor is separated by a semicolon. When entering them in a *DescFile*, each descriptor is on a separate line. No more than 512 disks can be defined for a file system.

Each descriptor contains positional parameters that apply to the IBM Virtual Shared Disk, separated by colons (:).

You must provide a descriptor for each GPFS IBM Virtual Shared Disk to be created or passed to the GPFS file system. Each descriptor can contain the following positional parameters:

Disk Name
: The device name, for example, `hdisk1`, of any disk you wish to use to create an IBM Virtual Shared Disk, or the name of an existing IBM Virtual Shared Disk.

  GPFS performance and recovery processes function best with one disk per IBM Virtual Shared Disk. If you want to create IBM Virtual Shared Disks with more than one disk, refer to Managing Shared Disks. Then pass the existing IBM Virtual Shared Disks to GPFS using the VSD names.

Server Name
The server name, when creating an IBM Virtual Shared Disk. This can be in any recognizable form.

Backup Server Name
The backup server name, when creating an IBM Virtual Shared Disk. This can be in any recognizable form.

Disk Usage
Specify one of the following or allow to default:

- **dataAndMetadata** (default)
- **dataOnly**
- **metadataOnly**

Failure Group
A number identifying the failure group to which this disk belongs. All disks that have a common point of failure, such as all disks that are attached to the same VSD server node, should be placed in the same failure group. GPFS uses this information during data and metadata placement to assure that no two replicas of the same block are written in such a way as to become unavailable due to a single failure. You can specify any value from -1 (where -1 indicates that the disk has no point of failure in common with any other disk) to 4000. If you do not specify a failure group, the value defaults to the server node number plus 4000.

Any parameter that is intentionally omitted or allowed to default must be explicitly skipped. For example, if you had no backup server and wanted to use your disk for both data and metadata (the default), you could create a disk descriptor that looked like this:

```
hdisk1:sp1sw05:::5
```

If you also wanted to allow the last parameter, Failure Group, to default, you could omit the last three delimiting colons:

```
hdisk1:sp1sw05
```

or leave them in if you prefer:

```
hdisk1:sp1sw05:::
```

Trailing delimiters are acceptable.

Options

-a Specifies whether **mmadddisk** should wait for rebalancing to complete before returning. If this flag is specified, **mmadddisk** runs asynchronously and returns after the file system descriptor is updated and the rebalancing scan is started, but it does not wait for rebalancing to finish. If no rebalancing is requested (**-r** option not specified), this option has no effect.

-r Rebalance all existing files in the file system to make use of new disks.

-v yes | **no** Verify that specified disks do not belong to an existing file system. The default is **yes**. Specify **no** only when you want to reuse disks that are no longer part of an existing file system.

mmadddisk

Description

Use the **mmadddisk** command to add disk(s) and logical volume groups to a GPFS file system. The current maximum number of disks that can be defined for any single file system is 512.

This command optionally rebalances an existing file system after adding disks. **mmadddisk** does not require the file system to be unmounted; it can be run while the file system is in use.

Notes on specifying disk descriptor parameters:

Notes:

1. To use an existing IBM Virtual Shared Disk volume group in the file system, only the *Disk Name* (the IBM Virtual Shared Disk name) need be specified in the disk descriptor. The *Disk Usage* and *Failure Group* parameters are optional and may be allowed to default.
2. To create an IBM Virtual Shared Disk with one disk and use it in a file system, you must specify the *Disk Name* and *Server Name* parameters in the descriptor. *Backup Server*, *Failure Group*, and *Disk Usage* are optional.

Results

Upon successful completion of **mmadddisk** the following tasks are performed for each disk added to the GPFS file system:

- The volume group is created
- The IBM Virtual Shared Disk is configured and started
- New disks are added to the **/etc/filesystems** stanza

If **mmadddisk** is unsuccessful because a node is down, the added disk(s) will be propagated when that node comes up.

Files

- **/etc/filesystems**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have root authority to run **mmadddisk**.

Examples

To create a new IBM Virtual Shared Disk volume group for new disk, hdisk48 and add it to file system **fs0** by entering descriptor input on the command line, enter:

```
mmadddisk fs0 "hdisk48:sp1sw05:sp1sw06:dataOnly"
```

This command creates a new IBM Virtual Shared Disk with the default failure group of 4005 . The IBM Virtual Shared Disk will store only data, not metadata.

Related Information

To add disks through SMIT panels, see “Add Disks Using SMIT” on page 50.

“mmchdisk” on page 86

“mmdeldisk” on page 105

“mmlsdisk” on page 125

mmaddnode

Purpose

Adds node(s) to GPFS configuration.

Syntax

mmaddnode *Newnodes*

Operands

Newnodes

List of nodes, delineated by commas. Nodes can be specified in any of the following hostname forms:

Format	Example
Short Hostname	k145s02
Long Hostname	k145s02.dpd.ibm.com
IP Address	9.119.19.102

Regardless which form you choose, specify the switch hostname.

Description

Use the **mmaddnode** command to add nodes to an existing GPFS configuration. **mmaddnode** must be run on an existing node in the GPFS subsystem.

Whenever you add nodes to the GPFS configuration, you must consider the impact to the quorum. The number of original nodes that are down (not operational), added to the number of new nodes, should not be equal to or greater than the new quorum requirement. The safest way to add nodes is in small increments. The following examples illustrate the consequences of adding nodes to an existing configuration.

- **Adding Nodes — Example 1 (Successful)**

  GPFS is currently configured on eight nodes, all of which are up and running. Five nodes are required for a quorum. If two nodes are added for a total of ten nodes, the quorum changes to six. Since eight nodes are already running, the quorum requirement is met. All that remains to be done is to stop and restart GPFS on all existing nodes in order for the change to be propagated across the GPFS configuration. This can be done gradually, as the workload allows.

- **Adding Nodes — Example 2 (Successful)**

  GPFS is currently configured on eight nodes, five of which are up and running. Five nodes are required for a quorum. If two nodes are added for a total of ten nodes, the quorum changes to six. With only five nodes currently running, the quorum requirement is not met. The five running nodes automatically stop and restart the GPFS daemons to load the new node information. Then they enter and remain in a wait state until GPFS is started on another node, whether old or new, in order to meet quorum. The first time GPFS is started on the new nodes, it must be done manually. Until quorum is met, no file system operations can be performed.

- **Adding Nodes — Example 3 (Unsuccessful)**

GPFS is currently configured on eight nodes, five of which are up and running. Five nodes are required for a quorum. If four nodes are added, the new total is twelve nodes and the quorum changes to seven. A problem can occur if the network partitions before the quorum changes, and if the original five nodes where GPFS is running are in one partition and the remaining seven nodes are in the other partition. You will then be able to start GPFS on the seven nodes, but the two partitions will not be able to communicate and file system corruption can result.

You can avoid this situation by assuring that the total of new nodes, when added to existing nodes where GPFS is not running, does not satisfy the new quorum. Alternatively, simply stop GPFS on all configured nodes before adding nodes.

Note: When you add nodes, the original nodes in the configuration recognize new ones only after a GPFS restart. Until then, they do not allow the new nodes to become part of the configuration.

Results

Upon successful completion of **mmaddnode** the following tasks are performed for each node added to the configuration:

- Copy **/var/mmfs/etc/mmfs.cfg**, **/var/mmfs/etc/tm.cfg**, **/etc/cluster.nodes**.
- Create mount point directory and character mode device for each GPFS file system
- Update **/etc/filesystems** with stanzas for GPFS file systems

Upon successful completion of **mmaddnode** the following is performed for each pre-existing node in the configuration:

- Update **/etc/cluster.nodes** to reflect newly added nodes.
- Update quorum to reflect new requirement

If the **mmaddnode** command is unsuccessful because any existing nodes in the file system are down, the added node(s) will be propagated as each node comes up.

Files

- **/etc/filesystems**
- **/etc/cluster.nodes**
- **/var/mmfs/etc/mmfs.cfg**
- **/var/mmfs/etc/tm.cfg**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

mmaddnode must be run on an existing node in the GPFS configuration where new node(s) are to be added.

You must have root authority to run **mmaddnode**.

Examples

1. To add node4 and node5 in the k145 system to the GPFS subsystem, enter:

```
mmaddnode k145n04,k145n05
```

Related Information

To add nodes through SMIT panels, see “Adding Nodes Using SMIT” on page 58.

“mmchconfig” on page 83

“mmconfig” on page 94

“mmdelnode” on page 110

mmchattr

Purpose

Changes attributes of one or more GPFS files.

Syntax

mmchattr [**-m** *MetadataReplicas*] [**-r** *DataReplicas*] *Filename* ...

Options

-m *MetadataReplicas*
: Specifies how many copies of the file system's metadata to create. Enter a value of 1 or 2, but no larger than the value of the *MaxMetadataReplicas* attribute of the file.

-r *DataReplicas*
: Specifies how many copies of the file data to create. Enter a value of 1 or 2, but no larger than the value of the *MaxDataReplicas* attribute of the file.

Operands

Filename
: The name of one or more files to be changed. Each filename should be delineated by a space. Wildcard characters are supported in filenames, for example, project*.sched.

Description

Use **mmchattr** to change the replication attributes of files in the GPFS file system. The replication factor must be less than or equal to the maximum replication factor for the file. If insufficient space is available in the file system to increase the number of replicas to the value requested, **mmchattr** ends. Some blocks of the file, however, may have their replication factor increased after **mmchattr** ends.

If additional free space becomes available in the file system at a later time (when, for example, you add another disk to the file system), you can then issue the **mmrestripefs** command with the **-r** or **-b** option to complete the replication of the file. You can then use the **mmlsattr** command to display the replication values.

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

Write permission for the specified files is required to run this command. The **mmchattr** command, however, does not require exclusive access to the file to run; the file can be in use.

Examples

1. To change the metadata replication factor to 2 and the data replication factor to 2 for the **project7.resource** file in file system fs1, enter:

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

Related Information

To change file attributes through SMIT panels, see “Change File System Attributes Using SMIT” on page 46.

“mmcrfs” on page 97

“mmlsattr” on page 123

“mmlsfs” on page 127

mmchconfig

Purpose

Changes GPFS configuration parameters when node is next rebooted or the daemon is restarted.

Syntax

mmchconfig *Attribute=value* [*node_list*]

Operands

Attribute

The name of the attribute to be changed to the specified *value*. The following attribute changes take effect when the GPFS daemon is restarted:

maxFilesToCache

This is the number of i–nodes to cache for recently used files that have been closed. Storing a file's i–node in cache permits faster reaccess to the file. The default is 200 but increasing this number may improve throughput for workloads with high file reuse.

Each cached file requires space in the mallocpool for an i–node plus approximately 800 bytes of metadata. GPFS will not use more than 50% of the mallocpool for this purpose. If you increase **maxFilesToCache**, you should also increase the **mallocsize** parameter to a value equal to 2 x (maximum i–node size for the file system + 800) x **maxFilesToCache**. For example, to set **maxFilesToCache** to 1000 for a file system with a maximum i–node size of 4096, **mallocsize** should be approximately 10MB.

dataStructureDump

This specifies a path for the storage of dumps. The default is to store dumps in /**tmp**/**mmfs**. Specify **no** to store no dumps. This option sets two values in the **mmfs.cfg** file, **dataStructureDump** and **dataStructureDumpOnSGPanic**.

pagepool

This is the size of the cache on each node. It can range from a minimum of 4MB to a maximum of 512MB per node. This value should be specified with the character **M**, for example, `60M`.

mallocsize

This area is used exclusively for GPFS control structures. It can range from a minimum of 2MB to a maximum of 128MB. This value should be specified with the character **M**, for example, `8M`.

Note: The sum of **pagepool** and **mallocsize** must not exceed 80% of real memory.

priority

Sets the UNIX dispatching priority of the daemon process to the specified value. The default is 40.

The next 4 attribute changes take effect when the nodes are rebooted:

autoload
Starts GPFS automatically whenever the nodes are rebooted. Values are **yes** or **no**.

client_ports
The number of communication ports that are reserved for requesting file access tokens.

server_kprocs
The number of kernel processes that are dedicated to processing file access token requests.

server_port_number
Two consecutive UDP ports are defined starting at this port number. The token manager receives token requests through these ports.

Note: You cannot change this attribute if you are providing a partial list of target nodes with a *node_list*. You can only modify this attribute when all nodes are targeted.

node_list
list of target nodes, delineated by commas, specified for customized configuration change. If *node_list* is not specified, the results are propagated to all GPFS nodes. If a list of target nodes is specified, the configuration changes affect only the GPFS nodes listed.

The list can cite nodes in any form of hostname shown in the following examples:

Format	Example
Short Hostname	k145s02
Long Hostname	k145s02.dpd.ibm.com
IP Address	9.119.19.102

Regardless which form you select, you should specify the switch hostname.

Description

Use the **mmchconfig** command to change the GPFS configuration on a single node, a list of nodes, or to make global changes to all nodes in the configuration. Although the configuration may be changed at any time, the changes take effect only when the daemons are restarted or the target nodes are rebooted, depending on the attributes changed. All target nodes need not be rebooted at the same time.

Note: The **server_port_number attribute** must be the same on all nodes. Reboot all nodes in the GPFS subsystem if this attribute is changed.

Results

Upon successful completion of **mmchconfig**, the **mmfs.cfg** or **tm.cfg** file is updated on all GPFS nodes to include the configuration changes made to the target nodes.

Files

- **/var/mmfs/etc/mmfs.cfg**
- **/var/mmfs/etc/tm.cfg**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have root authority to run **mmchconfig**.

Examples

1. To change the priority of the file system daemon to 2 on all nodes in the configuration, enter:

   ```
   mmchconfig priority=2
   ```

2. To change pagepool only on node 6 and node 7 in the k145s system, enter

   ```
   mmchconfig pagepool=380M k145sn6,k145sn7
   ```

Related Information

To change configuration through SMIT panels, see “Changing Configuration Using SMIT” on page 33.

“mmconfig” on page 94

mmchdisk

Purpose

Changes state or parameters of one or more disks.

Syntax

mmchdisk *Device* {**suspend** | **resume** | **stop** | **start** | **change**} **-d** *DiskDescList*

Options

Device

The name of the file system to which the disk(s) belong. File system names need not be fully qualified. **fs0** is as acceptable as /**dev**/**fs0**.

suspend

Instructs GPFS to stop allocating space on the specified disk. Place a disk in this state when you are preparing to restripe the file system off this disk because of faulty performance. This is a user-initiated state that GPFS will never use without an explicit command to change disk state.

Note: A disk remains suspended until it is explicitly resumed. Restarting GPFS or rebooting nodes does not restore normal access to a suspended disk.

resume

Informs GPFS that a disk previously suspended is now available for allocating new space. If the disk is currently in a stopped state, it remains stopped until you specify the **start** option. Otherwise, normal read and write access to the disk resumes.

stop

Instructs GPFS to stop any attempts to access the specified disk(s). Use this option to tell the file manager that a disk has failed or is currently inaccessible because of maintenance.

Note: A disk remains stopped until it is explicitly started by the **mmchdisk** command with the **start** option. Restarting the GPFS Server daemon or rebooting does not restore normal access to a stopped disk.

start

Informs GPFS that disk(s) previously stopped are now accessible. This is accomplished by first changing the disk availability from `down` to `recovering`. The file system metadata is then scanned and any missing updates (replicated data that was changed while the disk was down) are repaired. If this operation is successful, the availability is then changed to `up`. If the metadata scan fails, availability is set to `unrecovered`. This could occur if too many other disks are down. The metadata scan can be reinitiated at a later time by issuing the **mmchdisk start** command again..

Note: If you want to start more than one disk, specify them all in the same **mmchdisk start** command. If you start them separately while metadata is stored on any disk that remains down, the **mmchdisk start** command will fail.

change
Instructs GPFS to change the *Disk Usage* parameter, the *Failure Group* parameter, or both, according to the value(s) specified in *DiskDescList*.

-d *DiskDescList*
A descriptor for each disk to be changed.

Specify only disk names when using the **suspend**, **resume**, **stop**, or **start** options. Separate multiple disk names with semicolons and enclose list in quotation marks. For example, `"gpfs1n12;gpfs2n12"`

When using the **change** option, include the disk name and any new *Disk Usage* and *Failure Group* positional parameter values in the descriptor just as you would for the **mmadddisk** command, except leave the *Server Name* and *Backup Name* positional parameters unspecified. Separate descriptors with semicolons and enclose list in quotation marks, for example, `"gpfs1n12:::dataOnly;gpfs2n12:::metadataOnly:12"`. See "mmadddisk" on page 74 for more on disk descriptors.

Description

Use the **mmchdisk** command to change the state or parameters of one or more disks in a GPFS file system.

The state of a disk can be determined from its status and availability fields, displayable with the **mmlsdisk** command. A disk's state is a combination of its *availability* and *status*. Disk status is either ready or suspended. (Transitional status of replacing, replacement, or being emptied might also appear if a disk is being deleted or replaced.)

A suspended disk is one that the user has decided not to place any new data on. Existing data on a suspended disk may still be read or updated. Typically, a disk is suspended prior to restriping a file system. Suspending a disk tells the **mmrestripefs** command that data should be migrated off that disk. See "mmrestripefs" on page 139. Data may be read from a suspended disk.

Disk availability can be one of three values:

1. Up

   Normal read and write operations are possible

2. Down

   Normal read and write operations are not possible either due to a system-detected failure or an explicit user command that takes the disk offline. A disk with this status is not used.

3. Recovering

   Transitional availability between down and up, during which disk contents are validated.

   When disk availability remains `recovering` after **mmchdisk** has been issued, only partial recovery was achieved. It may be that another disk has an availability of `down` or an I/O error has occurred. Repair all disks and paths to disks before reissuing **mmchdisk** and specifying all `down` disks. If this is unsuccessful and some disks remain in `recovering`, run **mmfsck** to repair the data structure on those disks.

4. Unrecovered

   The disk was not successfully brought up.

You can use the **mmchdisk** command to change the status of a disk (suspend an active disk or resume a suspended disk), or to change its availability (start or stop a disk).

Be sure to use **stop** before you take a disk off-line for maintenance. You should also use **stop** when a disk has become temporarily inaccessible due to a disk failure that is repairable without loss of data on that disk (for example, an adapter failure or a failure of the disk electronics).

The **change** option operates on the *Disk Usage* (**dataOnly**, **metadataOnly**, **dataAndMetadata**) and *Failure Group* parameters of a disk. **mmchdisk change** does not move data or metadata that resides on the disk. After changing disk parameters, in particular, *Disk Usage*, you may have to issue the **mmrestripefs** command with the **-r** option to relocate data so that it conforms to the new disk parameters.

mmchdisk can be issued for a mounted or unmounted file system. When maintenance is complete or the failure has been repaired, use the **mmchdisk** command with the **start** option. If the failure cannot be repaired without loss of data, you can use the **mmdeldisk** command.

Return Codes

0	Successful completion.
1	A failure has occurred.

Restrictions

Root privileges are required to run this command.

Examples

1. To suspend active disk gpfs1n365, enter:

   ```
   mmchdisk /dev/fs0 suspend –d gpfs1n365
   ```

2. To stop suspended disk gpfs1n365, enter:

   ```
   mmchdisk /dev/fs0 stop –d gpfs1n365
   ```

3. To start stopped disk gpfs1n365, enter:

   ```
   mmchdisk /dev/fs0 start –d gpfs1n365
   ```

4. To resume allocation on started disk gpfs1n365, enter:

   ```
   mmchdisk /dev/fs0 resume –d gpfs1n365
   ```

5. To specify that metadata should no longer be stored on disk gpfs1n365, enter:

   ```
   mmchdisk /dev/fs0 change –d "gpfs1n365:::dataOnly"
   ```

Related Information

“mmadddisk” on page 74

“mmdeldisk” on page 105

“mmlsdisk” on page 125

“mmrpldisk” on page 141

mmcheckquota

Purpose

Checks file system i-node and space usage.

Syntax

mmcheckquota [**-v**] {**-a** | *Device*...}

Options

-a Checks all GPFS file systems with disk quotas.

-v Reports discrepancies between calculated and recorded disk quotas.

Operands

Device Device name of the file system to be checked. File system names need not be fully qualified. **fs0** is as acceptable as /**dev**/**fs0**.

Description

The **mmcheckquota** command counts i-node and space usage in a file system by user and group, and writes the collected data into quota files. This command need only be issued when quotas are first installed, after executing **mmedquota**, or when quota information is lost due to node failure and users are denied disk space that their quotas should allow.

mmcheckquota is an I/O-intensive command and should be run when the system load is light.

Files

- *filesystem_mountpoint*/**user.quota**
- *filesystem_mountpoint*/**group.quota**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have root authority to run **mmcheckquota**.

Examples

1. To check quotas for file system **fs0**, enter:

   ```
   mmcheckquota fs0
   ```

2. To check quotas for all file systems, enter:

   ```
   mmcheckquota -a
   ```

Related Information

“mmedquota” on page 116

“mmlsquota” on page 129

“mmquotaon” on page 135

“mmquotaoff” on page 133

“mmrepquota” on page 137

mmchfs

Purpose

Changes the attributes of a GPFS file system.

Syntax

mmchfs *Device* [**-A yes** | **no**] [**-Q yes** | **no**] [**-s** StripeMethod] [**-m** *DefaultMetadataReplicas*] [**-r** *DefaultDataReplicas*]

Options

-A yes | **no**
: Change the automatic mount option for the file system. This specifies whether the file system should be automatically mounted when the GPFS daemons start.

-m *DefaultMetaDataReplicas*
: Change the default number of replicas for metadata. Allowable values are 1 and 2 but cannot exceed the values of *MaxMetaDataReplicas* set when the file system was created.

-Q yes | **no**
: Activate quotas automatically when the file system is mounted.

-r *DefaultDataReplicas*
: Change the default number of replicas for data. Allowable values are 1 and 2 but cannot exceed the values of *MaxDataReplicas* set when the file system was created.

-s *StripeMethod*
: Change the stripe method for this file system. Possible values are:

  roundRobin
  : Places one block on all of the disks in the file system before repeating one, but uses the same permutation of disks repeatedly as the file grows.

  random
  : Places the next block randomly, subject only to the constraint that no two replicas of a block be in the same failure group.

  balancedRandom
  : Randomly places one block on all of the disks in the file system before repeating one.

Operands

Device
: The name of the file system device to be changed. File system names need not be fully qualified. **fs0** is as acceptable as /**dev**/**fs0**.

Description

This command need only be executed on one node in the GPFS configuration. The changes will be propagated through all nodes in the configuration.

Results

Upon successful completion of **mmchfs** the following tasks are performed:

- /**etc/filesystems** is updated, if the **-A** or **-Q** option is specified.
- Quotas are activated on subsequent starts of the GPFS daemons, if the **-Q** option is activated.
- All files created after execution of **mmchfs** take new attributes. Existing files are not affected. Use the **mmchattr** command to change the replication factor of existing files. To apply a new stripe method to existing files you must run the **mmrestripefs** command.

Files

- /**etc/filesystems**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have root authority to run **mmchfs**.

Examples

To change the default replicas for metadata to 2 and the default replicas for data to 2 for new files created in the **fs0** file system, enter:

```
mmchfs fs0 -m 2 -r 2
```

Related Information

To change file system attributes through SMIT panels, see “Change File System Attributes Using SMIT” on page 46.

“mmcrfs” on page 97

“mmdelfs” on page 108

“mmdf” on page 112

“mmfsck” on page 118

“mmlsfs” on page 127

mmconfig

Purpose

Configures GPFS prior to creating file system.

Syntax

mmconfig [**-A**] [**-c** *config_file*] [**-D** *dataStructureDump*] [**-K** *server_kprocs*] [**-m** *mallocsize*] [**-M** *maxFilesToCache*] [**-n** *node_file*] [**-p** *pagepool*] [**-P** *client_ports*] [**-r** *priority*] [**-S** *server_port_number*]

Options

-A

Specifies that GPFS daemons are to be automatically started when nodes come up. The default is not to start daemons automatically.

-c *config_file*

The name of a file containing global configuration information. GPFS copies the data specified in *config_file* to **/var/mmfs/etc/mmfs.cfg** on all nodes to be configured. If this option is not specified, the sample configuration file, **/usr/lpp/mmfs/samples/mmfs.cfg.sample** from the install package is used.

-D *dataStructureDump*

This specifies a path for the storage of dumps. The default is to store dumps in **/tmp/mmfs**. Specify no to store no dumps. This option sets two values in the **mmfs.cfg** file, **dataStructureDump** and **dataStructureDumpOnSGPanic**.

-K *server_kprocs*

The number of kernel processes that are dedicated to processing file access token requests. This determines how efficiently multiple parallel processes may access a single file.

-m *mallocsize*

This area is used exclusively for GPFS control structures. It can range from a minimum of 2MB to a maximum of 128MB. This value should be specified with the character **M**. The default is 4M. If the **-c** option is specified, then **-m** is not allowed.

-M *maxFilesToCache*

This is the number of i–nodes to cache for recently used files that have been closed. Storing a file's i–node in cache permits faster reaccess to the file. The default is 200 but increasing this number may improve throughput for workloads with high file reuse. Increasing it where file reuse is not great will waste kernel heap storage. Do not increase this parameter if you are in doubt.

Each cached file requires space in the mallocpool for an i–node plus approximately 800 bytes of metadata. GPFS will not use more than 50% of the mallocpool for this purpose. If you increase **maxFilesToCache**, you should also increase the **mallocsize** parameter to a value equal to 2 x (maximum i–node size for the file system + 800) x **maxFilesToCache**. For example, to set **maxFilesToCache** to 1000 for

a file system with a maximum i–node size of 4096, **mallocsize** should be approximately 10MB.

-n *node_file*
Copies the list of nodes specified in *node_file* to the /**etc**/**cluster.nodes** file, which defines the cluster on all nodes. *node_file* consists of a list of hostnames, one per line. The list can cite nodes in any of the hostname forms shown in the following examples:

Format	Example
Short Hostname	k145s02
Long Hostname	k145s02.dpd.ibm.com
IP Address	9.119.19.102

Regardless which form you select, you should specify switch hostnames. If this option is not specified, all nodes in the System Data Repository are added to the /**etc**/**cluster.nodes** file.

-p *pagepool*
This is the size of the cache on each node. It can range from a minimum of 4MB to a maximum of 512MB per node. This value should be specified with the character **M**. The default is 20M. If the **-c** option is specified, then **-p** is not allowed.

Note: The sum of **pagepool** and **mallocsize** must not exceed 80% of real memory.

-P *client_ports*
The number of communication ports that are reserved for requesting file access tokens.

-r *priority*
Sets the UNIX dispatching priority of the daemon process to the value specified. The default is 4. If the **-c** option is specified, then **-r** is not allowed.

-S *server_port_number*
Two consecutive UDP ports are defined starting at this port number. The token manager receives token requests through these ports.

Description

The **mmconfig** command allows you to specify configuration information on the command line or supply it in a *config_file* with the **-c** option. The use of a *config_file* provides you with a future reference and input should you decide to change the configuration at a later date. You might find it easier to input the specifications to a file, rather than in the more pressured situation that sometimes exists when entering long strings of data on the command line.

Results

Upon successful completion of **mmconfig** the following tasks are performed on all specified nodes:

- The configuration files are copied to/**var**/**mmfs**/**etc**/**mmfs.cfg** and /**var**/**mmfs**/**etc**/**tm.cfg**
- The /**etc**/**cluster.nodes** file is created and copied
- The GPFS vfs number is added to the /**etc**/**vfs** file

Files

- **/etc/cluster.nodes**
- **/var/mmfs/etc/mmfs.cfg**
- **/var/mmfs/etc/tm.cfg**
- **/etc/vfs**

Return Codes

0 Successful completion.

1 A failure has occurred and an error message is printed to stdout.

Restrictions

You must have root authority to run **mmconfig**.

You cannot create a second GPFS configuration on an SP system. Use **mmchconfig** instead if you want to change a current GPFS configuration.

Examples

1. To configure all the nodes listed in the file called **system_1**, and have GPFS automatically started when the nodes come up, enter:

   ```
   mmconfig –n system_1 -A
   ```

Related Information

To configure GPFS through SMIT panels, see “Configure Using SMIT” on page 31.

“mmaddnode” on page 78

“mmchconfig” on page 83

“mmdelnode” on page 110

mmcrfs

Purpose

Creates GPFS file system.

Syntax

mmcrfs *Mountpoint Device* { *DiskDescList* | **-F** *DescFile*} [**-A** **yes** | **no**] [**-B** *BlockSize*] [**-i** *InodeSize*] [**-I** *IndirectSize*] [**-k** **yes** | **no**] [**-m** *DefaultMetadataReplicas*] [**-M** *MaxMetadataReplicas*] [**-n** *NumNodes*] [**-N** *NumInodes*] [**-Q** **yes** | **no**] [**-r** *DefaultDataReplicas*] [**-R** *MaxDataReplicas*] [**-s** *StripeMethod*] [**-v** **yes** | **no**]

Operands

Mountpoint
: The mountpoint directory of the GPFS file system.

Device
: The device name of the file system to be created. File system names need not be fully qualified. **fs0** is as acceptable as /**dev**/**fs0**.

-F *DescFile*
: Points to a file containing a list of disk descriptors (see “Disk Descriptors”), one per line. This file eliminates the need for command line entry of these descriptors using the *DiskDescList* parameter. A sample file can be found in /**usr**/**lpp**/**mmfs**/**samples**/**diskdesc**.

DiskDescList
: A descriptor for each IBM Virtual Shared Disk to be added. Each descriptor (see “Disk Descriptors”) is separated by a semicolon (;). The entire *DiskDescList* must be enclosed in quotation marks (' or ").

Disk Descriptors

Whether you use a descriptor file or disk descriptor list, the positional parameters of the disk descriptor are the same. The only difference is how the disk descriptors are delimited. When entering them on the command line using a disk descriptor list, each descriptor is separated by a semicolon. When entering them in a descriptor file, specify each descriptor on a separate line. No more than 512 disks can be defined for a single file system.

Each descriptor contains positional parameters that apply to the IBM Virtual Shared Disk, separated by colons (:).

You must provide a descriptor for each GPFS IBM Virtual Shared Disk to be created or passed to the GPFS file system. Each descriptor can contain the following positional parameters:

Disk Name
: The device name, for example, `hdisk1`, of any disk you wish to use to create an IBM Virtual Shared Disk, or the name of an existing IBM Virtual Shared Disk.

  GPFS performance and recovery processes function best with one disk per IBM Virtual Shared Disk. If you want to create IBM Virtual Shared

Disks with more than one disk, refer to Managing Shared Disks. Then pass the existing IBM Virtual Shared Disks to GPFS using the VSD names.

Server Name
The server name, when creating an IBM Virtual Shared Disk. This can be in any recognizable form and is unnecessary when specifying a predefined IBM Virtual Shared Disk.

Backup Server Name
The backup server name, when creating an IBM Virtual Shared Disk. This can be in any recognizable form and is unnecessary when specifying a predefined IBM Virtual Shared Disk.

Disk Usage
Specify one of the following or allow to default:

- **dataAndMetadata** (default)
- **dataOnly**
- **metadataOnly**

.

Failure Group
A number identifying the failure group to which this disk belongs. All disks that have a common point of failure, such as all disks that are attached to the same VSD server node, should be placed in the same failure group. GPFS uses this information during data and metadata placement to assure that no two replicas of the same block are written in such a way as to become unavailable due to a single failure. You can specify any value from -1 (where -1 indicates that the disk has no point of failure in common with any other disk) to 4000. If you do not specify a failure group, the value defaults to the server node number plus 4000.

Any parameter that is intentionally omitted or allowed to default must be explicitly skipped. For example, if you had no backup server and wanted to use your disk for both data and metadata (the default), you could create a disk descriptor that looked like this:

```
hdisk1:sp1sw05:::5
```

If you also wanted to allow the last parameter, *Failure Group*, to default, you could omit the last three delimiting colons:

```
hdisk1:sp1sw05
```

or leave them in if you prefer:

```
hdisk1:sp1sw05:::
```

Trailing delimiters are acceptable.

Options

-A yes | no
Automatic Mount option. Indicates whether the file system should automatically mount when the GPFS daemon starts. The default is **yes**.

-B *BlockSize*
Size of data blocks. Must be 16KB, 64KB, or 256KB (the default).

-i *InodeSize*
The number of bytes in an i-node. The default is 512 bytes, the size of one logical sector. This can be as large as 4KB.

-I *IndirectSize*
Number of bytes in an indirect block. This value must be a multiple of one subblock (*BlockSize*/32), and cannot exceed 32KB or *BlockSize*, whichever is smaller. Recommended values are *BlockSize*/*n*, where *n* is a power of 2. The default is *BlockSize*/16.

-k yes | **no**
Keep the **restart.desc** file regardless of whether **mmcrfs** completes successfully. This file contains a list of all IBM Virtual Shared Disks in the file system, whether generated or not, and can be helpful in later attempts to create the file system and avoid recreating any IBM Virtual Shared Disks already completed. See “Error Recovery” on page 101.

-m *DefaultMetadataReplicas*
Default number of copies of i-nodes and indirect blocks for a file. Allowable values are 1 and 2 but cannot exceed the value of *MaxMetadataReplicas*. The default is 1.

-M *MaxMetadataReplicas*
Default maximum number of copies of i-nodes and indirect blocks for a file. Can only be overridden by a system call when the file has length 0. Determines the maximum possible file size, since space is reserved in the i-node for all possible copies of pointers to indirect blocks. Allowable values are 1 and 2 but cannot be lower than *DefaultMetadataReplicas*. The default is 1.

-n *NumNodes*
The estimated number of nodes on which the file system will be mounted. This is used as a best guess for the initial size of some file system data structures. The default is 32.

-N *NumInodes*
The estimated number of i-nodes to allocate in the file system. This value defaults to the size of the file system at creation, divided by 1MB, and can be specified with a suffix, for example 8K or 2M.

-Q yes | **no**
Activate quotas automatically when the file system is mounted. The default is **no**.

-r *DefaultDataReplicas*
Default number of copies of each data block for a file. Allowable values are 1 and 2 but cannot exceed *MaxDataReplicas*. The default is 1.

-R *MaxDataReplicas*
Default maximum number of copies of data blocks for a file. Can only be overridden by a system call when the file has length 0. Determines the maximum possible file size, since space is reserved in the i-node and indirect blocks for all possible copies of pointers to data blocks. Allowable values are 1 and 2 but cannot be lower than *DefaultDataReplicas*. The default is 1.

-s *StripeMethod*
Set the stripe method for this file system. Possible values are:

roundRobin
Places one block on all of the disks in the file system before writing to the first disk again, repeating the same sequence of disks as the file grows. This provides the best performance if the file system never changes the number of disks, but requires more data movement when restriping. This is the default.

random
Places the next block randomly, subject only to the constraint that no two replicas of a block be in the same failure group.

balancedRandom
Randomly places one block on all of the disks in the file system before repeating one.

-v yes | no
Verify that specified disks do not belong to an existing file system. The default is **yes**. Specify **no** only when you want to reuse disks that are no longer needed for an existing file system.

Description

Use the **mmcrfs** command to create a GPFS file system. The block size, indirect size, i-node size, and replication factors affect file system performance as well as maximum potential file size. Refer to Table 1 on page 11 for details.

Notes on specifying disk descriptor parameters::

1. To use an existing IBM Virtual Shared Disk volume group in the file system, only the *Disk Name* (the IBM Virtual Shared Disk name) need be specified in the disk descriptor. The *Disk Usage* and *Failure Group* parameters are optional and may be allowed to default.
2. To create an IBM Virtual Shared Disk with one disk and use it in a file system, you must specify the *Disk Name* and *Server Name* parameters in the descriptor. *Backup Server*, *Failure Group*, and *Disk Usage* are optional.

When specifying the *DiscDescList* parameter, enclose the entire string in quotation marks so that the shell does not interpret the semicolon delimiters. The current maximum number of disks that can be defined for any file system is 512.

When issuing the **mmcrfs** command, the size of the command must not exceed the shell limit. If the shell limit is insufficient to specify the file system on the command line, shorten the command string by using the **-F** option and specifying the disk descriptors in a file.

Although you can use SMIT to create the GPFS file system on each node, the following parameters can only be set using the **mmcrfs** command:

-m *DefaultMetadataReplicas*

-r *DefaultDataReplicas*

-M *MaxMetadataReplicas*

-R *MaxDataReplicas*

Results

Upon successful execution of **mmcrfs** the following tasks are completed on all GPFS nodes:

- IBM Virtual Shared Disks are created and configured.
- Mount point directory is created.
- File system is formatted.
- GPFS stanza is added to /**etc/filesystems** file.

If GPFS is not active on the node where **mmcrfs** is issued, the command fails.

If the node specified as either the server or backup server is down, the **mmcrfs** command fails because it will be unable to create IBM Virtual Shared Disks.

If any other node is down, the command executes, provided enough nodes are up to qualify as a quorum (see “Quorums” on page 6.) The command completes the operation when the node comes back up.

Error Recovery: **mmcrfs** generates an error recovery disk descriptor file called **restart.desc**. If the creation of an IBM Virtual Shared Disk fails due to a system or user error, **mmcrfs** is designed to keep going rather than force a restart of the time-intensive process of IBM Virtual Shared Disk creation. The **restart.desc** file is stored in the directory where **mmcrfs** is issued. It contains a list of disk descriptors corresponding to those specified to the **mmcrfs** command. For IBM Virtual Shared Disk that are successfully created, the descriptor reflects the new VSD name. The only descriptors remaining in *disk* format are those that failed. If **mmcrfs** is successful, then the **restart.desc** file is deleted, unless **-k yes** is specified.

To use the **restart.desc** file, check the error messages for the IBM Virtual Shared Disks that failed. If the error is a user error, fix the problem in the disk, the descriptor, or wherever the problem occurred. If the failure was transient, such as failing to get a lock from IBM Recoverable Virtual Shared Disk, then the descriptor may be retried without modification. The subsequent **mmcrfs** command will generate a new **restart.desc** file, so rename the current one before using it. Use the renamed **restart.desc** file as the descriptor file in the **-F** option of **mmcrfs** in place of the original list of descriptors. If there is a problem with a descriptor that cannot be fixed, consider deleting that descriptor from the list. If the problem can be fixed later, the disk may be added using **mmadddisk**.

If the **mmcrfs** command is interrupted, the **restart.desc** file may not contain all descriptors. Append the missing lines to the file from the corresponding descriptors in the original list if you want to use the **restart.desc** file.

Files

- **/etc/filesystems**

Return Codes

0	Successful completion.
1	A failure has occurred.

Restrictions

You must have root authority to run **mmcrfs**.

Examples

1. Start with a basic disk descriptor that takes advantage of GPFS defaults:

   ```
   hdisk5:sp1sw06:sp1sw07::
   ```

   This descriptor translates as follows:

   Disk Name hdisk5

   Server Name Switch node 6

   Backup Server Switch node 7

   Disk Usage Defaults to **dataAndMetadata**, allowing both

   Failure Group Defaults to 4006 (server node number + 4000)

   This descriptor would create an IBM Virtual Shared Disk with the name **gpfs1n6** on hdisk5.

2. If we add another descriptor to the file,

   ```
   hdisk6:sp1sw04::dataOnly:28
   ```

   which translates as follows:

   Disk Name hdisk6

   Server Name Switch node 4

   Backup Server None

   Disk Usage Data, not metadata

   Failure Group 28

   GPFS would also create an IBM Virtual Shared Disk with the name **gpfs2n4** on hdisk6.

3. This descriptor,

   ```
   hdisk9:sp1sw04::metadataOnly:-1
   ```

   translates to:

   Disk Name hdisk9

   Server Name Switch node 4

   Backup Server none

   Disk Usage Metadata, not data

   Failure Group Creates a unique failure group containing only this device

   This descriptor tells GPFS to also create an IBM Virtual Shared Disk with the name **gpfs3n4** on hdisk9.

4. One additional descriptor,

```
vsdx3n12::::
```

which could also be entered simply as

```
vsdx3n12
```

translates to:

Disk Name vsdx3n12, the name of an existing IBM Virtual Shared Disk

Server Name Predefined

Backup Server Predefined

Disk Usage Defaults to **dataAndMetadata**, allowing both

Failure Group Defaults to server node number + 4000

This descriptor specifies that an existing IBM Virtual Shared Disk with the name **vsdx3n12** be added to the GPFS file system.

To create a file system named **fs1** by entering the four disk descriptors on the command line and allowing all other parameters to default:

```
mmcrfs /gpfs/fs1 fs1 "hdisk4:sp1sw06:sp1sw07;hdisk5:sp1sw06::dataOnly:28;hdisk6:sp1sw04::metadataOnly:-1;vsdx3n12"
```

Instead of keying in the disk descriptors on the command line, we could enter them, one per line, in a file named **fs1disc.desc** and identify it with the **-F** option.

```
hdisk4:sp1sw06:sp1sw07
hdisk5:sp1sw06::dataOnly:28
hdisk6:sp1sw04::metadataOnly:-1
vsdx3n12
```

This example creates a file system called **fs1** with a block size of 16KB, an indirect block size 1KB, and an i-node size 1KB. The file system will be mounted on 16 nodes and accommodate an estimated 2500 files. It will be automatically mounted when the nodes boot up, with quota enforcement enabled. The maximum replication for data and metadata in this file system will be 2, although neither will be initially replicated by default.

```
mmcrfs /gpfs/fs1 fs1 -F fs1disc.desc -A yes -B 16K -I 1K -M 2 -n 16 -N 2500 -Q yes -R 2
```

Related Information

To create a GPFS file system through SMIT panels, see “Create File System Using SMIT” on page 39.

“mmchfs” on page 92

“mmdelfs” on page 108

“mmfsck” on page 118

“mmdf” on page 112

“mmlsfs” on page 127

mmdelacl

Purpose

Delete GPFS access control list.

Syntax

mmdelacl [**-d**] *File*

Operands

File

The path name of the file or directory for which the ACL is to be deleted. If the -d option is specified, *file* must contain the name of a directory.

Options

-d

Specifies that the default ACL of a directory is to be deleted.

Description

Use the **mmdelacl** command to delete the extended entries of an access ACL of a file or directory, or to delete the default ACL of a directory.

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

mmdelacl can only be executed by the file or directory owner.

Examples

1. To delete the default ACL for a directory named **project2**, enter:

```
mmdelacl –d project2
```

Related Information

"mmeditacl" on page 114

"mmgetacl" on page 122

"mmputacl" on page 131

mmdeldisk

Purpose

Removes a disk from GPFS file system.

Syntax

mmdeldisk *Device DiskNames* [**-a**] [**-n**] [**-p**] [**-r**]

Operands

Device

The device name identifying the file system where the disk(s) are to be deleted. file system names need not be fully qualified. **fs0** is as acceptable as /**dev**/**fs0**.

DiskNames

The names of the IBM Virtual Shared Disks to be deleted. Specify the name generated when it was added. Use the **mmlsdisk** command ("mmlsdisk" on page 125) to display disk names. If there is more than one disk to be deleted, delineate each name with a semicolon (;) and enclose the list of disk names in quotation marks.

Options

-a

Specifies that the **mmdeldisk** command should NOT wait for rebalancing to complete before returning. When this flag is specified, **mmdeldisk** runs asynchronously and returns after the file system descriptor is updated and the rebalancing scan is started, but it does not wait for rebalancing to finish. If no rebalancing is requested (**-r** option is not specified), this option has no effect.

-n

Preserve the deleted VSDs. If this option is not specified, any VSD generated by a disk descriptor is deleted.

-p

Indicates that the disk(s) are permanently damaged or no longer accessible, and instructs the **mmdeldisk** command not to attempt to read any data or metadata from the disk. Instead of migrating the data, **mmdeldisk** invalidates all references to the deleted disks in the i-nodes and indirect blocks. Run **mmfsck** after using this option.

Any subsequent request to read data from a file that was stored on a deleted disk and not sufficiently replicated returns an I/O error (EIO).

-r

Rebalance all existing files in the file system to make more efficient use of the remaining disks.

Description

The **mmdeldisk** command migrates all data that would otherwise be lost to the remaining disks in the file system. It then removes the disks from the file system descriptor. **mmdeldisk** optionally rebalances the file system after removing the disks.

Run **mmdeldisk** when system demand is low.

If a replacement for a failing disk is available, use the **mmrpldisk** command in order to keep the file system balanced. Otherwise, use one of the following procedures to delete a disk:

- If the disk is not failing and the GPFS Server can still read from it:
  1. Suspend the disk
  2. Restripe to rebalance all data onto other disks
  3. Delete the disk
- If the disk is permanently damaged and the data cannot be recovered, delete the disk if the file system is not replicated. If the file system is replicated:
  1. Suspend the disk
  2. Restripe and restore replication for the file system, if possible
  3. Delete the disk from the file system

  Later, when the disk is repaired or a replacement is available,
  4. Add the repaired or new disk back into the file system
  5. Restripe the file system

Note: **mmdeldisk**will not delete a disk in a stopped state. A stopped disk must be started before it can be deleted using **mmdeldisk**.

Results

Upon successful execution of **mmdeldisk** the following tasks are completed:

- Unreplicated data from the target disk(s) is migrated to other disks in the file system.
- IBM Virtual Shared Disk is stopped and unconfigured.
- Stanza with the deleted disks is updated in the /**etc**/**filesystems** file if the disk names appear on the options line. If the options line subsequently contains too few disk listings, more will be added from the full list in the SDR version of the stanza.
- Remaining disks are rebalanced, if specified.
- The IBM Virtual Shared Disk is deleted, if it was created from a disk descriptor and the **—n** option is not specified.

Files

- **/etc/filesystems**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have root authority to run **mmdeldisk**.

Examples

1. To delete gpfs2n3 and gpfs3n4 from file system **fs0** and rebalance the files across the remaining disks, enter:

```
mmdeldisk fs0 "gpfs2n3;gpfs3n4" -r
```

Related Information

To delete disks through SMIT panels, see “Delete Disks Using SMIT” on page 51.

“mmadddisk” on page 74

“mmchdisk” on page 86

“mmlsdisk” on page 125

“mmrpldisk” on page 141

mmdelfs

Purpose

Removes a GPFS file system.

Syntax

mmdelfs *Device* [**-n**]

Operands

Device

Device name of the file system, as shown in /**etc**/**filesystems**. File system names need not be fully qualified. **fs0** is as acceptable as /**dev**/**fs0**.

Options

-n

Preserve the deleted IBM Virtual Shared Disks. If this option is not specified, any IBM Virtual Shared Disk generated by a disk descriptor is deleted.

Description

The **mmdelfs** command removes all the structures for the specified file system from the nodes in the configuration.

Before you can delete a file system using **mmdelfs**, you must unmount it on all nodes.

Results

Upon successful execution of **mmdelfs** the following tasks are completed on all nodes:

- Removes the stripe group
- Removes all IBM Virtual Shared Disks created from disk descriptors, if the **-n** option is not specified
- Removes the stanza for the file system from /**etc**/**filesystems** file
- Deletes the character device entry from /**dev**
- Removes the mount point directory where the file system had been mounted

Files

- /**etc**/**filesystems**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have root authority to run **mmdelfs**.

mmdelfs must be run on a node in the GPFS configuration.

Examples

1. To delete file system **fs0**, enter:

```
mmdelfs fs0
```

Related Information

To delete a file system through SMIT panels, see “Delete a File System Using SMIT” on page 43.

“mmcrfs” on page 97

“mmchfs” on page 92

“mmdf” on page 112

“mmlsfs” on page 127

mmdelnode

Purpose

Removes one or more nodes from GPFS configuration.

Syntax

mmdelnode [**-c**] {*Delnodes* | **-a**}

Options

-a Delete all nodes.

-c Compress the node list contained in /**etc**/**cluster.nodes** file. This option removes `localhost` entries. The GPFS daemons must be stopped on all nodes (not only the ones specified in *Delnodes*) before you can run this option. Afterward, you must restart GPFS on all nodes at the same time in order for nodes with different versions of the /**etc**/**cluster.nodes** file to communicate.

Operands

Delnodes

List of nodes to be deleted, delineated by commas. The list can cite nodes in any of the hostname forms shown in the following examples:

Format	Example
Short Hostname	k145s02
Long Hostname	k145s02.dpd.ibm.com
IP Address	9.119.19.102

Description

Use the **mmdelnode** command to delete one or more nodes from the GPFS configuration. Issue **mmdelnode** on any GPFS node.

Before you can use **mmdelnode** to remove a node, you must unmount the file system and stop GPFS on the node to be deleted.

Note: Exercise caution when deleting nodes from the GPFS configuration. If the number of remaining nodes falls below the requirement for a quorum, you will be unable to perform GPFS file system operations. See “Quorums” on page 6 for more information.

Results

Upon successful execution of **mmdelnode** the following tasks are completed:

- Mount point directories and character mode devices are removed from the deleted nodes.
- /**etc**/**filesystems** is updated on deleted nodes by removing the GPFS stanzas.
- /**etc**/**cluster.nodes**, **mmfs.cfg**, and **tm.cfg** are removed from deleted nodes.
- /**etc**/**cluster.nodes** updated on remaining nodes, to replace deleted nodes with `localhost` entries.
- The quorum requirement is updated

If the **mmdelnode** command is unsuccessful because one or more nodes are down, it will complete execution as the node(s) come up.

Files

- **/etc/cluster.nodes**
- **/etc/filesystems**
- **/var/mmfs/etc/mmfs.cfg**
- **/var/mmfs/etc/tm.cfg**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have root authority to run **mmdelnode**.

Examples

1. To delete all nodes, enter:

   ```
   mmdelnode -a
   ```

2. To delete nodes k145s12, k145s13, and k145s14, enter:

   ```
   mmdelnode k145s12,k145s13,k145s14
   ```

3. To compress the **/etc/cluster.nodes** file by removing all `localhost` entries, enter:

   ```
   mmdelnode -c
   ```

Related Information

To delete a node through SMIT panels, see “Deleting Nodes Using SMIT” on page 59.

“mmaddnode” on page 78

“mmconfig” on page 94

“mmchconfig” on page 83

“mmlsfs” on page 127

mmdf

Purpose

Queries available file space on a GPFS file system.

Syntax

mmdf *Device* [**-d** | **-m**]

Operands

Device
: The name of the GPFS file system, for example, **/dev/fs0**. File system names need not be fully qualified. **fs0** is as acceptable as **/dev/fs0**.

Options

-d List only disks that can hold data.

-m List only disks that can hold metadata.

The default is to list all disks.

Description

Use the **mmdf** command to display the available space on each disk, as well as the total space available to the GPFS file system.

mmdf can be run against a mounted or unmounted file system.

Displayed values are rounded down to a multiple of 1024 bytes. If the fragment size used by the file system is not a multiple of 1024 bytes, then the displayed values may be lower than the actual values. This can result in the display of a total value that exceeds the sum of the rounded values displayed for individual disks. The individual values are accurate if the fragment size is a multiple of 1024 bytes.

Results

For each disk in the GPFS file system, **mmdf** displays:

- The size of the disk
- The failure group of the disk
- Whether the disk is used to hold data, metadata, or both
- Available space in full blocks
- Available space in fragments

mmdf displays the disks by failure group.

Return Codes

0 Successful completion.

1 A failure has occurred.

Examples

1. To query all disks in the **gpfssa** file system, enter:

   ```
   mmdf gpfssa
   ```

   Information is returned in the following format:

```
                                                        free KB   free KB
disk            disk size  failure holds    holds     in full        in
name                in KB    group metadata data       blocks fragments
--------------- --------- -------- -------- ------ --------- ---------
g05ssa06vsd1n5    2097152      561 yes      yes      1774592      1312
g05ssa06vsd6n5    2097152      561 yes      yes      1775104       896
g05ssa06vsd2n5    2097152      561 yes      yes      1774080       856
g05ssa06vsd5n5    2097152      561 yes      yes      1773056       784
g05ssa06vsd3n5    2097152      561 yes      yes      1773312       680
g05ssa06vsd4n5    2097152      561 yes      yes      1773312      1064
g07ssa08vsd3n7    2097152      781 yes      yes      1772800       928
g07ssa08vsd4n7    2097152      781 yes      yes      1773056       624
g07ssa08vsd2n7    2097152      781 yes      yes      1775104       864
g07ssa08vsd5n7    2097152      781 yes      yes      1772544      1480
g07ssa08vsd1n7    2097152      781 yes      yes      1774336      1096
g07ssa08vsd6n7    2097152      781 yes      yes      1774848      1368
                 ---------                         --------- ---------
(total)          25165824                          21286144     11952
```

Related Information

“mmchfs” on page 92

“mmcrfs” on page 97

“mmdelfs” on page 108

“mmlsfs” on page 127

mmeditacl

Purpose

Create or change GPFS access control list.

Syntax

mmeditacl [**-d**] *File*

Operands

File

The path name of the file or directory for which the ACL is to be edited. If the -d option is specified, *file* must contain the name of a directory.

Options

-d

Specifies that the default ACL of a directory is to be edited.

Description

Use the **mmeditacl** command for interactive editing of the ACL of a file or directory. This command uses the default editor, specified in the EDITOR environment variable, to display the current access control information, and allows the file owner to change it. The command verifies the change request with the user before making permanent changes.

The EDITOR environment variable must contain a complete path name, for example,

```
EDITOR=/usr/bin/vi
```

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

mmeditacl can be used to display an ACL by any user with access to the file or directory, but only the owner or root user can change an ACL.

Examples

1. To edit the ACL for a file named **project2.history** enter,

   ```
   mmeditacl project2.history
   ```

2. To edit the default ACL for a directory named **project2** enter,

   ```
   mmeditacl –d project2
   ```

Related Information

“mmdelacl” on page 104

“mmgetacl” on page 122

“mmputacl” on page 131

mmedquota

Purpose

Sets quota limits.

Syntax

mmedquota {**-u** [**-p** *ProtoUser*] *User*... | **-g** [**-p** *ProtoGroup*] *Group*... | **-t** {**-u** | **-g**}}

Operands

User Name or user id of target user for quota editing.

Group Name of target group for quota editing.

Options

-g Sets quota limits or grace times for groups.

-p When invoked with the **-u** option, uses *ProtoUser*, a set of limits that you can apply, by name, to any user without entering the values manually. This makes it easy to set the same limits for all users. When invoked with the **-g** option, **mmedquota** uses quotas established for prototypical group specified in *ProtoGroup* for each target group. You can specify any user as a prototype for another user, or any group as a prototype for another group.

-u Sets quota limits or grace times for users.

-t Sets grace period during which quotas can exceed the soft limit before it is imposed as a hard limit. The default grace period is one week.

Description

Use the **mmedquota** command to set or change quota limits for users and groups. **mmedquota** displays the current values for the following limits, if any, and prompts you to enter new values using your default editor:

- Current Block usage (display only)
- Current I-node usage (display only)
- I-node soft limit
- I-node hard limit
- Block soft limit
- Block hard limit

Note: A block or inode limit of 0, indicates no limit.

The **mmedquota** command waits for the edit window to be closed before checking and applying new values. If an invalid entry is made, you must reissue the command and enter the correct values.

You can also use **mmedquota** to change the file system-specific grace periods for block and file usage if the default of one week is unsatisfactory. The grace period is the time during which users can exceed the soft limit. If the user or group does not reduce usage below the soft limit before the grace period expires, the soft limit becomes the new hard limit.

Files

- *filesystem_mountpoint*/**user.quota**
- *filesystem_mountpoint*/**group.quota**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

The file system must be mounted before **mmedquota** can be run.

You must have root authority to run **mmedquota**.

Examples

1. To set user quotas for user paul, enter:

   ```
   mmedquota -u paul
   ```

   You receive the following prompt in your default editor:

   ```
   *** Edit quota limits for USR paul
   NOTE: block limits will be rounded up to the next multiple of the block size.
   gpfs0:  blocks in use (in KB) : 25600, limits (soft = 0, hard = 0)
            inodes in use: 3, limits (soft = 0, hard = 0)
   ```

2. To change the grace periods for all users, enter:

   ```
   mmedquota -t -u
   ```

   You receive the following prompt in your default editor:

   ```
   *** Edit grace times:
   Time units may be : days, hours, minutes, or seconds
   Grace period before enforcing soft limits for USRs:
   gpfs0: block grace period: 7 days, file grace period: 7 days
   ```

3. To set the quotas for the group, blueteam, to the prototype values in modelteam, enter:

   ```
   mmedquota –g –p modelteam blueteam
   ```

Related Information

“mmcheckquota” on page 90

“mmlsquota” on page 129

“mmquotaon” on page 135

“mmquotaoff” on page 133

“mmrepquota” on page 137

mmfsck

Purpose

Checks and repairs GPFS file system.

Syntax

mmfsck *Device* [**-n** | **-y**] [**-c**] [**-o**] [**-t** *Directory*]

The file system must be unmounted before you can run **mmfsck** with any option other than **-o**.

Operands

Device
The name of the GPFS file system to be checked, for example, **fs0**. File system names need not be fully qualified. **fs0** is as acceptable as /**dev**/**fs0**.

Note: The file system must be unmounted before **mmfsck** can repair file system inconsistencies.

Options

-c When the file system log has been lost and the file system is replicated, this option specifies that **mmfsck** attempt corrective action by comparing the replicas of metadata and data. If this error condition occurs, it is indicated by an error log entry.

-n Specifies a **no** response to all prompts from **mmfsck**. **-n** reports inconsistencies but it does not change the file system. To save this information, redirect it to an output file when you enter the **mmfsck** command. If neither **-n** nor **-y** is specified, **mmfsck** runs interactively, prompts you for permission to repair each consistency error as reported.

Recommendation: In all but the most severely damaged file systems, you should run interactively (the default).

-y Specifies a **yes** response to all prompts from **mmfsck**. Use this option only on severely damaged file systems. It allows **mmfsck** to take any action necessary for repairs. If you do not enter **-n** or **-y**, you are prompted for permission to repair each consistency error as reported.

-o Specifies that the file system can be mounted during the operation of the **mmfsck** command. Online mode does not perform a full file system consistency check, but blocks marked as allocated that do not belong to a file are recovered.

-t *Directory* Specifies the directory to be used for temporary storage during **mmfsck** processing. The default directory is /**tmp**. The minimum space required is equal to 8 bytes x the maximum number of i-nodes in the file system.

Description

The occurrence of input/output errors, or the appearance of a message telling you to run the **mmfsck** command, may indicate file system inconsistencies. Should either situation occur, use the **mmfsck** command to check file system consistency and interactively repair the file system.

mmfsck checks for the following inconsistencies:

- Blocks marked allocated that do not belong to any file. The corrective action is to mark the block free in the allocation map.
- Files for which an i-node is allocated and no directory entry exists (orphaned files). The corrective action is to create directory entries for these files in a **lost+found** subdirectory at the root of this file system. The index number of the i-node is assigned as the name. If you do not allow **mmfsck** to reattach an orphaned file, it asks for permission to destroy the file.
- Directory entries pointing to an i-node that is not allocated. The corrective action is to remove the directory entry.
- Ill-formed directory entries. A directory file contains the i-node number and the generation number of the file to which it refers. When the generation number in the directory does not match the generation number stored in the file's i-node, the corrective action is to remove the directory entry.
- Incorrect link counts on files and directories. The corrective action is to update them with accurate counts.
- Cycles in the directory structure. The corrective action is to break any detected cycles. If the cycle was a disconnected cycle, the new top level directory is moved to the **lost+found** directory.

Results

If the file system is inconsistent, **mmfsck** displays information about the inconsistencies and (depending on the option entered) may prompt you for permission to repair them. **mmfsck** tries to avoid actions that may result in loss of data. In some cases, however, it may recommend the destruction of a damaged file.

If there are no file system inconsistencies to detect, **mmfsck** reports the following information for the file system:

- Number of files
- Used blocks
- Free blocks

All corrective actions, with the exception of recovering lost disk blocks (blocks that are marked as allocated but do not belong to any file), require that the file system be unmounted on all nodes. If **mmfsck** is run on a mounted file system, it will recover lost blocks but any other inconsistencies will only be reported, not repaired.

If a bad disk is detected, **mmfsck** stops the disk and writes an entry to the error log. The operator must manually start and resume the disk when the problem is fixed.

Files

- **/etc/filesystems**
- **/etc/cluster.nodes**

Return Codes

0 Successful completion.

2 The command was interrupted before it complete checks or repairs.

4 The command changed the file system and it must now be restarted.

8 The file system contains unrepaired damage.

Restrictions

You must have root authority to run **mmfsck**.

Examples

1. To run **mmfsck** on the **fs1** file system, receive a report, but not fix inconsistencies, enter:

   ```
   mmfsck fs1 -n
   ```

   You receive information similar to the following:

   ```
   Checking "fs1"
   Checking inodes
   Checking directories and files
   Checking log files
   Processing inodes

           7680 inodes
            907   allocated
              0   repairable
              0   repaired
              0   damaged
              0   deallocated
              0   orphaned
              0   attached

         823296 subblocks
          38211   allocated
              0   unreferenced
              0   deletable
              0   deallocated


   File system is clean
   ```

   mmfsck found no inconsistencies in this file system.

2. To run **mmfsck** on the /dev/fs2 file system, receive a report, and fix inconsistencies, enter:

   ```
   mmfsck /dev/fs2 -y
   ```

   You receive information similar to the following:

```
Checking "/dev/fs2"
Checking inodes
block allocation map is modified.
         429 allocated subblocks (actual)
        1805 allocated subblocks (on disk)
Flush to disk? yes
Checking directory
Checking log files
Processing inodes
Freeing deallocated inodes
        2555 inodes
          65   allocated
           0   repairable
           0   repaired
           0   damaged
           0   deallocated
           0   orphaned
           0   attached

         46080 subblocks
           461   allocated
          4736   unreferenced
             0   deletable
             0   deallocated

            55 directory files
             0   deletable
             0   deleted
file system has been repaired
```

In this example, inconsistencies are identified and repaired.

Related Information

"mmcrfs" on page 97

"mmdelfs" on page 108

"mmdf" on page 112

"mmlsfs" on page 127

mmgetacl

Purpose

Display GPFS access control list.

Syntax

mmgetacl [**-d**] [**-o** *outfile*] *File*

Operands

File
: The path name of the file or directory for which the ACL is to be displayed. If the **-d** option is specified, *file* must contain the name of a directory.

Options

-d
: Specifies that the default ACL of a directory is to be displayed.

-o *outfile*
: The path name of a file to which the ACL is to be written.

Description

Use the **mmgetacl** command to display the ACL of a file or directory.

Return Codes

0 Successful completion.

1 A failure has occurred.

Examples

1. To display the ACL for a file named **project2.history** , enter

   ```
   mmgetacl project2.history
   ```

   Output similar to the following is returned:

   ```
   #owner:paul
   #group:design
   user::rwx
   group::r-x
   other::r-x
   ```

Related Information

"mmeditacl" on page 114

"mmdelacl" on page 104

"mmputacl" on page 131

mmlsattr

Purpose

Queries file attributes.

Syntax

mmlsattr *FileName...*

Operands

FileName
: The name of the file to be queried. You must enter at least one filename. Wildcard characters are supported for filenames, for example, **project*.sched**.

Description

Use the **mmlsattr** command to display replication attributes of a file.

Results

mmlsattr lists the following attributes of the specified file:

- Current number of copies of data in a file and the maximum value
- Number of copies of the metadata for a file and the maximum value

Files

- **/etc/filesystems**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have read access to run **mmlsattr**.

Examples

- To list the attributes of a file named **project4.sched**, enter:

  ```
  mmlsattr /mmfs/fs0/project4.sched
  ```

  The following information is returned:

  ```
  replication factors
  metadata(max) data(max)  file
  ------------- ---------  ----
        1 (  2)   1 (  2)  /mmfs/fs0/project4.sched
  ```

- To show the attributes for all files in file system **fs0**, enter:

  ```
  mmlsattr /mmfs/fs0/*
  ```

  The following information is returned:

```
replication factors
metadata(max) data(max)   file
------------- ---------   ----
      1 (  2)   1 (  2)   /mmfs/fs0/project4.sched
      1 (  2)   1 (  2)   /mmfs/fs0/project4.hist
      1 (  2)   1 (  2)   /mmfs/fs0/project5.plan
```

- To show the attributes for all files in file system **fs1**, enter:

  ```
  mmlsattr /mmfs/fs1/*
  ```

  The following information is returned:

  ```
  replication factors
  metadata(max) data(max)  file
  ------------- ---------  ----
        2 (  2)   1 (  2)  /mmfs/fs1/groupA.resource
        2 (  2)   1 (  2)  /mmfs/fs1/groupA.budget
        2 (  2)   1 (  2)  /mmfs/fs1/groupA.oview
  ```

Related Information

To query file replication attributes through SMIT panels, see “Querying File Replication Using mmlsattr” on page 48.

“mmchattr” on page 81

mmlsdisk

Purpose

Displays current state of disks in file system.

Syntax

mmlsdisk *Device* [**-d** *DiskName* ...]

Options

-d *DiskName*

The IBM Virtual Shared Disk name of the disk for which you want to display the state. When you enter multiple *DiskNames*, you must separate them with semicolons and enclose the entire string of disk names in quotation marks (for example "gpfs3n12;gpfs4n12;vsdx5n12").

Operands

Device

The name of the file system to which the disk(s) belong, for example, **fs0**. **fs0** and /**dev**/**fs0** are equally acceptable.

mmlsdisk can be run against a mounted or unmounted file system.

Description

Use the **mmlsdisk** command to display the current state of the disks in the file system.

For each disk in the list, **mmlsdisk** shows:

- Current Status (whether the disk is suspended or replaced, for example)
- Availability (whether the disk is stopped)

Return Codes

0 Successful completion.

1 A failure has occurred.

Examples

1. To display the current state of gpfs3n12, enter:

   ```
   mmlsdisk /dev/fs0 -d gpfs3n12
   ```

   Information similar to the following is displayed:

```
disk         driver     sector     failure      holds         holds       status     availability
name         type       size       group        metadata      data
gpfs3n12     disk       512        -1           yes           yes         ready      up
```

2. To display the current states of gpfs3n12, gpfs4n12, and vsdx5n12, enter:

   ```
   mmlsdisk /dev/fs0 -d "gpfs3n12;gpfs4n12;vsdx5n12"
   ```

```
disk         driver     sector     failure     holds        holds      status     availability
name         type       size       group       metadata     data
gpfs3n12     disk       512        -1          yes          yes        ready      up
gpfs4n12     disk       512        -1          yes          yes        ready      up
vsdx5n12     disk       512        -1          yes          yes        ready      up
```

Related Information

To display the state of file system disks through SMIT panels, see “Display Disk States Using SMIT” on page 55.

“mmadddisk” on page 74

“mmchdisk” on page 86

“mmdeldisk” on page 105

“mmrpldisk” on page 141

mmlsfs

Purpose

Displays file system attributes.

Syntax

mmlsfs *Device* [**-a**] [**-B**] [**-d**] [**-f**] [**-i**] [**-l**] [**-m**] [**-M**] [**-n**] [**-Q**] [**-r**] [**-R**] [**-s**]

Options

-a Display the estimated average file size, in bytes

-B Display the block size, in bytes

-d Display the names of all disks in the file system

-f Display the minimum fragment size, in bytes

-i Display i-node size, in bytes

-l Display the indirect block size, in bytes

-m Display the default number of replicas for metadata

-M Display the maximum number of replicas for metadata

-n Display the estimated number of nodes that will mount file system

-Q Display which quotas are currently enforced on the file system

-r Display the default number of replicas for data

-R Display the maximum number of replicas for data

-s Display the stripe method for this file system

Operands

Device

The name of the file system, for example, **fs0**. **fs0** and /**dev**/**fs0** are equally acceptable.

Description

Use the **mmlsfs** command to list the attributes of a file system.

Results

If you do not specify any options, all attributes of the file system are listed. When you use options, only those attributes specified are listed, in the order issued in the command. Some parameters are preset for optimum performance and, although they display in **mmlsfs** output, you cannot change them.

Return Codes

0 Successful completion.

1 A failure has occurred.

Examples

1. If you issue the **mmlsfs** command with no options, for example:

   ```
   mmlsfs gpfssa
   ```

   The following information is returned:

```
| flag value          description
| ---- -------------- ----------------------------------------------------
|  -s  roundRobin     Stripe method
|  -f  8192           Minimum fragment size in bytes
|  -i  512            Inode size in bytes
|  -I  16384          Indirect block size in bytes
|  -m  2              Default number of metadata replicas
|  -M  2              Maximum number of metadata replicas
|  -r  1              Default number of data replicas
|  -R  1              Maximum number of data replicas
|  -a  102400         Estimated average file size
|  -n  8              Estimated number of nodes that will mount file system
|  -d  g05ssa06vsd1n5;g07ssa08vsd1n7;g05ssa06vsd2n5;g07ssa08vsd2n7;g05ssa06vsd3n5;
| g07ssa08vsd3n7;g05ssa06vsd4n5;g07ssa08vsd4n7;g05ssa06vsd5n5;g07ssa08vsd5n7;
| g05ssa06vsd6n5;g07ssa08vsd6n7  Disks in file system
|  -B  262144         Block size
|  -Q  none           Quotas enforced
```

Note: Output appears in the order specified in the command.

Related Information

"mmcrfs" on page 97

"mmchfs" on page 92

"mmdelfs" on page 108

mmlsquota

Purpose

Displays quota information for a user or group.

Syntax

mmlsquota [**-u** *User* | **-g** *Group*] [**-v** | **-q**] [**-e**]

Options

-g *Group* Displays the quotas of the user group specified in the *Group* parameter.

-u *User* Displays user quotas for name or user id specified in the *User* parameter.

If neither **-g** nor **-u** is specified, **mmlsquota** displays quota data for the user who entered the command.

-e Specifies that **mmlsquota** is to collect updated quota usage data from all nodes before displaying results. If this option is not specified, the command displays the last quota data collected by the quota server.

-q Prints a terse message containing only information about file systems with usage over quota.

-v Displays quotas on file systems with no allocated storage.

Description

The **mmlsquota** command displays information about quota limits and current usage. The default is to display only user quotas for the user who issues the command.

Files

- *filesystem_mountpoint*/**user.quota**
- *filesystem_mountpoint*/**group.quota**

Return Codes

0 Successful completion.

1 A failure has occurred.

Examples

1. To display his own quota data, user paul enters:

   ```
   mmlsquota
   ```

   Information similar to the following is displayed:

```
                         Block Limits                                    File Limits
Name type    KB       quota    limit in_doubt    grace    files   quota    limit in_doubt    grace
paul USR     16384    17920    19968 16384       none     2       45       50    3           none
```

This output shows the quotas for user paul set to a soft limit of 17920KB, and a hard limit of 19968KB. 16384KB is currently allocated to him. 16384KB is also in doubt, meaning that the quota manager has not yet been updated as to whether this space has been used by the nodes, or whether it is still available.

The soft limit for files (i-nodes) is set at 45 and the hard limit is 50. Two files are currently allocated to this user, and the quota manager does not yet know whether the three in doubt have been used or are still available.

No grace period appears because the user has not exceeded his quotas. If he had exceeded 17920KB or 45 files, the grace period would be set and the user would have that amount of time to bring his usage below the quota values. If he failed to do so, he would not be allocated any more space.

Related Information

“mmcheckquota” on page 90

“mmedquota” on page 116

“mmrepquota” on page 137

“mmquotaon” on page 135

“mmquotaoff” on page 133

mmputacl

Purpose

Set GPFS access control list.

Syntax

mmputacl [**-d**] [**-i** *Infile*] *File*

Operands

File
: The path name of the file or directory for which the ACL is to be set. If the -d option is specified, *file* must contain the name of a directory.

Options

-d
: Specifies that the default ACL of a directory is to be set.

-i *Infile*
: The path name of a source file from which the ACL is to be read.

Description

Use the **mmputacl** command to set the ACL of a file or directory.

Note: If the **-i** option is not used, the command expects the input to be supplied through standard input, and will await your response to the prompt.

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must be the file or directory owner or have root authority to run **mmputacl**.

Examples

1. To use the entries in a file named **standard.acl** to set the ACL for a file named **project2.history**, enter:

   ```
   mmputacl –i standard.acl project2.history
   ```

   where **standard.acl** contains the following:

   ```
   user::rwx
   group::rwx
   other::--x
   mask::rw-
   user:alpha:rwx
   group:audit:rwx
   group:system:-w-
   ```

Related Information

“mmeditacl” on page 114

“mmdelacl” on page 104

“mmgetacl” on page 122

mmquotaoff

Purpose

Deactivates quota limit checking.

Syntax

mmquotaoff [**-u** | **-g**] [**-v**] {**-a** | *Device* ...}

Options

-a Deactivates all GPFS file system quotas, as indicated in the **/etc/filesystems** file. When used in combination with the **-g** option, only group quotas listed in **/etc/filesystems** are deactivated. When used in combination with the **-u** option, only user quotas listed in **/etc/filesystems** are deactivated.

-g Specifies that only group quotas are to be deactivated.

-u Specifies that only user quotas are to be deactivated.

-v Prints a message for each file system in which quotas are deactivated.

Operands

Device Specifies the device name of the file system. **fs0** is as acceptable as **/dev/fs0**.

Description

Before issuing the **mmquotaoff** command to turn quota checking off, the file system must be mounted.

If neither the **-u** nor the **-g** option is specified, **mmquotaoff** disables both.

If the **-a** option is not specified, *device* must be the last parameter entered.

Files

- *filesystem_mountpoint*/**user.quota**
- *filesystem_mountpoint*/**group.quota**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have root authority to run **mmquotaoff**.

Examples

1. To deactivate user quotas on file system **fs0**, enter:

   ```
   mmquotaoff –u /dev/fs0
   ```

2. To deactivate group quotas on all read-write file systems, enter:

   ```
   mmquotaoff –g –a
   ```

3. To deactivate all quotas on fs0, enter:

   ```
   mmquotaoff fs0
   ```

Related Information

“mmcheckquota” on page 90

“mmedquota” on page 116

“mmlsquota” on page 129

“mmquotaon” on page 135

“mmrepquota” on page 137

mmquotaon

Purpose

Activates quota limit checking.

Syntax

mmquotaon [**-u** | **-g**] [**-v**] {**-a** | *Device* ...}

Options

-a Activates all GPFS file system quotas, as indicated in the **/etc/filesystems** file. When used in combination with the **-g** option, only group quotas listed in **/etc/filesystems** are activated. When used in combination with the **-u** option, only user quotas listed in **/etc/filesystems** are activated.

-g Specifies that only group quotas are to be activated.

-u Specifies that only user quotas are to be activated.

-v Prints a message for each file system in which quotas are activated.

Operands

Device Specifies the device name of the file system. **fs0** is as acceptable as **/dev/fs0**.

Description

Before issuing the **mmquotaon** command to turn quota checking on, the file system must be mounted.

If neither the **-u** nor the **-g** option is specified, **mmquotaon** enables both.

If the **-a** option is not used, *device* must be the last parameter specified.

Files

- *filesystem_mountpoint*/**user.quota**
- *filesystem_mountpoint*/**group.quota**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have root authority to run **mmquotaon**.

Examples

1. To activate user quotas on file system **fs0**, enter:

   ```
   mmquotaon –u /dev/fs0
   ```

2. To activate group quotas on all read-write file systems, enter:

   ```
   mmquotaon –g –a
   ```

3. To activate all quotas on fs0, enter:

   ```
   mmquotaon fs0
   ```

Related Information

"mmcheckquota" on page 90

"mmedquota" on page 116

"mmlsquota" on page 129

"mmquotaoff" on page 133

"mmrepquota" on page 137

mmrepquota

Purpose

Displays file system user and group reports.

Syntax

mmrepquota [**-e**] [**-g**] [**-q**] [**-u**] [**-v**] {**-a** | *Device*...}

Options

-a Lists quotas for all file systems.

-e Specifies that **mmrepquota** is to collect updated quota usage data from all nodes before displaying results. If this option is not specified, the command displays the last quota data collected by the quota server.

-g List only group quotas.

-q Show whether quota enforcement is active.

-u List only user quotas.

-v Print a header line.

Operands

Device Device name of the file system to be listed. **fs0** is as acceptable as /**dev**/**fs0**.

Description

The **mmrepquota** command lists (reports on) users and groups in a file system. If neither the **-g** nor the **-u** option is specified, then both user and group quotas are listed.

If the **-a** option is not specified, *device* must be the last parameter entered.

Files

- *filesystem_mountpoint*/**user.quota**
- *filesystem_mountpoint*/**group.quota**

Return Codes

0 Successful completion.

1 A failure has occurred.

Examples

1. To report on user quotas for file system **fs0**, enter:

   ```
   mmrepquota –u fs0
   ```

2. To report on the group quotas for all file systems, enter:

   ```
   mmrepquota –g –a
   ```

3. To report on quota enforcement for **fs2**, enter:

```
mmrepquota –q fs2
```

Related Information

“mmcheckquota” on page 90

“mmedquota” on page 116

“mmlsquota” on page 129

“mmquotaoff” on page 133

“mmquotaon” on page 135

mmrestripefs

Purpose

Rebalances or restores the replication factor of all files in a file system.

Syntax

mmrestripefs *Device* {**-m** | **-r** | **-b**}

Options

-b Rebalances all files across all disks that are not suspended, even if they are stopped. Although blocks are allocated on a stopped disk, they are not written to a stopped disk, nor are reads allowed from a stopped disk, until that disk is started and replicated data is copied onto it. **mmrestripefs** rebalances and restripes according to the stripe method for this file system. Use this option to rebalance the file system after adding, changing, or deleting disks in a file system.

-m Migrates all critical data off any suspended disk in this file system. Critical data is all data that would be lost if currently suspended disks were removed.

-r Migrates all data off suspended disks. It also restores all replicated files in the file system to their designated degree of replication when a previous disk failure or removal of a disk has made some replica data inaccessible. Use this parameter either immediately after a disk failure to protect replicated data against a subsequent failure, or before taking a disk off-line for maintenance to protect replicated data against failure of another disk during the maintenance process.

Operands

Device Device name of the file system to be restriped. **fs0** is as acceptable as /**dev**/**fs0**.

Description

Use the **mmrestripefs** command to rebalance or restore the replication factor of all files in a file system. **mmrestripefs** moves existing file system data between different disks in the file system based on changes to the disk state made by the **mmchdisk**, **mmadddisk**, and **mmdeldisk** commands.

mmrestripefs attempts to restore the metadata or data replication factor of any file in the file system.

You must specify one of the three options (**-b**, **-m**, or **-r**) to indicate how much file system data to move. You can issue this command against a mounted or unmounted file system.

Files

/etc/filesystems

Return Codes

0	Successful completion.
1	A failure has occurred.

Restrictions

You must have root authority to run **mmrestripefs**.

Examples

1. To restripe and move all critical data from any suspended disk in file system **fs0**, enter:

   ```
   mmrestripefs fs0 -m
   ```

   This command also rebalances the data in the file system.

   Messages similar to these are generated:

   ```
   MMFS: 6027-560 Scanning file system metadata ...
   MMFS: 6027-565 Scanning user file metadata ...
   3 % completed
       .
       .
       .
   100% completed
   MMFS: 6027-552 Scan completed successfully.
   ```

2. To rebalance all files in file system **fs0** across all defined, accessible disks that are not stopped or suspended, enter:

   ```
   mmrestripefs fs0 -b
   ```

   Messages are generated similar to those in the first example.

Related Information

"mmadddisk" on page 74

"mmchdisk" on page 86

"mmdeldisk" on page 105

"mmrpldisk" on page 141

mmrpldisk

Purpose

Replaces specified disk.

Syntax

mmrpldisk *Device OldDiskName DiskDesc* [**-n**] [**-v** **yes** | **no**]

Operands

Device

The name of the file system device where the disk is to be replaced. File system names need not be fully qualified. **fs0** is as acceptable as /**dev**/**fs0**.

OldDiskName

The name of the disk to be replaced. This can be a name previously generated by the **mmcrfs**, **mmadddisk**, **mmrpldisk** command, or one that was passed to one of these commands. You can have the entire list of disk names displayed using the **mmlsdisk** command.

DiskDesc

A descriptor for the replacement IBM Virtual Shared Disk. The disk descriptor contains the following positional parameters that apply to the IBM Virtual Shared Disk, separated by colons (:)

Disk Name

The device name, for example, `hdisk1`, of any disk you wish to use to create an IBM Virtual Shared Disk, or the name of an existing IBM Virtual Shared Disk.

GPFS performance and recovery processes function best with one disk per IBM Virtual Shared Disk. If you want to create IBM Virtual Shared Disks with more than one disk, refer to *Managing Shared Disks*. Then pass the existing IBM Virtual Shared Disks to GPFS using the VSD names.

Server Name

The server name, when creating an IBM Virtual Shared Disk. This can be in any recognizable form.

Backup Server Name

The backup server name, when creating an IBM Virtual Shared Disk. This can be in any recognizable form.

Disk Usage

Specify the same as the disk being replaced or omit parameter and allow to default to the value of the original disk:

- **dataAndMetadata**
- **dataOnly**
- **metadataOnly**

.

Failure Group
A number identifying the failure group to which this disk belongs. Specify the same as the disk being replaced, or omit and allow to default to the value of the original disk.

Any parameter that is intentionally omitted or allowed to default must be explicitly skipped. Trailing delimiters are acceptable.

Options

-n Preserve the deleted IBM Virtual Shared Disks. If this option is not specified, any VSD generated by a disk descriptor is deleted.

-v yes | **no** Verify that the new IBM Virtual Shared Disk does not belong to an existing file system. The default is **yes**. Specify **no** only when you want to reuse a disk that is no longer needed for an existing file system.

Description

Use the **mmrpldisk** command to replace an existing disk in the GPFS file system with a new one. All data on the old disk is migrated to the new one.

Notes:

1. To create a replacement IBM Virtual Shared Disk and add it to the file system, you must specify the Disk Name and Server Name parameters in the descriptor. The remaining positional parameters can be allowed to default.
2. You cannot replace a disk when it is the only remaining disk in the file system.
3. The replacement disk must have the same values for the Disk Usage and Failure Group parameters as the one it replaces. This means that if the old disk took the default failure group, the replacement disk must also take the default and have the same server.

The file system need not be unmounted before the **mmrpldisk** command can be run.

Results

Upon successful execution of **mmrpldisk** the following tasks are completed:

- IBM Virtual Shared Disk described in the descriptor is created and configured.
- /**etc**/**filesystems** is updated on all nodes in SP configuration to reflect disk replacement.
- The disk is replaced in the file system and data is copied to the new disk without restriping.
- SDR version of file systems stanza is updated with new /**etc**/**filesystems** data.
- Old disk is deleted, if so specified.

Files

- /**etc**/**filesystems**

Return Codes

0 Successful completion.

1 A failure has occurred.

Restrictions

You must have root authority to run **mmrpldisk**.

Examples

1. To replace disk gpfs1n12 in **fs0** with a new IBM Virtual Shared Disk consisting of new disk hdisk48:

   ```
   mmrpldisk fs0 gpfs1n12 hdisk48:node12:::
   ```

   This command creates a new IBM Virtual Shared Disk with node12 as the server and no backup. The Disk Usage and the Failure Group parameters default to the corresponding values of gpfs1n12.

Related Information

To replace a disk through SMIT panels, see “Replacing Disks Using SMIT” on page 52.

“mmadddisk” on page 74

“mmchdisk” on page 86

“mmlsdisk” on page 125

“mmrestripefs” on page 139

mmrpldisk

Appendix A. Considerations for GPFS Applications

GPFS is designed so that most applications written to the X/Open standard for file system calls can access GPFS data with no modification, however, there are some exceptions.

Exceptions to X/Open Standards

The calls that provide the capability to use memory mapped files are not supported in this release. These include

1. **mmap**
2. **munmap**
3. **msync**
4. **shmat**

IBM intends to support these calls in a future release of GPFS.

Applications that depend on exact reporting of changes to the following fields returned by the **stat** call may not work as expected:

1. **mtime**
2. **ctime**
3. **atime**

Providing exact support for these fields would require significant performance degradation to all applications executing on the system. These fields are guaranteed accurate when the file is closed.

Application Support

Applications access GPFS data through the use of standard AIX system calls and libraries. Support for larger files is provided through the use of AIX 64–bit forms of these libraries. See the AIX product documentation for details.

This release of GPFS does not provide any extensions to the standard calls.

© Copyright IBM Corp. 1998

Glossary

F

failover. The assuming of server responsibilities by the node designated as backup server, when the primary server fails.

failure group. A collection of disks that share common access paths or adaptor connection, and could all become unavailable through a single hardware failure.

fragment. The space allocated an amount of data (usually the end of a file) too small to require a full block, consisting of one or more subblocks (one thirty-second of block size).

I

IBM Virtual Shared Disk. A subsystem that allows application programs executing on different nodes access a raw logical volume as if it were local at each node.

i-node. The internal structure that describes an individual file to AIX. An i-node contains file size and update information, as well as the addresses of data blocks, or in the case of large files, indirect blocks that, in turn, point to data blocks. One i-node is required for each file.

J

journaled file system. The local file system within a single instance of AIX.

L

Logical Volume Manager. Manages disk space at a logical level. It controls fixed-disk resources by mapping data between logical and physical storage, allowing data to be discontiguous, span multiple disks, replicated, and dynamically expanded.

M

metadata. Data structures that contain access information about file data. These might include inodes, indirect blocks, and directories. These data structures are used by GPFS but are not accessible to user applications.

mirroring. The creation of a mirror image of data to be preserved in the event of disk failure.

P

primary server. When physical disks are connected to two nodes (twin-tailed), this is the node that normally maintains and controls local access to the disk.

Q

quorum. The minimum number of nodes that must be operational in order for GPFS to allow file system access. This is one node more than half the nodes in the GPFS configuration.

quota. The amount of disk space and number of i-nodes assigned as upper limits for a specified user or group of users.

R

RAID. Redundant Array of Independent Disks. A set of physical disks that are act as a single physical volume and use parity checking to protect against disk failure.

recovery. The process of restoring access to file system data when a failure has occurred. This may involve reconstructing data or providing alternative routing through a different server.

replication. The practice of creating and maintaining multiple file copies to ensure availability in the event of hardware failure.

S

SSA. Serial Storage Architecture. An expanded storage adapter for multi-processor data sharing in UNIX-based computing, allowing disk connection in a high-speed loop.

SCSI. Small Computer Systems Interface. An adapter supporting attachment of various direct-access storage devices.

secondary server. The second node connected to a twin-tailed disk. This node assumes control of local access if the primary server fails.

stripe group. A file system written across many disks, which are connected to multiple nodes.

striping. A method of writing a file system, in parallel, to multiple disks instead of to single disks in a serial operation.

© Copyright IBM Corp. 1998

sub-block. The smallest unit of data accessible in an I/O operation, equal to one thirty-second of a data block.

T

token management. A system for controlling file access in which each application performing a read or write operation is granted exclusive access to a specific block of file data. This ensures data consistency and controls conflicts.

Index

A

access control lists 65
  changing 67
  creating 131
  deleting 68, 104
  displaying 67, 122
  editing 114
  setting 66
activate quotas 63, 135
add disk 50, 74
add node 57, 78
administration 3, 29
applications 9, 12, 145
autoload 6
automount 18

B

backup server 15
block size 8
  tuning 69
buddy buffers 12

C

cache 6
change configuration 83
change disk 86
change disk state 55
change file system attributes 92
 change quota 61
change replication 81
changing access control lists 67
changing replication 48
check file system 43, 118
check quotas 62, 90
client_ports 70
configuration 30
  changing 33, 83
  sample 18
  setting 94
configuring GPFS 5
consistency 2
creat file system 97
create quota reports 64, 137
creating access control lists 131

D

data availability 2
dataStructureDump 83
deactivate quotas 64, 133
delete disk 51, 105
delete file system 43, 108
delete node 58, 110
deleting access control lists 68, 104
disk
  adding 50, 74
  changing 86
  deleting 51, 105
  name 35
  replacing 52, 141
  state
    changing 55
    displaying 53, 125
  usage 17, 36
disk descriptors
  building 35, 36
disk failure 14
disks 13
  connectivity 13
  free space 13
display disk state 53, 125
display file system attributes 127
displaying access control lists 67, 122

E

edit quota 116
editing access control lists 114
establish quota 61
exceeded quota 65
exceptions 145

F

failure
  disk 14
  group 17, 36
  node 14
  server 14
failure group 16, 17
file size
  maximum 9
file system
  access control 65
  attributes
    changing 92
    displaying 127
    listing 45
    modifying 46
  checking 43, 118
  creating 35, 97

© Copyright IBM Corp. 1998

file system *(continued)*
deleting 43, 108
mounting 41
repairing 44
restriping 48, 139
sample 19
space
querying 49, 112
unmounting 42
file system size 8
maximum 10
files
maximum 10
fragment 8

I

i-node 9
i-nodes
maximum 10
IBM Recoverable Virtual Shared Disks 12
IBM Virtual Shared Disk
tune 26
tuning 71
IBM Virtual Shared Disks
defining 13
sample 20
images 24
indirect block 9
installation 23

K

kerberos authentication 29

L

list attributes 45
list quotas 62, 129

M

malloc size 70
mallocsize 6
maxFilesToCache 7, 83
maximum
file size 9
file system size 10
number files 10
memory
tuning 69
mirroring 15, 16
mmadddisk 74
mmaddnode 78
mmchattr 81
mmchconfig 83
mmchdisk 86
mmcheckquota 90
mmchfs 92
mmconfig 94
mmcrfs 97
mmdelacl 104
mmdeldisk 105
mmdelfs 108
mmdelnode 110
mmdf 112
mmeditacl 114
mmedquota 116
mmfsck 118
mmgetacl 122
mmlsattr 123
mmlsdisk 125
mmlsfs 127
mmlsquota 129
mmputacl 131
mmquotaoff 133
mmquotaon 135
mmrepquota 137
mmrestripefs 139
mmrpldisk 141
modify attributes 46
mount file system 41

N

node
adding 57, 78
deleting 58, 110
node failure 14
nodes, count 5
nodes, list 5
nodes, planning 5

P

page pool 69
pagepool 6
path 23
performance 1, 7
performance tuning 69
primary server 15
priority 7, 70

Q

query space 49, 112
querying replication 47, 123
quorum 6
quotas
activating 63, 135
changing 61, 116
checking 62, 90

quotas *(continued)*
deactivating 64, 133
establishing 61, 116
exceeding 65
installing 60
listing 62, 129
managing 59
reports
creating 64, 137
uninstalling 61

R

RAID 8, 14, 16, 17
recoverability 13, 14, 17
Redundant Array of Independent Disks 14
repair file system 44
replace disk 52, 141
replication 9, 15, 17
changing 48, 81
querying 47, 123
requirements 23
restripe file system 48, 139
root authority 29

S

server
name 35
server failure 14
server_kprocs 71
server_port_number 71
servers 12
set configuration 94
setting access control lists 66
size, file system 8
SMIT 29
standards
exceptions 145
starting 32
starting automatically 6
stopping 33
stripe method 18
subblock 8
switch 26
tuning 71
sysctl 26

T

token management 70
tuning 69
twin-tailed disk 16
twin-tailed disks 15

U

unmount file system 42

Communicating Your Comments to IBM

General Parallel File System for AIX
Installation and Administration

Publication No. SA22-7278-01

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

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

If you are mailing an RCF from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF at the back of this book.
- If you prefer to send comments by FAX, use this number:
  - FAX: (International Access Code)+1+914+432-9405
- If you prefer to send comments electronically, use this network ID:
  - IBMLink: (United States customers only): KGNVMC(MHVRCFS)
  - IBM Mail Exchange: USIB6TC9 at IBMMAIL
  - Internet e-mail: mhvrcfs@us.ibm.com
  - World Wide Web: http://www.s390.ibm.com/os390

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies

Optionally, if you include your telephone number, we will be able to respond to your comments by phone.

Reader's Comments — We'd Like to Hear from You

General Parallel File System for AIX
Installation and Administration

Publication No. SA22-7278-01

You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

Today's date: _____

What is your occupation?

Newsletter number of latest Technical Newsletter (if any) concerning this publication:

How did you use this publication?

[ ]	As an introduction	[ ]	As a text (student)
[ ]	As a reference manual	[ ]	As a text (instructor)
[ ]	For another purpose (explain)		

_____

_____

Is there anything you especially like or dislike about the organization, presentation, or writing in this manual? Helpful comments include general usefulness of the book; possible additions, deletions, and clarifications; specific errors and omissions.

Page Number: Comment:

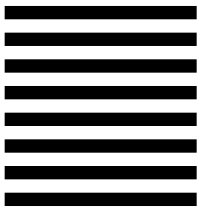
Name	Address
Company or Organization	
Phone No.	

Reader's Comments — We'd Like to Hear from You
SA22-7278-01

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department 55JA, Mail Station P384
522 South Road
Poughkeepsie NY 12601-5400

Fold and Tape **Please do not staple** Fold and Tape

Cut or Fold
Along Line

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SA22-7278-01