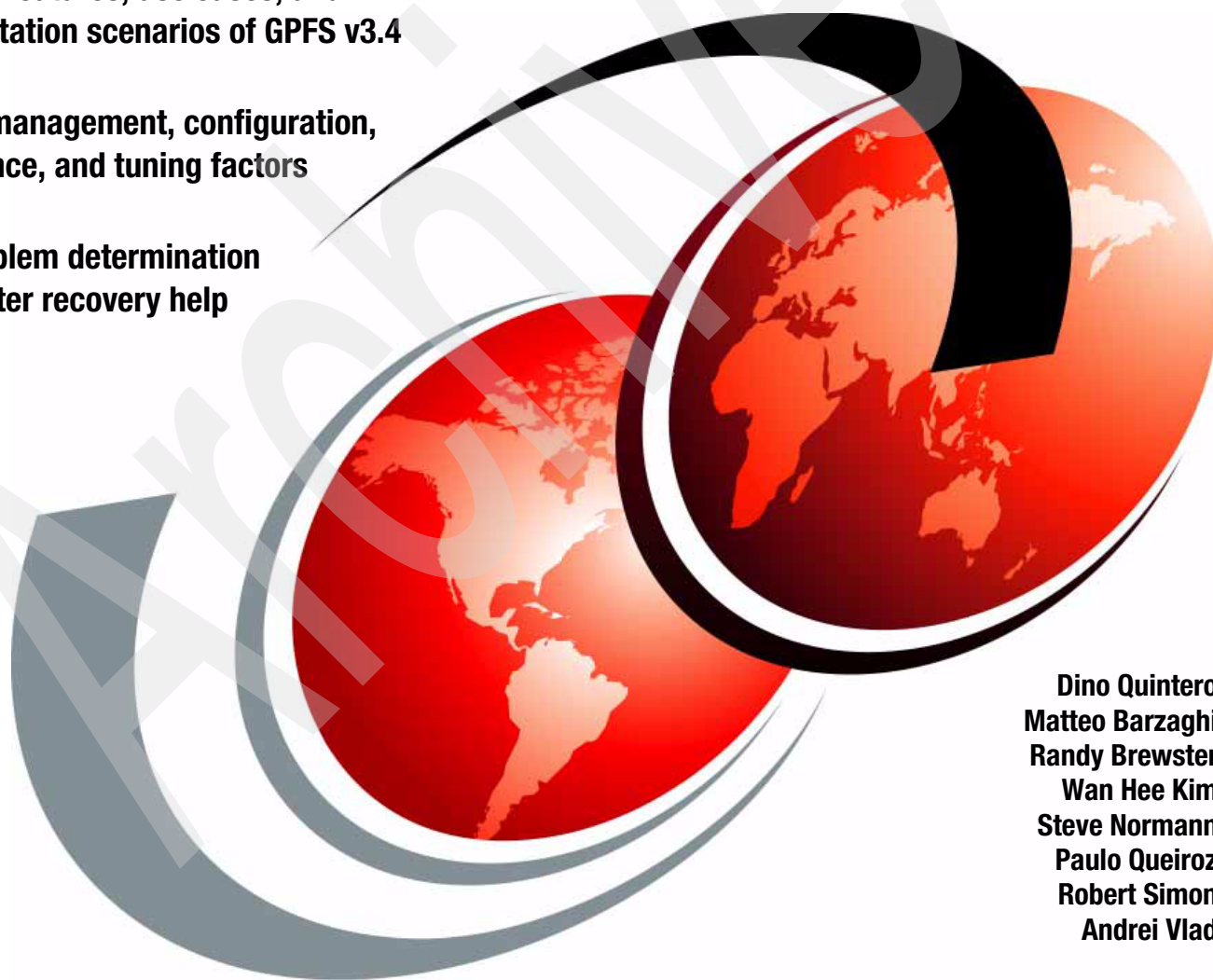


Implementing the IBM General Parallel File System (GPFS) in a Cross-Platform Environment

Describes features, use cases, and implementation scenarios of GPFS v3.4

Explains management, configuration, performance, and tuning factors

Gives problem determination and disaster recovery help



Dino Quintero
Matteo Barzaghi
Randy Brewster
Wan Hee Kim
Steve Normann
Paulo Queiroz
Robert Simon
Andrei Vlad

Redbooks



International Technical Support Organization

**Implementing the IBM General Parallel File System
(GPFS) in a Cross-Platform Environment**

June 2011

Archived

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

First Edition (June 2011)

This edition applies to IBM AIX 6.1 TL05, IBM Virtual IO Server 2.1.3.10-FP23, IBM General Parallel File System 3.4.0.1, RedHat Enterprise Linux 5.5.

Contents

| | |
|---|------|
| Notices | vii |
| Trademarks | viii |
| Preface | ix |
| The team who wrote this book | ix |
| Now you can become a published author, too! | xi |
| Comments welcome | xi |
| Stay connected to IBM Redbooks | xii |
| Chapter 1. Introduction | 1 |
| 1.1 Overview and strategy | 2 |
| 1.2 Features | 2 |
| 1.3 Licensing | 3 |
| 1.4 Operating systems support | 3 |
| 1.5 Hardware support | 3 |
| 1.6 Contact information | 4 |
| Chapter 2. Infrastructure planning | 5 |
| 2.1 Network design | 6 |
| 2.1.1 The 1 Gigabit Ethernet and 10 Gigabit Ethernet | 6 |
| 2.1.2 InfiniBand (IP over IB/RDMA) | 6 |
| 2.1.3 Configuring the NIC | 9 |
| 2.1.4 IBM data center networking (DCN) products | 10 |
| 2.2 Storage design | 11 |
| 2.2.1 Host bus adapter (HBA) | 12 |
| 2.2.2 Multipath driver | 13 |
| 2.2.3 The characteristic of storage hardware | 16 |
| 2.2.4 Tape library | 20 |
| 2.3 InfiniBand Storage DCS9900 | 22 |
| 2.4 Industry solution design | 23 |
| 2.4.1 High-Performance Computing | 23 |
| 2.4.2 RDBMS | 25 |
| 2.4.3 Web application server | 33 |
| 2.4.4 Media contents delivery | 33 |
| 2.4.5 Commercial industry | 35 |
| 2.5 GPFS planning guidance | 38 |
| 2.5.1 Operating system and file systems with multiplatform GPFS | 40 |
| 2.5.2 Security | 45 |
| 2.5.3 High availability | 46 |
| 2.5.4 Network Shared Disk (NSD) creation considerations | 54 |
| 2.5.5 GPFS considerations | 60 |
| 2.5.6 Tivoli Storage Manager for GPFS | 67 |
| 2.6 Summary | 68 |
| Chapter 3. Scenarios | 69 |
| 3.1 ITSO laboratory environment | 70 |
| 3.1.1 Diagram of ITSO laboratory | 70 |
| 3.1.2 ITSO laboratory environment host file | 70 |
| 3.2 Three-node GPFS cluster using internal disks | 73 |

| | | |
|-------------------|---|------------|
| 3.2.1 | Requirements: Hardware, software, network, storage | 73 |
| 3.2.2 | GPFS configuration | 74 |
| 3.2.3 | Diagram of GPFS three-node cluster with internal disks | 75 |
| 3.2.4 | Setting up and configuring a three-node cluster | 75 |
| 3.3 | Recovery on the three-node cluster | 85 |
| 3.4 | Linux InfiniBand cluster with RDMA and Linux for System x clients | 90 |
| 3.4.1 | Requirements: Hardware and software | 91 |
| 3.4.2 | Ethernet GPFS cluster | 91 |
| 3.4.3 | InfiniBand GPFS cluster | 94 |
| 3.5 | Cross-platform cluster: Windows servers, Linux/AIX clients | 97 |
| 3.5.1 | Requirements: Hardware, software, network, storage | 97 |
| 3.5.2 | GPFS cluster configuration | 97 |
| 3.5.3 | GPFS cluster diagram | 98 |
| 3.5.4 | Installing Windows Server 2008 R2 | 98 |
| 3.5.5 | Install the IBM GPFS Windows version | 108 |
| 3.5.6 | GPFS Windows cluster setup | 109 |
| 3.5.7 | Simulating a GPFS failure on the cluster | 115 |
| 3.5.8 | Adding x86 Linux node to the Windows cluster | 117 |
| 3.6 | DB2 pureScale InfiniBand cluster on AIX | 120 |
| 3.6.1 | Requirements: Hardware, software, network, storage | 120 |
| 3.6.2 | GPFS configuration | 121 |
| 3.6.3 | GPFS cluster diagram | 122 |
| 3.6.4 | Cluster setup and configuration | 122 |
| 3.6.5 | Use the mmpmon tool to collect statistics from each node | 141 |
| 3.6.6 | Simulating a GPFS failure on member 0 (algeria) | 145 |
| 3.6.7 | Troubleshooting DB2 pureScale issues | 147 |
| 3.7 | Multi-cluster configuration | 149 |
| 3.7.1 | Requirements: Hardware, software, network, storage | 149 |
| 3.7.2 | GPFS configuration | 149 |
| 3.7.3 | GPFS multi-cluster diagram | 150 |
| 3.7.4 | Multi-cluster setup and configuration | 150 |
| 3.7.5 | Multi-cluster scenario | 150 |
| 3.7.6 | Add security at the GPFS communication network on Bronx cluster | 155 |
| 3.7.7 | Verify GPFS daemon is active on all nodes of both clusters | 157 |
| 3.8 | Disaster recovery using GPFS replication | 162 |
| 3.8.1 | Requirements: Hardware, software, network, storage | 162 |
| 3.8.2 | GPFS configuration | 162 |
| 3.8.3 | GPFS configuration diagram | 163 |
| 3.8.4 | Set up and configure GPFS DR cluster | 164 |
| Chapter 4. | Management and maintenance | 189 |
| 4.1 | Migration and update | 190 |
| 4.1.1 | Migrating GPFS from 3.2 or 3.3 to 3.4 | 190 |
| 4.1.2 | Applying corrective fixes to GPFS | 193 |
| 4.1.3 | Migrating data to new storage | 195 |
| 4.1.4 | Reinstalling a node (mmsrdfs) | 199 |
| 4.2 | Managing a GPFS cluster | 200 |
| 4.2.1 | Adding a node to the cluster | 200 |
| 4.2.2 | Adding a disk to a file system | 203 |
| 4.2.3 | GPFS network usage | 206 |
| 4.2.4 | Adding a remote cluster | 207 |
| 4.2.5 | Adding or removing a file system | 210 |
| 4.2.6 | Enabling GPFS replication | 213 |

| | |
|---|------------|
| 4.2.7 Exporting or importing a file system | 217 |
| 4.3 Reducing file system fragmentation: The mmdefragfs command | 222 |
| 4.4 Optimizing extended attributes: The fastea option | 224 |
| 4.5 Setting up GPFS callback functionality: The callback commands | 224 |
| 4.6 Monitoring GPFS configuration status: The SNMPD protocol | 226 |
| 4.7 SSH configuration | 227 |
| Chapter 5. Performance and tuning | 229 |
| 5.1 GPFS architecture | 230 |
| 5.2 NSD considerations | 231 |
| 5.3 GPFS parametrization | 231 |
| 5.3.1 File system parametrization | 231 |
| 5.3.2 Block allocation methods | 233 |
| 5.3.3 GPFS general parametrization | 234 |
| 5.4 Network parametrization | 237 |
| 5.5 Parametrization | 247 |
| 5.5.1 Controlling the bandwidth | 247 |
| 5.5.2 Controlling GPFS parallelism | 248 |
| 5.5.3 Linux considerations | 249 |
| 5.5.4 Storage device considerations | 250 |
| 5.6 Monitoring performance | 250 |
| Chapter 6. Problem determination | 253 |
| 6.1 Problem determination process | 254 |
| 6.1.1 Defining the problem | 254 |
| 6.1.2 Gathering information from the user | 254 |
| 6.1.3 Gathering application, GPFS, and system information | 255 |
| 6.2 GPFS debug commands | 258 |
| 6.2.1 Data collection commands | 258 |
| 6.2.2 Data analysis commands and scripts | 263 |
| 6.2.3 mmfs logs | 265 |
| 6.3 GPFS problem scenarios | 266 |
| 6.3.1 Considerations | 266 |
| 6.3.2 Scenario 1: Upgrading GPFS | 268 |
| 6.3.3 Scenario 2: Analyzing waiters | 269 |
| 6.3.4 Scenario 3: Application failure | 271 |
| 6.3.5 Scenario 4: GPFS file system hang | 273 |
| 6.3.6 Scenario 5: File system unmounting | 275 |
| Chapter 7. IBM Power Systems virtualization and GPFS | 279 |
| 7.1 IBM Power Systems (System p) | 280 |
| 7.1.1 Introduction | 280 |
| 7.1.2 Virtual I/O Server (VIOS) | 281 |
| 7.1.3 VSCSI and NPIV | 283 |
| 7.1.4 Virtual SCSI target adapters (VSCSI) | 283 |
| 7.1.5 N_Port ID Virtualization (NPIV) | 300 |
| 7.2 Shared Ethernet Adapter and Host Ethernet Adapter | 306 |
| 7.2.1 Shared Ethernet Adapter (SEA) | 306 |
| 7.2.2 Host Ethernet Adapter (HEA) | 313 |
| Chapter 8. Information lifecycle management (ILM) | 317 |
| 8.1 Explaining the ILM concept | 318 |
| 8.1.1 Snapshot management tasks | 318 |
| 8.1.2 Storage pools | 319 |

| | | |
|-------|--|------------|
| 8.1.3 | File sets | 322 |
| 8.1.4 | Policies and rules | 326 |
| 8.1.5 | ILM data flow | 328 |
| 8.2 | Tivoli Storage Manager | 329 |
| 8.2.1 | Preparing the Tivoli Storage Manager server | 330 |
| 8.2.2 | Tivoli Storage Manager installation | 332 |
| 8.2.3 | Administrator Center installation | 339 |
| 8.2.4 | Tivoli Storage Manager server configuration | 342 |
| 8.2.5 | Tivoli Storage Manager client configuration | 344 |
| 8.3 | Information lifecycle management scenarios | 346 |
| 8.3.1 | Working with snapshots | 347 |
| 8.3.2 | Working with storage pools and policies | 350 |
| 8.3.3 | Working with external storage pool and GPFS policies | 360 |
| 8.3.4 | Working with hierarchical storage management | 363 |
| 8.3.5 | Working with hierarchical storage management and GPFS policies | 367 |
| 8.3.6 | End-to-end implementation of ILM and HSM | 371 |
| 8.4 | Backup and restore | 376 |
| 8.4.1 | GPFS backup tools | 376 |
| 8.4.2 | GPFS advanced backup tools | 382 |
| 8.4.3 | Conclusions | 384 |
| 8.5 | Advanced external pool scenario | 384 |
| 8.5.1 | Off-line compressed storage | 385 |
| 8.5.2 | Implementation scenario | 385 |
| | Chapter 9. Disaster recovery using GPFS | 393 |
| 9.1 | Disaster recovery solution using GPFS replication | 394 |
| 9.1.1 | Configuration | 394 |
| 9.1.2 | Characteristics of this DR configuration | 395 |
| 9.2 | The GPFS mmfscctl command | 396 |
| | Related publications | 397 |
| | IBM Redbooks | 397 |
| | Other publications | 397 |
| | Online resources | 398 |
| | How to get Redbooks | 399 |
| | Help from IBM | 399 |
| | Index | 401 |

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

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

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.


Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AFS®
AIX 5L™
AIX®
BladeCenter®
Blue Gene®
Cognos®
DB2®
DS4000®
Enterprise Storage Server®
ESCON®
FlashCopy®
GPFS™
HACMP™

IBM®
InfoSphere™
LoadLeveler®
NetView®
POWER Hypervisor™
Power Systems™
POWER6®
PowerHA™
PowerPC®
PowerVM™
POWER®
pSeries®
pureScale™

Redbooks®
Redbooks (logo) ®
System p®
System Storage®
System x®
System z®
Tivoli®
TotalStorage®
WebSphere®
XIV®
z/OS®
zSeries®

The following terms are trademarks of other companies:

Intel Xeon, Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linear Tape-Open, LTO, Ultrium, the LTO Logo and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Microsoft, Windows NT, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Snapshot, and the NetApp logo are trademarks or registered trademarks of NetApp, Inc. in the U.S. and other countries.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbooks® publication provides a documented deployment model for IBM GPFS™ in a cross-platform environment with IBM Power Systems™, Linux, and Windows servers. With IBM GPFS, customers can have a planned foundation for file systems management for cross-platform access solutions.

This book examines the functional, integration, simplification, and usability changes with GPFS v3.4. It can help the technical teams provide file system management solutions and technical support with GPFS, based on Power Systems virtualized environments for cross-platform file systems management.

The book provides answers to your complex file systems management requirements, helps you maximize file system availability, and provides expert-level documentation to transfer the how-to skills to the worldwide support teams.

The audience for this book is the technical professional (IT consultants, technical support staff, IT architects, and IT specialists) who is responsible for providing file system management solutions and support for cross-platform environments that are based primarily on Power Systems.

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.



Front row (left to right): Wan Hee Kim, Paulo Queiroz, Andrei Vlad, Matteo Barzaghi

Back row (left to right): Randy Brewster, Dino Quintero (Project Leader), Robert Simon, Steve Normann

Dino Quintero is a technical Project Leader and a IT Generalist with the International Technical Support Organization (ITSO) in Poughkeepsie, NY. His areas of expertise include enterprise continuous availability planning and implementation, enterprise systems management, virtualization, and clustering solutions. He is currently an Open Group Master Certified IT Specialist - Server Systems. He holds a Masters degree in Computing Information Systems, and a Bachelor of Science in Computer Science from Marist College.

Matteo Barzaghi is a Support Center Representative in Italy. He has 13 years of experience at IBM, mainly working with the Power Systems software products (AIX®, PowerHA™). He worked for four years in the field for a large bank customer as an AIX System Administrator. He is a former PowerHA Virtual Front End Team member, and now works for Virtual Front End Poughkeepsie European Team supporting all products, mainly GPFS, for all Europe, Africa, and Middle East customers. He has written extensively about GPFS maintenance.

Randy Brewster is a GPFS Software Test Specialist at IBM in Poughkeepsie, New York. He has 18 years of experience in pSeries® systems and clustering software. Prior to his work on pSeries systems, he spent over 10 years on zSeries® systems. He has worked at IBM for over 28 years. His areas of expertise include clustering software, GPFS, HAGEO, HACMP™ and Blue Gene®. He has written extensively on GPFS and DB2®. He hold a Master of Science in Computer Science from the City University of New York.

Wan Hee Kim is a Technical Sales Specialist in Korea. He has 5 years of experience with IBM. He holds LPI, OCP, and IBM Certification 061,086,071. His areas of expertise include open source technology with e1350 solution, based on System x® and BladeCenter® System. In particular, he is a technical sales focal point of the GPFS solution in Korea. He has vast experience in complex GPFS solutions with various storage systems in the public and media industries. He also hosted a major IBM Business Partner Residency Program regard GPFS, xCAT, and Virtualization. Recently, his focus is on cross-platform technology, system management, and cloud computing infrastructure.

Steve Normann is a Senior Software Engineer in MTS working in Poughkeepsie, New York. He has worked with IBM since 1984. He currently is a Team Leader for GPFS in the Software Technical Support Group, which supports the High Performance Clustering software (LoadLeveler®, CSM, GPFS, RSCT, and PPE) and High Performance Clustering software.

Paulo Queiroz is an IT Specialist with IBM in Brazil. He has 10 years of experience in UNIX and Linux, ranging from systems design and development to systems support. His areas of expertise include AIX Power Systems, AIX, GPFS, RHCS, KVM, and Linux. He is a Certified Advanced Technical Expert for Power Systems with AIX (CATE) and a Red Hat Certified Engineer (RHCE).

Robert Simon is a Senior Software Engineer in STG working in Poughkeepsie, New York. He has worked with IBM since 1987. He currently is a Team Leader in the Software Technical Support Group, which supports the High Performance Clustering software (LoadLeveler, CSM, GPFS, RSCT and PPE). He has extensive experience with IBM System p® hardware, AIX, HACMP, and High Performance Clustering software. He has participated in the development of four other IBM Redbooks publications.

Andrei Vlad is an IT Specialist with IBM in Romania, working in the Server and Storage department since 2002. His areas of expertise include Linux performance and clustering, GPFS, CSM, xCAT, and HPC infrastructure. He has implemented several GPFS and CSM clusters, and provided worldwide customer training. Currently, his focus is on developing HPC courses and workshops. He is certified in AIX and Linux, and has a Masters degree in Electronic Engineering from Polytechnical University in Bucharest, Romania. He is currently pursuing a Ph.D. in Electronic Engineering.

Thanks to the following people for their contributions to this project:

David Bennin, Ella Buslovich, Richard Conway, Diane Sherman
International Technical Support Organization, Poughkeepsie Center

Gordon McPheeters, Heena Raval, Kuei-Yu Wang-Knop, Puneet Chaudhary, Scott Fadden,
Sridhar Murthy, Scott Denham
IBM USA

Octavian Lascu, Andrei Socoliuc
IBM Romania

Steve Schormann
IBM Canada

Rik Foote
IBM Australia

Andre Posthuma
IBM UK

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author - all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

ibm.com/redbooks

- Send your comments in an email to:

redbooks@us.ibm.com

- Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- ▶ Find us on Facebook:
<http://www.facebook.com/IBMRedbooks>
- ▶ Follow us on twitter:
<http://twitter.com/ibmredbooks>
- ▶ Look for us on LinkedIn:
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:
<http://www.redbooks.ibm.com/rss.html>

Introduction

This chapter contains the following topics:

- ▶ 1.1, “Overview and strategy” on page 2
- ▶ 1.2, “Features” on page 2
- ▶ 1.3, “Licensing” on page 3
- ▶ 1.4, “Operating systems support” on page 3
- ▶ 1.5, “Hardware support” on page 3
- ▶ 1.6, “Contact information” on page 4

1.1 Overview and strategy

Growth of data, transactions, and digitally-aware devices are straining IT infrastructure and operations; storage costs and user expectations are increasing. As users are added and more data is stored, file-level data availability becomes more difficult to achieve and management can become more complex.

To address these data management challenges, you need a cost-effective alternative that can help you move beyond simply adding storage to optimizing your data management. A single file server does not scale and multiple file servers are not flexible enough to provide dynamic 24x7 data access needed in today's competitive digital marketplace.

The IBM General Parallel File System (GPFS), which is a high-performance enterprise file, can help you move beyond simply adding storage to optimizing data management. GPFS is a high-performance shared-disk file management solution that provides fast, reliable access to a common set of file data, online storage management, scalable access and tightly integrated information life cycle tools capable of managing petabytes of data and billion of files.

IBM GPFS currently powers many of the world's largest scientific supercomputers and commercial applications that require high-speed access to large volumes of data such as the following examples:

- ▶ Digital media
- ▶ Engineering design
- ▶ Business intelligence
- ▶ Financial analytics
- ▶ Seismic data processing
- ▶ Geographic information systems
- ▶ Scalable file serving

1.2 Features

GPFS provides online storage management, scalable access, and integrated information life-cycle management tools that are capable of managing petabytes of data and billions of files. Virtualizing your file storage space and allowing multiple systems and applications to share common pools of storage provides you the flexibility to transparently administer the infrastructure without disrupting applications, thereby improving cost and energy efficiency, and reducing management overhead.

Massive namespace support, seamless capacity and performance scaling, along with proven reliability features and flexible architecture of GPFS helps your company foster innovation by simplifying your environment and streamlining data workflows for increased efficiency.

High-performance enterprise file management with GPFS provides the following features:

- ▶ Seamless capacity expansion to handle the explosive growth of digital information and improve efficiency through enterprise wide, interdepartmental information sharing
- ▶ High reliability and availability to eliminate production outages and provide disruption-free maintenance and capacity upgrades
- ▶ Performance to satisfy the most demanding applications

- ▶ Policy-driven automation tools to ease information lifecycle management (ILM)
- ▶ Extensible management and monitoring infrastructure to simplify file system administration
- ▶ Cost-effective disaster recovery and business continuity

1.3 Licensing

See the following announcements for pricing, licensing, and entitlement structure for Version 3.4:

<http://www.ibm.com/common/ssi/cgi-bin/ssialias?subtype=ca&infotype=an&appname=iSource&supplier=897&letternum=ENUS210-167>

GPFS has two types of licenses, a Server license and a Client license (licenses are priced per processor core). For each node in a GPFS cluster, the customer determines the appropriate number of GPFS Server licenses or GPFS Client licenses that correspond to the way GPFS is used on that node (a node is defined as one operating system instance on a single computer or running in a virtual partition). For further information, see the related GPFS FAQs:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.cluster.gpfs.doc/gpfs_faqs/gpfs_faqs.html

1.4 Operating systems support

GPFS is supported on AIX, Linux, and Windows. Consult the GPFS FAQ information center for the latest list of AIX environments, Linux distributions, Linux kernel versions, OpenSSL levels, and Windows environments:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.cluster.gpfs.doc/gpfs_faqs/gpfsclustersfaq.html

1.5 Hardware support

GPFS for POWER® is supported on both AIX and Linux. The GPFS for Linux Multiplatform and x86 Architecture products run Linux clusters that are based on selected servers. GPFS for Windows Multiplatform is supported on Windows Server 2008 on 64-bit architectures (AMD x64 / EM64T).

For further information regarding the use of GPFS in your clusters, see the *General Parallel File System Concepts, Planning, and Installation Guide Version 3 Release 4*, GA76-0413.

1.6 Contact information

Depending on the nature of your question, you may ask it in one of several ways, if you want to correspond with IBM regarding GPFS:

- ▶ If your question concerns a potential software error in GPFS and you have an IBM software maintenance contract, contact 1-800-IBM-SERV in the United States, or your local IBM Service Center in other countries.
- ▶ If you have a question that can benefit other GPFS users, you may post it to the GPFS technical discussion forum at:

http://www.ibm.com/developerworks/forums/dw_forum.jsp?forum=479&cat=13

- ▶ The FAQ is continually being enhanced. To contribute possible questions or answers, send them to:

<mailto:gpfs@us.ibm.com>

- ▶ If you want to interact with other GPFS users, the San Diego Supercomputer Center maintains a GPFS user mailing list:

<mailto:gpfs-general@sdsc.edu>

You may subscribe to the list at the following location:

<https://lists.sdsc.edu/mailman/listinfo/gpfs-general>

Infrastructure planning

This chapter introduces the hardware architecture and solution for the IBM General Parallel File System (GPFS). As part your infrastructure planning, this chapter includes input/output (I/O) patterns and special requirements for each industry solution.

In 2010, many fabric device manufacturers announced new versions of products such as 10 Gbps Ethernet, Fibre Channel over Ethernet, and new versions of host bus adapters. This chapter describes architecture of the devices, how they are used by IBM GPFS, and considerations for connectivity with server systems.

This chapter contains the following topics:

- ▶ 2.1, “Network design” on page 6
- ▶ 2.2, “Storage design” on page 11
- ▶ 2.3, “InfiniBand Storage DCS9900” on page 22
- ▶ 2.4, “Industry solution design” on page 23
- ▶ 2.5, “GPFS planning guidance” on page 38
- ▶ 2.6, “Summary” on page 68

2.1 Network design

The IBM GPFS solution works with various network devices. GPFS requires remote shell and Network Shared Disk (NSD) design based on TCP/IP. This section also explains characteristics of Network File System (NFS) and Common Internet File System (CIFS) operation with GPFS.

2.1.1 The 1 Gigabit Ethernet and 10 Gigabit Ethernet

This section describes characteristics and configuration options of the networks that can be used while deploying GPFS.

1 Gigabit Ethernet

This traditional standard uses an RJ45 or SX type of Ethernet connection, which might require that the two networks (RJ45 and SX network types) be separated. Consider the following points regarding Ethernet configuration:

- ▶ Jumbo frames: To optimize the performance for GPFS and your network, be sure to enable jumbo frames if your switch supports it. Also, the maximum jumbo frame size differs by network interface card (NIC) adapter maker. Be sure to change the TCP window scale according to your operating system for operation with the jumbo frame.
- ▶ Configure NIC: Usually, the host bus adapter (HBA) speed is higher than the 1 Gbps per NIC. It can be 4 Gbps or 8 Gbps, therefore, the NIC aggregation function is useful for getting more bandwidth. If you configure aggregation (*teaming*) with two adapters, GPFS with full bandwidth uses 2 Gbps. The teaming rule function operates with the teaming driver and the Ethernet switch module setting.

10 Gigabit Ethernet

As mentioned, the recent trend is for customers to consider the 10 Gbps NIC for their own infrastructures. This option is useful for the media industry, which requires more high-performance solutions.

The key option for GPFS is the adapter. The 10 Gbps NIC adapter has a higher bandwidth than 8 Gbps HBA. Therefore, this option does not require teaming for increased bandwidth.

2.1.2 InfiniBand (IP over IB/RDMA)

GPFS can exploit IP over InfiniBand and Remote Direct Memory Access (RDMA) to provide access to the file system. This section describes the InfiniBand architecture.

Introducing InfiniBand

InfiniBand is an industry-standard for server I/O and inter-server communication. InfiniBand is an open set of interconnect standards and specifications which can be found in the following web location:

<http://www.infinibandta.org>

This solution is useful for simplifying high speed and integration. Usually, this architecture is used for High Performance Computing solutions because they require high speed communication and low latencies. However, this technology has increased its popularity for commercial solutions, for example multimedia, storage, databases, and it is also useful in GPFS clusters.

InfiniBand is based on a switched fabric architecture of serial point-to-point links, as shown in Figure 2-1. The InfiniBand links can be connected to either host channel adapters (HCAs) that are used primarily in servers, or target channel adapters (TCAs) that are used primarily in storage subsystems.

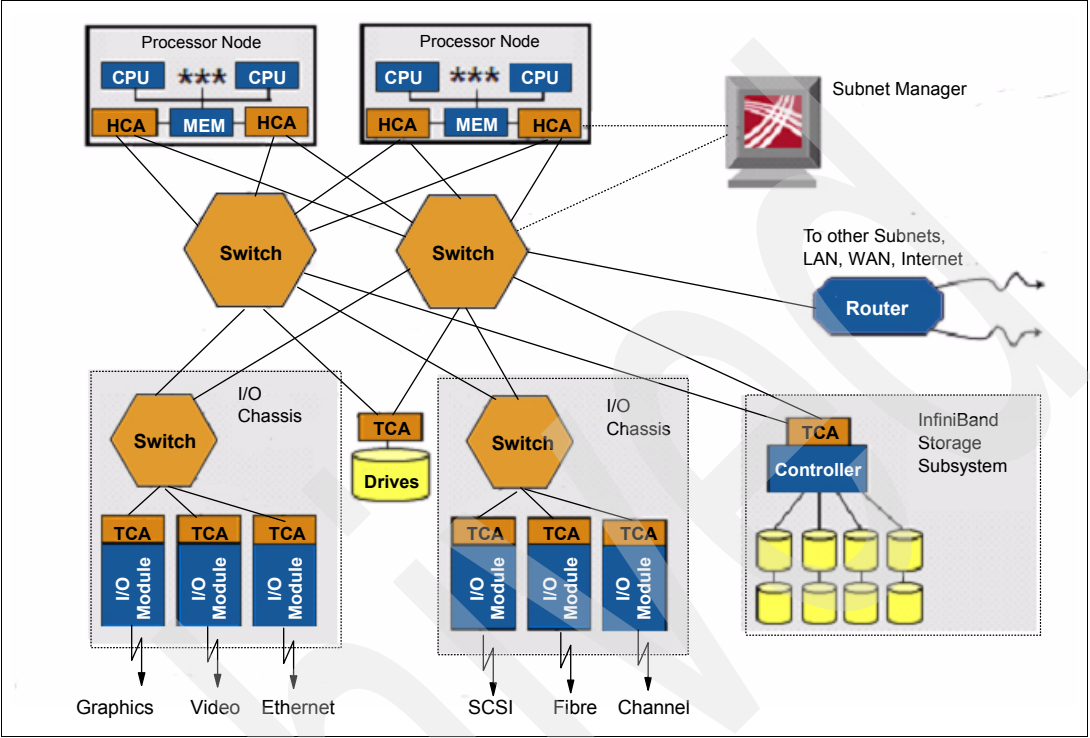


Figure 2-1 InfiniBand architecture

Figure 2-2 shows the InfiniBand driver and hardware stack.

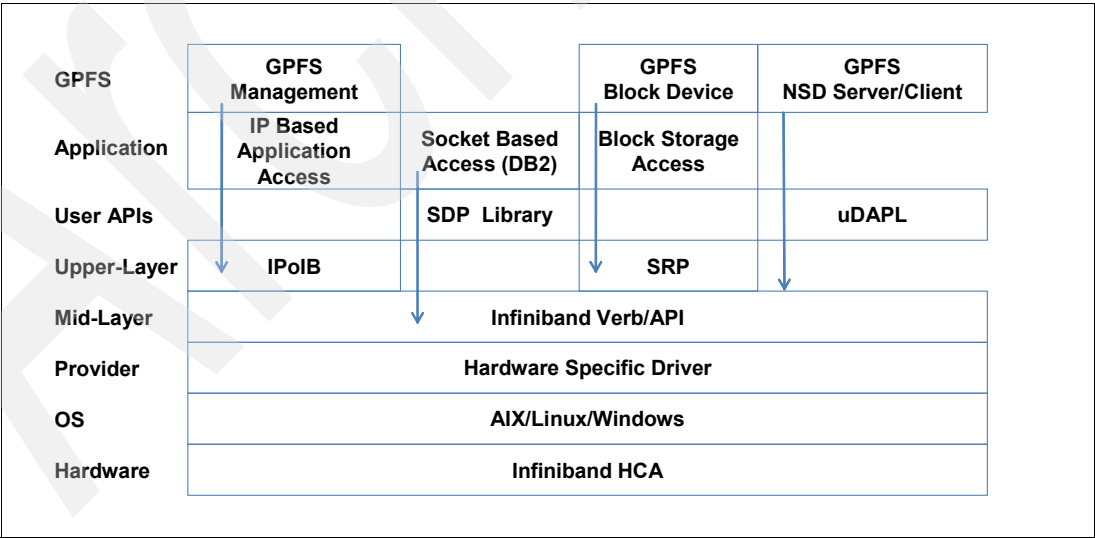


Figure 2-2 InfiniBand driver stack diagram with GPFS daemon

Overview of InfiniBand protocols

Upper-level protocols such as IP over InfiniBand (IPoIB), Socket Direct Protocol (SRP), SCSI RDMA Protocol (SDP), iSCSI Extensions for RDMA (iSER) and so on, facilitate standard data networking, storage, and file system applications to operate over InfiniBand. Except for IPoIB, which provides a simple encapsulation of TCP/IP data streams over InfiniBand, the other upper-level protocols transparently enable higher bandwidth, lower latency, lower CPU utilization, and end-to-end service using field-proven RDMA and hardware-based transport technologies available with InfiniBand. Configuring GPFS to exploit InfiniBand helps make the network design simple because InfiniBand integrates the network and the storage together (each server uses a single adapter utilizing different protocols), as in the following examples:

- ▶ GPFS cluster management can use IPoIB.
- ▶ GPFS can use SDP.
- ▶ GPFS storage attached can use SRP.
- ▶ GPFS Network Shared Disk (NSD) communication can use RDMA.

IP over InfiniBand (IPoIB)

IPoIB running over high-bandwidth InfiniBand adapters can provide an instant performance boost to any IP-based applications. IPoIB support tunneling of IP packets over InfiniBand hardware. This method of enabling IP applications over InfiniBand is effective for management, configuration, setup or control plane related data where bandwidth and latency are not critical. Because the application continues to run over the standard TCP/IP networking stack, the application are completely unaware of the underlying I/O Hardware.

Socket Direct Protocol (SDP)

For applications that use TCP sockets, the SDP delivers a significant boost to performance and reduces CPU utilization and application latency. The SDP driver provides a high-performance interface for standard socket applications and a boost in performance by bypassing the software TCP/IP stack, implementing zero copy and asynchronous I/O, and transferring data using efficient RDMA and hardware-based transport mechanisms.

SCSI RDMA Protocol (SRP)

SRP was defined by the ANSI T11 committee to provide block storage capable for the InfiniBand architecture. SRP is protocol that uses the InfiniBand reliable connection and RDMA capabilities to provide a high-performance transport for the SCSI protocol. SRP is extremely similar to Fibre Channel Protocol (FCP), which uses Fibre Channel to provide SCSI over Fibre Channel. This allows one host driver to use storage target devices from various storage hardware.

Remote Direct Memory Access (RDMA)

RDMA used to allow various servers on the InfiniBand fabric to access the memory of another server directly. An example of this is a GPFS Cluster or a database server cluster. GPFS has `verbPorts` and `verbRdma` options for the InfiniBand RDMA function, and the database server cluster adds an RDMA agent to its core functionality, which allows two database instances running on different nodes to communicate directly with each other, bypassing all kernel-level communication operations, thus reducing the number of times that the data is copied from persistent storage into the RAM memory of cluster nodes.

2.1.3 Configuring the NIC

This section provides information about how to take advantages of features within each operating system and the network interface cards.

Bonding on Linux

The bonding module provides seven modes of operation:

- ▶ 0 for balance round robin (RR)
- ▶ 1 for active-backup
- ▶ 2 for balance-selected transmit hash policy (XOR)
- ▶ 3 for broadcast
- ▶ 4 for 802.3ad
- ▶ 5 for balance-transmit load balancing (TLB)
- ▶ 6 for balance-adaptive load balancing (ALB) change arp (CHARP)

When configuring the bonding operation module on Linux, set it up on the network switch side. All modules are included in all Linux distributions.

The following options can be used while configuring the network adapters:

- ▶ Round robin: The output device is selected based on the next available subordinate, regardless of the source or destination of the packet.
- ▶ XOR: The XOR parameter selects the same subordinate for each destination hardware address.
- ▶ Active-backup: This policy ensures that one (and only one) device will transmit at any given moment. Active-backup policy is useful for implementing high availability solutions using two hubs. Usually, 10 Gbps NIC devices are configured with failover options, and no load balancing enabled.

Teaming on Windows

The Windows operating system does not include teaming software so you must download it from Intel, Broadcom, or another vendor. The teaming function differs slightly from each network supplier, for example as follows:

- ▶ Intel
 - Adapter Fault Tolerance (AFT)
 - Switch Fault Tolerance (SFT)
 - Adaptive Load Balancing (ALB)
 - Virtual Machine Load Balancing (VMLB)
 - Intel Link Aggregation (LA), Fast EtherChannel (FEC), and Gig EtherChannel (GEC)
 - IEEE 802.3ad
- ▶ Broadcom
 - Smart Load Balancing (SLB)
 - Link Aggregation (802.3ad)
 - Generic Link Aggregation (Trunking)
 - Failover Teaming

EtherChannel on AIX

EtherChannel is a network port aggregation technology that allows several Ethernet adapters to be put together to form a single pseudo-Ethernet device on AIX. EtherChannel has four modes:

- ▶ **Standard:** In this mode, the EtherChannel uses an algorithm to choose on which adapter it will send the packets out. The algorithm consists of taking a data value, dividing it by the number of adapters in the EtherChannel, and using the remainder.
- ▶ **Round_robin:** In this mode, the EtherChannel rotates through the adapters, giving each adapter one packet before repeating. The packets may be sent out in a slightly different order than they are given to the EtherChannel, but the EtherChannel makes the best use of its bandwidth.
- ▶ **Netif_backup:** This mode is available only in AIX 5.1 and AIX 4.3.3. In this mode, the EtherChannel activates only one adapter at a time. The intention is that the adapters are plugged into separate Ethernet switches, each of which is capable of getting to any other machine on the subnet or network.
- ▶ **802.3ad:** This mode enables the use of the IEEE 802.3ad Link Aggregation Control Protocol (LACP) for automatic link aggregation.

2.1.4 IBM data center networking (DCN) products

The IBM data center networking products are important for the GPFS solution. This offering provides the entire network solution and the network management infrastructure. When you design a large cluster with GPFS, consider as a design point the network and how to manage the cluster within the network. The following items are part of the data center networking product solution:

- ▶ 1 Gbps Switch, 10 Gbps Switch
- ▶ 10 Gbps FCoE Switch
- ▶ Backbone Switch
- ▶ Network Management Software

For more details, see Figure 2-3 on page 11.

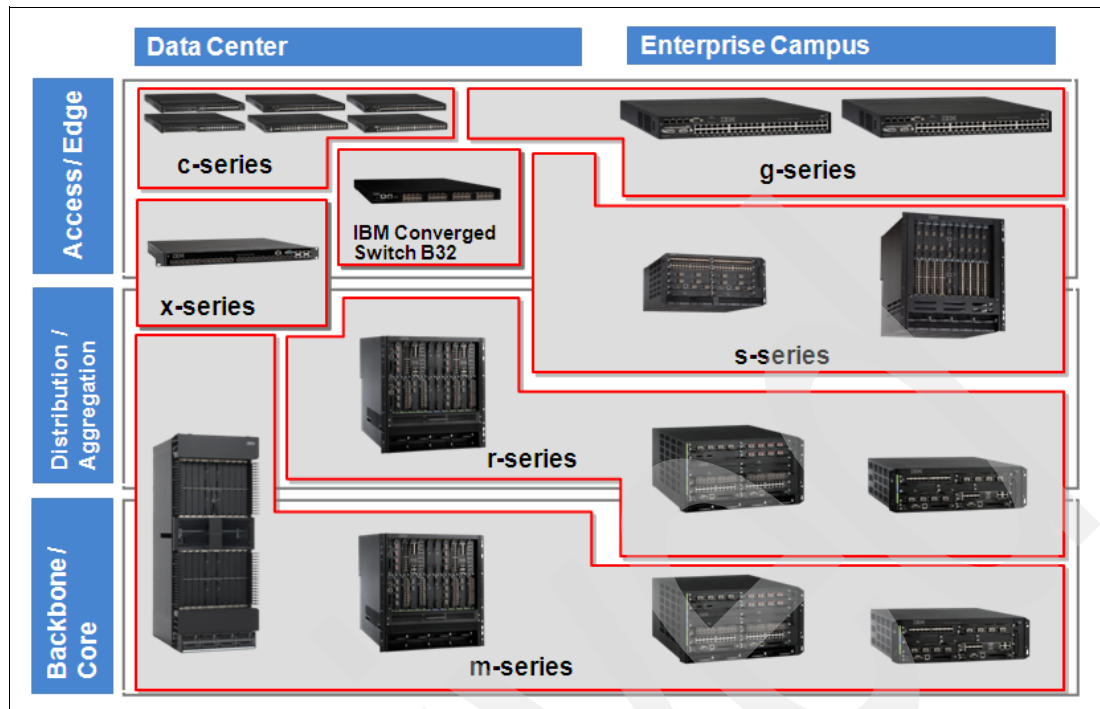


Figure 2-3 Overview of DCN products

Note: See the following resource for more information:

<http://www.ibm.com/systems/networking/>

2.2 Storage design

There are several aspects to consider when designing the storage configuration for a GPFS cluster. For example, storage solutions deployed with disk interfaces are an important consideration because of good performance characteristics. Thus, select the best storage solution to help you achieve the performance characteristics for your cluster solution. The following list represents several storage solutions you might consider when designing your storage solution:

- ▶ Host bus adapter: FibreChannel HBA, SAS HBA
- ▶ RAID control: SAS/SATA
- ▶ Storage controller and configuration
- ▶ Disk type: SAS/SATA/FC/SSD
- ▶ SAN Switch and Multipath I/O

2.2.1 Host bus adapter (HBA)

This section provides details about the characteristics of the HBA.

4 Gbps or 8 Gbps host bus adapter

The HBA is most often used to refer to a Fibre Channel interface. The two types of worldwide names (WWNs) on a host bus adapter are as follows:

- ▶ A worldwide node name (WWNN), which is shared by all ports on a host bus adapter
- ▶ A worldwide port name (WWPN), which is unique to each port

HBA models have various speeds: 1 Gbps, 2 Gbps, 4 Gbps, and 8 Gbps.

For GPFS, use latest release of firmware and driver.

You may choose the following HBA options:

- ▶ Single port 4 Gbps or 8 Gbps PCI Express adapter
- ▶ Dual port 4 Gbps or 8 Gbps PCI Express type adapter

N_Port ID Virtualization (NPIV) architecture

Figure 2-4 shows how to assign a volume on each virtual machine or logical partition. The layers for NPIV configuration differ on System x and System p.

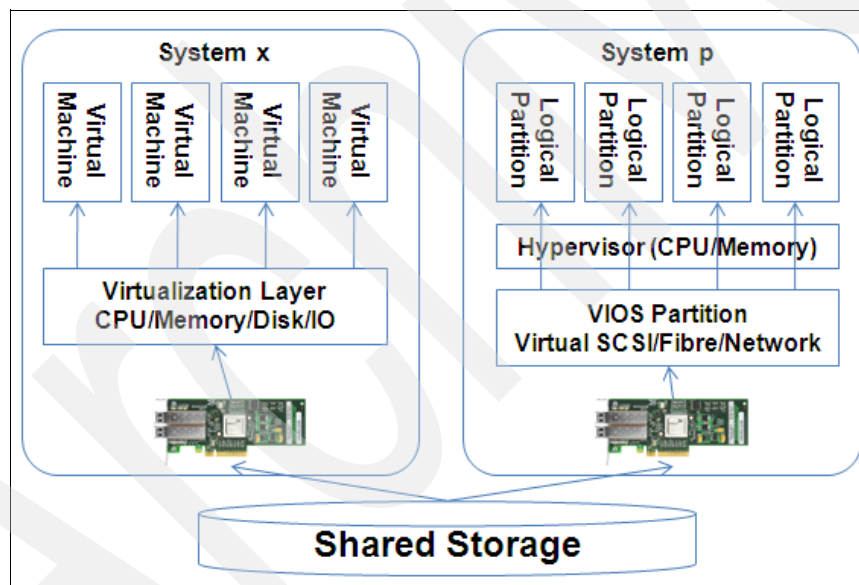


Figure 2-4 Managed system configured to use NPIV on System x and System p

This technology was introduced with System z® in 2007. Although the technology seems to operate like vNIC on Ethernet, it now functions to use all HBA with the server system. NPIV provides a simple, high-availability capability. NPIV uses HBA technology, which creates virtual HBA ports on a host by abstracting the underlying physical port. This support enables a single physical Fibre Channel HBA port to function as multiple logical ports, each with its own identity.

For the following reasons, consider using NPIV for a virtualization environment:

- ▶ Virtual server requires secure access to storage the same way as physical server does.
- ▶ Without NPIV, a single physical server connection is unable to provide independent storage access to an individual virtual server.
- ▶ Without NPIV, all storage port and logical unit numbers (LUNs) are exposed to all virtual machines, reducing securities and manageability.
- ▶ NPIV is an ANSI standard that is designed to solve this problem.
- ▶ NPIV devices, connected to the same switch port, must have a unique 24-bit address, and a unique device WWPN.

Failover driver on Linux x64

Many HBA suppliers provide failover drivers only on Linux. The following situations might occur:

- ▶ The vendor does not provide the multipath driver for the storage product.
- ▶ The vendor does not support mixed multipath driver provided for the storage product.
- ▶ The solution cannot use dual HBAs because the operation is configured as active-passive mode.

2.2.2 Multipath driver

The following section provides characteristics and details of the multipath driver on System p and System x.

Multipath driver MPIO on System p

Multiple Path I/O (MPIO) helps to increase availability of virtual SCSI resources by providing redundant paths to the resource. This topic describes how to set up MPIO for AIX client logical partitions. To provide MPIO to AIX client logical partitions, you must have two Virtual I/O Server (VIOS) logical partitions configured on your system.

Figure 2-5 shows volume mapping with multipath driver on System p.

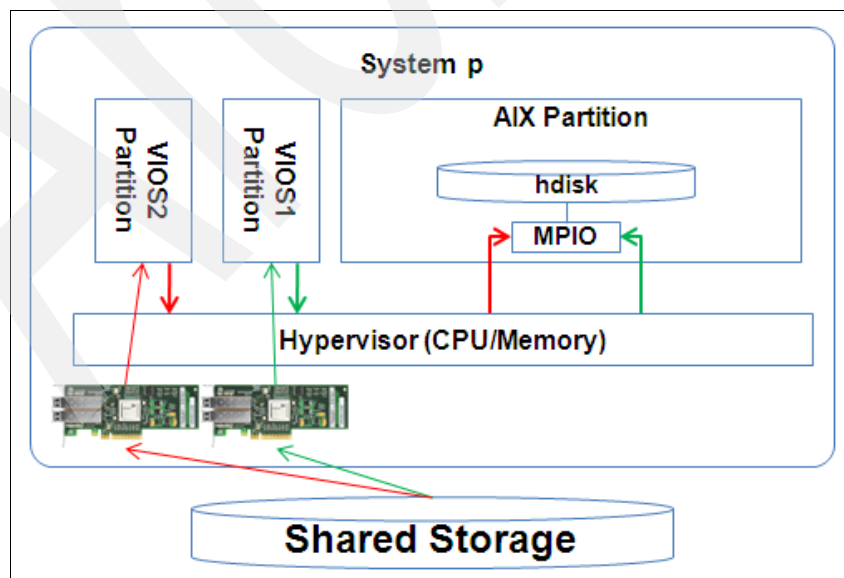


Figure 2-5 External storage disk configuration with MPIO and redundant VIOS

System p has two options for multipath configuration:

- ▶ With MPIO, only one path per controller is activated.
- ▶ With the IBM Subsystem Device Driver Path Control Module (SDDPCM), the full path is activated but cannot be used over four-path.

Multipath driver RDAC and DM-Multipath driver on Linux

The multipath driver RDAC and the DM-multipath driver for Linux are available as follows:

- ▶ RDAC is provided by the IBM DS3000, DS4000®, DS5000 series storage solutions.
- ▶ The DM-multipath driver is provided by the open source community.
- ▶ Multipath drivers provide solutions to the following functions:
 - HBA failure
 - FC cable failure
 - SAN switch failure
 - Array controller port failure

RDAC

The Redundant Disk Array Controller (RDAC) component contains a multipath driver and hot-add support. It must be installed on the host system, and it provides redundant paths to the storage server when both RAID controllers are installed. If a RAID controller fails or becomes inaccessible because of connectivity problems, RDAC reroutes the I/O requests through another RAID controller. The hot-add part of RDAC enables you to dynamically register new logical drives to the operating system.

To configure RDAC, you must disable the auto volume transfer (AVT) function and the non-failover driver:

1. An important option on RDAC is to use the following parameter, which disables the failback function on the storage side (the parameter is in the `/etc/mpp.conf` file):
`DisableLUNRebalance=3`
2. Next, use the HBA *driver failover time minimize option*. For example, if you have QLogic HBA, use the following parameter to minimize failover time (the parameter is in the `/etc/modprobe.conf` file):
`qlport_down_retry=3`

Also consider enabling on the RDAC and the HBA driver modules the failover options.

Important: RDAC must be loaded, even if you have only one HBA on the host.

DM-Multipath

DM-Multipath can provide failover in an active-passive configuration. In an active-passive configuration, only half the paths are used at any time for I/O. If any element of an I/O path (the cable, switch, or controller) fails, DM-Multipath switches to an alternate path. It can be configured in active-active mode, where I/O is spread over the paths in a round-robin fashion. In certain configurations, DM-Multipath can detect loading on the I/O paths and dynamically rebalance the load. With DM-Multipath configured, a failure at any of these points causes DM-Multipath to switch to the alternate I/O path.

Figure 2-6 shows the device name of mapping information about each of multipath driver on Linux system. This device name is used to configure NSDs for the GPFS cluster.

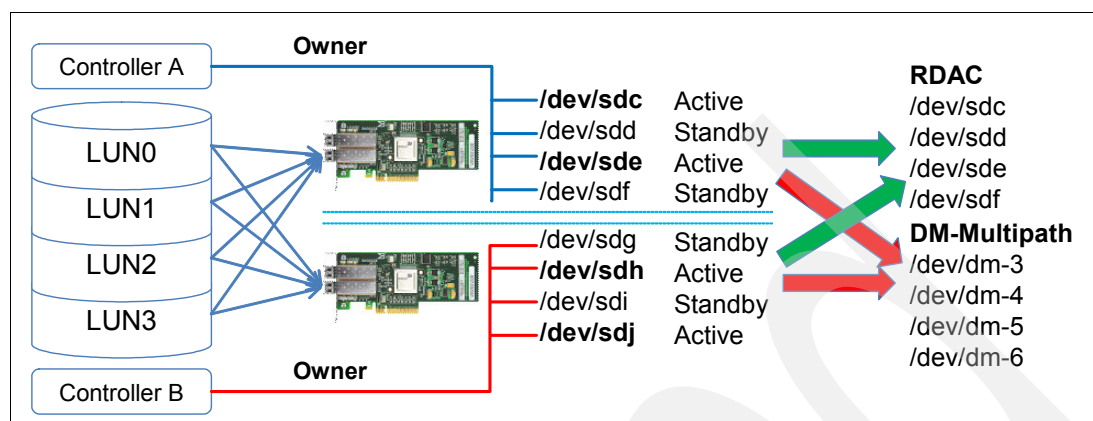


Figure 2-6 Mapping diagram on Linux with each of multipath drive

Note: The following information relates to the Multipath driver with Storage Auto-Logical Drive Transfer (ADT):

- ▶ ADT is not a failover driver. ADT provides storage systems with the flexibility to work with certain third-party failover software.
- ▶ In ADT-disabled mode, you are required to issue a redistribution command manually to get the LUNs balanced across the controllers.
- ▶ In ADT mode, RDAC automatically redistributes the LUNs to their preferred path after the failed path is again operational.

Multipath driver on Windows

MPIO is a Driver Development Kit (DDK) from Microsoft for developing code that manages multipath devices. It contains a core set of binary drivers, which are installed with the DS5000 Device Specific Module (DSM) to provide a transparent system architecture. The architecture relies on Microsoft Plug and Play to provide LUN multipath functionality and maintain compatibility with existing Microsoft Windows device driver stacks.

Note: The MPIO Driver is included in the Storage Manager software package for Windows and supports Microsoft Windows 2008 R2 x64 systems. In Windows 2008 R2, MPIO is already part of the operating system.

Multipath driver with the IBM SVC

The IBM System Storage® SAN Volume Controller (SVC) provides virtualized volume with heterogeneous storage. For example, a single service point of view can be configured with the multipath driver provided by the storage vendor. If you want to configure a multipath environment with multiple vendor storage controllers, the SVC multipath driver can be installed in your system because separate drivers cannot be installed in one system. This virtualizes the controllers and provides a multi path environment using multiple vendor storage systems.

Figure 2-7 shows the SVC architecture and the physical and logical views.

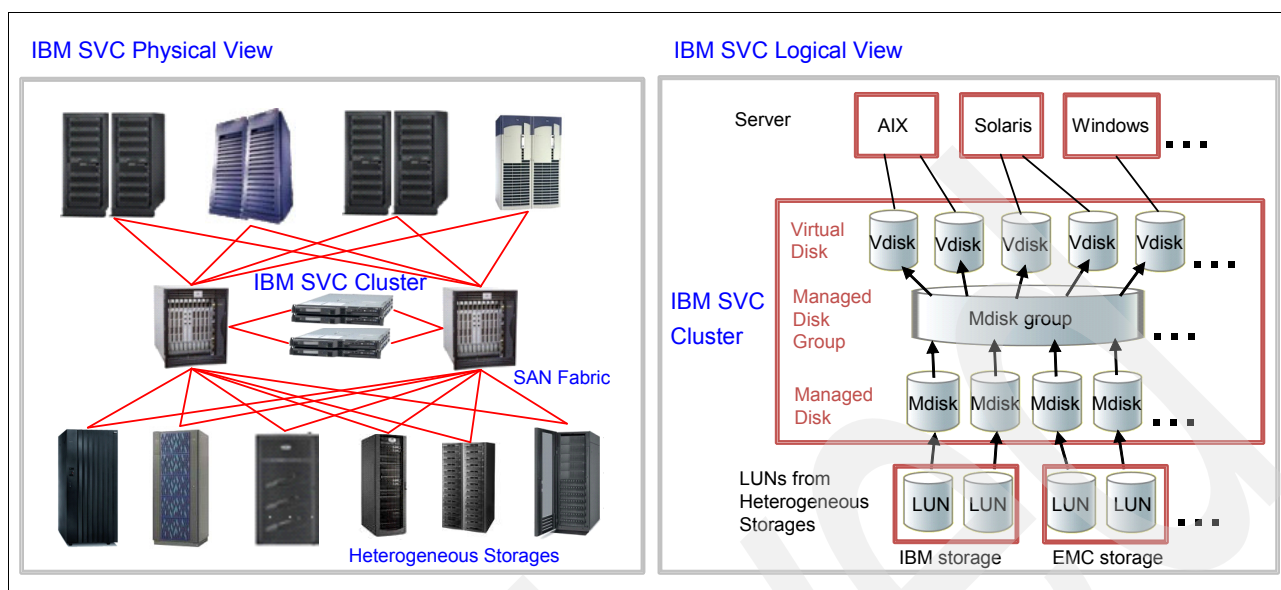


Figure 2-7 IBM SVC physical and logical view

Be sure to use zoning to limit the number of paths to four. The hosts must run a multipathing device driver to show a single device. The multipathing driver supported and delivered by SVC is the IBM Subsystem Device Driver (SDD). Native MPIO drivers on selected hosts are supported.

Notes: For OS-specific information about MPIO support, go to the following address:

<http://www.ibm.com/systems/storage/software/virtualization/svc/interop.html>

To download Subsystem Device Driver Device Specific Module (SDDDSM) information, see the following address:

http://www.ibm.com/support/docview.wss?rs=540&context=ST52G7&dc=D430&uid=s5g1S4000350&loc=en_US&cs=utf-8&lang=en#SVC

2.2.3 The characteristic of storage hardware

This section provides information about the hardware storage functions.

RAID level

Select a RAID protection level for your array. For most situations, you need a degree of availability protection, which eliminates RAID 0. You also have to select the best RAID level according to your applications and business needs. To help you with the selection, use Table 2-1 on page 17 as a guideline for selecting RAID protection for GPFS.

Table 2-1 Guidelines for selecting RAID protection for GPFS

| Description | RAID 5 | RAID 6 | RAID 10 | Advantage |
|----------------------------------|--------|--------|---------|--------------|
| Random write performance | 2 | 3 | 1 | RAID 10 |
| Sequential write performance | 1 | 2 | 3 | RAID 5 |
| Availability | 3 | 1 | 2 | RAID 6 |
| Space efficiency | 1 | 2 | 3 | RAID 5 |
| RAID level for GPFS | Yes | Yes | No | RAID 5 and 6 |
| Configuration for GPFS | 4+P | 8+2P | N/A | N/A |
| Minimum number of disks per LUNs | 5 | 10 | 8 | N/A |

Cache policy

This cache policy is an area of tuning where there are significant differences based on the style of storage server and the I/O workload. Generally, consider the following guidelines:

- ▶ Sequential I/O workloads: Enable read cache, disable any read prefetch
- ▶ Random I/O workloads: Enable read and write cache
- ▶ Metadata Disks: Enable read and write cache

It may be more apparent why you would enable cache for random workloads but not so clear why you disable read cache on sequential workloads.

With GPFS, disable any read prefetch mechanisms for the caching of data in the storage server. Although it works well for high concurrency environments and is high performance, read-prefetch at the storage must be disabled because the GPFS mechanism for allocating space is difficult for storage algorithms to “predict.” GPFS can compensate by prefetching data. To a storage controller, this read access does not look sequential, so the read-ahead that is provided in most storage servers makes incorrect assumptions, therefore actually degrading read-performance. In fact, up to a 20% penalty can occur on certain models of storage.

The following DS4000/DS5000 cache settings are supported:

- ▶ Read cache enabled or disabled
- ▶ Write cache enabled or disabled

The cache mirroring is enabled or disabled (depending upon the write cache mirroring setting). The performance benefits of the read or the write caching depend on the application. Because the cache settings can be easily adjusted from the Storage Manager (SM), preferably carry out your performance tests during the implementation. Use the following sample configuration as a starting point for the NSD servers or other GPFS nodes that are directly attached to a SAN over a Fibre Channel network; use the following cache settings:

- ▶ Read cache = enabled
- ▶ Read ahead multiplier = 0
- ▶ Write cache = disabled
- ▶ Write cache mirroring = disabled
- ▶ Write block size = 16 KB

Note: In the event of a power failure or hard controller failure, a risk of file system corruption exists when enabling write cache.

Block size and calculation guidelines

To optimize disk I/O performance, consider the following options for NSD servers or other GPFS nodes that are directly attached to a SAN over a Fibre Channel network. When the storage server disks are configured for RAID 5, certain configuration settings can affect GPFS performance. These settings are as follows:

- ▶ GPFS block size
- ▶ Maximum I/O size of host Fibre Channel (FC) host bus adapter (HBA) device driver
- ▶ Storage server RAID 5 stripe size

Figure 2-8 describes GPFS Block, RAID level and storage segment size points to consider.

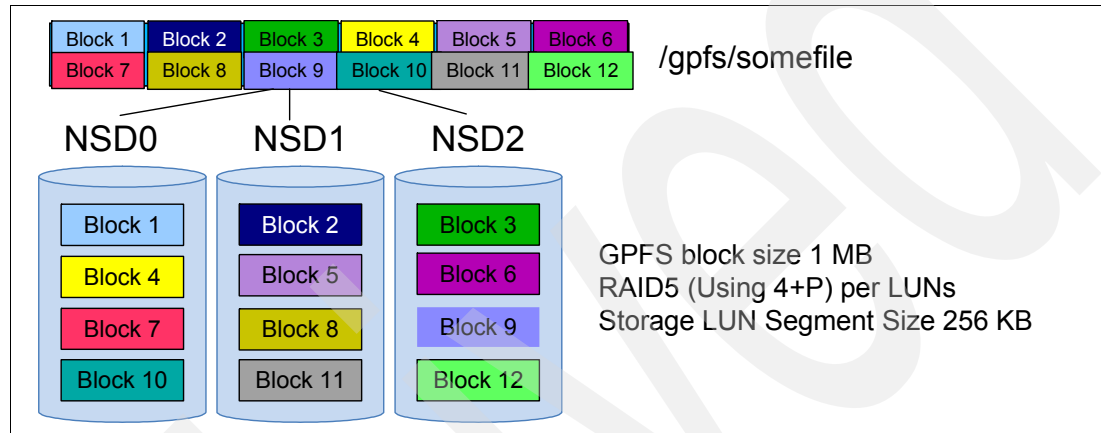


Figure 2-8 GPFS block and storage LUN segment guideline

For optimal performance, GPFS block size should be a multiple of the maximum I/O size of the FC HBA device driver. In addition, the maximum I/O size of the FC HBA device driver must be a multiple of the RAID 5 stripe size. These suggestions might avoid the performance penalty of read-modify-write at the storage server for GPFS write operations. Examples of the suggested settings are as follows:

- ▶ 8+P RAID 5
 - GPFS block size = 512 KB
 - Storage Server RAID 5 segment size = 64 KB (RAID 5 stripe size=512 KB)
 - Maximum I/O size of FC HBA device driver = 512 KB
- ▶ 4+P RAID 5
 - GPFS block size = 256 KB
 - Storage Server RAID 5 segment size = 64 KB (RAID 5 stripe size = 256 KB)
 - Maximum I/O size of FC HBA device driver = 256 KB

For the example settings using 8+P and 4+P RAID 5, the RAID 5 parity can be calculated from the data written and prevents reading from disk to calculate the RAID 5 parity. The maximum I/O size of the FC HBA device driver can be verified by using the `iostat` command or the Storage Server performance monitor. In certain cases, the device driver might have to be patched to increase the default maximum I/O size.

I/O balance

I/O balance is key when you design a high performance GPFS system. This balance is most important when you are expecting to get the best sequential I/O performance from a solution. To better understand how balance can affect performance we describe an example.

GPFS reads and writes to all disks at the same time when performing sequential I/O operations. Therefore, with this file system configuration (assuming the servers have enough capacity) you can get a total throughput of 400 MBps.

When a new disk is added, the overall performance of the file system now drops below the original 400 MBps. Why does this happen? GPFS reads and writes to all disks concurrently, which means if the streaming operations are long enough, all the queues (adapter, disk and so on) will fill up and start emptying at the same rate. This process slows the throughput of all the disks to that of the slowest disk. As a result, instead of 400 MBps you have 300 MBps.

Therefore, if you want to mix disks of different performance characteristics in the same file system, either you must understand the overall effect on performance or you have to use storage pools. GPFS storage pools allow you to keep a slow disk from affecting faster disks in a single file system. These same principles of balance apply to all levels of the I/O path including the Fibre Channel HBAs, host ports on the storage, and the number of arrays used for each controller.

Considerations for DS4000/5000 configuration with GPFS

Consider the following information about GPFS configurations:

- ▶ Recoverable Virtual Shared Disk (RVSD) clusters can support up to two IBM Virtual Shared Disks and RVSD servers for each DS4000 or DS5000 partition.
- ▶ Single node quorum is not supported in a dual-node GPFS cluster with DS4000 or DS5000 disks in the configuration.
- ▶ Heterogeneous configurations are not supported.

Considerations DCS9550 and DCS9900 with GPFS

When using a DCS9900 with multiple NSD servers running multipathing software, ensure that the LUNS, for which each NSD server is primary, are spread across both controllers. As an example, we look at one of the NSD servers, which is the primary for two LUNS. We verify that each LUN is being shared from a separate controller for optimum performance. This information is from a Red Hat 5.3 Linux system using the recommended built-in multipathing software. Consider the following information:

- ▶ Use the `mm1nsd` command to get the device name that is associated with the NSD because this storage does not provide a multipath driver.
- ▶ Disable write cache on DCS9900 models that do not have mirrored cache.
- ▶ Set cache prefetch in the storage side to 0. GPFS obtains the best read performance when the controller is not doing any prefetching.
- ▶ Use one LUN per tier by assigning, at a minimum, one tier to GPFS as a single LUN. Do not split the tier into multiple LUNS.
- ▶ Use four or eight HBAs or InfiniBand connections to balance I/O over all four couplet channels.
- ▶ Use a separate Fibre Channel or SAS disk storage (DS3500 for example) for the file system metadata.

Considerations for XIV storage with GPFS

One of the benefits of the IBM XIV® Storage System is that there is no tuning required for most workloads, and this applies to GPFS also. When configuring a system with XIV and GPFS, be sure multiple LUNS are available so you can have multiple SCSI devices available in the operating system for best performance. For example, create eight LUNS or 16 LUNS for a file system. In the XIV architecture, LUNS are spread over all available disks so there are no

RAID configuration, segment size, or LUN layout considerations. Consider the following information:

- ▶ Be sure all server HBAs are zoned to see all XIV controller ports.
- ▶ Create multiple LUNS so that the operating system (Linux for example) can see multiple targets for better SCSI performance.
- ▶ Because XIV uses a 1 MB allocation size, for sequential data access, a GPFS file system block size of 1 MB can provide optimal performance.
- ▶ Use multiple LUNS in each GPFS file system so you can best utilize multiple SAN paths and device queues in the operating systems.

Solid state drive (SSD) storage

The world of storage is no longer only about capacity; it is now also about the need for extremely fast and reliable access to data. The ever-increasing demand on today's servers, combined with the exponential growth of processor performance, has exposed the limitations of today's aging storage architectures.

IBM offers solid state storage today with the new solid state PCI Express (PCIe) adapters. Designed around silicon-based storage architecture, the PCI Express adapter has clustering technology, with performance comparable to DRAM and storage capacity on par with today's hard disks, giving you the power to improve both memory capacity and storage performance. Examples are as follows:

- ▶ SSD SAS, SATA Interface
- ▶ Solid State PCIe Adapter Type

The SSD-based storage system can be useful for applications exploiting the GPFS solution. For example, GPFS provides a storage pool architecture where GPFS populates each of file on special storage group such as SSDs, HDDs, and SATA HDDs. Depending on how fast the store or retrieval speed is required, the fastest is the medium to where the file can be populated. Examples are as follows:

- ▶ GPFS metadata: If you use SSD for the metadata area of the file system, you can gain response time on the file system, especially concurrently accessing a group of files.
- ▶ SSD storage pool: This example is useful for RDBMS index database and real-time access database file.

2.2.4 Tape library

Although automated tape libraries have one or more tape drives, they are typically used with at least two tape drives. All tape cartridges are accessible to all drives, therefore making concurrent reading and writing operations possible by Tivoli® Storage Manager or Hierarchical storage management solution in the GPFS environment, which means GPFS cannot directly handle this tape library. This section explains two types of tape drives for information life management (ILM). Usually, single-port tape drive is used for simple backup; dual-port tape drive is used for the ILM environment.

You can increase throughput by adding more drives. Libraries can exchange tapes in several seconds, substantially improving file-restore response times. Tape libraries also offer the feature of enabling a drive to take over if another drive fails.

Multi-drive automated tape libraries and ultra-scalable tape libraries combined with storage-management software, including concurrent backup, archive, and hierarchical storage management (HSM), offer the most robust solution to manage and protect huge amounts of corporate data. Automated tape libraries allow random access to large numbers

of tape cartridges and the concurrent use of two or more drives, rather than manually loading one tape after another or using a single-drive sequential autoloader.

IBM offers the following tape library models: TS3100, TS3200, TS3310, TS3400 and TS3500. These models support two types of tape drives:

- ▶ Linear Tape-Open (LTO)
- ▶ TS1100 Series

Tape drives (LTO series)

Two LTO formats (Ultrium and Accelis) were originally introduced in 2000, and licenses for the new technology were made available. Since then, manufactures have not pursued the Accelis format because the Ultrium format meets market needs. The LTO-sponsoring companies have taken steps to protect client investment by providing a six-generation roadmap to illustrate native capacity, shown in Figure 2-9, and establishing an infrastructure to enable compatibility between products.

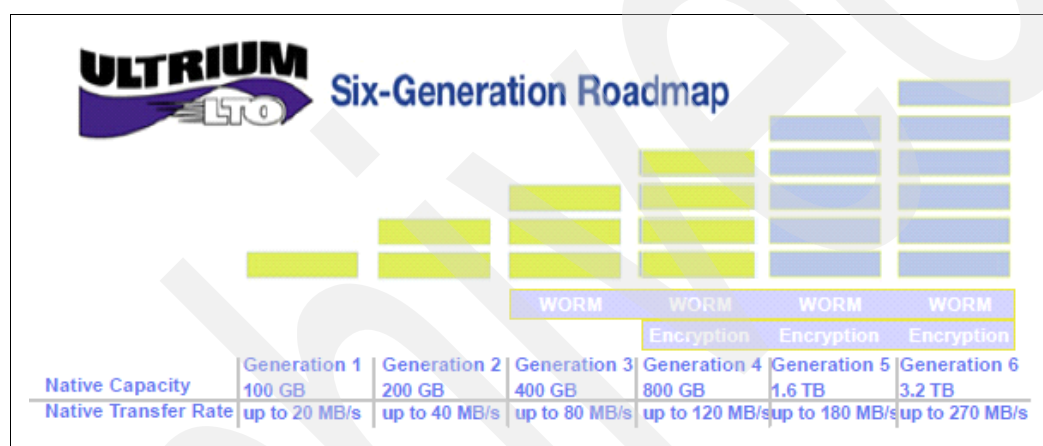


Figure 2-9 LTO Ultrium native capacity roadmap

TS1100 series tape drive

The IBM System Storage TS1120 and TS1130 tape drives (machine types 3592-E05 and 3592-E06/EU6) offer a design of high capacity, performance, and high reliability for storing mission-critical data. With the 2003 introduction of the first generation of the new family of tape drives, IBM advanced its high-end half-inch cartridge tape technology. The second generation of the 3592 family enhanced the capacity and performance characteristics, which have now become increased by the IBM System Storage TS1130 Model E06/EU6 Tape Drive, the third generation of the 3592 family. This generation provides the capacity of 1 TB of uncompressed data.

Data path failover provides a failover mechanism in the IBM device driver, enabling you to configure multiple redundant paths in a SAN environment. In the event of a path or component failure, the failover mechanism automatically provides error recovery to retry the current operation using an alternate, pre-configured path without canceling the current job in progress. This capability allows you flexibility in SAN configuration, availability, and management.

LAN-free backup (SAN backup)

A technology provides an alternative path for data movement between the Tivoli Storage Manager client and the server. Shared storage resources (disk, tape) are accessible to both the client and the server through the SAN. Data movement is off-loaded from the LAN and from the server processor and allows for greater scalability.

Figure 2-10 shows that the storage agent handles the communication with the Tivoli Storage Manager server over the LAN but sends the data directly to SAN-attached tape devices, relieving the Tivoli Storage Manager server from the actual I/O transfer.

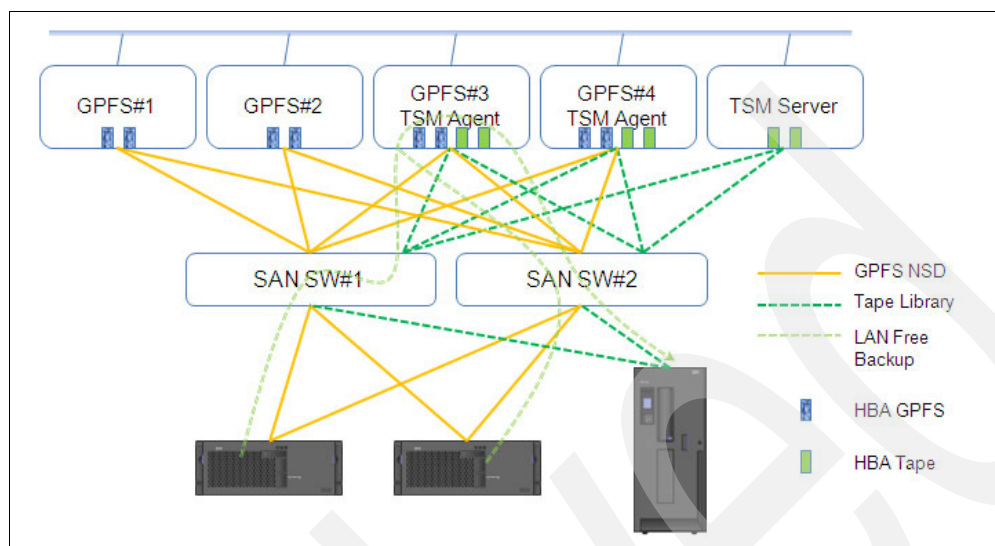


Figure 2-10 GPFS TSM/HSM Configuration and Lan Free Backup route.

LAN-free backups decrease the load on the LAN by introducing a storage agent. The storage agent can be thought of as a small Tivoli Storage Manager server (without a database or recovery log) that is installed and run on the Tivoli Storage Manager client machine.

2.3 InfiniBand Storage DCS9900

The IBM System Storage DCS9900 Storage System is designed for applications with high-performance streaming data requirements served by Deep Computing systems, IBM System p High Performance Computing (HPC) systems, and System x 1350 Linux Cluster systems.

Applications such as those in HPC, rich media, life sciences and government research require high performance, reliable access to streaming data and extreme capacity and density to simplify management and reduce cost. Examples of such applications include weather forecasting, seismic processing, reservoir modeling, high definition (HD) creation and distribution, proteomics, structural analysis, fluid dynamics, and gene sequencing. With its parallel storage solution architecture, the DCS9900 is specifically designed to address those needs.

With hardware-enabled RAID 6, and online parity checking and correction, performance is assured even in the event of disk-drive or enclosure failures. Silent data errors are resolved dynamically, and disk-rebuilds and even missing enclosures are handled without affecting performance.

The DCS9900 Controller is in a 4U rack-mount chassis, providing dual controllers with 5 GB cache (2.5 GB cache per controller), up to eight fiber channel adapters or InfiniBand 4x DDR host ports and twenty 3-Gbps SAS disk expansion ports to support up to 1200 disk drives, and supports up to 2.4 PB of physical storage capacity and 1024 LUNs per DCS9900 system.

Product highlights are as follows;

- ▶ A 2 TB SATA disk drive (7200 RPM) or 600 GB SAS disk drive (15000 RPM)
- ▶ Support for a three-enclosure configuration with SAS/SATA intermix in the same enclosure
- ▶ High-performance sequential I/O: Reads and writes at up to 5.9 GBps with no RAID 6 write penalty.
- ▶ Dense packaging and capacity, with up to 1.2 PB in a single rack and 2.4 PB in only two floor tiles
- ▶ High availability: Data access is ensured access independent of disk, enclosure or controller failures.
- ▶ Extreme reliability: Data transfer rates are sustained, independent of disk or enclosure failures, dynamic detection and correction of SATA silent data-corruption errors, and RAID 6 reliability.
- ▶ Fiber channel 8 Gbps 8 ports or InfiniBand 4x DDR (20 Gbps) 8 ports, InfiniBand model direct connect to InfiniBand Switch

2.4 Industry solution design

This section introduces several representative industry solutions and how to configure and use GPFS with the solutions.

Items to consider regarding industry application performance are as follows:

- ▶ Server system processor performance
- ▶ Memory channel bandwidth and performance
- ▶ Network bandwidth performance
- ▶ System bus architecture and bandwidth on each adapter
- ▶ File system block size
- ▶ Single and multiple streams with large data blocks
- ▶ Checkpoint and restart options with large and small I/O requests
- ▶ Checkpoint and restart on large file count
- ▶ Parallel checking through the directory trees
- ▶ Random stat() system call to files in the file system

2.4.1 High-Performance Computing

This section provides solution details for the High Performance Computing industry.

Introduction to the Abaqus solution

The Abaqus FEA suite of software is known for its high performance, quality, and ability to solve more kinds of challenging simulations than any other software. The Abaqus suite consists of four core products: Abaqus/Standard, Abaqus/Explicit, and Abaqus/CFD, and Abaqus/CAE. Each package offers optional modules that address specialized capabilities that customer might need. The core products are described as follows:

- ▶ Abaqus/CFD provides advanced computational fluid dynamics capabilities with extensive support for pre-processing and post-processing provided in Abaqus/CAE. These scalable parallel CFD simulation capabilities address a broad range of nonlinear coupled fluid-thermal and fluid-structural problems.
- ▶ Abaqus/Standard provides analysis technology for the solution of traditional implicit finite element analyses (FEA), including static or dynamic stress/displacement, heat transfer,

mass diffusion, and acoustics. All are modeled with the widest range of contact and nonlinear material options. Abaqus/Standard also offers optional add-on and interface products to address design sensitivity analysis, offshore engineering, and integration with third-party software to perform calculations such as plastic injection molding analysis.

- ▶ Abaqus/Explicit implements an explicit method for high-speed, nonlinear, transient response and multiphysics applications. It is appropriate for many applications such as drop-test, and crushing and manufacturing processes.
- ▶ Abaqus/CAE provides a complete modeling and visualization environment for Abaqus analysis products. With direct access to CAD models, advanced meshing and visualization, and with an exclusive view towards Abaqus analysis products, Abaqus/CAE is the modeling environment of choice for many Abaqus users.

Disk requirements

The I/O requirement for Abaqus is high because of the application's use of scratch file space during computation. Therefore, adequate I/O (such as disk) capacity and performance are critical to ensure efficient operation. For clusters of all sizes, a high-performance file storage system (such as external Fibre Channel or NAS systems, or striped local disks on the head node) must be provided to ensure adequate I/O capability.

Introduction to the Fluent solution

Computational Fluid Dynamics (CFD) is the science of predicting fluid flow, heat and mass transfer, chemical reactions, and related phenomena by numerically solving the set of governing mathematical equations (conservation of mass, momentum, energy, and others). FLUENT is a leading computer program for modeling fluid flow and heat transfer in complex geometries. FLUENT provides complete mesh flexibility, solving flow problems with unstructured meshes that can be generated about complex geometries with relative ease.

Disk requirements

The FLUENT computation requires 100 MB of disk storage per million grid-cells on the head node that acts as a file server for FLUENT runs on the cluster. For example, 5 GB is required for a 50-million-cell model. Typically, users maintain several versions of each model throughout a project. Therefore, providing significantly more disk storage than is required for a single model is desirable. For the small or medium size business cases, be sure that 1 TB of disk storage is provided on the head node. In most cases, the compute nodes can be configured with a smaller amount of disk storage; often a standard amount of 160 GB per compute node is sufficient. If consultation with the customer suggests that additional archival and storage capability is needed, propose a more robust external storage.

Introduction to the petroleum industry solution

The larger energy companies have their own research personnel and perform much of their own code development, although the companies might place that code on top of an infrastructure system that is provided by an independent software vendor (ISV). Many algorithms are employed, but those that are most compute-intensive are the migration or imaging codes, which account for probably 90% of the compute-cycles used.

Disk requirements

Although the algorithms are floating-point intensive and require modest per-node I/O rates, many independent streams can be processed in parallel, leading to a workload that looks fairly random to GPFS. Most upstream seismic space is dominated by large block streaming sequential I/O, but several algorithms require non-sequential access (some are strided, some are pseudo-random, using block sizes in the 8 - 32 KB range. Although clusters are generally large (hundreds to thousands of nodes), there is relatively little *true* parallel access (such as

multiple nodes writing the same file). File sizes are very large (tens to hundreds of gigabytes per file); reuse is low, so cache algorithms are generally not helpful.

2.4.2 RDBMS

This section provides details for when the industry solution calls for a database implementation with GPFS.

OLTP and OLAP (decision support systems)

Online transaction processing (OLTP) databases are among the most mission-critical and widely deployed. The primary defining characteristic of OLTP systems is that the transactions are processed in real time or online and often require immediate response back to the user.

Examples include:

- ▶ A point of sale terminal in a retail business
- ▶ An automated teller machine (ATM) that is used for bank transactions
- ▶ A telemarketing site that processes sales orders and reviews the inventories

From a workload perspective, OLTP databases typically have the following characteristics:

- ▶ Process a large number of concurrent user sessions.
- ▶ Process a large number of transactions using simple SQL statements.
- ▶ Process a single database row at a time.
- ▶ Are expected to complete transactions in seconds, not minutes or hours.

OLTP systems process the daily operations of businesses and, therefore, have strict user response and availability requirements. They also have extremely high throughput requirements and are characterized by large numbers of database inserts and updates. They typically serve hundreds, or even thousands, of concurrent users.

Online analytical processing (OLAP) differs from the typical transaction-oriented systems in that they most often use data that is extracted from multiple sources for the purpose of supporting user decision-making. The types of processing are as follows:

- ▶ Data analysis applications using predefined queries
- ▶ Application-generated queries
- ▶ Ad hoc user queries
- ▶ Reporting requirements

OLAP systems typically deal with substantially larger volumes of data than OLTP systems because of their role in supplying users with large amounts of historical data. Whereas 100 GB of data is considered large for an OLTP environment, a large OLAP system might be 1 TB of data or more. The increased storage requirements of OLAP systems can also be attributed to the fact that they often contain multiple, aggregated views of the same data. Although OLTP queries are mostly related to one specific business function, OLAP queries are often substantially more complex. The need to process large amounts of data results in many CPU-intensive database sort and join operations. The complexity and variability of these types of queries must be given special consideration when estimating the performance of an OLAP system.

Considerations of GPFS for DB2

Consider the following information when implementing GPFS for DB2:

- ▶ Where performance and reliability is a concern, use two GPFS file systems because of the widely varying access patterns between data and log access. Create one for database data and the other for the database logs.
- ▶ Where simplicity of administration is the priority, use a single GPFS file system. Using a single GPFS file system for data minimizes the administrative overhead. A single file system combined with DB2 automatic storage allows you to dynamically add storage capacity and even additional hosts without having to create another file system.
- ▶ Use a GPFS file system block size of 1 MB. Larger block sizes use the page pool more efficiently.
- ▶ Use direct I/O (DIO). DB2 version 9.7 can directly exploit direct I/O in GPFS. As of version 9.7, all DB2 files, including the active log files, are opened with DIO, allowing for faster access to the disks and avoiding double buffering of data in DB2 and the file system cache. This way, DB2 chooses the files that must be accessed and the data that must be placed in cache. Allowing DB2 to open the files using DIO is preferred over mounting the entire file system in DIO mode.
- ▶ Enable higher concurrency. Increase the value of the GPFS worker1Threads parameter from the default of 40 to 256. This parameter controls the maximum number of concurrent file operations on a GPFS file system. The sum of the prefetchThreads parameter value and the worker1Threads parameter value must be less than 550.
- ▶ Increase file system cache. The GPFS page pool is used to cache user data, file system metadata and other internal data. Increase the size of the page pool by changing the value of the page pool parameter from the default of 64 MB to at least 256 MB. The page pool is pinned in memory, so make sure that there is enough free memory available on the system, otherwise this change can result in system paging and can negatively affect overall performance.

IBM pureScale solution

In December 2009, the IBM DB2 pureScale™ feature for enterprise server edition was introduced. This solution takes advantage of an active-active shared-disk database implementation based on the DB2 for z/OS® data sharing architecture. It takes advantage of proven technology from DB2 database software on the mainframe to bring the active-active shared-disk technology to open systems.

The DB2 pureScale feature meets the needs of many customers by providing the following key benefits:

- ▶ Virtual unlimited capacity
 - Provides practically unlimited capacity by enabling the additional and removal of members on demand.
 - Scales to 128 member server.
 - Offers highly efficient and centralized management facility.
 - Takes advantage of a technology called Remote Direct Memory Access (RDMA)
- ▶ Application transparency
 - Does not need to have any knowledge of the difference members.
 - Automatically routes applications to the members deemed most appropriate.
 - Offers native support for syntax that order databases vendor use.

- Continuous availability
 - Offers a fully active-active configuration such that if one member goes down, processing continues at the remaining active member.
 - During a failure, only data being modified on the failing member is temporarily unavailable.
- Reduced total cost of ownership

The DB2 pureScale interfaces easily handle the deployment and maintenance of components integrated within the DB2 pureScale feature.

Figure 2-11 shows that the client can connect to any member. The DB2 pureScale feature can automatically locally balance the client across the separate members based on the usage of each computer. If any host fails in the configuration, the DB2 pureScale feature redirects clients among the active members on the remaining hosts.

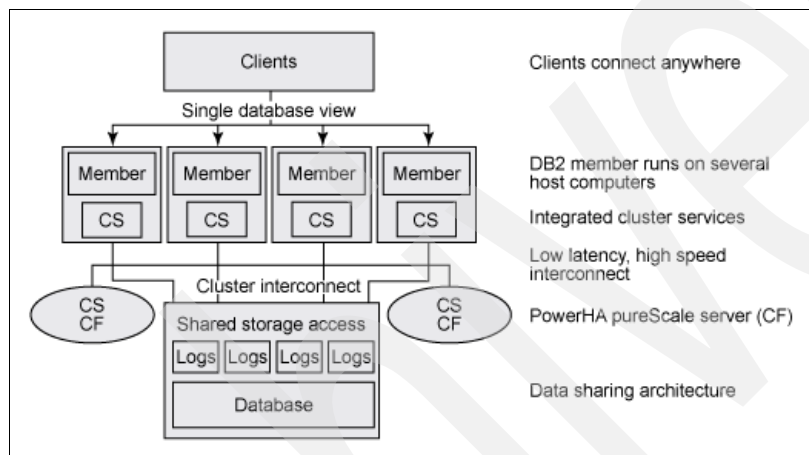


Figure 2-11 DB2 pureScale feature topology overview

Extract, transform, and load (ETL) for data warehousing

ETL processing can be optimized using a GPFS infrastructure to manage the data. GPFS can be used to provide high performance data access to a common set of data for grid ETL processing and loading into the data warehouse.

Figure 2-12 shows the architecture of the ETL solution with GPFS.

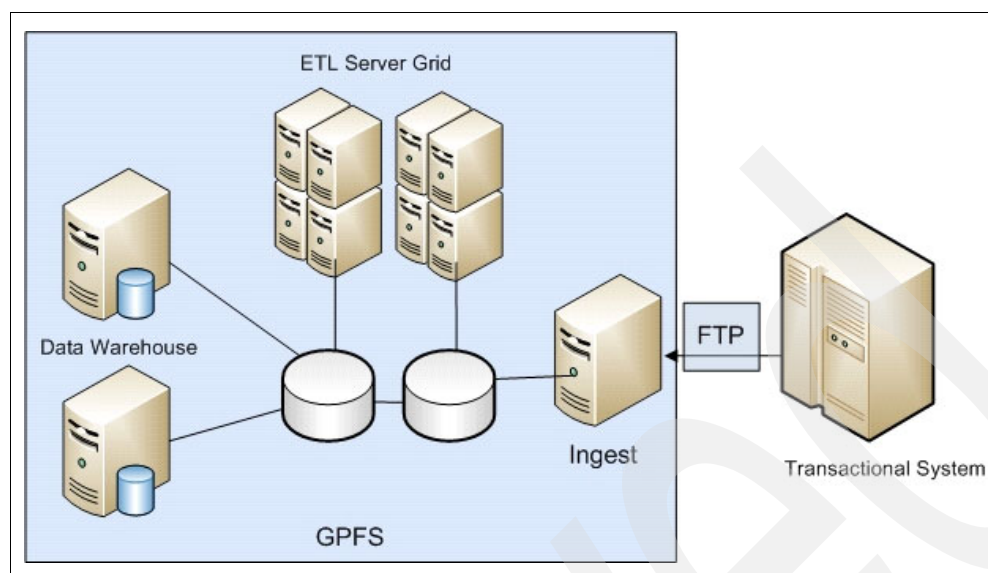


Figure 2-12 Design ETL solution with GPFS

A typical scenario includes a transaction processing system that is generating the data. One or more ETL systems are transforming the transactional data into a data mining format. Then, finally the data is loaded into a data warehouse for data mining. Many solutions today include a group of loosely connected machines that are processing the data. GPFS can greatly improve the efficiency of ETL processing by enabling effective grid ETL processing, without requiring duplication of data sets and direct load into the data warehouse.

IBM Smart Analytics System solution

IBM Smart Analytics System offers faster business value at lower cost by combining the simplicity and rapid deployment characteristics of an appliance, and retains the flexibility of a custom integration approach to take advantage of ongoing hardware and software enhancements without full system replacement.

IBM Smart Analytics System was developed based on customer data analysis needs. It maintains the high availability and mission-critical reliability that are essential for today's environment. It is designed to simplify the delivery of information to the people in the business that need it the most, helping to minimize the cost of building a complex architecture to support analytical applications.

Several key features of IBM Smart Analytics System are as follows:

- ▶ Powerful and flexible analytics in a package that simplifies deployment and speeds business results, meeting a broad spectrum of line-of-business analytic needs, and lowering the IT costs to do so
- ▶ Modular and scalable approach that is based on an integrated solution combining IBM leading database management software, Power Systems, and IBM storage
- ▶ Comprehensive, integrated solution consisting of a platform with optional modules depending on client requirements

All systems are optimized and pretested and packaged in several preconfigured sizes to simplify ordering and delivery.

- ▶ Highly available and reliable system with built-in failover and near zero planned downtime incorporated from the beginning.

- ▶ Modular Analytics capacity
- ▶ Modular data warehouse processing capacity
- ▶ Modular storage capacity
- ▶ Modular user and administration capacity
- ▶ Advanced workload management and monitoring

IBM Smart Analytics System offers a wide range of analytics capabilities, enabling you to consume information in the most digestible format, gain insight, and make smarter decisions today and in the future. At the core of IBM Smart Analytics System are a powerful warehouse and storage optimization capabilities. This foundation manages the data store, and is essential for speeding system deployment and enabling advanced analytics.

The IBM Smart Analytics System and Software Stack are as follows:

- ▶ AIX
- ▶ GPFS
- ▶ Tivoli System Automation
- ▶ InfoSphere™ Warehouse
- ▶ DB2
- ▶ Cognos®
- ▶ WebSphere® Application Server

Note: See the following resource for more information about IBM Smart Analytics System:

<http://www.ibm.com/software/data/infosphere/support/smart-analytics-system/>

Oracle RAC with GPFS solution

GPFS is a high-performance shared-disk file system that provides data access from all nodes in a cluster environment. Parallel and serial applications can access the files on the shared disk space using standard UNIX file system interfaces. The same file can be accessed concurrently from multiple nodes (single name space). GPFS is designed to provide high availability through logging and replication. It can be configured for fail over from both disk and server malfunctions.

GPFS greatly simplifies the installation and administration of Oracle 10g RAC. Because GPFS is a shared file system, all database files can be placed in one common directory, and database administrators can use the file system as a typical journaled file system (JFS) and enhanced JFS (JFS2). Allocation of new data files or resizing existing files does not require system administrator intervention. Free space on GPFS is seen as a traditional file system that is easily monitored by administrators.

Moreover, with GPFS you can keep a single image of Oracle binary files and share them between all cluster nodes. This single image applies both to Oracle database binaries (ORACLE_HOME) and Oracle Clusterware binary files. This approach simplifies maintenance operations, such as applying patch sets and unique patches, and keeps all sets of log files and installation media in one common space.

For clients running Oracle Applications (eBusiness Suite) with multiple application tier nodes, using GPFS as a shared APPL_TOP file system is also possible and convenient.

In GPFS Version 2.3, IBM introduces cluster topology services within GPFS. Thus, for GPFS configuration, other clustering layers, such as HACMP or RSCT, are no longer required.

Advantages of running Oracle 11g RAC on a GPFS file system are as follows:

- ▶ Simplified installation and configuration
- ▶ Possibility of using AUTOEXTEND option for Oracle data files, similar to JFS/JFS2 installation
- ▶ Ease of monitoring free space for data files storage
- ▶ Ease of running cold backups and restores, similar to a traditional file system
- ▶ Capability to place Oracle binaries on a shared file system, thus making the patching process easier and faster

You can locate every Oracle 11g RAC file type (for database and clusterware products) on the GPFS, which includes the following types:

- ▶ Clusterware binaries
- ▶ Clusterware registry files
- ▶ Clusterware voting files
- ▶ Database binary files
- ▶ Database initialization files (init.ora or spfile)
- ▶ Control files
- ▶ Data files
- ▶ Redo log files
- ▶ Archived redo log files
- ▶ Flashback recovery area files

You can use the GPFS to store database backups. In that case, you can perform the restore process from any available cluster node. You can locate other non-database-related files on the same GPFS also.

Figure 2-13 shows a diagram of Oracle RAC on GPFS architecture. All files that are related to Oracle Clusterware and database are located on the GPFS.

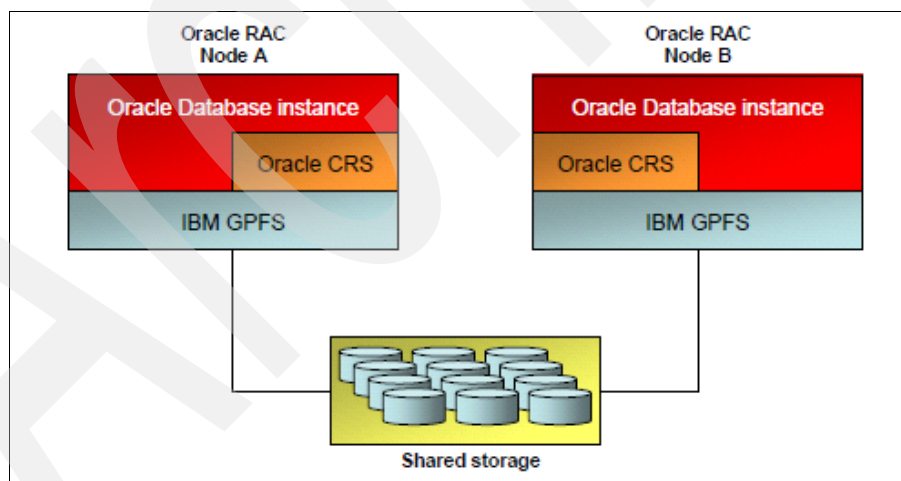


Figure 2-13 Oracle RAC on GPFS architecture

Note: Oracle Cluster Ready Services (CRS) configuration and vote devices are located with GPFS.

CRS and vote data must be located on the GPFS or raw device. When located on the GPFS volume, they are more easy to control. When located on raw devices, there are more management points such as dump this raw device and store on GPFS volume or tape drives.

Figure 2-14 shows the architecture with CRS devices outside the GPFS.

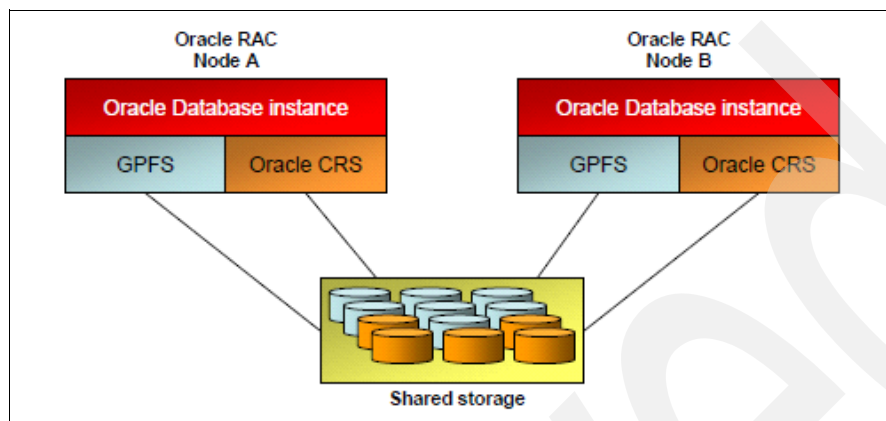


Figure 2-14 Oracle RAC on GPFS architecture with Raw device

Note: CRS configuration and vote devices located in raw physical volumes.

SAP NetWeaver Business Warehouse (BW) Accelerator

IBM and SAP offer an integrated solution that enables customers to deploy SAP NetWeaver BW Accelerator in a cost-effective way as a snap-in solution to an existing SAP NetWeaver BW landscape. Included with Quad-Core Intel Xeon processor-based blade servers, the IBM Systems Solution for SAP NetWeaver BW Accelerator provides the solution to achieve near real-time analytics for the SAP NetWeaver Business Warehouse. The IBM Systems solutions for SAP NetWeaver BW Accelerator crunches through terabytes of SAP NetWeaver BW data in a matter of seconds, without the need for dedicated database administrators building and maintaining predefined aggregates or spending valuable time on query performance tuning. IBM automated High Availability solution for BW Accelerator allows customers to implement BW Accelerator as a mission-critical system, at minimal effort using standard functionality.

Figure 2-15 shows IBM BladeCenter technology-based SAP Netweaver BW Accelerator appliance, consisting of SUSE Enterprise Linux, GPFS, and SAP BWA software stack.

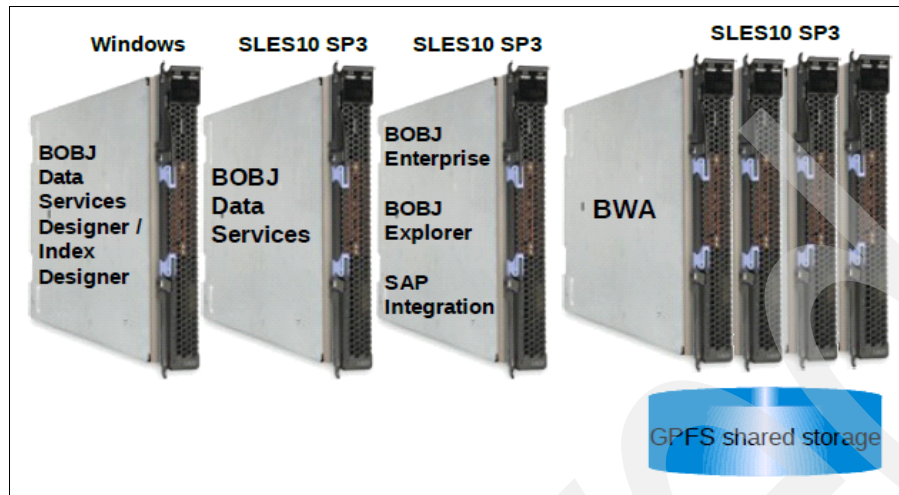


Figure 2-15 SAP Netweaver BW Accelerator appliance Design

SAP NetWeaver BW Accelerator clearly extends traditional SAP NetWeaver BW software by delivering the following benefits:

- ▶ Offers query processing up to 150 times faster than traditional approaches
- ▶ Offers data compression combined with In-memory processing and search engine-based parallel processing delivered with scalability, speed, flexibility and business insight.
- ▶ Empowers SAP NetWeaver BW users with turbo-charged insights and decision support.

For SAP BW Accelerator, node quorum with tiebreaker disks allows running with as little as one quorum node available, as long as having access to a majority of the quorum disks is possible. Switching to quorum with tiebreaker disks is accomplished by indicating a list of one to three disks as values for the tiebreakerDisks parameter, defined or changed with the `mmchconfig` command. Be sure to use tiebreaker disk (or disks) and configure two quorum nodes, both to be used as primary and secondary GPFS nodes, and for master roles in SAP NetWeaver BW Accelerator also.

Because GPFS does not recognize increasing of array size, adding more arrays and configuring GPFS to recognize the new disks make more sense. Use the additional disks for data only.

Note: For more information, see the following resource:

<http://www.ibm.com/solutions/sap/us/en/>

2.4.3 Web application server

The representative web application solutions are the IBM WebSphere, the Red Hat JBoss and the Oracle WebLogic. These solutions have almost the same I/O pattern with a web server. Consider the following parameters for GPFS random I/O:

- ▶ **pagepool:** The default pagepool parameter size is typically not sufficient for random I/O or workloads involving a large number of small files. In certain cases, allocating 4 GB, 8 GB or more memory can improve workload performance.

Note: This is the pagepool on the GPFS client and not the GPFS server.

- ▶ **prefetchPct:** This parameter defaults to 20% of pagepool. GPFS uses this value as a guideline to limit how much pagepool space will be used for prefetch or write-behind buffers in the case of active sequential streams. The default works well for many applications. Alternatively, if the workload is mostly sequential (video serving and ingest) with little caching of small files or random I/O, increase this number up to its 60% maximum so that each stream can have more buffers available for prefetch and write-behind operations.
- ▶ **worker1Threads:** This parameter controls the maximum number of concurrent file operations at any one instant. If there are more requests than that, the excess waits until a previous request has finished. This attribute is primarily used for random read or write requests that cannot be pre-fetched, random I/O requests, or small file activity. The default value is 48. The minimum value is 1. The maximum value of prefetchThreads plus worker1Threads is 550 on 64-bit kernels.
- ▶ **blocksize:** A smaller block size is likely to offer better performance for small file, small random read and write, and metadata-intensive workloads.

2.4.4 Media contents delivery

This section provides details about media delivery contents in a GPFS cluster environment.

3D animation development infrastructure

How do you design and build this type of system? If you start with no studio and want to create a full-length feature animation studio, what kind of infrastructure will you need?

Figure 2-16 shows the basic architecture for a 3D animation development studio.

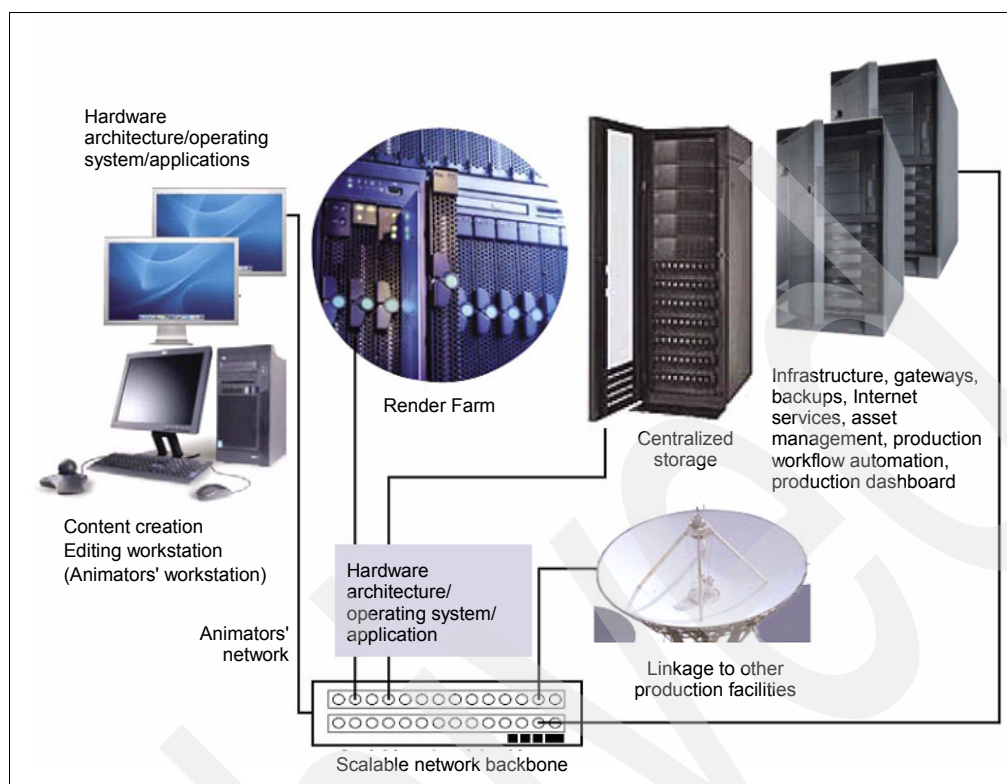


Figure 2-16 Basic architecture for an animation studio

The workflow infrastructure consists of four subsystems:

- Content creation, editing workstations

Workstations are high-performance desktops, and in some cases, mobile computers. As mentioned, the process of creating rich digital content, whether it is video for TV, cinematic film, or graphics for interactive entertainment (games) can be divided into two basic activities, content creation and content editing:

- *Content creation* is the process of creating 2D or 3D images, animations, backgrounds, textures, and so forth.
- *Content editing* is the processing and sequencing of motion sequences into a final form. Both are I/O-intensive graphics processes.

- Render farm

A *render farm* is an industry term for what computer scientists might call a loosely coupled supercomputer. Render farm technology arose as a way to meet the needs of organizations attempting to produce compelling content in a reasonable amount of time. The basics are fairly simple to grasp. Special effects and animation are collections of individual pictures or frames. The process of turning a model into a fully lit 2D or 3D picture is called rendering, which is a highly compute-intensive task. Although this area usually runs on a Linux platform, 3D media content creation is usually a Windows platform. Previously, NFS was used for sharing data. GPFS is now a good solution because the GPFS cluster provides a mixed platform solution with Linux and Windows servers.

- **Centralized storage**

Rich content consumes massive amounts of storage. Centralizing storage offers many operational benefits. For instance, when assets are stored centrally, they can easily be tracked, monitored and reused. This leads to improved production efficiencies, which lead to shorter production times. Projects can be delivered in a more predictable time frame.

GPFS provides file data access from all nodes participating in the cluster by providing a global name space for files. Applications can efficiently access files using standard UNIX file system interfaces; GPFS supplies the data to any location in the cluster. All application compute nodes appear directly connected to all storage and any client can access information using the combined bandwidth of all storage devices. GPFS allows all compute nodes to have coherent and concurrent access to all storage, which is important in video production.

- **Network**

Binding these components together is the network. However, the importance of the network is often underestimated. Video editing alone can easily consume all available network bandwidth. Although many companies understand that their operations require a 10 Gb backbone, they balk at the initial cost of the equipment. So they buy cheap network equipment at the beginning of a project, only to find they have to rip and replace their entire network infrastructure after production of a project moves into full swing.

2.4.5 Commercial industry

This section provides details of how GPFS provides a solution for customers in the commercial arena.

SONAS Solution

Scale out Network Attached Storage (SONAS) can address the new storage challenges that are posed by the continuing explosion of data. Taking advantage of mature technology from IBM High Performance Computing experience, and based on the IBM General Parallel File System (GPFS), SONAS is an easy-to-install, turnkey, modular, scale-out NAS solution that provides the performance, clustered scalability, high availability and functionality that are essential to meeting strategic *petabyte age* and cloud storage requirements.

Figure 2-17 on page 36 shows that SONAS supports a full complement of standards-based network protocols, including Common Internet File System (CIFS), Network File System (NFS), Secure Copy Protocol (SCP), Hypertext Transfer Protocol (HTTP), and File Transfer Protocol (FTP).

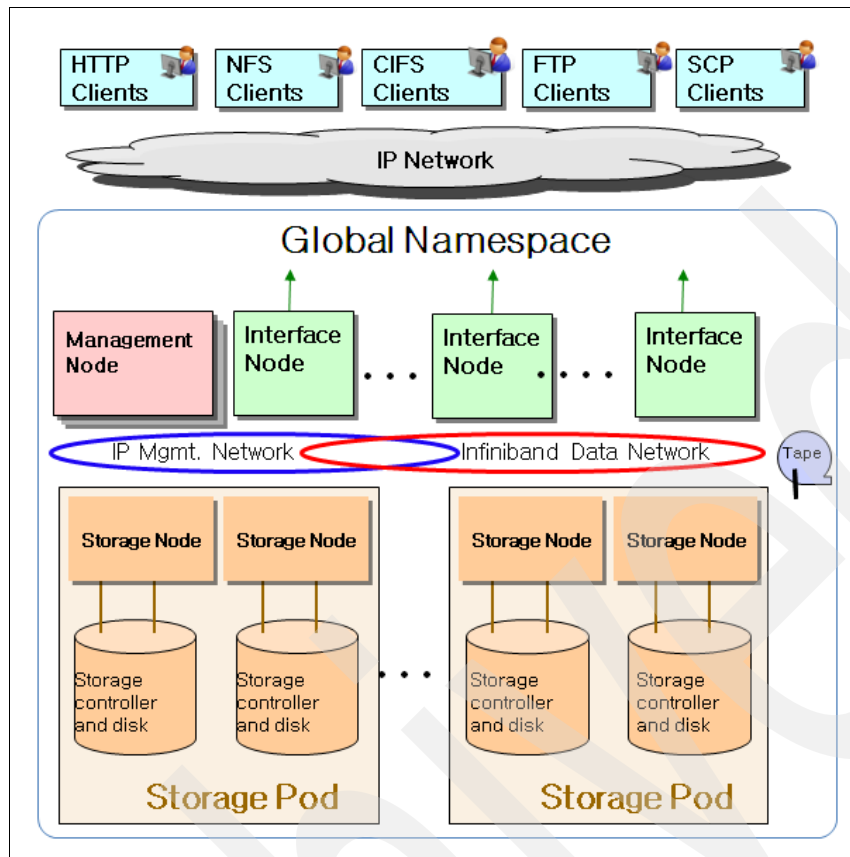


Figure 2-17 IBM SONAS solution architecture

The high-density, high-performance SONAS can help organizations consolidate and manage data affordably, reduce crowded floor space, and reduce management expense that is associated with administering an excessive number of disparate storage systems. With its advanced architecture, SONAS virtualizes and consolidates multiple files into a single, enterprise-wide file system, which can translate into reduced total cost of ownership, reduced capital expenditure, and enhanced operational efficiency.

SONAS provides a global namespace that enables your storage infrastructure to scale to extreme amounts of data, from terabytes to petabytes. Within the solution, centralized management, provisioning, control, and automated information lifecycle management (ILM) are integrated as standard features to provide the foundation for a truly cloud storage-enabled solution.

In addition, the system supports traditional NAS features and functions:

- ▶ Snapshots
- ▶ Quotas (user, group, and file set level)
- ▶ Integration with user directory servers such as Microsoft Active Directory (AD) and Lightweight Directory Access Protocol (LDAP)
- ▶ Command-line interface (CLI) and browser-based graphical user interface (GUI)
- ▶ Call-home and remote support capabilities

In addition to these traditional features, the system supports advanced features, including integrated Tivoli Storage Manager backup client.

SONAS provides extreme scale out capability, a globally clustered NAS file system built upon IBM GPFS. The global namespace is maintained across the entire global cluster of multiple storage pods and multiple interface nodes. All interface nodes and all storage nodes share equally in the cluster to balance workloads dynamically and provide parallel performance to all users and storage, ensuring high availability and automated failover.

GMAS

As doctors and patients demand more frequent scans, the number of medical images is growing exponentially, dramatically increasing storage and administrative costs for providers. Medical facilities that have not yet transitioned to digital formats must grapple with the time and labor costs of handling film. Those enterprises that have made the transition to picture archiving and communication systems (PACS) face exploding storage volumes, rising expenses and slowing image retrieval time, even as pressure grows to retain physicians, meet tightening margins and provide remote access to patient images.

The IBM Healthcare and Life Science's Grid Medical Archive Solution (GMAS) employs an unparalleled combination of features to cost-effectively deliver enterprise-wide medical image access with real-time business continuity. GMAS is a storage solution that intelligently manages the interaction between an application and its storage resources through client business rules. GMAS is an implementation of Grid Access Manager within the health-care industry. GMAS consists of the following components:

- ▶ Grid Access Manager software
- ▶ IBM System x Servers
- ▶ IBM Storage and GPFS
- ▶ IBM GTS Services

Figure 2-18 shows the IBM Grid Medical Archive Solution Infrastructure diagram.

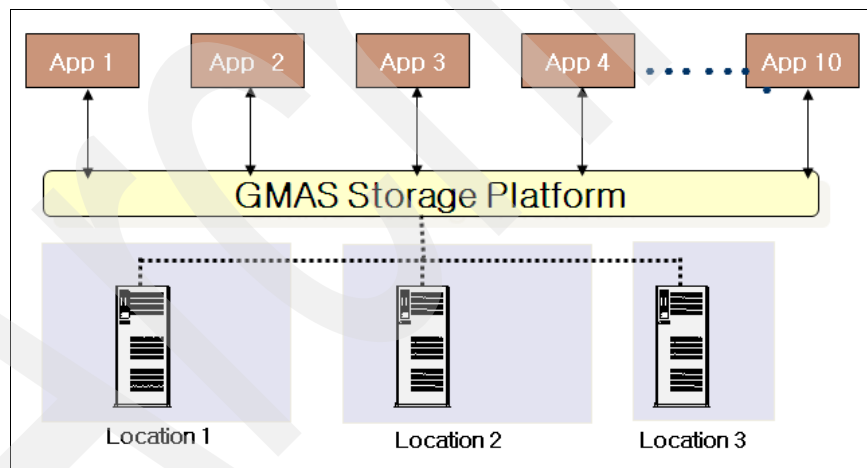


Figure 2-18 IBM GMAS storage platform with medical application such as PACS

These components combine to deliver to customers a completely operational solution for reference data storage requirements. The most common implementation of GMAS is one in which IBM provides all of the components. GMAS protects data and simplifies the delivery and operation of reference data storage systems, as follows:

- ▶ Protects data for life
 - Uses digital signatures and proactive checking; no data is lost.
- ▶ Provides an enterprise solution.
 - Supports all sites (LAN/WAN), applications, and storage tiers.

- ▶ Improves availability and uptime of applications.
 - Uses real time failover, automated rebuild, and self healing.
 - Prevents downtime; changes are transparent to applications.
- ▶ Automates storage administration.
 - Includes ILM, data replication, upgrades, and data migrations.

Desktop virtualization solution

Virtualization technology is useful for the finance and design industry. It can simplify the company's IT infrastructure, resolve security problems, and provides a centralized management infrastructure. This solution works with Intel-based x64 server system. GPFS provides a single namespace. Also, live migration technology can be used with a shared storage solution.

Figure 2-19 shows a virtual desktop infrastructure with the GPFS solution.

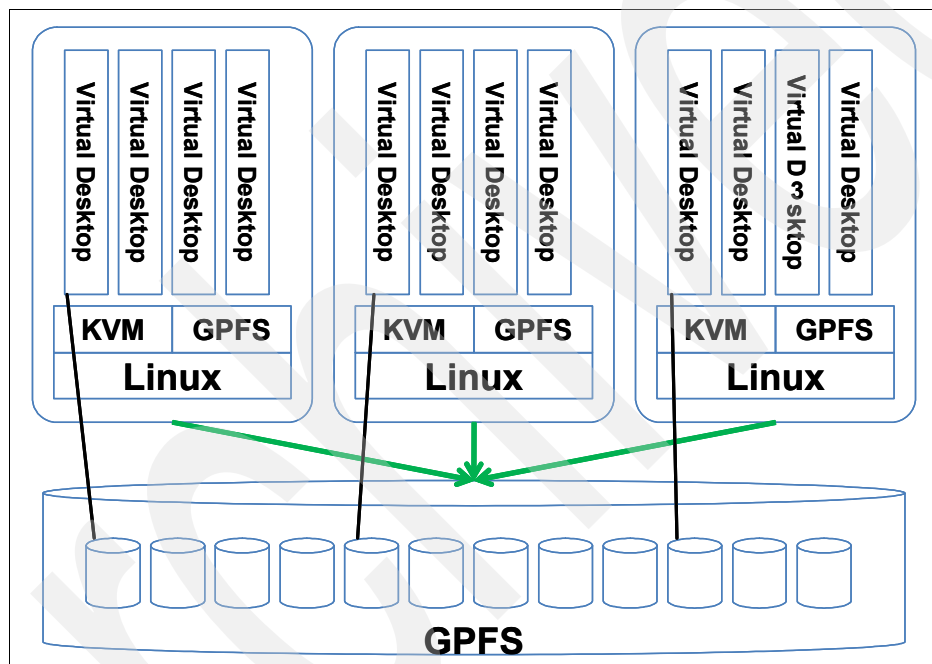


Figure 2-19 Design of virtual desktop based on KVM with GPFS

Note: IBM GPFS does not working on XEN and VMware based kernels.

2.5 GPFS planning guidance

Previous sections described fabric technologies and several industry solutions. GPFS can provide various configuration for customer solutions. The following three configuration types are sample GPFS solutions, which are classified by the location of the customer application on GPFS:

- ▶ Running on network-based GPFS client
- ▶ Running on SAN-based GPFS server
- ▶ Running on mixed NSD server and client

Figure 2-20 shows the customer application running on GPFS NSD client side, which means this application accesses GPFS volume through network-based architecture. Application performance depends on the interconnect Ethernet fabric speed.

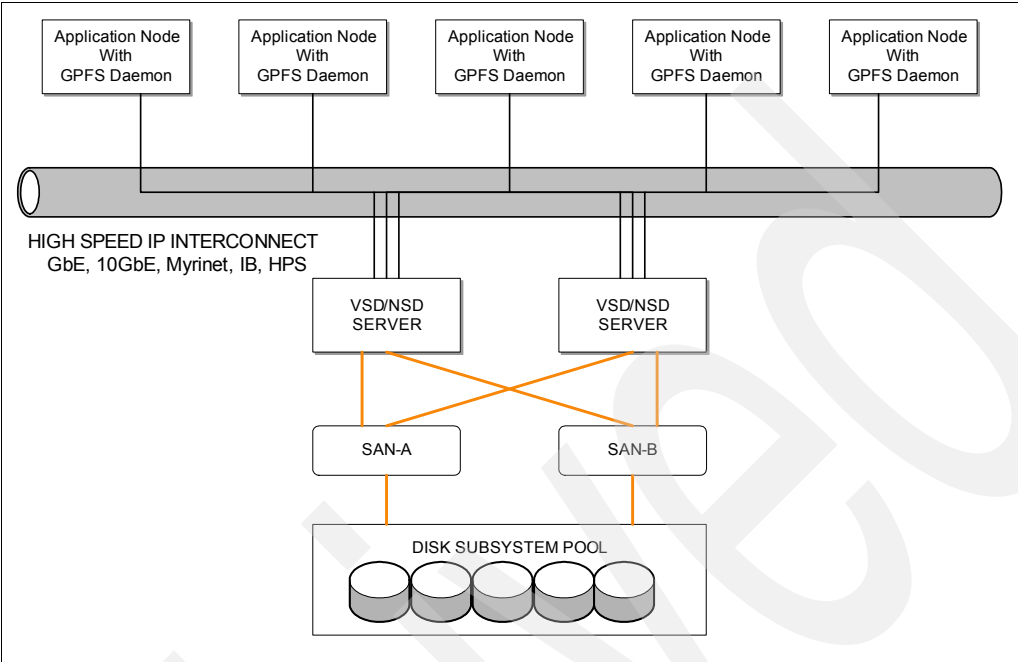


Figure 2-20 Network storage access configuration

Figure 2-21 shows the customer application running on GPFS server with no NSD configuration, which means this application accesses the GPFS volume through SAN. This case application uses full storage performance and provides faster recovery time, when a node failure occurs, than another architecture.

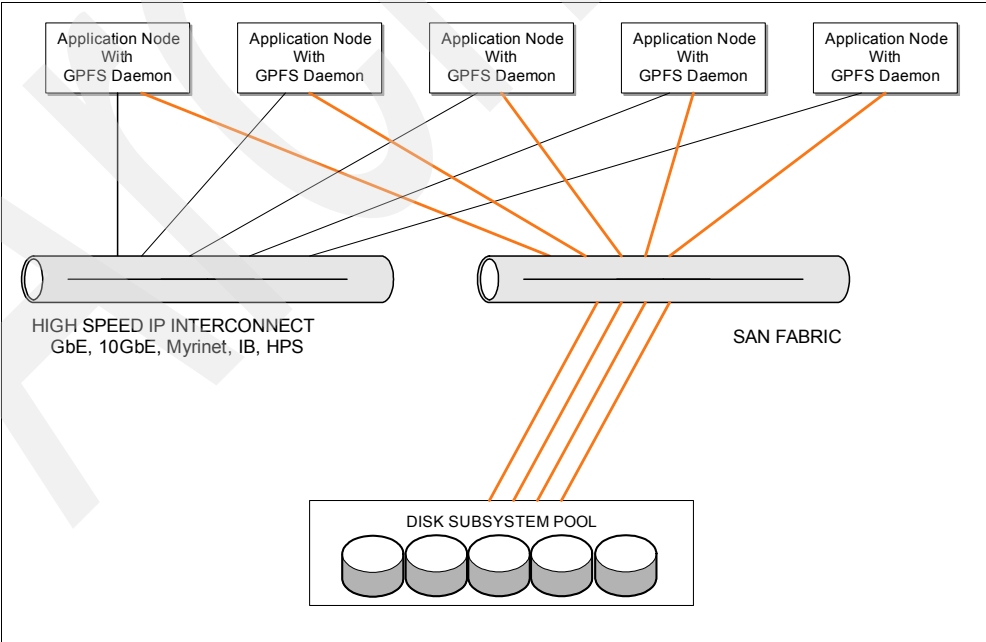


Figure 2-21 Direct storage access configuration

Figure 2-22 shows the customer application running on GPFS NSD server and client; this application accesses the SAN and the network concurrently. This is assigned by the characteristic of the application. A good example of this application is Tivoli Storage Manager. In this case, Tivoli Storage Manager can directly back up and restore all GPFS file systems of this cluster, which means that the Tivoli Storage Manager can access all the GPFS volumes. Therefore, design the GPFS solution from a customer application perspective.

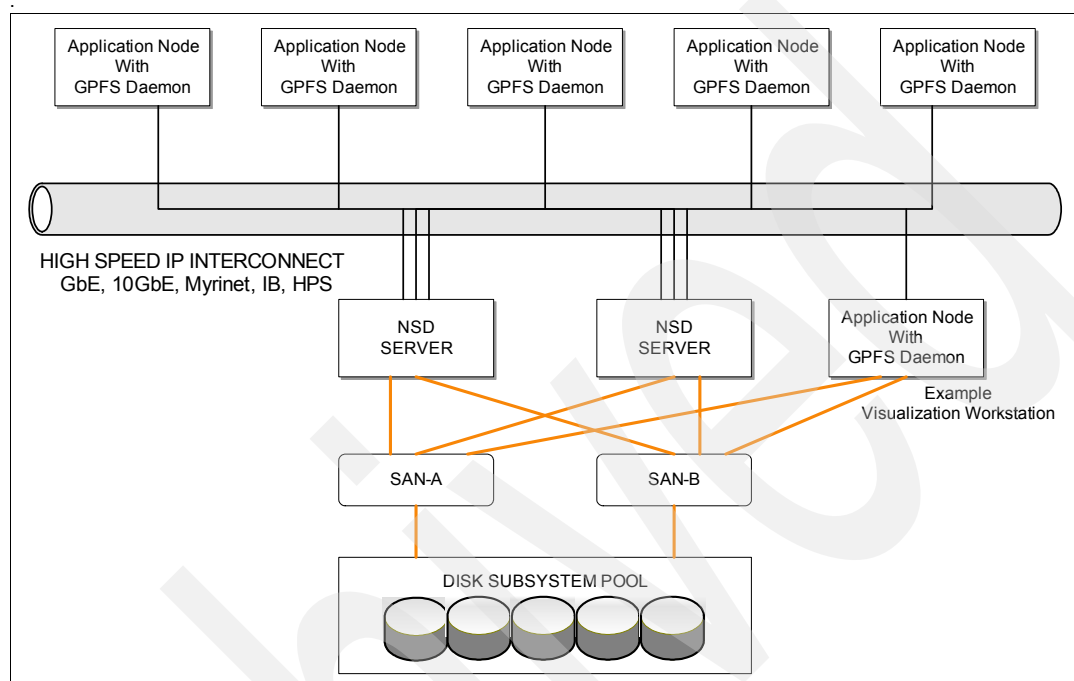


Figure 2-22 Mixed configuration with NSD server/client

2.5.1 Operating system and file systems with multiplatform GPFS

From the GPFS point of view, all configuration commands in a mixed operating system environment are the same. In an AIX and Linux environment, the commands are almost the same; in a Windows environment not all the cluster commands are provided for the GPFS management functions. This section describes the supported operating systems and what GPFS functions are disabled in a mixed cluster.

AIX

GPFS v3.4 supports the following AIX versions:

- ▶ AIX V6.1
- ▶ AIX 5L™ 5.3

Requirements for GPFS are as follows:

- ▶ The following additional file sets are required by GPFS V3.4:
 - xlc.aix50.rte (C Set ++ Runtime for AIX 5.0), version 8.0.0.0 or later
 - xlc.rte (C Set ++ Runtime), version 8.0.0.0 or later
- ▶ Enhancements to the support of Network File System (NFS) V4 in GPFS are only available on AIX V5.3 systems with the minimum technology level of 5300-04 applied or on AIX V6.1 with GPFS V3.4.
- ▶ The version of OpenSSL included with certain releases of AIX V6.1 and AIX V5.3 do not work with GPFS because of a change in how the library is built.

Linux on System p and System x

GPFS V3.4 supports the following distributions:

- ▶ RHEL4, RHEL5, RHEL6
- ▶ SLES9, SLES10, SLES11

Requirements for GPFS are as follows:

- ▶ RHEL 5.0 and later on POWER requires GPFS V3.2.0.2 or later; With RHEL5.1, the automount option is slow. This issue will be addressed in a subsequent kernel.
- ▶ GPFS for Linux on POWER does not support mounting of a file system with a 16 KB block size when running on either RHEL 5 or SLES 11.
- ▶ The GPFS V3.2 GPL build requires **imake** makefile generator.
- ▶ If **imake** was not installed on the SLES 10 or SLES 11 system, install the `xorg-x11-devel-*.rpm` file.
- ▶ GPFS V3.2.1.13 or later supports SLES 11.
- ▶ GPFS V3.3 supports SLES 9 SP 3 or later.
- ▶ GPFS V3.3.0-5 or later supports SLES 10 SP3.
- ▶ GPFS does not support SLES 10 SP3 on POWER 4 machines.
- ▶ Service is required for Linux kernels 2.6.30 or later, or on RHEL5.4 (2.6.18-164.11.1.el5).
- ▶ Not supported are the following Linux kernels: RHEL hugemem, RHEL largesmp, RHEL uniprocessor (UP), SLES xen.

Windows x64 platform

GPFS V3.4 supports all editions of the following operating systems:

- ▶ Windows 2008 R2 x64 Edition
- ▶ Windows 2008 SP2 x64 Edition

Also, the following Windows GPFS cluster characteristics apply:

- ▶ When the configuration is created by using the defaults administrator account. Also, remote shell operations are `administrator@nodename`, using **ssh** or **winscp** commands.
- ▶ This configuration cannot be mixed with other operating systems because UNIX and Linux do not have an administrator account, and do not require adding root user.
- ▶ The member server with Windows Active Directory infrastructure or Windows stand alone server will join Windows GPFS Cluster.
- ▶ The Windows GPFS node will be the NSD Server or the NSD Client.

In a GPFS mixed operating system cluster, the following characteristics apply:

- ▶ Add root user running as administrator, which requires changes to the operating system policy.
- ▶ The GPFS daemon is running as root user.
- ▶ The **ssh** daemon requires root user because the UNIX system default admin user is root.
- ▶ When adding a Windows GPFS node to a UNIX/Linux GPFS cluster, check which function is disabled in the UNIX/Linux side of the GPFS cluster.

GPFS considerations on Windows

GPFS for Windows does not fully support all of GPFS features that are available on AIX and Linux. Some of these limitations constrain how you can configure a GPFS cluster when it

includes Windows nodes. The remaining limitation only pertain to Windows nodes rather than the whole cluster.

The following GPFS commands and features are not supported on Windows:

- ▶ **Commands**
 - mmapplypolicy, mmbackup, mmbackupconfig, mmdefedquota, mmdelacl, mmeditacl, mmedquota, mmgetacl, mmlsquota, mmpmon, mmputacl, mmrepquota, mmrestoreconfig
- ▶ **Features**
 - The ILM/HSM functions are not supported.
 - GPFS does not support Microsoft Windows NT File System (NTFS) features such as encrypted files and directories, Transactional NTFS (TxF), defragmentation and error-checking tools.
 - Must use IPoIB with InfiniBand, GPFS 3.4 does not support RDMA.

Note: If you have a GPFS cluster that is based on Linux or AIX nodes, the cluster cannot join the Windows GPFS cluster. The reason is because a difference exists between the POSIX file system and the NTFS file systems, which the Windows GPFS version has as a limitation.

Virus scan and monitor program on Windows GPFS

If more than one GPFS Windows node is running antivirus software that scans directories and files, shared files only need to be scanned by one GPFS node. Scanning shared files more than once is not necessary. When you run antivirus scans from more than one node, schedule the scans to run at separate times to allow better performance of each scan, and to avoid any conflicts that might arise because of concurrent exclusive access attempts by the antivirus software from multiple nodes.

Notes:

- ▶ Enabling real-time antivirus protection for GPFS volumes might significantly degrade GPFS performance and cause excessive resource consumption
- ▶ Consider using a single, designated Windows node to perform all virus scans.

Difference between GPFS and NTFS

GPFS differs from the Microsoft Windows NT File System (NTFS) in its degree of integration into the Windows administrative environment, Windows Explorer, and the desktop. The differences are as follows:

- ▶ Manual refreshes are required with GPFS if you want to see any updates to the GPFS namespace. You cannot use the recycle bin.
- ▶ You cannot use distributed link tracking with GPFS. This is a technique through which shell shortcuts and OLE links continue to work after the target file is renamed or moved. Distributed link tracking can help you locate the link sources in case the link source is renamed or moved to another folder on the same or different volume on the same computer, or moved to a folder on any computer in the same domain.
- ▶ You cannot use NTFS change journaling with GPFS. This also means that you cannot use the Microsoft Indexing 3 Service or Windows Search Service to index and search files and folders on GPFS file systems.

GPFS does not support the following NTFS features:

- ▶ File compression (on individual files or on all files within a folder)
- ▶ Encrypted files and directories
- ▶ Quota management (GPFS quotas are administered through GPFS-specific commands)
- ▶ Reparse points, defragmentation, and error-checking tools
- ▶ Alternate data streams and a change journal for file activity
- ▶ The assignment of an access control list (ACL) for the entire drive
- ▶ The scanning of all files or directories that a particular SID owns (FSCTL_FIND_FILES_BY_SID)
- ▶ Generation of AUDIT and ALARM events that are specified in a System Access Control List (SACL)

GPFS is capable of storing SACL content, but will not interpret it.

- ▶ Windows sparse files API and TxF

Identity management on Windows

GPFS allows file sharing among AIX, Linux, and Windows nodes. AIX and Linux rely on 32-bit user and group IDs for file ownership and access control purposes; Windows uses variable-length security identifiers (SIDs). The difference in the user identity description models presents a challenge to any subsystem that allows for heterogeneous file sharing.

GPFS uses 32-bit ID name space as the canonical name space, and Windows SIDs are mapped into this name space as needed. Two mapping algorithms are used (depending on system configuration):

- ▶ GPFS built-in auto-generated mapping
- ▶ User-defined mappings that are stored in the Microsoft Windows Active Directory using the Microsoft Identity Management for UNIX (IMU) component

Summary of GPFS support functions on operating systems

Table 2-2 lists the functions that are supported on each operating system.

Table 2-2 GPFS support functions on each OS

| Function | AIX | Linux | Windows | Mixed OS |
|-------------------------|------|--------------------|---------|------------------|
| SCSI-III PR | Yes | Yes | No | No |
| DMAPI | Yes | Yes | No | Yes ^a |
| ILM/HSM | Yes | Yes | No | Yes ^a |
| InfiniBand RDMA | Yes | Yes | No | Yes ^a |
| InfiniBand IPoIB | Yes | Yes | Yes | Yes |
| NSD: partition | Yes | Yes | No | - |
| NSD: LUN (block device) | Yes | Yes | Yes | - |
| NSD: server/client | Yes | Yes | Yes | Yes |
| Multipath driver | MPIO | Multipathd RDAC | MPIO | Yes ^b |

| Function | AIX | Linux | Windows | Mixed OS |
|---------------------------------|------|-------|----------------------------|------------------|
| Minimum required GPFS version | v3.0 | v3.0 | v3.2.1.5 | v3.2.1.5 |
| Allows upgrade from old version | Yes | Yes | No | Yes ^a |
| mmbackup | Yes | yes | No | Yes ^a |
| mmquota | Yes | Yes | No | Yes ^a |
| mmacl | Yes | Yes | No | Yes ^a |
| mmpmon | Yes | Yes | No | Yes ^a |
| GPFS APIs | Yes | Yes | No | Yes ^a |
| Default administrator name | root | root | administrator ^c | root |

a. Not supported with Windows GPFS.

b. Must use SVC solution with multi storage vendor.

c. On multicluster, use root user; require Active Directory and Identity Management for UNIX.

Cluster file system

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

The unique differentiation points for GPFS versus other files systems are as follows:

- ▶ GPFS is not a client server model such as NFS, CIFS, AFS® or DFS. GPFS is not single-server bottleneck, thus there is no protocol overhead for data transfer.
- ▶ One metanode exists per open file. The *metanode* is responsible for maintaining file metadata integrity. In almost all cases, the node that has had the file open for the longest continuous period of time is the metanode. All nodes accessing a file can read and write data directly, but updates to metadata are written only by the metanode. The metanode for each file is independent of that for any other file and can move to any node to meet application requirements.
- ▶ GPFS commands can be executed from any node in the cluster. If tasks must be performed on another node in the cluster, the command automatically redirects the request to the appropriate node for execution. For administration commands to be able to operate, passwordless remote shell communications between the nodes is required. This task can be accomplished by using a standard shell program, such as `rsh` or `ssh`, or a custom application that uses the same semantics. To provide additional security, you can control the scope of passwordless access by allowing administration tasks to be performed from either of the following sources:
 - From all nodes running GPFS (by using `mmchconfig adminMode=allToall`)
 - From a subset of nodes (by using `mmchconfig adminMode=central`)
- ▶ GPFS consists of a single installation package. For GPFS installation, install only one GPFS software package; it does not require additional software for configuration. Other cluster file system solutions require additional software packages such as cluster server, volume manager and file system software. This is a strong point which helps reduce complexity for installation and administration of the cluster file system.
- ▶ GPFS provides multiple features that improve the reliability of your file system. This improved reliability includes automatic features such as file system logging, and configurable features such as intelligently mounting file systems on startup and providing tools for flexible synchronous replication. Also, GPFS provides a heterogeneous

infrastructure with AIX, Linux, and Windows. However, other cluster file systems do not support this kind of architecture, but do take advantage of CIFS and NFS with their own shared file system.

2.5.2 Security

This section provides the security mechanism that GPFS utilizes to manage the cluster.

Remote shell with ssh

GPFS commands need to be able to communicate across all nodes in the cluster. To achieve this, the GPFS commands use the remote shell command that you specify on the `mmcrcluster` command or the `mmchcluster` command.

The default remote shell command is `rsh`. You can designate the use of a different remote shell command by specifying its fully-qualified path name with the `mmcrcluster` command or the `mmchcluster` command. The remote shell command you use must adhere to the same syntax as the `rsh` command, but can implement an alternate authentication mechanism. Clusters that include both UNIX and Windows nodes require the use of `ssh` for the remote shell command.

By default, you can issue GPFS administration commands from any node in the cluster. Optionally, you can choose a subset of the nodes that are capable of running administrative commands. In either case, the nodes that you plan to use for administering GPFS must be able to run remote shell commands on any other node in the cluster as user “root” without the use of a password and without producing any extraneous messages.

Remote cluster: subnets and firewalls rule

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

The firewall setting must allow the following rule (the network switch based firewall rule setting should be the same as the operating system firewall):

- ▶ Inbound/outbound, GPFS/1191port
- ▶ Inbound/outbound, SSH/22 port

Figure 2-23 shows a GPFS remote-cluster mount diagram.

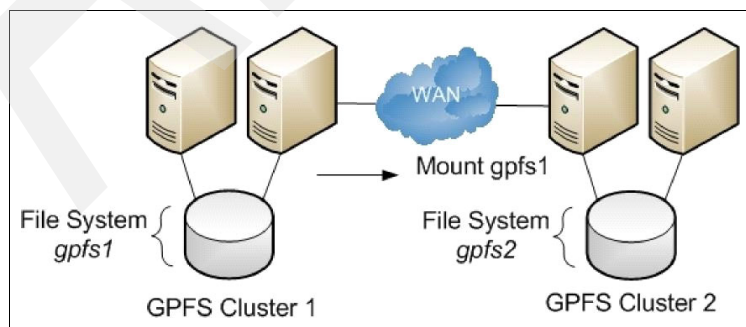


Figure 2-23 Multi-cluster configuration

SELinux configuration with GPFS

When running GPFS on Security Enhanced Linux (SELinux) you cannot allow the normal *init* scripts to start GPFS because doing so causes the GPFS daemons to run in the SELinux *initrc_t* domain and is too restricted for GPFS. To get GPFS to run fully unconfined, use the **runcon** command to set the security context.

Perform the following steps:

1. Run the following command in `/etc/rc.d/rc.local` directory:

```
runcon -t unconfined_t /usr/lpp/mmfs/bin/mmstartup
```
2. Disable automatic startup of GPFS by running the following command:

```
chkconfig gpfs off
```

Also be aware that GPFS does not support the SELinux file labels, so you must set a label on the file system at mount-time. Any files in the GPFS file system will then have the same label.

Perform the following steps:

1. Mount a GPFS file system with xen-images, disable normal GPFS automount of the file system by running the following command

```
mmchfs gpfsxen -A no
```
2. Manually mount the file system with the correct fscontext by adding the following lines to the `/var/mmfs/etc/mmfsup.scr` file:

```
mount /dev/gpfsxen -t gpfs -o \  
"fscontext=system_u:object_r:xen_image_t",rw,mtime,atime,dev=gpfsxen \  
/var/lib/xen/images
```

Note: Usually, if you run GPFS, disable SELinux on the Linux system. If there is a security problem, you cannot get support from IBM.

2.5.3 High availability

This section describes the high availability characteristics of a GPFS cluster.

Quorum

GPFS uses a cluster mechanism called *quorum* to maintain data consistency in the event of a node failure.

Quorum operates on the principle of majority rule. This means that a majority of the nodes in the cluster must be successfully communicating before any node can mount and access a file system. The quorum keeps any nodes that are cut off from the cluster (for example, by a network failure) from writing data to the file system.

During node failure situations, quorum must be maintained so that the cluster can remain online. If quorum is not maintained because of node failure, GPFS unmounts local file systems on the remaining nodes and attempts to reestablish quorum, at which point file system recovery occurs. For this reason be sure to carefully consider the set of quorum nodes.

Figure 2-24 shows how to configure the quorum and tiebreaker disks.

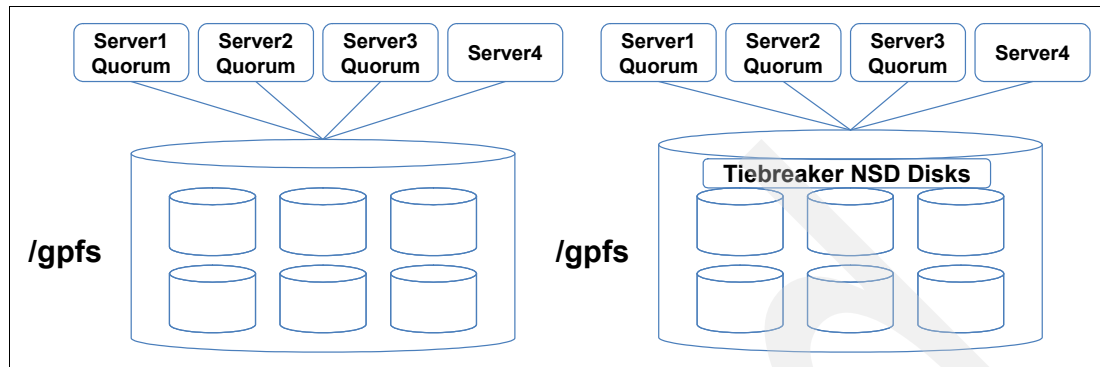


Figure 2-24 Design GPFS quorum and tiebreaker

GPFS quorum must be maintained within the cluster for GPFS to remain active. If the quorum semantics are broken, GPFS performs recovery in an attempt to achieve quorum again.

GPFS can use any one of the following methods for determining quorum:

- ▶ **Node quorum:** This algorithm is the default for GPFS. Note the following information about node quorum:
 - Quorum is defined as one plus half of the explicitly defined quorum nodes in the GPFS cluster.
 - There are no default quorum nodes; you must specify which nodes have this role.
- ▶ **Node quorum with tiebreaker disks:** When running on small GPFS clusters, you might want to have the cluster remain online with only one surviving node. To achieve this, add a tiebreaker disk to the quorum configuration. Note the following information about this algorithm:
 - Allows you to run with as little as one quorum node available if you have access to a majority of the quorum disks.
 - Enabling these disks starts by designating one or more nodes as quorum nodes. Then, define one to three disks as tiebreaker disks by using the `tiebreakerDisks` parameter with the `mmchconfig` command.
 - Designate any disk in the file system to be a tiebreaker.

Note: Tiebreaker disk rules:

- ▶ Although you may have one, two, or three tiebreaker disks, a good practice is to use an odd number of tiebreaker disks.
- ▶ Among the quorum node groups that appear after an interconnect failure, only those having access to a majority of tiebreaker disks can be candidates to be the survivor group.
- ▶ Tiebreaker disks must be connected to all quorum nodes.

A file system descriptor quorum is initially written to every disk in the file system and is replicated on a subset of the disks as changes to the file system occur, such as the adding or deleting of disks.

Based on the number of failure groups and disks, GPFS creates one to five replicas of the descriptor (this is the same file system descriptor that is created with the `mmcrnsd` command):

- ▶ If at least five separate failure groups exist, five replicas are created; if at least five separate replicas exist, GPFS can tolerate a loss of two of the five replicas.
- ▶ If at least three separate disks exist, three replicas are created; if at least three replicas exist, GPFS can tolerate a loss of one of the three replicas.
- ▶ If only one or two disks exist, a replica is created on each disk; if fewer than three replicas exist, a loss of one replica might cause the descriptor to be inaccessible.

Figure 2-25 describes the file system descriptor with replication and storage pool. The diagram shows three tiebreakers for one GPFS cluster, and three descriptor for one GPFS file system. This configuration allows the loss of one failure group.

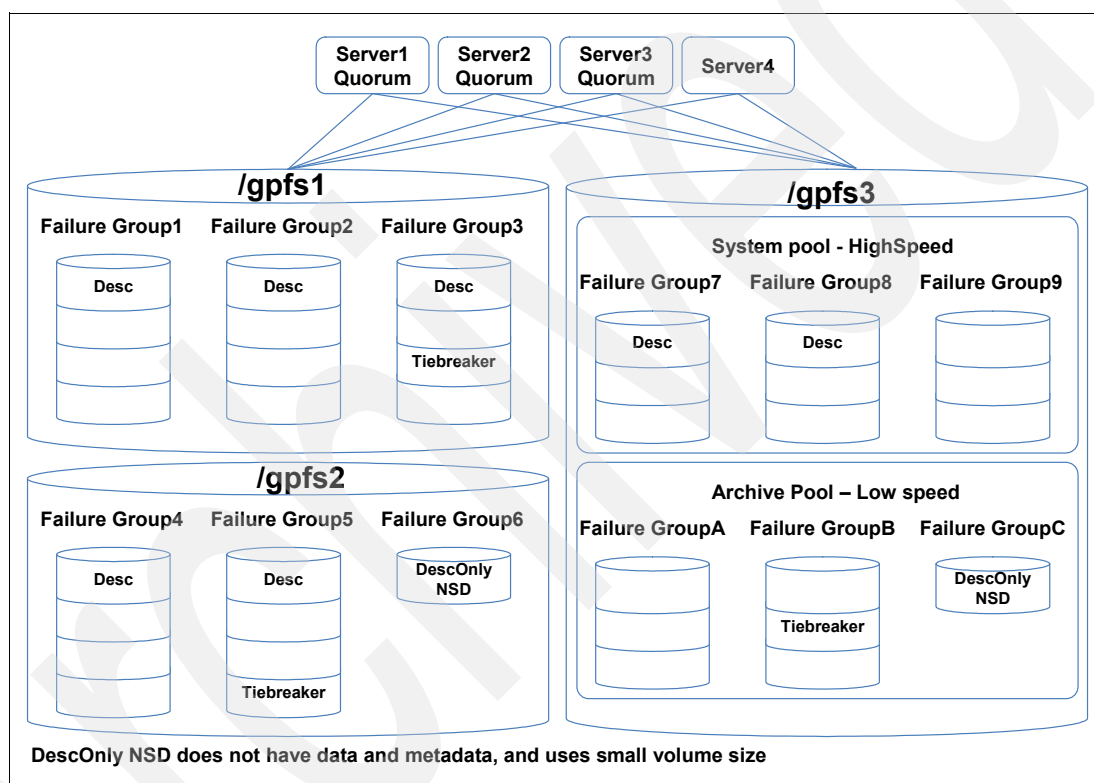


Figure 2-25 File system descriptor with replication and storage pool

After it decides how many replicas to create, GPFS picks disks to hold the replicas, so that all replicas are in separate failure groups, if possible, to reduce the risk of multiple failures. In picking replica locations, the current state of the disks are considered. Stopped or suspended disks are avoided. Similarly, when a failed disk is brought back online, GPFS might modify the subset to rebalance the file system descriptors across the failure groups. The disks that are used to hold the file system descriptor replicas can be seen by running the following command and looking for the string `desc` in the remarks column:

```
mmldisk fsname -L
```

The loss of all disks in a disk failure group might cause a majority of file system descriptors to become unavailable and inhibit further file system operations. For example, if your file system is backed up by three or more disks that are assigned to two separate disk failure groups, one of the failure groups will be assigned two of the file system descriptor replicas; the other failure group will be assigned only one replica. If all of the disks in the disk failure group that

contains the two replicas become unavailable, the file system also becomes unavailable. To avoid this particular scenario, consider introducing a third disk failure group consisting of a single disk that is designated as a descOnlydisk.

Note: A single disk that is designated as a descOnly disk and It would not contain any file system data. Also this disk can be as small as 4 MB.

NSD server and disk failure

The three most common reasons why data becomes unavailable are as follows:

- ▶ Disk failure
- ▶ Disk server failure with no redundancy (configure local storage)
- ▶ Failure of a path to the disk

In the event of a disk failure in which GPFS can no longer read or write to the disk, GPFS discontinues the use of the disk until it returns to an available state. You can guard against loss of data availability from disk failure by using the following items:

- ▶ Hardware data protection as provided by a RAID
- ▶ The GPFS data and metadata replication features

Figure 2-26 describes the NSD server failover architecture.

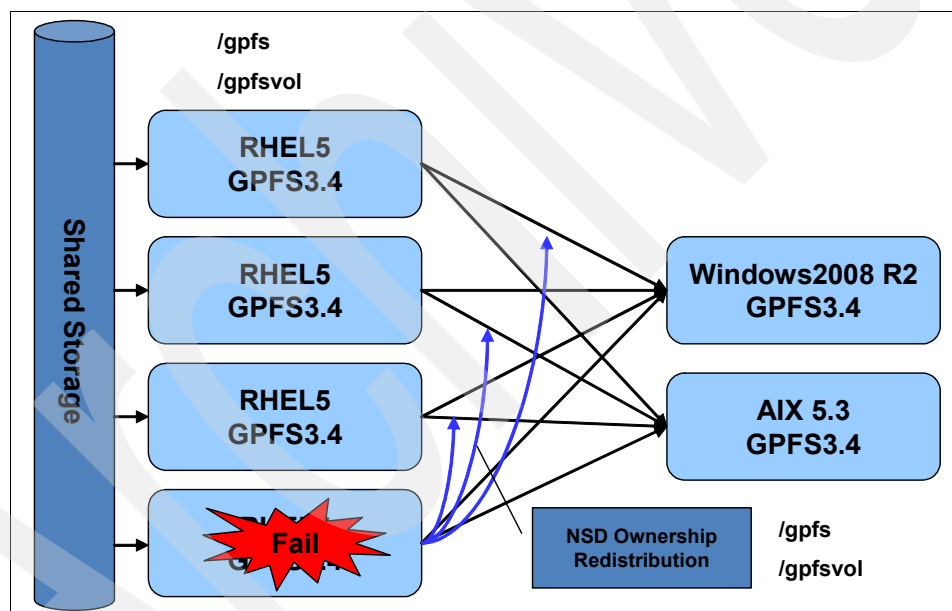


Figure 2-26 NSD server failure and rebalancing NSD volume ownership on GPFS server

Consider RAID as the first level of redundancy for your data and add GPFS replication if you want additional protection.

In the event of an NSD server failure in which a GPFS client can no longer contact the node that provides remote access to a disk, GPFS discontinues the use of the disk. You can guard against loss of an NSD server availability by using common disk connectivity on multiple NSD server nodes and specifying multiple NSD servers for each common disk.

Note: If a path to a disk fails, GPFS reports a disk failure and marks the disk as being down. To one bring the disk back online, first follow the directions supplied by your storage vendor to determine and one repair the failure

Minimize recovery time with SCSI-3 PR

Persistent Reserve (PR) provides a mechanism for reducing recovery times from node failures. To enable PR and to obtain recovery performance improvements, your cluster requires a specific environment:

- ▶ All disks must be PR-capable.
- ▶ All disks must be hdisks.
- ▶ If the disks have defined NSD servers, all NSD server nodes must be running AIX or Linux.
- ▶ If the disks are SAN-attached to all nodes, all nodes in the cluster must be running AIX or Linux.

You must explicitly enable PR by using the `usePersistentReserve` option with the `mmchconfig` command. If you set the option as follows, GPFS attempts to set up PR on all PR-capable disks:

```
usePersistentReserve=yes
```

All subsequent NSDs are created with PR enabled if they are PR-capable. However, PR is only supported in one home cluster. Therefore, access to PR-enabled disks from another cluster must be through an NSD server that is in the home cluster and not directly to the disk (for example, through a SAN).

From the PR perspective, PR is part of the SCSI-3 standard, which must be identical across the operating system. However, considering the difference between the Linux device driver and AIX device driver, a mixed AIX and Linux environment is not supported. Which raises the following statement and question: only if all NSD servers are either all AIX or all Linux, will SCSI-3 PR be supported? If there is any NSD server defined, all nodes in the cluster have to be either all AIX or all Linux to support SCSI-3 PR.

Note: A mixed AIX and Linux environment is not supported with SCSI-3 PR.

Disaster recovery

GPFS provides a number of features that facilitate the implementation of highly-available GPFS environments that are capable of withstanding catastrophic hardware failures. By maintaining a replica of the file system's data at a geographically-separate location, the system sustains its processing by using the secondary replica of the file system in the event of a total failure in the primary environment.

On a high level, a disaster-resilient GPFS cluster consists of two or three distinct, geographically separated, hardware sites that are operating in a coordinated fashion. Two of the sites consist of GPFS nodes and storage resources that are holding a complete replica of the file system. If a third site is active, it consists of a single node and a single disk that are used as a tiebreaker for GPFS quorum. In the event of a catastrophic hardware failure that disables the operation of an entire site, and assuming the tiebreaker site remains operational, file system services fail over to the remaining subset of the cluster and continue serving the data by using the replica of the file system that survived the disaster. However if the tiebreaker fails during the disaster, the remaining number of nodes and disks is insufficient to satisfy the quorum rules and the surviving site loses access to the GPFS file system. A manual procedure is needed to instruct GPFS to disregard the existing quorum assignments and continue operating with whatever resources are available.

The secondary replica is maintained by one of several methods:

- ▶ Synchronous mirroring by using GPFS replication

The data and metadata replication features of GPFS are used to implement synchronous mirroring between a pair of geographically separated sites. The use of logical replication-based mirroring offers a generic solution that relies on no specific support from the disk subsystem beyond the basic ability to read and write data blocks.

- ▶ Synchronous mirroring by using IBM TotalStorage® Enterprise Storage Server® (ESS) Peer-to-Peer Remote Copy (PPRC)

The PPRC feature of the ESS establishes a persistent mirroring relationship between pairs of Logical Units (LUNs) on two subsystems connected over an ESCON® or a fiber-channel link. All updates performed on the set of primary, or source, LUNs appear in the same order on the secondary, or target, disks in the target subsystem. The PPRC mechanism provides for an exact bitwise replica of the source's content as seen at the time of the failure on the target if the source volume fail.

- ▶ Synchronous mirroring using IBM TotalStorage ESS PPRC

Usage of synchronous IBM TotalStorage ESS PPRC, now extends to IBM TotalStorage Metro Mirror. Use of asynchronous PPRC now extends to IBM TotalStorage Global Mirror.

- ▶ Asynchronous mirroring using ESS FlashCopy®

Periodic point-in-time copies of the file system are taken using the facilities of ESS FlashCopy. The copy is subsequently transferred to a remote backup location using PPRC, written to tape, or both.

The primary advantage of both synchronous mirroring methods is the minimization of the risk of permanent data loss. Both methods provide two consistent, up-to-date replicas of the file system, each available for recovery if the other one fails. However, inherent to all solutions that synchronously mirror data over a WAN link is the latency penalty induced by the replicated write I/Os. This penalty makes both synchronous mirroring methods prohibitively inefficient for certain types of performance-oriented applications. The asynchronous method effectively eliminates this penalty. However, asynchronous mirroring might result in two distinct and not necessarily consistent replicas of the file system. There are no guarantees as to the validity of data that is kept in the snapshot beyond the fact that the file system's metadata is consistent and that the user data must have been valid at the time the snapshot was taken.

Synchronous mirroring utilizing IBM Total Storage ESS PPRC

PPRC is a function that continuously updates a secondary (target) copy of an ESS disk volume to match changes made to a primary (source) volume. Any pair of equal-sized ESS disks can be configured for a PPRC relationship, during which all write operations that are performed on the source are synchronously mirrored to the target device.

The PPRC protocol guarantees that the secondary copy is constantly up-to-date by ensuring that the primary copy is written only if the primary storage subsystem received acknowledgement that the secondary copy has been written. The paired volumes typically reside on two distinct and geographically separated ESS devices that are communicating over ESCON or over a Fibre Channel link.

A number of PPRC tasks are provided to facilitate recovery in the event of a site-wide failure. After the failure of the primary volume (or the failure of the entire storage subsystem), users execute the PPRC failover task, which suspends the PPRC relationship between the given pair of volumes and converts the target volume to a primary. When a volume enters the suspended state, a modification bitmap is established to keep track of the write operations performed on that volume to allow for an efficient resynchronization.

After the operation of the original primary volume has been restored, the PPRC fail-back task is executed to resynchronize the content of the two volumes. The original source volume is switched to the target mode, after which all modified data tracks (those recorded in the modification bitmap) are copied from the original target disk. The volume pair is then suspended again and another task is executed to reverse the volumes' roles, thus bringing the pair into its initial state.

The ESS Copy Services Web Interface, as described in the *IBM TotalStorage Enterprise Storage Server Web Interface User's Guide, SC26-7448*, is a GUI that allows users to establish and terminate PPRC pairs, and invoke failover, fail-back, and related PPRC functions. A Java-based CLI, described in the *IBM Enterprise Storage Server Command-Line Interfaces User's Guide, SC26-7494* provides another method of interaction.

A PPRC-based GPFS cluster can be established in two ways:

- ▶ A single GPFS cluster encompassing two sites and an optional tiebreaker site
- ▶ Two distinct GPFS clusters

Active-active GPFS cluster with PPRC

The high-level organization of PPRC-based active-active GPFS cluster is illustrated in Figure 2-27. A single GPFS cluster is created over three sites. The data is mirrored between two active sites with a cluster configuration server residing at each site and a tiebreaker quorum node installed at the third location. The presence of an optional tiebreaker node allows the surviving site to satisfy the node quorum requirement with no additional intervention. Without the tiebreaker, the failover procedure requires an additional administrative command to relax node quorum and allow the remaining site to function independently. Furthermore, the nodes at the recovery site have direct disk paths to the primary site's storage.

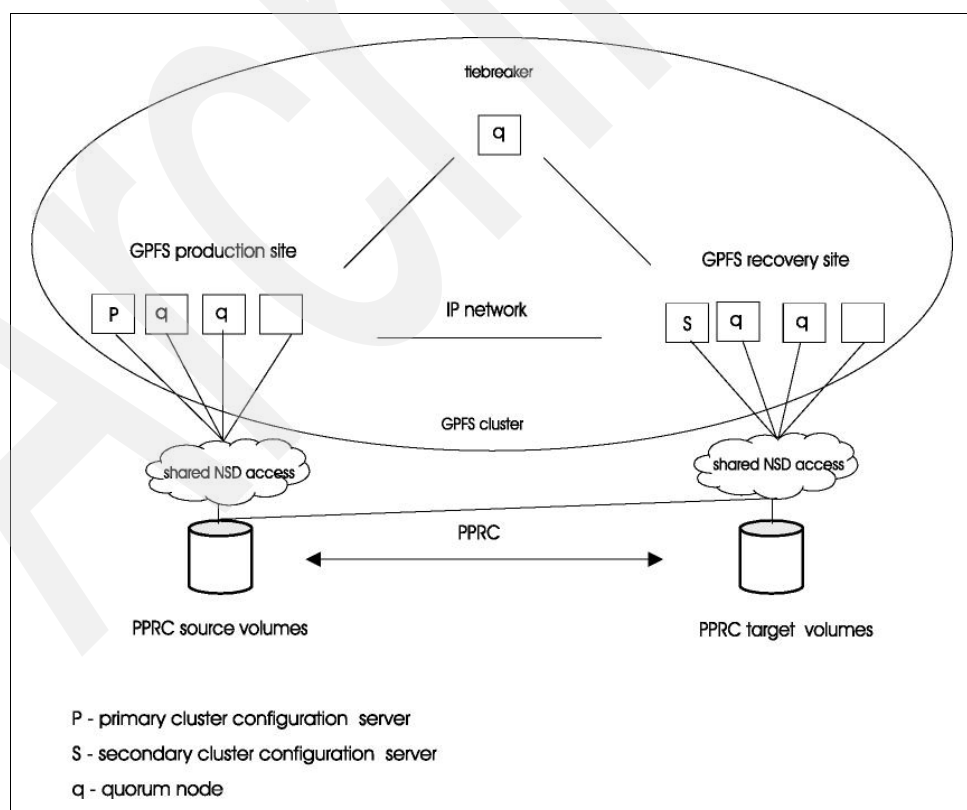


Figure 2-27 Synchronous active-active PPRC-based mirrored GPFS with a tiebreaker site

Active-passive GPFS cluster with PPRC

In an active-passive environment, two GPFS clusters are set up in two geographically distinct locations (the production and the recovery sites) and are referred to as *peer GPFS clusters*. A GPFS file system is defined over a set of disk volumes located at the production site; these disks are mirrored by using PPRC to a secondary set of volumes that is located at the recovery site. During normal operation, only the nodes in the production GPFS cluster mount and access the GPFS file system at any given time, which is the primary difference between a configuration of this type and the active-active model.

In the event of a catastrophe in the production cluster, the PPRC failover task is executed to enable access to the secondary replica of the file system located on the target PPRC disks. See *IBM TotalStorage Enterprise Storage Server Implementing ESS Copy Services in Open Environments*, SG24-5757, which explains PPRC failover and failback and the means of invoking these procedures in your environment.

The secondary replica is then mounted on nodes in the recovery cluster as a regular GPFS file system, allowing the processing of data to resume at the recovery site. At a latter point, after restoring the physical operation of the production site, the fail-back procedure is executed to resynchronize the content of the PPRC volume pairs between the two clusters and re-enable access to the file system in the production environment, as shown in Figure 2-28.

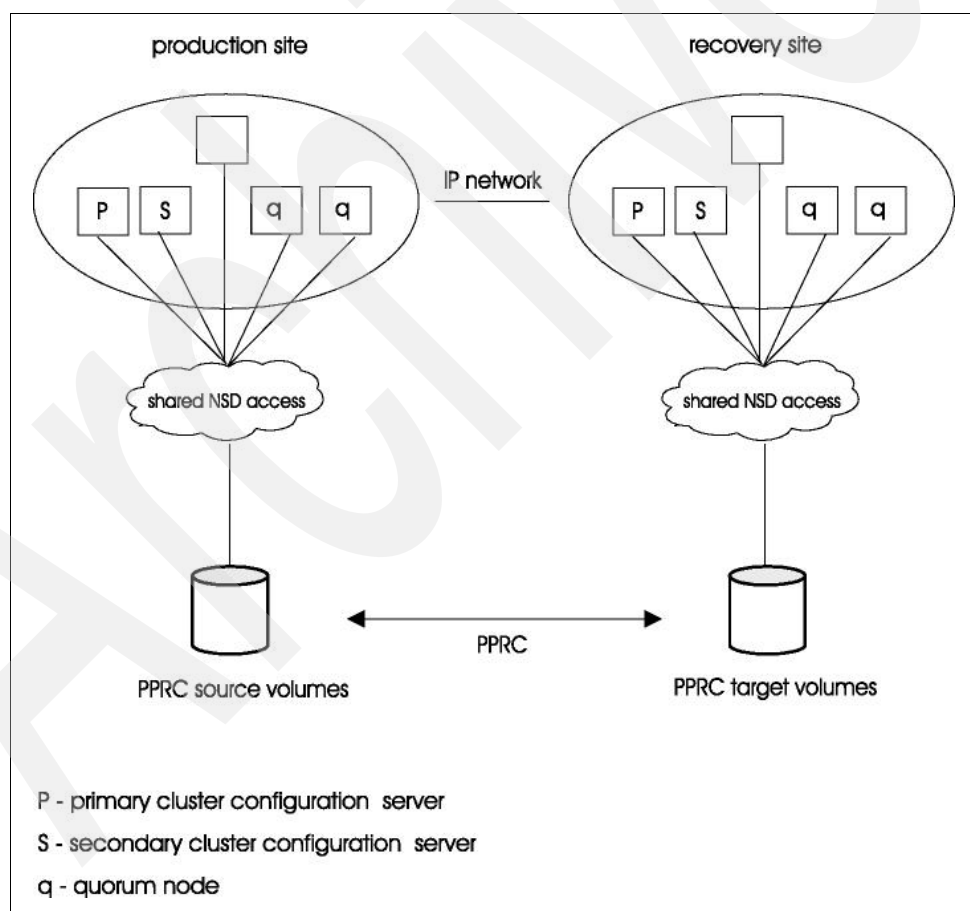


Figure 2-28 Synchronous active-passive PPRC-based GPFS without a tiebreaker site

2.5.4 Network Shared Disk (NSD) creation considerations

This section explains considerations when you create NSD on GPFS.

- ▶ Considerations for creating a NSD
- ▶ Operating NSD server and client

NSD configuration guidance

On UNIX and Linux, you must prepare each physical disk that you intend to use with GPFS by first defining it as an NSD by using the `mmcrnsd` command. NSDs can be created on the following types of physical disks:

- ▶ An hdisk or vpath on AIX
- ▶ A block disk device or a disk partition on Linux

On Windows, GPFS can create NSDs only from empty disk drives. The `mmcrnsd` command accepts Windows basic disks or unknown/not initialized disks. It always re-initializes these disks so that they become basic GPT disks with a single GPFS partition. NSD data is stored in GPFS partitions, allowing other operating system components to recognize that the disks are used. The `mmde1nsd` command deletes the partition tables that are created by that `mmcrnsd` command. Windows GPFS does not recognize partitions.

As input, the `mmcrnsd` command expects a file, `DescFile`, that contains a disk descriptor, one per line, for each of the disks to be processed. Disk descriptors have the following format:

```
DiskName:ServerList::DiskUsage:FailureGroup:DesiredName:StoragePool
```

Disk descriptors can be explained as follows:

- ▶ **DiskName**
 - On UNIX, this parameter is the block device name that appears in `/dev` for the disk you want to define as an NSD. Examples of disks that are accessible through a block device are directly attached disks. GPFS disk discovery looks for common device types, `/dev/sd` on Linux for example. If a `ServerList` parameter is specified, `DiskName` must be the `/dev` name for the disk device on the first NSD server node that is defined in the server list. The reason is because, when the `mmcrnsd` command is executed, the NSD descriptors are written by the first node in the `ServerList` parameter.
 - On Windows, this parameter is the disk number of the disk you want to define as an NSD. Disk numbers appear in Windows Disk Management console and the `DISKPART` command line utility. If a server node is specified, `DiskName` must be the disk number from the first NSD server node defined in the server list.

All device names under the same operating system point to the same LUN devices and cannot configure mixed operating systems with same LUN. For example, the NSD cannot use the same LUN with AIX and Linux for the following reasons:

- There is a different device name on each operating system
- There is a different host mapping information about the storage side.

- ▶ **ServerList**

This parameter is a comma-separated list of NSD server nodes and has the following form:

```
server1[,server2,...,server8]
```

You can specify up to eight NSD servers in this list. A GPFS node accessing the disk through an NSD server uses the first server in the list with which it can communicate. If the first server is not available, the node uses the next available server in the list. If you do not define a server list, GPFS assumes that the disk is SAN-attached to all nodes in the

cluster. If all nodes in the cluster do not have access to the disk, or if the file system to which the disk belongs is to be accessed by other GPFS clusters, you must specify a server list.

► **DiskUsage**

Use this parameter to specify disk usage or accept the default. This parameter is used at file system creation, so it is ignored by the `mmcrnsd` command and is passed unchanged to the output descriptor file produced by the `mmcrnsd` command. Possible values are as follows:

- `dataAndMetadata`: Indicates that the disk contains both data and metadata. This value is the default for the system storage pool.
- `dataOnly`: Indicates that the disk contains data and does not contain metadata. This value is the default and only allowed value for all storage pools other than the system pool.
- `metadataOnly`: Indicates that the disk contains metadata and does not contain data. Only NSDs in the system pool can be designated `metadataOnly`.
- `descOnly`: Indicates that the disk contains no data and no metadata. Such a disk is used solely to keep a copy of the file system descriptor. This disk usage is most commonly used as a third failure group in certain disaster recovery configurations.

► **FailureGroup**

This parameter is a number that identifies the failure group to which this disk belongs. You can specify any value in the range of -1 to 4000 (where -1 indicates that the disk has no point of failure in common with any other disk). If you do not specify a failure group, the value defaults to the node number plus 4000 for the first NSD server defined in the server list. If you do not specify an NSD server, the value defaults to -1. GPFS uses this information during data and metadata placement to ensure that no two replicas of the same block are written in such a way as to become unavailable because of a single failure. All disks that are attached to the same NSD server or adapter must be placed in the same failure group. This field is ignored and passed unchanged to the output descriptor file written by either the `mmcrnsd` command or the `mmcrvsd` command.

► **DesiredName**

This parameter specifies the name of the NSD to be created. This name must not already be used as another GPFS disk name, and it must not begin with the GPFS reserved string.

If a desired name is not specified, the NSD is assigned a name according to the following convention:

`gpfsNNnsd`

In this convention, NN is a unique nonnegative integer that is not used in any prior NSD.

Note: This name can contain only the following characters:

- A - Z
- a - z
- 0 - 9
- Underscore (`_`) character

All other characters are not valid.

► **StoragePool**

This parameter specifies the name of the storage pool to which the NSD is assigned. The following guidelines about providing storage pool names:

- Must be unique within a file system, but not across file systems.
- Must not be larger than 255 alphanumeric characters.
- Are case sensitive:

For example, “MYpool” and “myPool” are distinct storage pools. If this name is not provided, the default is system. Only the system pool may contain metadataOnly, dataAndMetadata, or descOnly disks.

Note: Be careful when configuring the GPFS NSD server because after creating the NSD and the file system, the NSD server-assigned information cannot be changed without bringing down the file system.

NSD server considerations

If you plan to use NSD servers to remotely serve disk data to other nodes, as opposed to having disks SAN-attached to all nodes, be sure to consider the total computing and I/O load on these nodes, as follows:

- Determine whether your NSD servers be dedicated servers or whether you will also be using them to run applications. If you will have non-dedicated servers, consider running fewer time-critical applications on these nodes. If you have runtime-critical applications on an NSD server, servicing disk requests from other nodes might conflict with the demands of these applications.
- Because the special functions of the file system manager consume extra processing time, if possible, avoid using a NSD server as the file system manager. The NSD server consumes both memory and processor cycles that can affect the operation of the file system manager.
- The actual processing capability required for NSD service is a function of the application I/O access patterns, the type of node, the type of disk, and the disk connection. You can later run the **iostat** command on the server to determine how much of a load your access pattern will place on an NSD server.
- Provide sufficient disks and adapters on the system to yield the required I/O bandwidth. Dedicated NSD servers must have sufficient disks and adapters to drive the I/O load you expect them to handle.
- Know approximately how much storage capacity you will need for your data.

Consider what you want as the default behavior for switching between local access and NSD server access in the event of a failure. To set this configuration, use the **-o useNSDserver** file system mount option of the **mmmount**, **mount**, **mmchfs**, and **mmremotefs** commands to control behavior:

- To specify the disk discovery behavior
- To limit or eliminate switching from either the local access to NSD server access or NSD server access to local access.

Consider specifying how long to wait for an NSD server to come online before allowing a file system mount to fail because the server is not available.

The **mmchconfig** command has the following options:

► **nsdServerWaitTimeForMount**

When a node is trying to mount a file system, which has disks that depend on NSD servers, this option specifies the number of seconds to wait for those servers to be up. If a server recovery is taking place, the wait time you specify with this option starts after recovery completes.

Note: The decision to wait for servers is controlled by the following option:

nsdServerWaitTimeWindowOnMount

► **nsdServerWaitTimeWindowOnMount**

This option specifies a window of time (in seconds) during which a mount can wait for NSD servers, as described for the **nsdServerWaitTimeForMount** option. The window begins when quorum is established (at cluster startup or subsequently), or at the last known failure times of the NSD servers that are required to perform the mount.

Notes:

- When a node rejoins a cluster, it resets all the failure times it knew about within that cluster.
- Because a node that rejoins a cluster resets its failure times within that cluster, the NSD server failure times are also reset.
- When a node attempts to mount a file system, GPFS checks the cluster formation criteria first. If that check falls outside the window, it then checks for NSD server fail times being in the window.

Figure 2-29 shows the NSD client performance, where the I/O balanced configuration is important. Each server must have its own NSD.

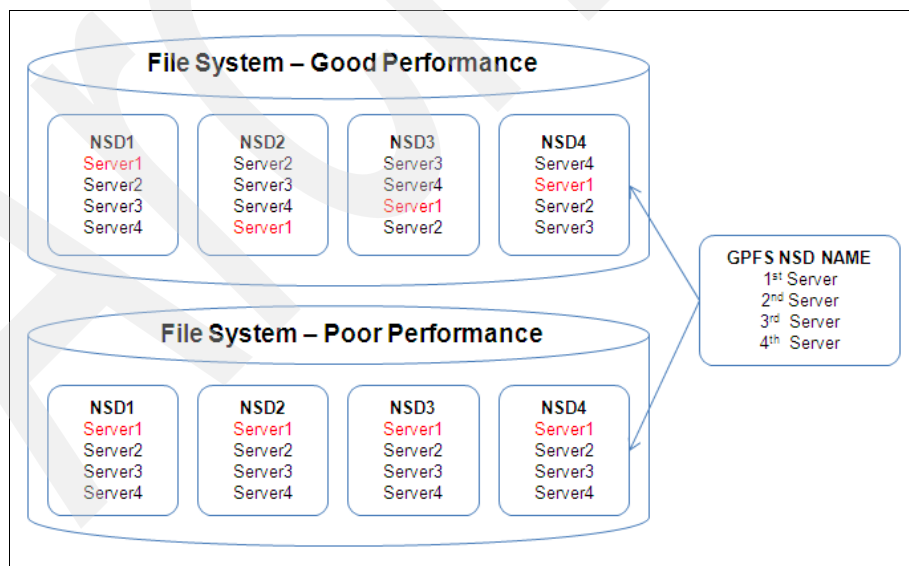


Figure 2-29 Server assignments on all NSDs

NSD server and client access operation

Usually, this configuration is based on shared storage and local storage. On the GPFS NSD client-side view, shared storage and local storage have TCP connections with all GPFS servers, and for writing a file, the work is divided across all GPFS servers. For example, the sample file is 300 MB and there are three GPFS servers. In this case, a 100-MB file is written to its own storage device for each.

Figure 2-30 shows how the operation goes on the GPFS NSD client when writing a file.

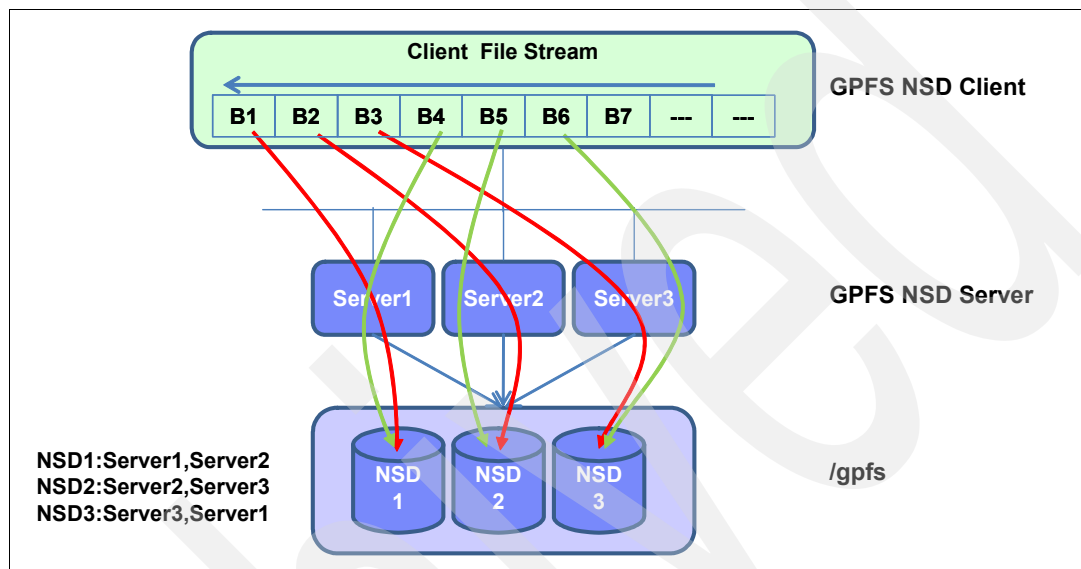


Figure 2-30 NSD client and server I/O stream

From Figure 2-30, the following points can be mentioned:

- ▶ GPFS NSD client: The GPFS volume access is through the GPFS NSD client's own network device.
- ▶ GPFS NSD server: GPFS exports this volume.

Direct NSD access operation

This configuration has direct access to all the LUNs in the storage system. Figure 2-31 describes how the operation on the GPFS server happens when writing a file.

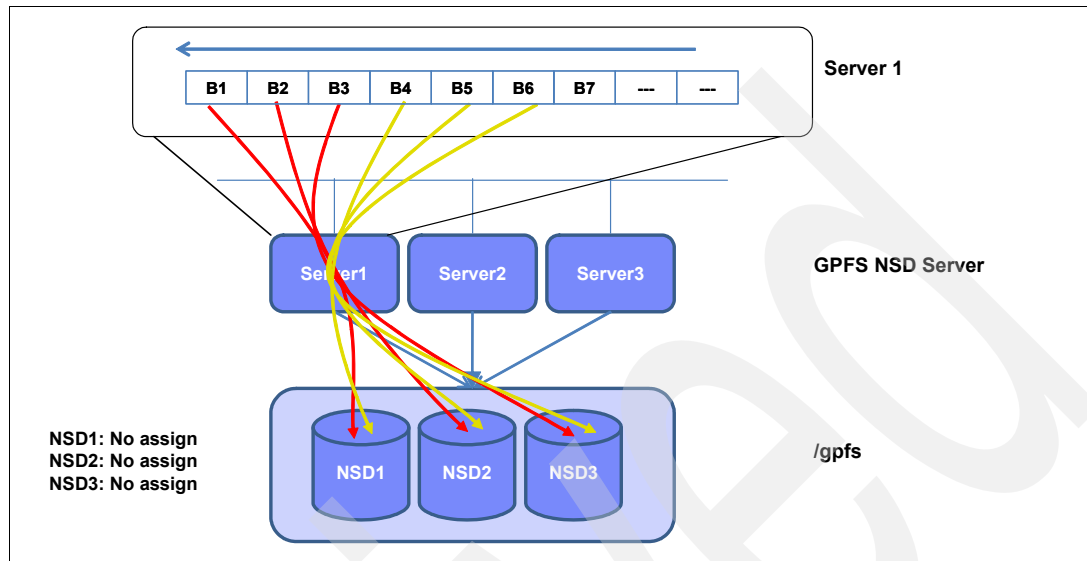


Figure 2-31 No NSD configuration I/O stream.

Disk access method: local versus shared disks

The GPFS NSD configuration requires a local disk or shared disk. The following configurations can be used for the disk access method:

- ▶ **Multi-platform:** This configuration is shared disk on the same operating system and heterogeneous environment on a single GPFS cluster.
- ▶ **Shared storage:** This design does not require an assigned NSD server, but cannot mix operating systems. Although it is an expensive solution, it provides more high availability, reliability, and high performance.
- ▶ **Local storage:** This configuration offers good performance. Performance depends on the network switch. When using this configuration, the NSD server must be define on local disk. If the node fails on this cluster, the file system will be shutdown. So, remember to take advantage of the local disk RAID configuration and the GPFS replication with a local disk tiebreaker configuration.

Figure 2-32 shows a diagram of three GPFS cluster configurations.

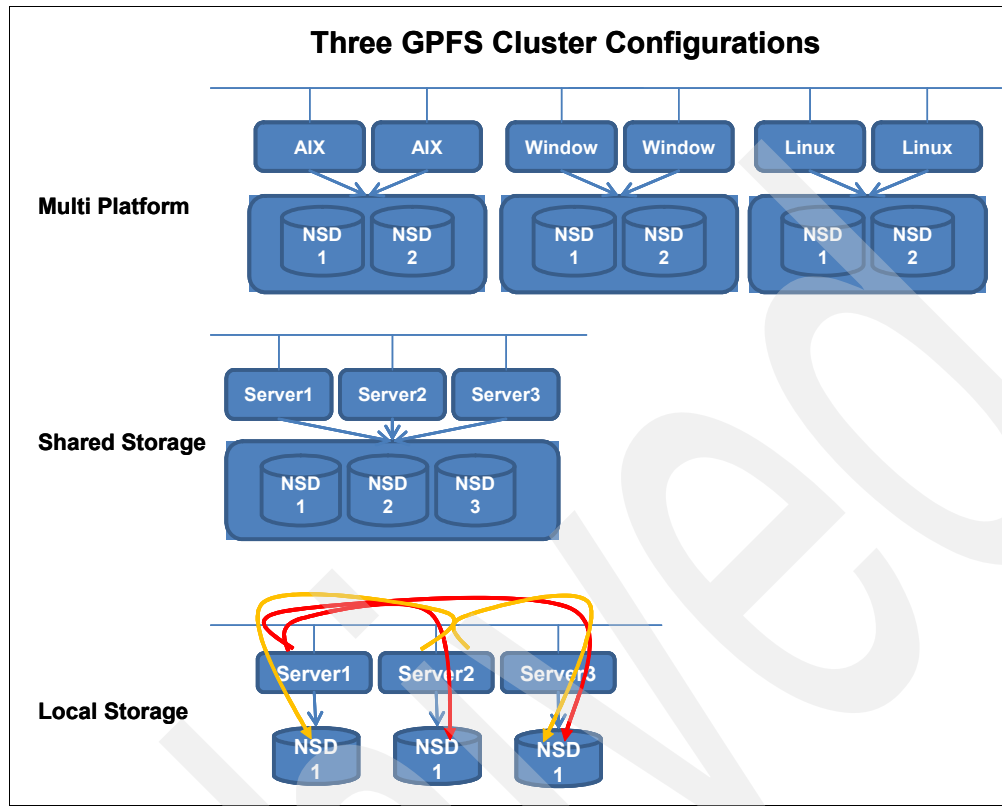


Figure 2-32 Three GPFS cluster configurations

Single node cluster

The GPFS cluster can be configured with 1 - 8192 nodes. If you want more high availability functions with your GPFS cluster, configure at least a three-node cluster. A single node GPFS cluster does not provide high availability functions.

For example with DR case, a third site is active. It consists of a single node and a single disk that used as a tiebreaker for GPFS quorum. In the event of a catastrophic hardware failure that disables the operation of an entire site, and assuming the tiebreaker site remains operational, file system services fail over to the remaining subset of the cluster and continue serving the data using the replica of the file system that survived the disaster.

2.5.5 GPFS considerations

This section has planning information and considerations for GPFS.

Planning a file system

When creating a file system there are two types of parameters: Those that can be changed dynamically and those that cannot. The key parameter that must be determined at file system creation is the file system block size. After it is set, the only way to change the block size is to re-create the file system, therefore, be sure to test the block size on a development system before production deployment.

GPFS supports block sizes in the range of 16 - 4 MB with a default of 256 KB. The block size of a file system is determined at creation using the **-B** parameter to the **mmcrfs** command. In

addition to using the **-B** parameter to create a file system with a block size greater than 1 MB, you must increase the value of the `maxblocksize` parameter. The default value of `maxblocksize` is 1 MB and the allowable range is 16 KB - 4 MB. For block sizes larger than 1 MB be sure that `maxblocksize` matches the value of the block size.

Note: GPFS for Linux on POWER does not support mounting of a file system with a 16 KB block size when running on either RHEL 5 or SLES 11.

How do you choose a block size for your file system? Of course it is best to test the impact of various blocksize settings with your application. If you cannot test various values of block sizes or you are looking only for a starting point, see Table 2-3, which provides basic guidance for what file system block sizes might be appropriate for various types of applications.

Table 2-3 Block size configuration guide, by application

| I/O type | Application examples | Block size |
|----------------------|--|------------|
| Large sequential I/O | Scientific computing, digital media | 1 ~ 4 MB |
| Relational database | DB2, Oracle | 512 KB |
| Small I/O sequential | General file service, file-based analytics | 256 KB |
| Small files | Email, web application servers | 64 KB |

The GPFS block size setting is the amount of data written to each disk in a file system before moving on to the next disk. The following characteristics are important to understand when you consider the appropriate value of block size:

- ▶ The block size is the largest size I/O that GPFS can issue to the underlying device
- ▶ A subblock is 1/32 of the block size. This value is the smallest allocation to a single file.
- ▶ Sector is 512 bytes. This number is the smallest I/O request size that GPFS issues to the underlying device.

This information means that, for example, if you use a block size of 1 MB, each file will use at least 32 KB ($1024\text{KB} / 32 = 32\text{KB}$).

What if you do not know your application I/O profile? Often, you do not have good information about the nature of the I/O profile, or the applications are so diverse that optimizing for one or the other is difficult. Generally, two approaches to designing for this type of situation separation or compromise are available:

▶ Separation

In this model, you create two file systems, one with a large file system block size for sequential applications and one with a smaller block size for small file applications. You can gain benefits from having file systems of two different block sizes even on a single type of storage. You can also use separate types of storage for each file system to further optimize to the workload.

In either case, the idea is that you provide two file systems to your users, for scratch space on a compute cluster for example. The users can run tests themselves by pointing the application to one file system or another to determine, by direct testing, which is best for their workload. In this situation, you may have one file system optimized for sequential I/O with a 1 MB block size and one for more random workloads at the 256 KB block size.

► **Compromise**

In this situation, you either do not have sufficient information about workloads (that is, users will not think about I/O performance) or enough storage for multiple file systems. In this case, a general suggestion is use a block size of 256 KB or 512 KB, depending on the general workloads and storage model used. With a 256 KB block size, you can still have good sequential performance (although not necessarily peak marketing numbers); you can also have good performance and space utilization with small files (256 KB has minimum allocation of 8 KB to a file). This configure is good for multi-purpose research workloads where the application developers are focusing on their algorithms more than I/O optimization.

Block allocation map

GPFS has two methods of allocating space in a system (cluster and scatter). You may use the `-j` option when creating a file system. This block allocation map type cannot be changed after the file system is created. Usually, GPFS first uses a round-robin algorithm to spread the data across all NSDs in the file systems.

Notes:

- After the disk is selected, the location of the data block on the disk is determined by the block allocation map type.
- The block allocation map mode (scatter/cluster) has nothing to do with the way data is spread across NSDs. This mode controls how data is distributed on a given NSD after it is selected by round robin.

Table 2-4 summarizes the block allocation map of each of option.

Table 2-4 Comparing types of block allocation

| Type | Cluster | Scatter |
|----------------|---------------------------|------------------------|
| Node configure | Eight or fewer nodes | More than eight nodes |
| Pattern | Sequential | Random |
| Benefit | Block allocation diminish | Consistent file system |

Replication

The metadata and data replication are set at the file system level and apply to all files. They are initially set for the file system when issuing the `mmcrfs` command. They can be changed for an existing file system by using the `mmchfs` command. When the replication parameters are changed, files created after the change are affected. To apply the new replication values to existing files in a file system, issue the `mmrestripefs` command.

Note: If you want to configure the replication function, before creating a file system, you must have a failure group number when creating the NSD and the data disk in each storage pool.

The following list provides information about the metadata replicas:

► **Default metadata replicas**

The default number of copies of metadata for all files in the file system may be specified at file system creation by using the **-m** option on the **mmcrfs** command or changed at a later time by using the **-m** option on the **mmchfs** command. This value must be equal to or less than MaxMetadataReplicas parameter, 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.

► **Maximum metadata replicas**

The maximum number of copies of metadata for all files in the file system can be specified at file system creation by using the **-m** option on the **mmcrfs** command. The default is 2. The allowable values are 1 or 2, but it cannot be lower than the value of DefaultMetadataReplicas parameter. This value cannot be changed.

► **Default data replicas**

The default replication factor for data blocks may be specified at file system creation by using the **-r** option on the **mmcrfs** command or changed at a later time by using the **-r** option on the **mmchfs** command. This value must be equal to or less than the MaxDataReplicas parameter, and the value cannot exceed the number of failure groups with disks that can store data. The allowable values are 1 and 2, with a default of 1.

If you want to change the data replication factor for the entire file system, the data disk in each storage pool must have a number of failure groups equal to or greater than the replication factor. For example, a failure error message is issued if you try to change the replication factor for a file system to two, when the storage pool has only one failure group.

► **Maximum data replicas**

The maximum number of copies of data blocks for a file can be specified at file system creation by using the **-r** option on the **mmcrfs** command. The default is 2. The allowable values are 1 and 2, but cannot be lower than the value of DefaultDataReplicas. This value cannot be changed.

In Figure 2-33, the shaded gray is the second image block and the white color is the original image block. The failure on failure group2 (Physical Storage Box or LUN Group), The GPFS File system operation has no problem. And after finish maintenance, re-join this file system, will start re-synchronize process automatically.

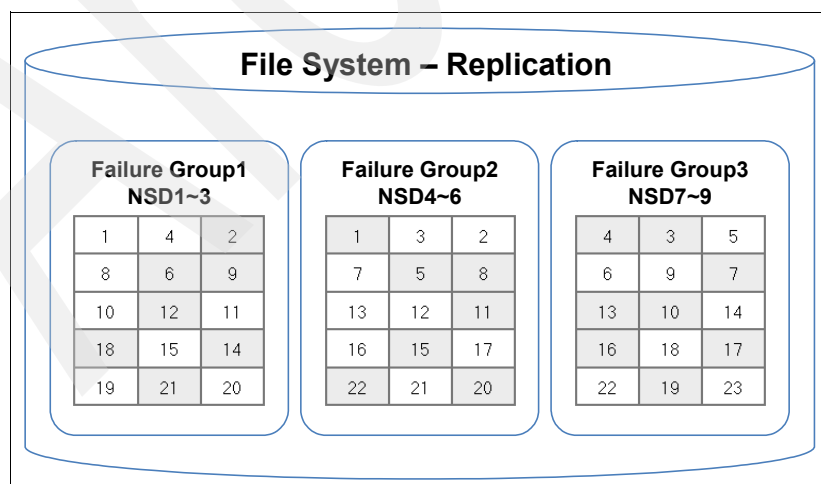


Figure 2-33 GPFS replication block diagram

Changing file replication attributes

Use the `mmchattr` command to change the replication attributes for one or more files. You can increase data and metadata replication only 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 after the file system has been created.

Storage pool with ILM

GPFS provides storage management based on the definition and use of storage pools, policies, and file sets.

Figure 2-34 shows operation GPFS storage pool with policy operation data flow.

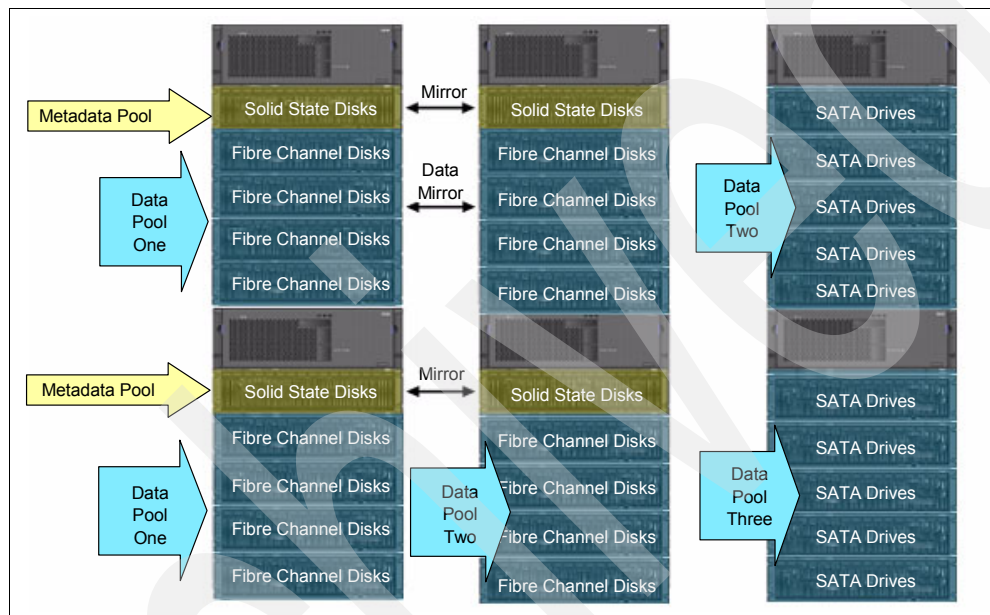


Figure 2-34 GPFS storage pool

Storage pools

A *storage pool* is a collection of disks or RAID configurations with similar properties that are managed together as a group. Storage pools provide a method to partition storage within a file system. As you plan how to configure your storage, consider factors such as the following factors:

- ▶ Improved price-performance by matching the cost of storage to the value of the data.
- ▶ Improved performance by reducing the following items:
 - Contention for premium storage
 - Effect of slower devices
- ▶ Improved reliability by providing for the following items:
 - Replication based on need
 - Better failure containment

Policies

Files are assigned to a storage pool based on defined *policies*:

- ▶ Placement policies:
 - Placing files in a specific storage pool when the files are created
- ▶ File management policies
 - Migrating files from one storage pool to another
 - Deleting files based on file characteristics
 - Changing the replication status of files
 - Taking a snapshot of metadata scans and file list creation

File sets

File sets provide a method for partitioning a file system and allow administrative operations at a finer granularity than the entire file system. For example file sets allow you to do the following tasks:

- ▶ Define data block and inode quotas at the file set level
- ▶ Apply policy rules to specific file sets

Snapshot

A *snapshot* of an entire GPFS file system can be created to preserve the contents of the file system at a single point in time. The storage overhead for maintaining a snapshot is keeping a copy of data blocks that would otherwise be changed or deleted after the time of the snapshot.

Snapshots of a file system are read-only; changes can be made only to the active (that is, normal, non-snapshot) files and directories.

The snapshot function allows a backup or mirror program to run concurrently with user updates and still obtain a consistent copy of the file system as of the time that the snapshot was created. Snapshots also provide an online backup capability that allows easy recovery from common problems such as accidental deletion of a file, and comparison with older versions of a file.

Notes:

- ▶ Because snapshots are not copies of the entire file system, they should not be used as protection against media failures. For information about protection against media failures, see Recoverability considerations in the *General Parallel File System Concepts, Planning, and Installation Guide Version 3 Release 4*, GA76-0413.
- ▶ A snapshot of a file creates a new file that captures the user data and user attributes from the original. The snapshot file is independent from the original file. For DMAPI-managed file systems, the snapshot of a file is not automatically managed by DMAPI, regardless of the state of the original file. The DMAPI attributes from the original file are not inherited by the snapshot. For more information about DMAPI restrictions for GPFS, see the *General Parallel File System Data Management API Guide Version 3 Release 4*, GA76-0414.

Export file system

This section lists aspects to consider when you export a GPFS file system to NFS. The operating system being used and the version of NFS might require special handling or considerations.

- ▶ Linux export considerations

For Linux nodes only, issue the **exportfs -ra** command to initiate a reread of the `/etc/exports` file. Starting with Linux kernel version 2.6, an `fsid` value must be specified for each GPFS file system that is exported on NFS. For example, the format of the entry in `/etc/exports` file for the `/gpfs/dir1` GPFS directory might look as follows:

```
/gpfs/dir1 cluster1(rw,fsid=745)
```

- ▶ AIX export considerations

AIX does not allow a file system to be exported by NFS V4 unless it supports NFS V4 ACLs.

- ▶ NFS export considerations for versions prior to NFS V4

For NFS exported file systems, the version of NFS you are running with might have an impact on the number of inodes you need to cache, as set by both the `maxStatCache` and `maxFilesToCache` parameters on the **mmchconfig** command.

The implementation of the **ls** command differs from NFS V2 to NFS V3. The performance of the **ls** command in NFS V3 in part depends on the caching ability of the underlying file system. Setting the cache large enough prevents rereading inodes to complete an **ls** command, but puts more of a CPU load on the token manager.

Also, the clocks of all nodes in your GPFS cluster must be synchronized. If this is not done, NFS access to the data, and other GPFS file system operations, might be disrupted.

- ▶ NFS V4 export considerations

For information about NFS V4, see the information at the NFSv4 General Information and References for the NFSv4 protocol website:

<http://www.nfsv4.org/>

- ▶ NFS V4, Samba, and Windows.

File systems supporting NFS V4 must set the **-D nfs4** parameter. The **-D posix** option allows NFS writes even in the presence of a deny-write open lock. If you intend to export the file system using NFS V4 or Samba, or mount your file system on Windows, you must use **-D nfs4**. For NFS V3 (or if the file system is not NFS exported at all) use the **-D posix** option.

File system performance monitor

Many options are available to monitor GPFS performance:

- ▶ Use SNMP service for the fabric hardware and the operating system.
- ▶ Use a monitoring tool such as **nmon** or **mmpmn**.
- ▶ Use GUI performance management that is provide by the hardware vendor.

Although GPFS provides an SNMP MIB file, the procedure is to use the SNMP base monitoring through the storage or the SAN switch hardware. In this case, monitoring the entire storage system is much easier, with no impact on GPFS Service.

2.5.6 Tivoli Storage Manager for GPFS

This section indicates considerations of Tivoli Storage Manager on GPFS.

Migration files overview

The HSM client provides both automatic and selective migration. After file migration begins, the HSM client sends a copy of your file to storage volumes on disk devices or devices that support removable media, such as tape, and replaces the original file with a stub file on your local file system.

The *stub file* is a small replacement file that causes the original file to appear as though it is on the local file system. It contains required information to locate and recall a migrated file and to respond to specific UNIX commands without recalling the file.

Automatic migration monitors space usage and automatically migrates eligible files according to the options and settings that you select. The HSM client provides two types of automatic migration: threshold migration and demand migration.

- ▶ Threshold migration maintains a specific level of free space on your local file system. When space usage reaches the high threshold that you set for your file system, eligible files are migrated to storage automatically. When space usage drops to the low threshold that you set for your file system, file migration stops.
- ▶ Demand migration responds to an out-of-space condition on your local file system. Selective migration moves specific files from your local file system to storage. For example, if you know that you will not be using a particular group of files for an extended time, you can migrate them to storage to free additional space on your local file system.

Considerations of HSM support for AIX GPFS and Linux x64 GPFS

Be aware of several limitations of HSM support for AIX GPFS and Linux x86/x86_64 GPFS systems:

- ▶ The management class information is for the default migration server only.
- ▶ The server options information is for the default migration server only.
- ▶ Every space management node must run the same HSM version.
- ▶ The backup-archive client cannot restore stub files to a GPFS file system that has more storage pools than the default system storage pool. Those stub files are restored to their resident state. GPFS stores the GPFS pool ID in extended attributes. The backup-archive client cannot store these extended attributes independent from the file content.

HSM support for AIX GPFS and Linux x86/x86_64 GPFS systems is not completely integrated with the backup-archive client support. For example, the HSM client refers to the following file to determine which server to use for a file system:

```
<fs>/SpaceMan/hsmfsconfig.xml
```

The client might contact a separate server for each file system. In contrast, the backup-archive client determines which server to use from the `dsm.opt` file, the `dsm.sys` file, or from a parameter that you specify on the command line when you start a backup-archive command line client. A backup-archive client process might back up, archive, restore, or retrieve from one server. If you need backup-archive client services for various servers, start a new backup-archive client process.

2.6 Summary

Fabric and storage technology is important when designing GPFS. Figure 2-35 shows a sample performance projection on GPFS. When you design a customer solution with GPFS, calculate bandwidth on all points to help you determine which point might become the bottleneck.

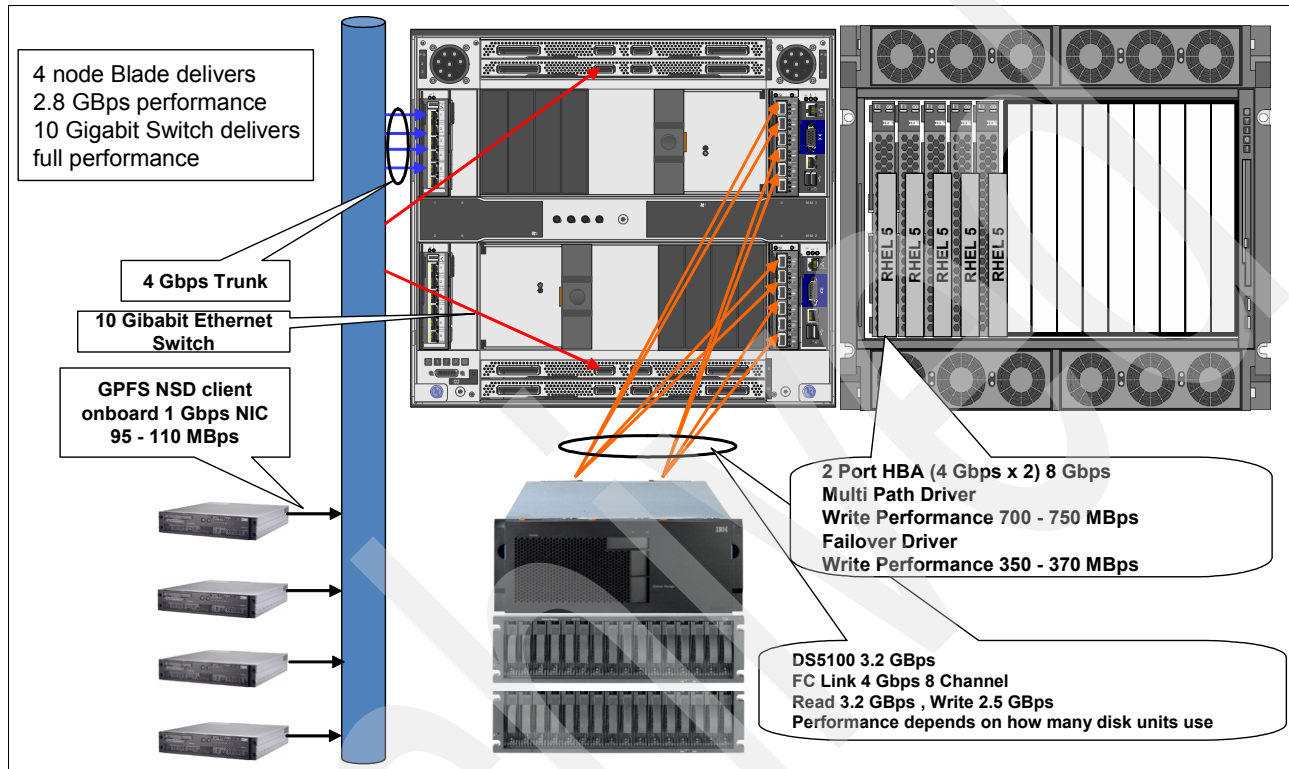


Figure 2-35 Considering all bandwidth hardware systems for GPFS

This solution can be used as the base infrastructure for all industry solutions. If there is a requirement for POSIX and shared file system for a customer solution, then GPFS might be a good solution. Many GPFS functions are for a single software stack. Many GPFS functions are for a single software stack, but the functionality depends on the operating system implemented. Therefore, be careful when configuring a heterogeneous GPFS cluster. The various industry solutions described in this chapter might help you with the GPFS design considerations.

Scenarios

This chapter describes several IBM General Parallel File System (GPFS) scenarios that we configured and tested in our ITSO lab environment. Each scenario focuses on a separate GPFS configuration or solution that customers can use within a typical enterprise.

First, the chapter describes our test environment:

- ▶ 3.1, “ITSO laboratory environment” on page 70

Then, the chapter presents the scenarios:

- ▶ 3.2, “Three-node GPFS cluster using internal disks” on page 73
- ▶ 3.3, “Recovery on the three-node cluster” on page 85
- ▶ 3.4, “Linux InfiniBand cluster with RDMA and Linux for System x clients” on page 90
- ▶ 3.5, “Cross-platform cluster: Windows servers, Linux/AIX clients” on page 97
- ▶ 3.6, “DB2 pureScale InfiniBand cluster on AIX” on page 120
- ▶ 3.7, “Multi-cluster configuration” on page 149
- ▶ 3.8, “Disaster recovery using GPFS replication” on page 162

3.1 ITSO laboratory environment

This section describes the ITSO laboratory testing environment.

The ITSO laboratory environment consists of IBM Power Systems, System x servers, SAN storage, InfiniBand and Gigabit Ethernet (GigE) networks. The hardware we used is as follows:

- ▶ Two IBM 550 servers
- ▶ Two IBM 570 servers
- ▶ Six x86_64 servers

3.1.1 Diagram of ITSO laboratory

Figure 3-1 shows the ITSO laboratory infrastructure and testing environment.

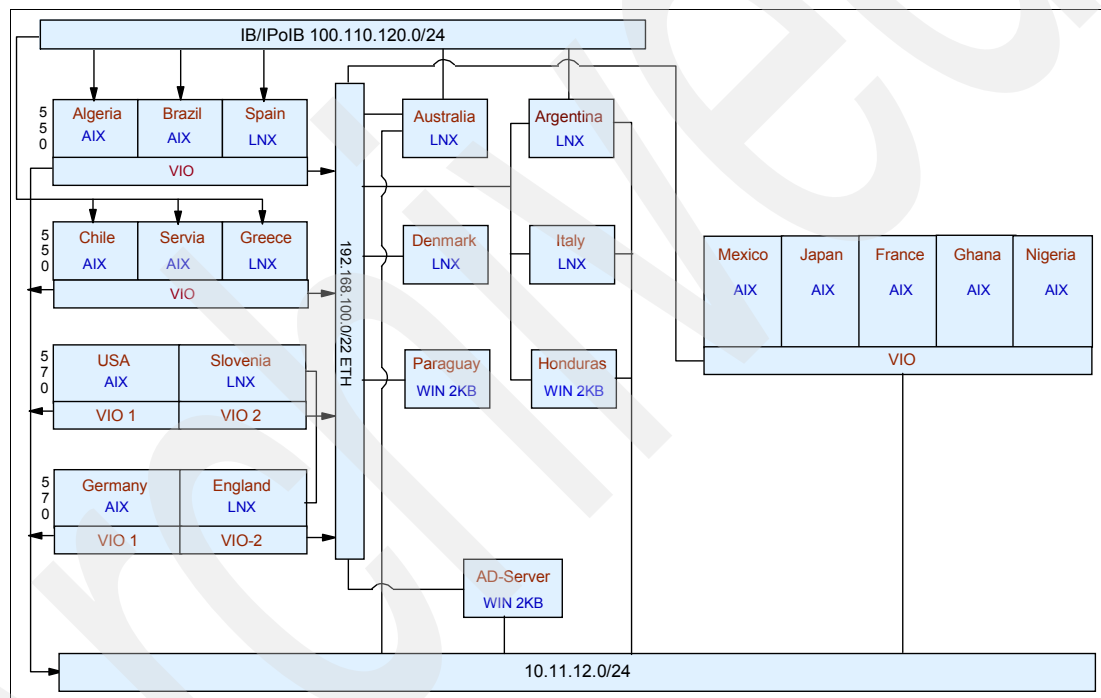


Figure 3-1 The ITSO laboratory infrastructure

3.1.2 ITSO laboratory environment host file

The /etc/hosts file, shown in Example 3-1, is used for network communications on all nodes in the ITSO lab environment.

Example 3-1 ITSO /etc/hosts file use for all scenarios in this chapter

```
# ETH - SERVICE - 192.168.100.0/22
# IPoIB          - 100.110.120.0/24
# ETH - GPFS     - 10.11.12.0/24

##### Back-end structure #####
192.168.100.4nimres2#root/itsoadmin NIM server for Residencies - LANROOM p550 4790E
192.168.100.109hmc4#hscroot/abc123 HMC for p6 570 MMAs and M50 POK12
192.168.100.110hmc3#hscroot/abc1234
```


POWER-PC SYSTEMS

9409-M50 4way 32GB # THIS WILL BE THE DR SITE !!!!!
192.168.101.120vio550-3# VIOS (already installed), 1 CPU, 4 GB RAM
192.168.102.121mexico# oracle DR node1, 0.5 CPU, 4 GB RAM
192.168.102.122japan# oracle DR node2, 0.5 CPU, 4 GB RAM
192.168.102.123france# db2 DR node1, 0.5 CPU, 4 GB RAM
192.168.102.124ghana# db2 DR node2, 0.5 CPU, 4 GB RAM
192.168.102.125nigeria# Linux DR node3 0.5 CPU, 4 GB RAM

550-1 8 way 64 GB RAM
192.168.101.100vio550-1# VIOS(already installed) 1CPU, 8GB RAM
192.168.101.101algeria# oracle node1, 2CPU, 16GB RAM
192.168.101.102brazil# db2 node1, 2CPU, 16GB RAM
192.168.101.103spain# Linux node1 with IB, 1CPU, 8GB RAM
192.168.101.104# Reserved

550-2 8 way 64 GB RAM
192.168.101.110vio550-2# VIOS (already installed) 1CPU, 8GB RAM
192.168.101.111chile# oracle node2, 2CPU, 16GB RAM
192.168.101.112serbia# db2 node2, 2CPU, 16GB RAM
192.168.101.113greece# Linux node2 with IB, 1CPU, 8GB RAM
192.168.101.104 # Reserved

570-1 4-way 32GB
192.168.101.130vio570-1# VIOS(already installed)0.5 CPU, 4 GB RAMp6 9117-MMA 0F170 - POK12
192.168.101.131vio570-11# VIOS , 0.5 CPU, 4 GB RAM
192.168.101.132 us# AIX node, 1 CPU, 8 GB RAM
192.168.101.133slovenia# Linux node, 1 CPU, 8 GB RAM

570-24-way 32GB
192.168.101.140vio570-2# VIOS(already installed)0.5 CPU, 4 GB RAMp6 9117-MMA 0F6A0 - POK12
192.168.101.141vio570-21# VIOS , 0.5 CPU, 4 GB RAM
192.168.101.142germany# AIX node, 1 CPU, 8 GB RAM
192.168.101.143england# Linux node, 1 CPU, 8 GB RAM

550-p5 4-way 32GB
192.168.101.150vio550-p5# VIOS (already installed)p5 9113-MMA

#####x86-64 SYSTEMS #####

192.168.101.150italy# TSM Linux node
192.168.101.151 argentina# Linux node1 with IB
192.168.101.152australia# Linux node2 with IB
192.168.101.153paraguay# Windows node1
192.168.101.154honduras# Windows node2
192.168.101.155denmark# KVM server
192.168.101.156 cameroon# KVM node2
192.168.101.157netherlands# KVM node1

Virtual machines

192.168.101.160 southafrica# windows VM node1
192.168.101.161 southkorea# windows VM node2
192.168.101.162slovakia# Linux VM node1
192.168.101.163newzealand# Linux VM node2

GPFS Network

```

10.11.12.221mexico-gpfs# oracle DR node1, 0.5 CPU, 4 GB RAM
10.11.12.222japan-gpfs# oracle DR node2, 0.5 CPU, 4 GB RAM
10.11.12.223france-gpfs# db2 DR node1, 0.5 CPU, 4 GB RAM
10.11.12.224ghana-gpfs# db2 DR node2, 0.5 CPU, 4 GB RAM
10.11.12.225nigeria-gpfs# Linux DR node3 0.5 CPU, 4 GB RAM

```

```

# 550-1 8 way 64 GB RAM
10.11.12.101algeria-gpfs# oracle node1, 2CPU, 16GB RAM
10.11.12.102brazil-gpfs# db2 node1, 2CPU, 16GB RAM
10.11.12.103spain-gpfs# Linux node1 with IB, 1CPU, 8GB RAM
10.11.12.104# Reserved

```

```

# 550-2 8 way 64 GB RAM

```

```

10.11.12.111chile-gpfs# oracle node2, 2CPU, 16GB RAM
10.11.12.112serbia-gpfs# db2 node2, 2CPU, 16GB RAM
10.11.12.113greece-gpfs# Linux node2 with IB, 1CPU, 8GB RAM
10.11.12.104 # Reserved

```

```

# 570-1 4-way 32GB

```

```

10.11.12.132us-gpfs# AIX node, 1 CPU, 8 GB RAM
10.11.12.133slovenia-gpfs# Linux node, 1 CPU, 8 GB RAM

```

```

# 570-24-way 32GB

```

```

10.11.12.142germany-gpfs# AIX node, 1 CPU, 8 GB RAM
10.11.12.143england-gpfs# Linux node, 1 CPU, 8 GB RAM

```

```

# 550-p5 4-way 32GB

```

```

#####x86-64 SYSTEMS #####

```

```

10.11.12.150italy-gpfs# TSM Linux node
10.11.12.151argentina-gpfs# Linux node1 with IB
10.11.12.152australia-gpfs# Linux node2 with IB
10.11.12.153paraguay-gpfs# Windows node1
10.11.12.154honduras-gpfs# Windows node2
10.11.12.155denmark-gpfs# KVM server
10.11.12.156cameroon-gpfs# KVM node2
10.11.12.157netherlands# KVM node2

```

```

##### Virtual machines ###

```

```

10.11.12.160southafrica# windows VM node1
10.11.12.161southkorea# windows VM node2
10.11.12.162slovakia# Linux VM node1
10.11.12.163newzealand# Linux VM node2

```

```

##### InfiniBand Network #####

```

```

100.110.120.221mexico-ib# oracle DR node1, 0.5 CPU, 4 GB RAM
100.110.120.222japan-ib# oracle DR node2, 0.5 CPU, 4 GB RAM
100.110.120.223france-ib# db2 DR node1, 0.5 CPU, 4 GB RAM
100.110.120.224ghana-ib# db2 DR node2, 0.5 CPU, 4 GB RAM
100.110.120.225nigeria-ib# Linux DR node3 0.5 CPU, 4 GB RAM

```

```

# 550-1 8 way 64 GB RAM
100.110.120.101algeria-ib# oracle node1, 2CPU, 16GB RAM
100.110.120.102brazil-ib# db2 node1, 2CPU, 16GB RAM

```

```

100.110.120.103spain-ib# Linux node1 with IB, 1CPU, 8GB RAM
100.110.120.104# Reserved

# 550-2 8 way 64 GB RAM

100.110.120.111chile-ib# oracle node2, 2CPU, 16GB RAM
100.110.120.112serbia-ib# db2 node2, 2CPU, 16GB RAM
100.110.120.113greece-ib# Linux node2 with IB, 1CPU, 8GB RAM
100.110.120.104 # Reserved

# 570-1 4-way 32GB

100.110.120.132us-ib# AIX node, 1 CPU, 8 GB RAM
100.110.120.133slovenia-ib# Linux node, 1 CPU, 8 GB RAM

# 570-24-way 32GB

100.110.120.142germany-ib# AIX node, 1 CPU, 8 GB RAM
100.110.120.143england-ib# Linux node, 1 CPU, 8 GB RAM

# 550-p5 4-way 32GB

#####x86-64 SYSTEMS #####
100.110.120.150italy-ib# TSM Linux node
100.110.120.151argentina-ib# Linux node1 with IB
100.110.120.152australia-ib# Linux node2 with IB
100.110.120.153paraguay-ib# Windows node1
100.110.120.154honduras-ib# Windows node2
100.110.120.155cameroon-ib# KVM node1
100.110.120.156denmark-ib# KVM node2

##### Virtual machines ###
10.11.12.160southafrica# windows VM node1
10.11.12.161southkorea# windows VM node2
10.11.12.162slovakia# Linux VM node1
10.11.12.163newzealand# Linux VM node2

```

3.2 Three-node GPFS cluster using internal disks

This scenario describes how to configure a three-node GPFS cluster solution, which uses internal disks for the GPFS file systems. GPFS data is striped across the internal disks on each node of the cluster for proper file system recovery. Without proper replication of the data, file system recovery will be impossible if a node failure occurs.

3.2.1 Requirements: Hardware, software, network, storage

The hardware requirements are as follows:

- ▶ x86_64 nodes
- ▶ Ethernet adapters
- ▶ InfiniBand adapters

The software requirements are as follows:

- ▶ Red Hat Enterprise Linux 5.5
- ▶ GPFS 3.4 for Linux on System x

The network is Gigabit Ethernet (GigE LAN), and the only storage requirement calls for internal disks.

3.2.2 GPFS configuration

Table 3-1 lists the nodes in the three-node cluster configuration.

Table 3-1 Three-node GPFS cluster node names and IP address

| Node type | Role | Name | Mgmt IP | GPFS IP |
|--------------|--------|----------------|-----------------|--------------|
| x86_64 Linux | Server | argentina-gpfs | 192.168.101.151 | 10.11.12.151 |
| x86_64 Linux | Server | australia-gpfs | 192.168.101.152 | 10.11.12.152 |
| x86_64 Linux | Server | denmark-gpfs | 192.168.101.155 | 10.11.12.155 |

In a GPFS three-node cluster that uses internal disks, having GPFS maintain the file systems that are mounted and online after a node or adapter failure is critical. GPFS uses several replication mechanisms to maintain the availability of the GPFS file system data:

- ▶ Data and metadata replication
- ▶ Log file replication
- ▶ Disk failure groups

To maintain data integrity and file system availability, the scenario uses the following GPFS features:

- ▶ Data and metadata replication on each file system
- ▶ One failure group, which is configured for the internal disks of each node
- ▶ At least three unique failure groups, which are defined on each file system

GPFS reserves a storage area on every disk that is part of a file system for the file system descriptor (FSDesc). By maintaining a quorum of file system descriptors (disks) and using disk failure groups, GPFS keeps the file system available. GPFS writes the FSDesc on three disks, which are part of that file system by default.

Disk failure groups, are used during metadata and data placement on the disks of a file system and ensures no two replicas of the same block become unavailable because of a single node failure.

GPFS *user-exit-based* event notification is used with this scenario to provide disks and file system recovery after network or node failure on any of the cluster nodes. The `mmaddcallback` command is used to register a user-defined command that GPFS executes when certain GPFS events occur.

3.2.3 Diagram of GPFS three-node cluster with internal disks

Figure 3-2 shows the three-node cluster configuration.

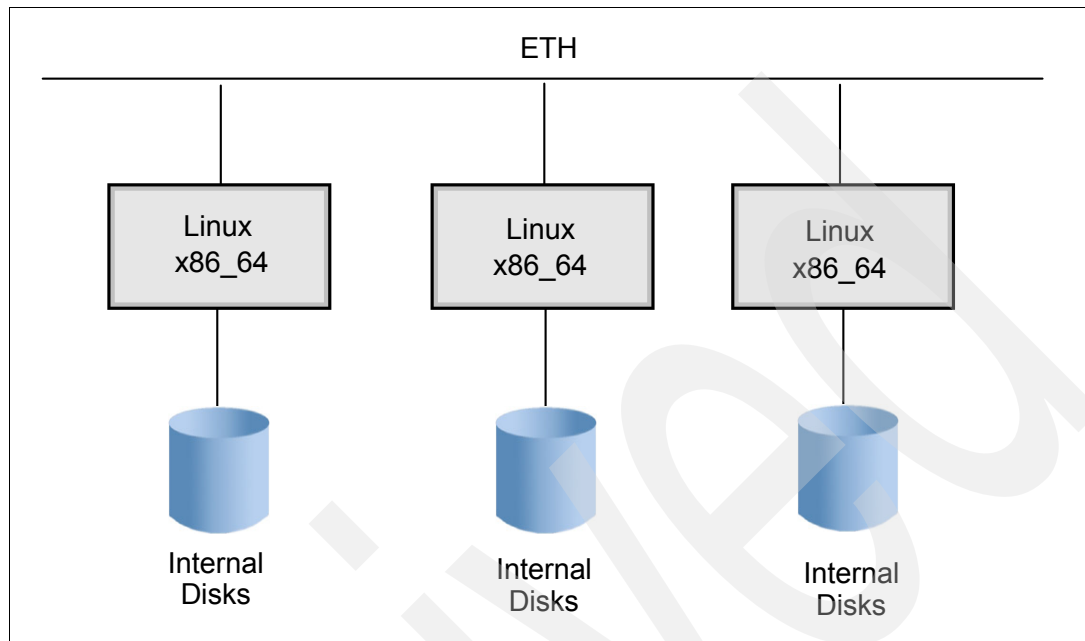


Figure 3-2 Three-node GPFS cluster using internal disks

Note: File system availability is guaranteed in a three-node GPFS cluster for network or node failure on a single node at a time.

3.2.4 Setting up and configuring a three-node cluster

The following two series of steps describe how to install GPFS, configure SSH remote shell, and get the three-node cluster configure and running. To set up and configure a three-node cluster, use the steps in the following sections:

- ▶ “Install the GPFS rpms file sets” on page 76
- ▶ “Configure SSH remote shell” on page 77
- ▶ “Compile and install the GPFS GPL layer binaries” on page 81
- ▶ “Verify GPL binaries are installed properly” on page 82
- ▶ “Create GPFS file system using NSD_Desc file” on page 82
- ▶ “Recover any down disks” on page 83

Install the GPFS rpms file sets

Prepare the nodes for the installation of the GPFS rpm file sets:

1. Check the OS level on each node in the cluster, as shown in Example 3-2.

Example 3-2 Check OS level on a Linux node

```
[root@argentina ~]# cat /etc/*release*  
cat: /etc/lsb-release.d: Is a directory  
Red Hat Enterprise Linux Server release 5.5 (Tikanga)  
[root@argentina
```

2. Check the internal disks on each node of the cluster, as shown in Example 3-3.

Example 3-3 Verify an internal disk is available on each node of the cluster

```
[root@argentina ~]# fdisk -l | grep dev  
Disk /dev/sda: 300.0 GB, 300000000000 bytes  
/dev/sda1 *          1          13      104391    83  Linux  
/dev/sda2            14         36472  292856917+  8e  Linux LVM  
Disk /dev/sdb: 300.0 GB, 300000000000 bytes  
Disk /dev/sdc: 300.0 GB, 300000000000 bytes  
Disk /dev/sdd: 300.0 GB, 300000000000 bytes  
Disk /dev/sde: 300.0 GB, 300000000000 bytes  
Disk /dev/sdf: 300.0 GB, 300000000000 bytes
```

3. Download and extract the GPFS rpms for the OS on the node, as shown in Example 3-4.

Example 3-4 Extract GPFS rpms for the OS on the node

```
[root@denmark]# ls -ltr  
total 12424  
-rw-r--r-- 1 root root 8658987 Jun 21 15:47 gpfs.base-3.4.0-0.x86_64.rpm  
-rw-r--r-- 1 root root 184447 Jun 21 15:48 gpfs.docs-3.4.0-0.noarch.rpm  
-rw-r--r-- 1 root root 452280 Jun 21 15:48 gpfs.gpl-3.4.0-0.noarch.rpm  
-rw-r--r-- 1 root root 84891 Jun 21 15:50 gpfs.msg.en_US-3.4.0-0.noarch.rpm  
[root@denmark]
```

4. Install GPFS rpms for the OS on each node as shown in Example 3-5.

Example 3-5 Install GPFS rpms for the OS on that node

```
[root@denmark]# rpm -ivh gpfs*  
Preparing... ##### [100%]  
 1:gpfs.base ##### [ 17%]  
 2:gpfs.docs ##### [ 33%]  
 3:gpfs.gpl ##### [ 50%]  
 4:gpfs.msg.en_US ##### [100%]  
[root@denmark]#
```

5. Check the GPFS rpms installed on each node as shown in Example 3-6.

Example 3-6 Check installed GPFS rpm on each node

```
[root@denmark]# rpm -qa | grep gpfs  
gpfs.base-3.4.0-0  
gpfs.msg.en_US-3.4.0-0  
gpfs.docs-3.4.0-0  
gpfs.gpl-3.4.0-0  
[root@denmark gpfs34_linux.100621.145015_amd64_sles]#
```

Configure SSH remote shell

Next, use the following steps to set up passwordless SSH remote shell execution between cluster nodes:

1. Generate public/private SSH key on one Linux node by using the following command:

```
ssh-keygen -t rsa -b 1024 -f /root/.ssh/id_rsa -N ''
```
2. Copy the following items to a Network File System (NFS) that is mounted on all nodes of the cluster:

```
/root/.ssh/id_rsa.pub  
/root/.ssh/id_rsa to a
```

3. Copy the following items to all other nodes in the cluster:

```
/root/.ssh/id_rsa.pub  
/root/.ssh/id_rsa to /root/.ssh
```

4. Create an `/NFSDIR/ssh/known_hosts` file in an NFS directory.
5. Append the `/etc/ssh/ssh_host_rsa_key.pub` from each node to the end of `/NFSDIR/known_hosts` file in the following format:

```
IF_hostname,IF_IP_address 'cat /etc/ssh/ssh_host_rsa_key.pub'
```

In the format, `IF_hostname` is the host name of that interface, and `IF_IP_address= IP` is the address of that interface.

6. Create `/NFSDIR/ssh/authorized_keys` in an NFS directory as follows:
 - a. From one node in the cluster, append `/root/.ssh/id_rsa.pub` to `/NFSDIR/ssh/authorized_keys`.
 - b. On each node of the cluster, copy `/NFSDIR/ssh/authorized_keys` to `/root/.ssh/` location.
 - c. Change the file permission of `/root/.ssh/authorized_keys` to '600' value.
 - d. Copy `/NFSDIR/ssh/known_hosts` to `/root/.ssh/` on each node of the cluster.
 - e. Change the file permission of `/root/.ssh/known_hosts` to '644' value.
 - f. Test the SSH configuration by running the following command between cluster nodes:

```
ssh hostname date
```

After configuring the SSH remote shell with these steps, continue with the configuration of the GPFS three-node cluster as follows:

7. Verify that the passwordless remote shell access works between all nodes, as shown in Example 3-7.

Example 3-7 ssh between nodes without password prompt

```
[root@argentina ~]# ssh denmark date  
Sun Jul  4 06:42:19 EDT 2010  
[root@argentina ~]# ssh australia date  
Sun Jul  4 07:46:59 EDT 2010  
[root@argentina ~]# ssh argentina date  
Sat Jul  3 08:10:00 EDT 2010  
[root@argentina ~]#
```

8. Create a three-node GPFS cluster, as shown Example 3-8.

Example 3-8 Use mmcrcluster to create ThreeNode Cluster

```
[root@argentina .ssh]# mmcrcluster -N
"argentina-gpfs:manager-quorum;australia-gpfs:quorum;denmark-gpfs:quorum" -p
argentina-gpfs -r /usr/bin/ssh -R /usr/bin/scp -C ThreeNode
mmcrcluster: Command successfully completed
mmcrcluster: Warning: Not all nodes have proper GPFS license designations.
Use the mmchlicense command to designate licenses as needed.
[root@argentina .ssh]#
```

Note: All nodes in the three-node cluster must be defined as quorum nodes to maintain node quorum in the cluster.

9. Verify the cluster definition, as shown Example 3-9.

Example 3-9 Use mmlscluster to check cluster definition

```
[root@argentina .ssh]# mmlscluster
=====
Warning:
This cluster contains nodes that do not have a proper GPFS license
designation. This violates the terms of the GPFS licensing agreement.
Use the mmchlicense command and assign the appropriate GPFS licenses
to each of the nodes in the cluster. For more information about GPFS
license designation, see the Concepts, Planning, and Installation Guide.
=====
```

GPFS cluster information

```
=====
GPFS cluster name:      ThreeNode.argentina-gpfs
GPFS cluster id:       723686009079829996
GPFS UID domain:       ThreeNode.argentina-gpfs
Remote shell command:  /usr/bin/ssh
Remote file copy command: /usr/bin/scp
```

GPFS cluster configuration servers:

```
-----
Primary server:   argentina-gpfs
Secondary server: (none)
```

| Node | Daemon node name | IP address | Admin node name | Designation |
|------|------------------|--------------|-----------------|----------------|
| 1 | argentina-gpfs | 10.11.12.151 | argentina-gpfs | quorum-manager |
| 2 | denmark-gpfs | 10.11.12.155 | denmark-gpfs | quorum-manager |
| 3 | australia-gpfs | 10.11.12.152 | australia-gpfs | quorum-manager |

```
[root@argentina .ssh]#
```


10. Add the GPFS server licenses for all nodes in the cluster, as shown in Example 3-10.

Example 3-10 Use mmchlicense to add server licenses to each node of the cluster

```
[root@argentina ~]# mmchlicense server --accept -N argentina-gpfs,australia-gpfs,denmark-gpfs
```

The following nodes will be designated as possessing GPFS server licenses:

```
    argentina-gpfs
australia-gpfs
    denmark-gpfs
```

mmchlicense: Command successfully completed

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

```
[root@argentina ~]#
```

11. Check the GPFS licenses on the cluster nodes, as shown in Example 3-11.

Example 3-11 Use mmlslicense -L to check licenses on the cluster

```
[root@argentina ~]# mmlslicense -L
```

| Node name | Required license | Designated license |
|----------------|------------------|--------------------|
| argentina-gpfs | server | server |
| australia-gpfs | server | server |
| denmark-gpfs | client | server |

Summary information

```
-----
Number of nodes defined in the cluster: 3
Number of nodes with server license designation: 3
Number of nodes with client license designation: 0
Number of nodes still requiring server license designation: 0
Number of nodes still requiring client license designation: 0
```

```
[root@argentina ~]#
```

12. Create one Network Shared Disk (NSD) for each internal disk on each node, as shown in Example 3-12.

Note: For proper data and metadata replication, define NSD as follows:

- ▶ Do not assign a secondary server to any NSD.
- ▶ Assign a separate failure groups for disks on each node.
- ▶ Assign all NSDs on one node to the same failure group.

Example 3-12 NSD Desc_File for the three-node cluster

```
[root@argentina mmcrnsd]# cat 3Node_nsd_input_save
# DiskName:ServerList::DiskUsage:FailureGroup:DesiredName:StoragePool
/dev/sdb:argentina-gpfs::dataAndMetadata:3001:argentina-gpfsNSD1::
/dev/sdc:argentina-gpfs::dataAndMetadata:3001:argentina-gpfsNSD2::
/dev/sdd:argentina-gpfs::dataAndMetadata:3001:argentina-gpfsNSD3::
/dev/sde:argentina-gpfs::dataAndMetadata:3001:australia-gpfsNSD4::
/dev/sdb:australia-gpfs::dataAndMetadata:3002:australia-gpfsNSD1::
/dev/sdc:australia-gpfs::dataAndMetadata:3002:australia-gpfsNSD2::
/dev/sdd:australia-gpfs::dataAndMetadata:3002:australia-gpfsNSD3::
/dev/sde:australia-gpfs::dataAndMetadata:3002:australia-gpfsNSD4::
/dev/sdb:denmark-gpfs::dataAndMetadata:3003:denmark-gpfsNSD1::
```

```
/dev/sdc:denamrk-gpfs::dataAndMetadata:3003:denmark-gpfsNSD2::  
/dev/sdd:denmark-gpfs::dataAndMetadata:3003:denmark-gpfsNSD3::  
/dev/sde:denmark-gpfs::dataAndMetadata:3003:denmark-gpfsNSD4::  
[root@argentina mmcrnsd]#
```

13. Create the NSD descriptor file with the **mmcrnsd** command, as shown in Example 3-13.

Example 3-13 Use mmcrnsd command to create NSDs

```
[root@argentina mmcrnsd]# mmcrnsd -F 3Node_nsd_input -v yes  
mmcrnsd: Processing disk sdb  
mmcrnsd: Processing disk sdc  
mmcrnsd: Processing disk sdd  
mmcrnsd: Processing disk sde  
mmcrnsd: Processing disk sdb  
mmcrnsd: Processing disk sdc  
mmcrnsd: Processing disk sdd  
mmcrnsd: Processing disk sde  
mmcrnsd: Processing disk sdb  
mmcrnsd: Processing disk sdc  
mmcrnsd: Processing disk sdd  
mmcrnsd: Processing disk sde  
mmcrnsd: Propagating the cluster configuration data to all  
affected nodes. This is an asynchronous process.  
[root@argentina mmcrnsd]#
```

14. Check NSD definitions in the NSD_Desc file.

Tip: After the **mmcrnsd** command completes successfully, the original NSD_Desc file is modified by GPFS and the NSD definitions are added.

Example 3-14 shows the content of the NSD_Desc file after the NSD are successfully created.

Example 3-14 Show the content of the NSD_Desc file after the mmcrnsd completes

```
[root@argentina mmcrnsd]# cat 3Node_nsd_input  
# DiskName:ServerList::DiskUsage:FailureGroup:DesiredName:StoragePool  
# /dev/sdb:argentina-gpfs::dataAndMetadata:3001:argentinaNSD1::  
argentinaNSD1::dataAndMetadata:3001::  
# /dev/sdc:argentina-gpfs::dataAndMetadata:3001:argentinaNSD2::  
argentinaNSD2::dataAndMetadata:3001::  
# /dev/sdd:argentina-gpfs::dataAndMetadata:3001:argentinaNSD3::  
argentinaNSD3::dataAndMetadata:3001::  
# /dev/sde:argentina-gpfs::dataAndMetadata:3001:argentinaNSD4::  
argentinaNSD4::dataAndMetadata:3001::  
# /dev/sdb:australia-gpfs::dataAndMetadata:3002:australiaNSD1::  
australiaNSD1::dataAndMetadata:3002::  
# /dev/sdc:australia-gpfs::dataAndMetadata:3002:australiaNSD2::  
australiaNSD2::dataAndMetadata:3002::  
# /dev/sdd:australia-gpfs::dataAndMetadata:3002:australiaNSD3::  
australiaNSD3::dataAndMetadata:3002::  
# /dev/sde:australia-gpfs::dataAndMetadata:3002:australiaNSD4::  
australiaNSD4::dataAndMetadata:3002::  
# /dev/sdb:denmark-gpfs::dataAndMetadata:3003:denmarkNSD1::  
denmarkNSD1::dataAndMetadata:3003::
```

```
# /dev/sdc:denmark-gpfs::dataAndMetadata:3003:denmarkNSD2::
denmarkNSD2:::dataAndMetadata:3003::
# /dev/sdd:denmark-gpfs::dataAndMetadata:3003:denmarkNSD3::
denmarkNSD3:::dataAndMetadata:3003::
# /dev/sde:denmark-gpfs::dataAndMetadata:3003:denmarkNSD4::
denmarkNSD4:::dataAndMetadata:3003::
[root@argentina mmcrnsd]#
```

15. List all defined NSD on the cluster, as shown in Example 3-15.

Example 3-15 Use mmlsnsd to list defined NSD on cluster

```
[root@argentina mmcrnsd]# mmlsnsd
```

| File system | Disk name | NSD servers |
|-------------|---------------|----------------|
| (free disk) | argentinaNSD1 | argentina-gpfs |
| (free disk) | argentinaNSD2 | argentina-gpfs |
| (free disk) | argentinaNSD3 | argentina-gpfs |
| (free disk) | argentinaNSD4 | argentina-gpfs |
| (free disk) | australiaNSD1 | australia-gpfs |
| (free disk) | australiaNSD2 | australia-gpfs |
| (free disk) | australiaNSD3 | australia-gpfs |
| (free disk) | australiaNSD4 | australia-gpfs |
| (free disk) | denmarkNSD1 | denmark-gpfs |
| (free disk) | denmarkNSD2 | denmark-gpfs |
| (free disk) | denmarkNSD3 | denmark-gpfs |
| (free disk) | denmarkNSD4 | denmark-gpfs |

```
[root@argentina mmcrnsd]#
```

Compile and install the GPFS GPL layer binaries

The GPFS General Public License (GPL) binaries are platform-dependent so they must be compiled and built in the current environment each time new GPFS code is installed. See Example 3-16.

Attention: Always check the compiler warning messages and the errors. Check for completion RC=0 messages, although there may be serious warning messages that might require you to contact IBM.

Example 3-16 Compile and install GPL binaries

```
[root@argentina mmcrfs]# cd /usr/lpp/mmfs/src
[root@argentina src]# make Autoconfig World InstallImages 2>/tmp/make_err 1>/tmp/make_out

[root@argentina src]# grep -i WAR /tmp/make_err
/usr/lpp/mmfs/src/gpl-linux/cxiCache.c:745: warning: assignment discards qualifiers from
pointer target type
/usr/lpp/mmfs/src/gpl-linux/cxiCache.c:910: warning: assignment discards qualifiers from
pointer target type
/usr/lpp/mmfs/src/gpl-linux/cxiCache.c:880: warning: âfileNamePâ may be used uninitialized
in this function
WARNING: could not find /usr/lpp/mmfs/src/gpl-linux/.mmfs.o_shipped.cmd for
/usr/lpp/mmfs/src/gpl-linux/mmfs.o_shipped
WARNING: could not find /usr/lpp/mmfs/src/gpl-linux/.libgcc.a_shipped.cmd for
/usr/lpp/mmfs/src/gpl-linux/libgcc.a_shipped
[root@argentina src]#
```

Verify GPL binaries are installed properly

Check in the `/lib/modules/<kernel-release>/extra` directory for the following modules and binaries:

- ▶ `tracedev.ko`
- ▶ `mmfslinux.ko`
- ▶ `mmfs26.ko`

Tip: GPL binaries are installed in the `/lib/modules/'uname -r'/extra` location, as shown in Example 3-17.

Example 3-17 List `/lib/modules/2.6.18-194.el5/extra`

```
[root@argentina ~]# ls -ltr /lib/modules/2.6.18-194.el5/extra
total 4876
-rw-r--r-- 1 root root 588322 Jun 23 08:48 tracedev.ko
-rw-r--r-- 1 root root 1670867 Jun 23 08:48 mmfslinux.ko
-rw-r--r-- 1 root root 2716822 Jun 23 08:48 mmfs26.ko
[root@argentina ~]#
```

Create GPFS file system using `NSD_Desc` file

The NSD descriptor file from the NSD creation step is used as input to the `mmcrfs` command to create the GPFS file systems. See Example 3-18.

Example 3-18 Use `mmcrfs` to create the GPFS file systems

```
[root@argentina mmcrfs]# cat mmcrfs_gpfs1.sh
mmcrfs /gpfs1 /dev/gpfs1 -F 3Node_nsd_input -B 256k -n 80 -v no -R 2 -M 2 -r 2 -m2 -A yes -Q no
[root@argentina mmcrfs]#
[root@argentina mmcrfs]# ./mmcrfs_gpfs1.sh
The following disks of gpfs1 will be formatted on node argentina:
  argentinaNSD1: size 292968750 KB
  argentinaNSD2: size 292968750 KB
  argentinaNSD3: size 292968750 KB
  argentinaNSD4: size 292968750 KB
  australiaNSD1: size 292968750 KB
  australiaNSD2: size 292968750 KB
  australiaNSD3: size 292968750 KB
  australiaNSD4: size 292968750 KB
  denmarkNSD1: size 292968750 KB
  denmarkNSD2: size 292968750 KB
  denmarkNSD3: size 292968750 KB
  denmarkNSD4: size 292968750 KB
Formatting file system ...

Disks up to size 5.6 TB can be added to storage pool 'system'.
Creating Inode File
Creating Allocation Maps
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool 'system'
  72 % complete on Wed Jun 23 09:43:19 2010
 100 % complete on Wed Jun 23 09:43:21 2010
Completed creation of file system /dev/gpfs1.
mmcrfs: Propagating the cluster configuration data to all
        affected nodes. This is an asynchronous process.
[root@argentina mmcrfs]#
```

The steps are as follows:

1. Mount the file system on all cluster nodes, as shown in Example 3-19.

Example 3-19 Use mmmount to mount gpfs1 file system on all nodes of the cluster

```
[root@argentina mmcrfs]# mmmount gpfs1 -a
Wed Jun 23 09:48:09 EDT 2010: mmmount: Mounting file systems ...
[root@argentina mmcrfs]#
```

2. Verify that the /gpfs1 directory is mounted on all three cluster nodes, as shown in Example 3-20.

Example 3-20 Use mmlsmount to verify the directory /gpfs1 is mounted on all cluster nodes

```
[root@argentina mmcrfs]# mmlsmount all -L

File system gpfs1 is mounted on 3 nodes:
10.11.12.151    argentina-gpfs
10.11.12.152    australia-gpfs
10.11.12.155    denmark-gpfs
[root@argentina mmcrfs]#
```

3. Check the state of the GPFS daemon on each node of the cluster, as shown in Example 3-21.

Example 3-21 Use mmgetstate to check the status of the GPFS daemon on each node

```
[root@argentina src]# mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|----------------|--------|----------|-------------|------------|-------------|
| 1 | argentina-gpfs | 2 | 3 | 3 | active | quorum node |
| 2 | denmark-gpfs | 2 | 3 | 3 | active | quorum node |
| 3 | australia-gpfs | 2 | 3 | 3 | active | quorum node |

Summary information

```
-----
Number of nodes defined in the cluster:      3
Number of local nodes active in the cluster:  3
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 3
Number of quorum nodes active in the cluster: 3
Quorum = 2, Quorum achieved

[root@argentina src]#
```

Recover any down disks

Set up and install a user-exit, and verify installation and recovery of down disks.

Set up a user-exit to recover down disks

Important: After GPFS recovers from a network or node failure, the internal disks on the node where the failure occurred are in a “down” state.

All disks that are in a down state must be brought back online (to an “up” state) for proper file system operation. Manual intervention is necessary to bring those disks online. To automate

the process of bringing the “down” disks to the up state, a user-exit is configured in the cluster nodes that recovers any down disks in all file systems.

Example 3-22 shows three disks in the “down” state after network or node failure.

Example 3-22 Example of down disks after failure on denmark

```
[root@denmark x86_64]# mmlsdisk gpfs1
```

| disk name | driver type | sector size | failure group | holds metadata | holds data | status | availability | storage pool |
|---------------|----------------|----------------|------------------|-------------------|---------------|--------|--------------|-----------------|
| <hr/> | | | | | | | | |
| argentinaNSD1 | nsd | 512 | 3001 | yes | yes | ready | up | system |
| argentinaNSD2 | nsd | 512 | 3001 | yes | yes | ready | up | system |
| argentinaNSD3 | nsd | 512 | 3001 | yes | yes | ready | up | system |
| argentinaNSD4 | nsd | 512 | 3001 | yes | yes | ready | up | system |
| australiaNSD1 | nsd | 512 | 3002 | yes | yes | ready | up | system |
| australiaNSD2 | nsd | 512 | 3002 | yes | yes | ready | up | system |
| australiaNSD3 | nsd | 512 | 3002 | yes | yes | ready | up | system |
| australiaNSD4 | nsd | 512 | 3002 | yes | yes | ready | up | system |
| denmarkNSD1 | nsd | 512 | 3003 | yes | yes | ready | down | system |
| denmarkNSD2 | nsd | 512 | 3003 | yes | yes | ready | up | system |
| denmarkNSD3 | nsd | 512 | 3003 | yes | yes | ready | down | system |
| denmarkNSD4 | nsd | 512 | 3003 | yes | yes | ready | down | system |

```
[root@denmark x86_64]#
```

Set up a user-exit (downDiskHandler)

The downDiskHandler user-exit is called after the GPFS startup local event. See Example 3-23.

Example 3-23 GPFS (downDiskHandler) user-exit

```
[root@australia tmp]# cat Down_Disk_Exit
downDiskHandler --command /tmp/recover_down_disk.sh --event startup -N all --sync --parms
"%eventName %eventNode.shortName %downNodes.shortName "
[root@australia tmp]#
```

Install a (downDiskHandler) user-exit

Use the **mmaddcallback** command to install User-Exit (downDiskHandler) which is triggered on the GPFS startup event. See Example 3-24.

Example 3-24 Set up user-exit (downDiskHandler)

```
[root@australia tmp]# mmaddcallback -S Down_Disk_Exit
mmaddcallback: Sun Jul 11 12:03:37 EDT 2010: Processing line downDiskHandler --command
/tmp/recover_down_disk.sh --event startup -N all --sync --parms "%eventName
%eventNode.shortName %downNodes.shortName "
mmaddcallback: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root@australia tmp]#
```

Verify proper installation and recovery of down disks

The GPFS startup event is triggered after a successful GPFS startup and when the node is ready for user initiated sessions.

For the (downDiskHandler) user-exit to function properly, the /tmp/recover_down_disk.sh script must be installed in the same location on all node of the cluster. Use the **mmcallback** command to verify (Example 3-25 on page 85).

Example 3-25 The mmlscallback command verifies (downDiskHandler) user-exit set up

```
[root@australia tmp]# mmlscallback
downDiskHandler
    command      = /tmp/recover_down_disk.sh
    sync         = true
    event        = startup
    node         = all
    parms        = %eventName %eventNode.shortName %downNodes.shortName

[root@australia tmp]#
```

The Korn shell (ksh) script /tmp/recover_down_disk.sh processes down disks in each file system on the cluster and starts the **mmrestripefs** command (to re-stripe the file systems with the disks incorporated into the GPFS cluster), as shown in Example 3-26.

Example 3-26 /tmp/recover_down_disk.sh ksh script to process down disks and run mmrestripefs

```
[root@australia tmp]# cat /tmp/recover_down_disk.sh
#!/bin/ksh
DAT=`date`
gdd='gpfs_down_disk_'
rsfs='RestripeFS_log'

/usr/lpp/mmfs/bin/mmlsconfig | grep dev >/tmp/gpfsFS

while read n1 n2
do
    /usr/lpp/mmfs/bin/mmlsdisk $n1 -e | grep -v grep | grep nsd | grep down | awk '{print $1}'
    >/tmp/$gdd$n1
    while read m1 m2
    do
        echo working on $n1 down disk $m1
        /usr/lpp/mmfs/bin/mmchdisk $n1 start -d "$m1"
        done</tmp/$gdd$n1
        nohup /usr/lpp/mmfs/bin/mmrestripefs $n1 -b >/tmp/$n1$rsfs 2>&1 &
    done </tmp/gpfsFS

[root@australia tmp]#
```

3.3 Recovery on the three-node cluster

The purpose of simulated errors on this GPFS three-node internal disks cluster is to show data integrity and availability of the file systems in the event of a node or network failure on any of the nodes. A simulated error is described in “Simulated GPFS Gigabit Ethernet adapter failure” on page 85.

Simulated GPFS Gigabit Ethernet adapter failure

This three-node GPFS cluster is configured to use Gigabit Ethernet LAN through the eth1 adapter. Verify that the network adapter, which is used by the GPFS daemon, is operational prior to any failure simulation. See Example 3-27 on page 86.

Example 3-27 Use ifconfig to check the status of the Gigabit Ethernet adapter (eth1)

```
[root@denmark x86_64]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:14:5E:B4:16:3A
          inet addr:192.168.101.155  Bcast:192.168.103.255  Mask:255.255.252.0
          inet6 addr: fe80::214:5eff:feb4:163a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1603776 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8751 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:159111632 (151.7 MiB)  TX bytes:1198737 (1.1 MiB)
          Interrupt:169 Memory:d9ff0000-da000000

eth1      Link encap:Ethernet  HWaddr 00:14:5E:B4:16:3B
          inet addr:10.11.12.155  Bcast:10.11.12.255  Mask:255.255.255.0
          inet6 addr: fe80::214:5eff:feb4:163b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:629870 errors:0 dropped:0 overruns:0 frame:0
          TX packets:157792 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:701535942 (669.0 MiB)  TX bytes:79160187 (75.4 MiB)
          Interrupt:169 Memory:d7ff0000-d8000000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:7311 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7311 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:3735251 (3.5 MiB)  TX bytes:3735251 (3.5 MiB)
[root@denmark x86_64]#
```

Verify that the file systems are mounted prior to interface being down

Prior to the start of any failure scenarios on any node in the cluster, verify all GPFS file systems are mounted and accessible on all cluster nodes. Use the `mmfsmount` command, as shown in Example 3-28.

Example 3-28 use mmfsmount command to verify file systems are mounted on all cluster nodes

```
[root@denmark ~]# mmfsmount all -L

File system gpfs1 is mounted on 3 nodes:
10.11.12.151    argentina-gpfs
10.11.12.152    australia-gpfs
10.11.12.155    denmark-gpfs
[root@denmark ~]#
```

Bring down GPFS daemon Gigabit Ethernet interface

Use the `ifconfig` command can be used to bring the Gigabit Ethernet (GigE) interface on a specific node down or up when simulating a GPFS network adapter failure. See Example 3-29.

Example 3-29 Use ifconfig to bring down the Gigabit Ethernet (eth1) interface

```
[root@denmark x86_64]# ifconfig eth1 down
```

Verify the GigE interface is down

When a network interface is down, the **ifconfig** command output does not show any data for that network interface because it is in the down state, as shown in Example 3-30.

Example 3-30 ifconfig command does not show data for eth1 when interface is down

```
[root@denmark x86_64]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:14:5E:B4:16:3A
          inet addr:192.168.101.155  Bcast:192.168.103.255  Mask:255.255.252.0
          inet6 addr: fe80::214:5eff:feb4:163a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1603875 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8790 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:159119334 (151.7 MiB)  TX bytes:1205077 (1.1 MiB)
          Interrupt:169 Memory:d9ff0000-da000000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:7311 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7311 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:3735251 (3.5 MiB)  TX bytes:3735251 (3.5 MiB)
```

```
[root@denmark x86_64]#
```

Expected results and symptoms after simulated adapter failure

After the GPFS daemon interface is down, a GPFS recovery operation begins soon after the down network adapter is detected. Eventually, all file systems on the node with the adapter failure are unmounted.

Two symptoms of an adapter failure on a cluster node are (see Example 3-31):

- ▶ The **df** command shows that the file systems are not mounted.
- ▶ The **df** command shows a “Stale NFS file handle” message.

Example 3-31 Use the df command to show GPFS file systems no longer mounted on that node

```
[root@denmark x86_64]# df
Filesystem          1K-blocks      Used Available Use% Mounted on
/dev/mapper/Vo1Group00-LogVol100
268790176 3472164 251444044    2% /
/dev/sda1            101086    13188    82679   14% /boot
tmpfs                4088764         0   4088764    0% /dev/shm
df: ~/gpfs1': Stale NFS file handle
[root@denmark x86_64]#
```

Verify file systems remain mounted on the other cluster nodes

When any node in the cluster experiences a network adapter or node failure, the GPFS file systems remain mounted, available and accessible on all other cluster. The **df** command and **mmfs mount** command can be used to verify that the GPFS file systems are mounted on the remaining nodes of the cluster, as shown in Example 3-32.

Example 3-32 Verify file systems remain mounted on all other cluster nodes

```
[root@australia mmcrfs]# df /gpfs1
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/gpfs1            3515624448    2008832 3513615616   1% /gpfs1
[root@australia mmcrfs]#

[root@argentina mmcrfs]# df /gpfs1
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/gpfs1            3515624448    2008832 3513615616   1% /gpfs1
[root@argentina mmcrfs]#

[root@argentina mmcrfs]# mmfs mount all -L
File system gpfs1 is mounted on 2 nodes:
    10.11.12.151    argentina-gpfs
    10.11.12.152    australia-gpfs
[root@argentina mmcrfs]#
```

Bring GPFS GigE interface back online

After several minutes, start the GigE interface on the node and bring it online with the **ifconfig** command, as shown in Example 3-33.

Example 3-33 Use ifconfig to bring eth1 interface online

```
[root@denmark x86_64]# ifconfig eth1 up
```

Check the state of the GigE interface

Verify the eth1 interface is now up and operational on that node by using the **ifconfig** command, as shown in Example 3-34.

Example 3-34 Verify GigE interface has come up and is operational

```
[root@argentina mmcrfs]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:11:25:3F:70:52
          inet addr:192.168.101.151 Bcast:192.168.103.255 Mask:255.255.252.0
          inet6 addr: fe80::211:25ff:fe3f:7052/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1974552 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1828944 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1874147257 (1.7 GiB) TX bytes:1906953413 (1.7 GiB)
          Interrupt:169 Memory:cbbf0000-cbc00000

eth1      Link encap:Ethernet  HWaddr 00:11:25:3F:70:53
          inet addr:10.11.12.151 Bcast:10.11.12.255 Mask:255.255.255.0
          inet6 addr: fe80::211:25ff:fe3f:7053/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5290219 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5939331 errors:0 dropped:0 overruns:0 carrier:0
```

```

collisions:0 txqueuelen:1000
RX bytes:7562930201 (7.0 GiB) TX bytes:8814999284 (8.2 GiB)
Interrupt:169 Memory:c9ff0000-ca000000

ib0      Link encap:InfiniBand HWaddr
80:00:00:48:FE:80:00:00:00:00:00:00:00:00:00:00:00:00
    inet addr:100.110.120.151 Bcast:100.110.120.255 Mask:255.255.255.0
    inet6 addr: fe80::202:c903:0:2d85/64 Scope:Link
    UP BROADCAST RUNNING MULTICAST MTU:65520 Metric:1
    RX packets:320636 errors:0 dropped:0 overruns:0 frame:0
    TX packets:320611 errors:0 dropped:19 overruns:0 carrier:0
    collisions:0 txqueuelen:256
    RX bytes:7219603650 (6.7 GiB) TX bytes:7220872817 (6.7 GiB)

lo       Link encap:Local Loopback
    inet addr:127.0.0.1 Mask:255.0.0.0
    inet6 addr: ::1/128 Scope:Host
    UP LOOPBACK RUNNING MTU:16436 Metric:1
    RX packets:9345 errors:0 dropped:0 overruns:0 frame:0
    TX packets:9345 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:0
    RX bytes:9696636 (9.2 MiB) TX bytes:9696636 (9.2 MiB)

```

```
[root@argentina mmcrfs]#
```

Verify GPFS file system is remounted

After GPFS detects the interface is back up, recovery of the node begins and all file systems (with the automount set parameter) are remounted on that node. See Example 3-35.

Example 3-35 Use mmlsmount to verify GPFS file systems are now re-mounted on denmark

```
[root@argentina mmcrfs]# mmlsmount all -L
```

```
File system gpfs1 is mounted on 3 nodes:
```

```

10.11.12.151    argentina-gpfs
10.11.12.152    australia-gpfs
10.11.12.155    denmark-gpfs

```

```
[root@argentina mmcrfs]#
```

3.4 Linux InfiniBand cluster with RDMA and Linux for System x clients

This scenario describes a GPFS Linux InfiniBand cluster with NSD servers on the Linux nodes and configured to use RDMA. This scenario uses the Spain, Greece, and Nigeria nodes, the marked nodes in Figure 3-3.

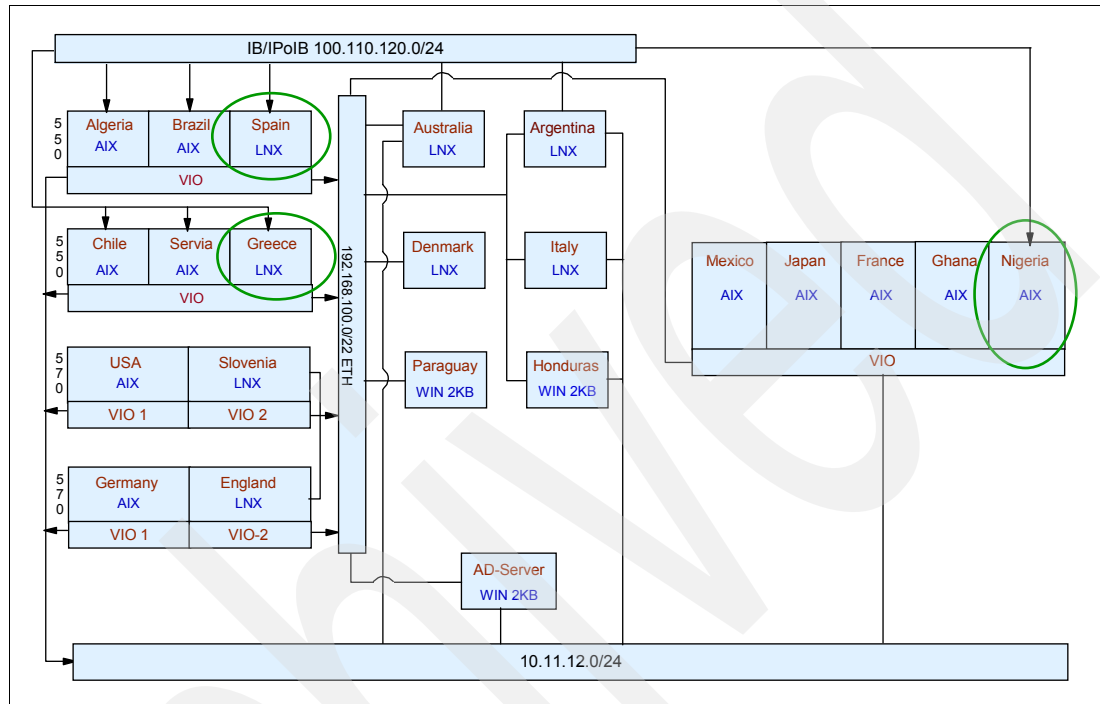


Figure 3-3 GPFS InfiniBand nodes

There are several advantages of using InfiniBand network:

- ▶ Is a low latency network.
- ▶ Is a fast network. In this configuration, the link is 20 Gbps, but a configuration that is using the latest hardware can provide 40 Gbps and 60 Gbps. The technology is improving
- ▶ Can transport IP layer and storage layer on the same hardware infrastructure by combining both communication technology (IP and storage) on the same network layer.
- ▶ Supports RDMA protocol, which has several major advantages:
 - Offers zero-copy technology that permits transferring data between the memory of separate nodes.
 - Lowers CPU utilization.

The targeted GPFS configuration is shown in Figure 3-4.

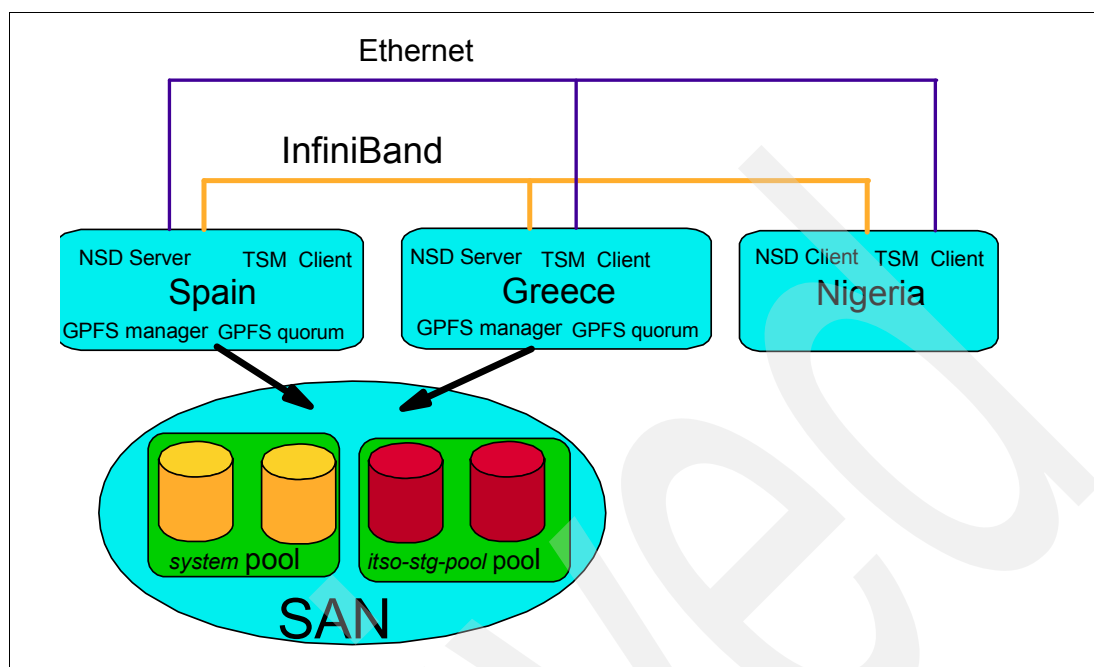


Figure 3-4 GPFS InfiniBand environment

3.4.1 Requirements: Hardware and software

The hardware requirements are as follows:

- ▶ Three Linux on IBM Power System LPARs
- ▶ Power Systems InfiniBand adapters
- ▶ InfiniBand switch and cables

The software requirements are as follows:

- ▶ Linux on Power Systems Red Hat Enterprise Linux 5.5
- ▶ GPFS 3.4 for Linux on Power Systems
- ▶ InfiniBand drivers
- ▶ Subnet Manager for InfiniBand

3.4.2 Ethernet GPFS cluster

Create a basic GPFS Linux on System p cluster by using the nodes spain, greece and nigeria as follows:

1. Install the gpfs3.4 code on all the nodes as shown in Example 3-36.

Example 3-36 Installing GPFS on Linux on System p

```
[root@spain ppc64_sles9]# rpm -ivh *.rpm
Preparing... ##### [100%]
T 1:gpfs.base ##### [14%]
   2:gpfs.docs ##### [29%]
   3:gpfs.gpl ##### [43%]
   4:gpfs.libsrc ##### [57%]
   5:gpfs.msg.en_US ##### [71%]
   6:gpfs.src ##### [86%]
```

2. Add GPFS binary into the PATH variable as shown in Example 3-37.

Example 3-37 GPFS PATH configuration

```
[root@spain ppc64_sles9]# cat ~/.bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin:/usr/lpp/mmfs/bin

export PATH
unset USERNAME
```

3. Set up SSH key exchange (See “Configure SSH remote shell” on page 77)
To check that the SSH key exchange is done properly, verify that every node can perform the following operations:
 - Connect to all other nodes without password prompt for the root user
 - Use **ssh** to itself without a password prompt for the root user.
 - Use **ssh** to its local host adapter without a password prompt for the root user.
4. Create the portability layer. Follow the instruction in the `/usr/lpp/mmfs/src/README` file, and run the following commands:

```
cd /usr/lpp/mmfs/src
make Autoconfig
make World
make InstallImages
```
5. Set up the system clock so all the nodes are using the same date from the Network Time Protocol (NTP) server.
6. Create the GPFS cluster and accept the license to continue, as shown in Example 3-38.

Example 3-38 Create the GPFS cluster and accept the license

```
[root@spain work]# cat nodes
spain-gpfs:manager-quorum:
greece-gpfs:manager-quorum:
nigeria-gpfs
[root@spain work]# mmcrcluster -N nodes -p spain-gpfs -s greece-gpfs -r /usr/bin/ssh -R /usr/bin/scp -C GPFS-InfiniBand
Fri Jun 25 12:13:21 EDT 2010: mmcrcluster: Processing node spain-gpfs
Fri Jun 25 12:13:22 EDT 2010: mmcrcluster: Processing node greece-gpfs
Fri Jun 25 12:13:23 EDT 2010: mmcrcluster: Processing node nigeria-gpfs
mmcrcluster: Command successfully completed
mmcrcluster: Warning: Not all nodes have proper GPFS license designations.
    Use the mmchlicense command to designate licenses as needed.
mmcrcluster: Propagating the cluster configuration data to all
    affected nodes. This is an asynchronous process.
root@spain work]# mmchlicense server --accept -N spain-gpfs,greece-gpfs
root@spain work]# mmchlicense client --accept -N nigeria-gpfs
```

7. Create the NSD disk, as shown in Example 3-39.

Example 3-39 Create the nsd disks

```
[root@spain work]# cat nsd
/dev/sdb:spain-gpfs,greece-gpfs::dataAndMetadata:1:::
/dev/sdc:greece-gpfs,spain-gpfs::dataAndMetadata:2:::
[root@spain work]# mmcrnsd -F nsd -v no
mmcrnsd: Processing disk sdb
mmcrnsd: Processing disk sdc
mmcrnsd: Propagating the cluster configuration data to all
        affected nodes. This is an asynchronous process.
```

8. Start the GPFS cluster.
9. Create the GPFS file system. See Example 3-40.

Example 3-40 Create the file system

```
mmcrfs /dev/gpfs-ib -F nsd -A yes -B 256K -M 2 -n 8 -R 2 -T /gpfs-ib
```

The following disks of gpfs-ib will be formatted on node spain:

gpfs1nsd: size 26214400 KB

gpfs2nsd: size 26214400 KB

Formatting file system ...

Disks up to size 219 GB can be added to storage pool 'system'.

Creating Inode File

Creating Allocation Maps

Clearing Inode Allocation Map

Clearing Block Allocation Map

Formatting Allocation Map for storage pool 'system'

Completed creation of file system /dev/gpfs-ib.

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

10. Mount the file system on all nodes by running the **mmmount all -a** command.
11. Test the performance of the GPFS cluster by running the **gpfsperf** command.

The main reason for this testing is to show the difference between sending data over Ethernet versus using RDMA on InfiniBand. The Ethernet test is shown in Example 3-41. Because this Ethernet network is also shared with other systems, the network has high traffic. We run the test on nigeria node twice for a consistent outcome. Notice that the nigeria node is an NSD client, so all the data goes through the network.

Note: Because this is not a benchmark test, the results themselves are not important, but the difference between them is. GPFS is using the default values.

Example 3-41 Sample GPFS tests

```
[root@nigeria perf]# ./gpfsperf create seq /gpfs-ib/gogo -r 256k -n 1024000000
-th 4
./gpfsperf create seq /gpfs-ib/gogo
recSize 256K nBytes 999936K fileSize 999936K
nProcesses 1 nThreadsPerProcess 4
file cache flushed before test
not using data shipping
not using direct I/O
```

```

offsets accessed will cycle through the same file segment
not using shared memory buffer
not releasing byte-range token after open
no fsync at end of test
Data rate was 49454.62 Kbytes/sec, thread utilization 0.969
[root@nigeria perf]# ./gpfsperf create seq /gpfs-ib/gogo -r 256k -n 1024000000
-th 4
./gpfsperf create seq /gpfs-ib/gogo
recSize 256K nBytes 999936K fileSize 999936K
nProcesses 1 nThreadsPerProcess 4
file cache flushed before test
not using data shipping
not using direct I/O
offsets accessed will cycle through the same file segment
not using shared memory buffer
not releasing byte-range token after open
no fsync at end of test
Data rate was 50919.39 Kbytes/sec, thread utilization 0.958

```

3.4.3 InfiniBand GPFS cluster

This section uses the previous cluster (from 3.4.2, “Ethernet GPFS cluster” on page 91) and modifies it to use the InfiniBand adapters.

Configure the InfiniBand network

Our InfiniBand network uses a QLogic switch 9024 and one port from each adapter. We use the Subnet Manager build in the switch itself.

For more information about setting up the InfiniBand infrastructure, see *HPC Clusters Using InfiniBand on IBM Power Systems Servers*, SG24-7767.

To set up GPFS to use InfiniBand, perform the following steps:

1. Create a symbolic link (symlink) for the libverbs library, use the following command:

```
ln -s /usr/lib64/libibverbs.so.1.0.0 /usr/lib64/libibverbs.so
```

Note: At the time of writing, the symlink is required because the InfiniBand libraries might not work otherwise. Check for the latest InfiniBand drivers and OpenFabrics Enterprise Distribution (OFED) libraries if this symlink is still required.

2. Identify the adapter name and port that will be used by GPFS, as shown in Example 3-42, where ehca0 is the adapter name. Also be sure the state port is *Active* and the physical state is *LinkUp*.

Example 3-42 Collecting infiniband information

```

root@spain work]# ibstat
CA 'ehca0'
CA type:
Number of ports: 1
Firmware version:
Hardware version:
Node GUID: 0x00025500403bd600
System image GUID: 0x0000000000000000

```


Port 1:

State: Active
Physical state: LinkUp
Rate: 20
Base lid: 7
LMC: 0
SM lid: 1
Capability mask: 0x02010068
Port GUID: 0x00025500403bd602

The current configuration of the GPFS cluster is shown in Example 3-43.

Example 3-43 Current cluster configuration

```
[root@spain work]# mmlsconfig
Configuration data for cluster GPFS-InfiniBand.spain-gpfs:
-----
clusterName GPFS-InfiniBand.spain-gpfs
clusterId 723685802921743777
autoload no
minReleaseLevel 3.4.0.0
dmapiFileHandleSize 32
adminMode central
```

3. Add the InfiniBand settings as shown in Example 3-44. The verbsPorts parameter requires the adapter name and the active port that is used for GPFS communication.

Example 3-44 Configuring the InfiniBand settings

```
[root@spain work]# mmchconfig verbsPorts="ehca0/1"
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

```
[root@spain work]# mmchconfig verbsRdma=enable
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

```
[root@spain work]# mmlsconfig
Configuration data for cluster GPFS-InfiniBand.spain-gpfs:
-----
clusterName GPFS-InfiniBand.spain-gpfs
clusterId 723685802921743777
autoload no
minReleaseLevel 3.4.0.0
dmapiFileHandleSize 32
verbsPorts ehca0/1
verbsRdma enable
adminMode central
```

File systems in cluster GPFS-InfiniBand.spain-gpfs:

/dev/gpfs-ib

- Restart the GPFS cluster by running the following commands:

```
mmumount all -a
mmshutdown -a
mmstartup -a
mmm mount gpfs-ib -a
```

- Check the `/var/mmfs/gem/mmfslog` file for InfiniBand connections as show in Example 3-45.

Example 3-45 InfiniBand connections

```
Fri Jun 25 13:40:43.631 2010: mmfsd initializing. {Version: 3.4.0.0   Built: Jun 10 2010
14:50:42} ...
Fri Jun 25 13:40:44.538 2010: VERBS RDMA starting.
Fri Jun 25 13:40:44.539 2010: VERBS RDMA library libibverbs.so (version >= 1.1) loaded and
initialized.
Fri Jun 25 13:40:44.540 2010: VERBS RDMA device ehca0 port 1 opened.
Fri Jun 25 13:40:44.541 2010: VERBS RDMA started.
Fri Jun 25 13:40:45.050 2010: Connecting to 10.11.12.103 spain-gpfs <c0p0>
Fri Jun 25 13:40:45.051 2010: Connected to 10.11.12.103 spain-gpfs <c0p0>
Fri Jun 25 13:40:45.056 2010: mmfsd ready
Fri Jun 25 13:40:45 EDT 2010: mmcommon mmfsup invoked. Parameters: 10.11.12.225 10.11.12.103 all
Fri Jun 25 13:40:45 EDT 2010: mounting /dev/gpfs-ib
Fri Jun 25 13:40:45.562 2010: Command: mount gpfs-ib
Fri Jun 25 13:40:45.655 2010: Connecting to 10.11.12.113 greece-gpfs <c0n1>
Fri Jun 25 13:40:45.656 2010: Connected to 10.11.12.113 greece-gpfs <c0n1>
Fri Jun 25 13:40:45.679 2010: VERBS RDMA connecting to 10.11.12.113 greece-gpfs on ehca0 port 1
Fri Jun 25 13:40:45.680 2010: VERBS RDMA connected to 10.11.12.113 greece-gpfs on ehca0 port 1
Fri Jun 25 13:40:45.701 2010: VERBS RDMA connecting to 10.11.12.103 spain-gpfs on ehca0 port 1
Fri Jun 25 13:40:45.703 2010: VERBS RDMA connected to 10.11.12.103 spain-gpfs on ehca0 port 1
Fri Jun 25 13:40:45.729 2010: Command: err 0: mount gpfs-ib
Fri Jun 25 13:40:45 EDT 2010: finished mounting /dev/gpfs-ib
```

- Rerun the test from the nigeria node, which is an NSD client, as shown in Example 3-46.

Example 3-46 InfiniBand gpfsperf test

```
[root@nigeria perf]# ./gpfsperf create seq /gpfs-ib/gogo -r 256k -n 1024000000
-th 4
./gpfsperf create seq /gpfs-ib/gogo
  recSize 256K nBytes 999936K fileSize 999936K
  nProcesses 1 nThreadsPerProcess 4
  file cache flushed before test
  not using data shipping
  not using direct I/O
  offsets accessed will cycle through the same file segment
  not using shared memory buffer
  not releasing byte-range token after open
  no fsync at end of test
  Data rate was 84967.91 Kbytes/sec, thread utilization 0.971
[root@nigeria perf]# ./gpfsperf create seq /gpfs-ib/gogo -r 256k -n 1024000000
-th 4
./gpfsperf create seq /gpfs-ib/gogo
  recSize 256K nBytes 999936K fileSize 999936K
  nProcesses 1 nThreadsPerProcess 4
  file cache flushed before test
  not using data shipping
```

not using direct I/O
offsets accessed will cycle through the same file segment
not using shared memory buffer
not releasing byte-range token after open
no fsync at end of test
Data rate was **87928.01** Kbytes/sec, thread utilization 0.965

Conclusions

Using GPFS with InfiniBand infrastructure has the following characteristics:

- ▶ Data and metadata is sent over infiniband using RDMA
- ▶ Daemon-to-daemon communication and `ssh` commands are sent over TCP/IP.
- ▶ Using Internet Protocol over InfiniBand (IPoIB) as the TCP/IP communication layer is possible. In this way, all the communication goes over InfiniBand.

3.5 Cross-platform cluster: Windows servers, Linux/AIX clients

This scenario describes how to configure a *cross-platform* GPFS cluster consisting of two Windows Server 2008 R2 nodes, a Linux on System p client node, and an AIX client node.

3.5.1 Requirements: Hardware, software, network, storage

The hardware requirements for these scenarios is as follows:

- ▶ x86 servers
- ▶ Linux on IBM Power Systems nodes on LPARs
- ▶ AIX nodes on LPARs

The software requirements for these scenarios is as follows:

- ▶ Red Hat Enterprise Linux 5.5
- ▶ AIX 6.1
- ▶ Windows 2008 R2
- ▶ GPFS 3.4

Network uses GigE LAN.

The storage requirements are as follows:

- ▶ SAN Storage (for Data)
- ▶ SAN disk (for the *tiebreaker* quorum)

3.5.2 GPFS cluster configuration

In the scenario, the Windows servers use the GigE LAN and are directly attached to SAN storage. Linux and AIX client nodes access the file systems on the Windows servers through NSD.

To maintain node quorum on the two Windows servers cluster, a separate NSD, which is accessible to both Windows nodes (designated as a *tiebreaker disk*) are defined and configured on the cluster.

After the cluster is fully configured, a simple (write/read) test application is run on the GPFS file system from both Windows nodes concurrently to show file system availability and data integrity.

In the test, concurrently simulated errors are injected on either of the Windows servers to verify the file system remains available from the surviving Windows node.

3.5.3 GPFS cluster diagram

Figure 3-5 shows the Windows server configuration.

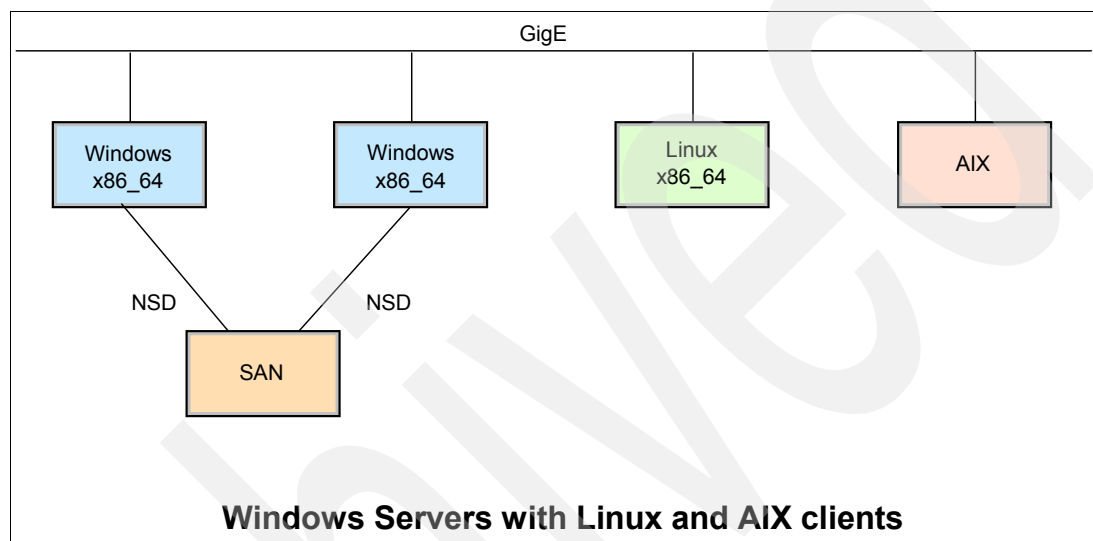


Figure 3-5 Windows server configuration

3.5.4 Installing Windows Server 2008 R2

Table 3-2 shows the Windows cluster nodes, names and IP addresses.

Table 3-2 Windows server configuration showing IP addresses

| Node type | Role | Name | Mgmt IP | GPFS IP |
|-----------|-------------|----------|-----------------|--------------|
| AD Server | DNS server | korea | 192.168.101.157 | 10.11.12.157 |
| Windows | GPFS server | paraguay | 192.168.101.153 | 10.11.12.153 |
| Windows | GPFS | honduras | 192.168.101.154 | 10.11.12.154 |

The basic installation steps for a Windows Server 2008 R2 on Node1 are as follows:

Set up the Active Directory and use itso.com as the domain name

The following procedure disables password complexity to an account when the Windows domain controller is running on Windows Server 2008 R2:

1. Open the Group Policy Management Editor (available under Administrative Tools).
2. In the console tree, select **Forest name** → **Domains** → **Domain name** → **Group Policy Objects**.
3. Right-click **Default Domain Policy** and select **Edit**.

4. In the Group Policy Management Editor (Figure 3-6), select **Computer Configuration** → **Policies** → **Windows Settings** → **Security Settings** → **Account Policies**. Then, select **Password Policy** to display the list of password policies.
5. Double-click **Password must meet complexity requirements**.
6. Select the **Disable** check box, and then click **OK**.

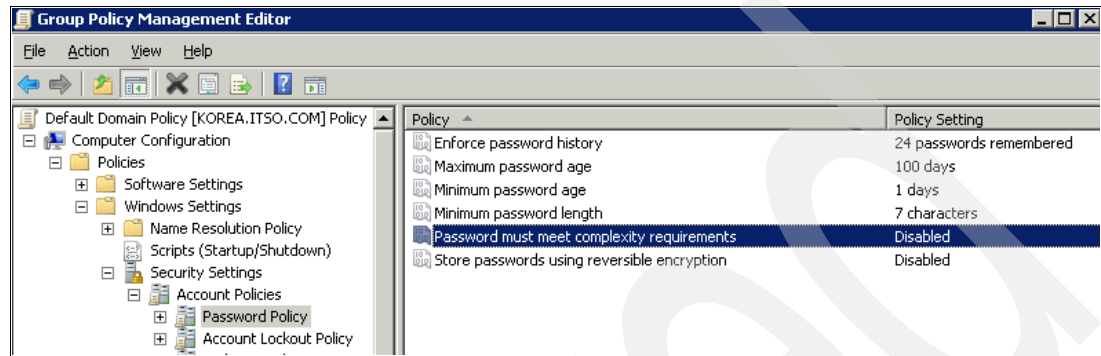


Figure 3-6 Group Policy Management Editor

Change the user account control

The following procedure changes User account control to an account that is used when the Windows domain controller is running on Windows Server 2008 R2:

1. Open the Group Policy Management Editor (available under Administrative Tools).
2. In the console tree, select **Forest name** → **Domains** → **Domain name** → **Group Policy Objects**.
3. Right-click **Default Domain Policy** and select **Edit**.
4. In the console tree, select **Computer Configuration** → **Policies** → **Windows Settings** → **Security Settings** → **Local Policies** → **Security Options**.
5. Double-click **User Account Control: Admin Approval Mode for the Built-in Administrator account**.
6. Select the Define this policy setting check box.
7. Select **Disable**, then click **OK**.
8. Double-click **User Account Control: Behavior of the elevation prompt for administrators in Admin Approval Mode**. See Figure 3-7 on page 100.
9. Select the **Define this policy setting** check box.
10. Select **Elevate without prompting**, then click **OK**.
11. Double-click **User Account Control: Detect application installations and prompt for elevation**.
12. Select the Define this policy setting check box.
13. Select **Disable**, then click **OK**.

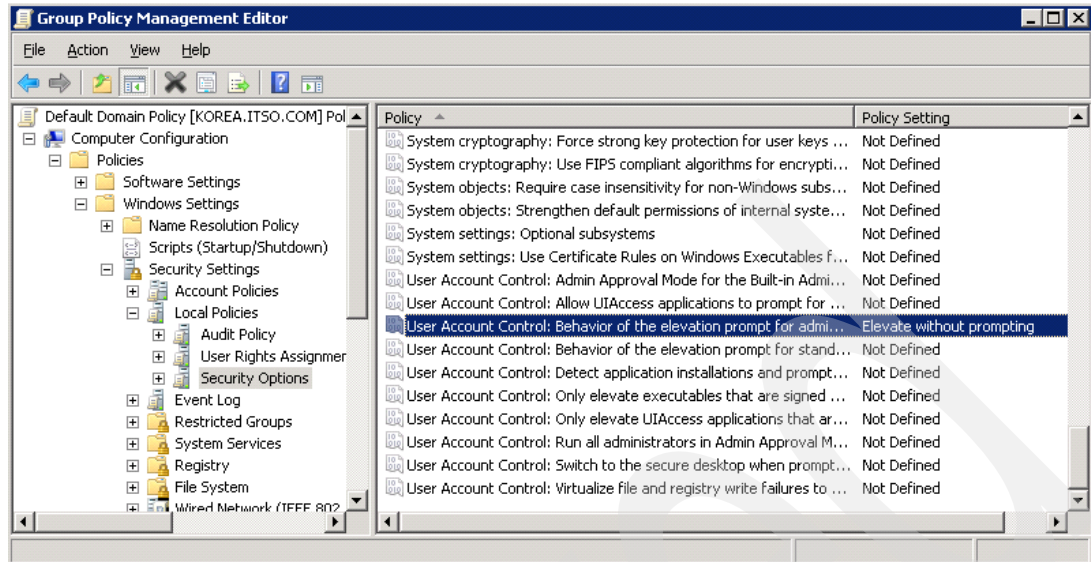


Figure 3-7 Use Account Control panel

Change the default firewall rule settings

The following procedure changes the default firewall rule setting to allow the Windows nodes to communicate in the GPFS cluster in a Windows domain controller running on Windows Server 2008 R2:

1. Open the Group Policy Management Editor (available under Administrative Tools).
2. In the console tree, select **Forest name** → **Domains** → **Domain name** → **Group Policy Objects**.
3. Right-click **Default Domain Policy** and select **Edit**.
4. In the console tree, select **Computer Configuration** → **Policies** → **Windows Settings** → **Security Settings** → **Windows Firewall and Advanced Security** → **Windows Firewall with Advanced Security - LDAP** → **Inbound Rules**, as shown in Figure 3-8 on page 101.
5. Right-click and then select to add the new rule. Add each of the following new rules (see Figure 3-8 on page 101):
 - GPFS TCP/UDP 1191 Port
 - ICMPv4
 - SSH 22

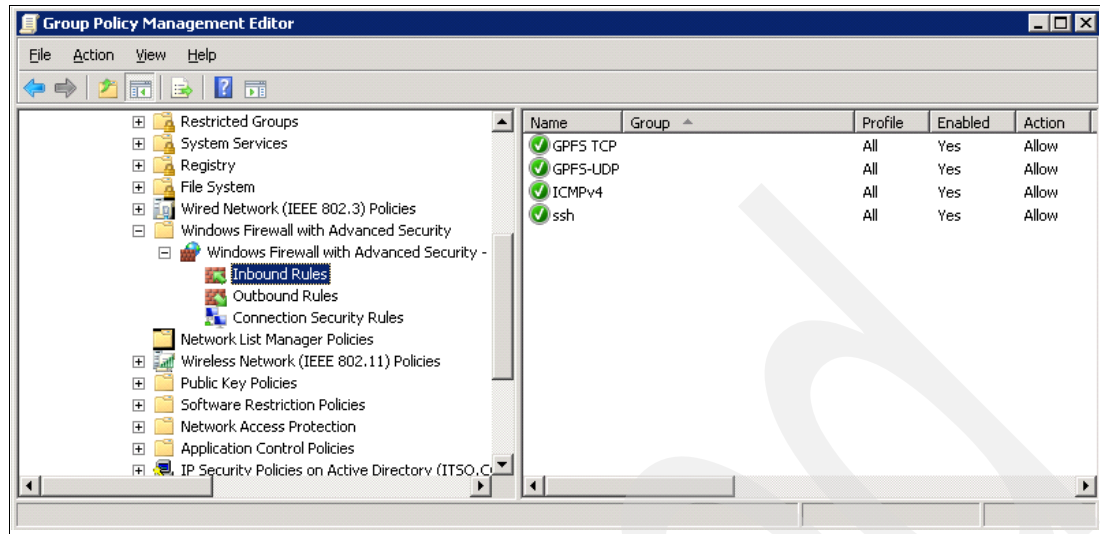


Figure 3-8 Inbound rules selection

Add the root user on the Active Directory Server

Add the root user on the Active Directory server, and add the Domain Admin Group (see Figure 3-9):

- ▶ Although the Profile directory is blank, by default, set it as follows:

C:\Users\root

- ▶ Set the Home directory as follows:

C:\root

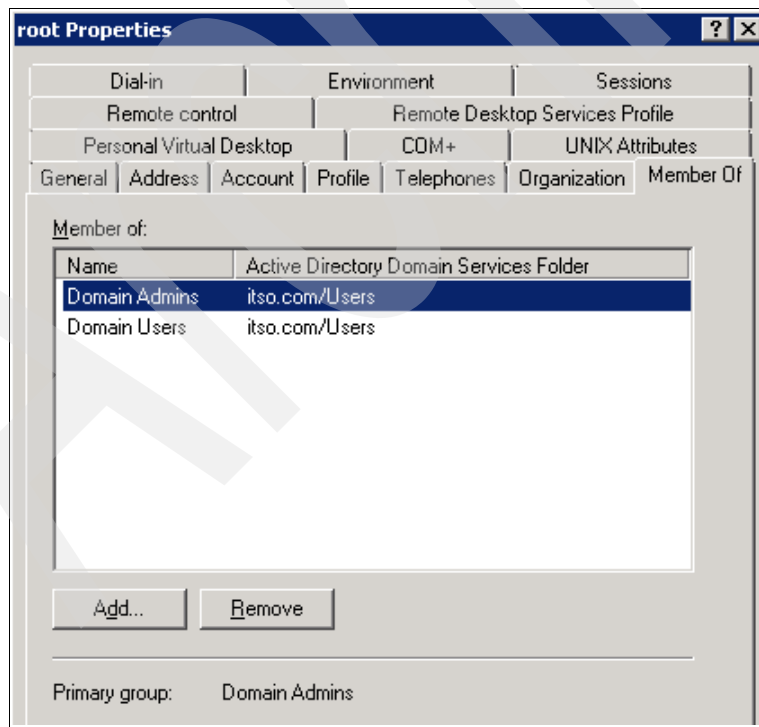


Figure 3-9 Add root server on Active Directory server

Assign the logon as a service right to an account

The following procedure assigns the logon as a service right to an account when the Windows domain controller is running on Windows Server 2008:

1. Open Group Policy Management Editor (available under Administrative Tools).
2. In the console tree, select **Forest name** → **Domains** → **Domain name** → **Group Policy Objects**.
3. Right-click **Default Domain Policy** and select **Edit**.
4. In the console tree, select **Computer Configuration/Policies** → **Windows Settings** → **Security Settings** → **Local Policies** → **User Rights Assignment**.
5. Double-click the **Log on as a service** policy (Figure 3-10).
6. Select the **Define this policy setting** check box.
7. Use **Add User or Group** to include the DomainName\root account in the policy, and then click **OK**.

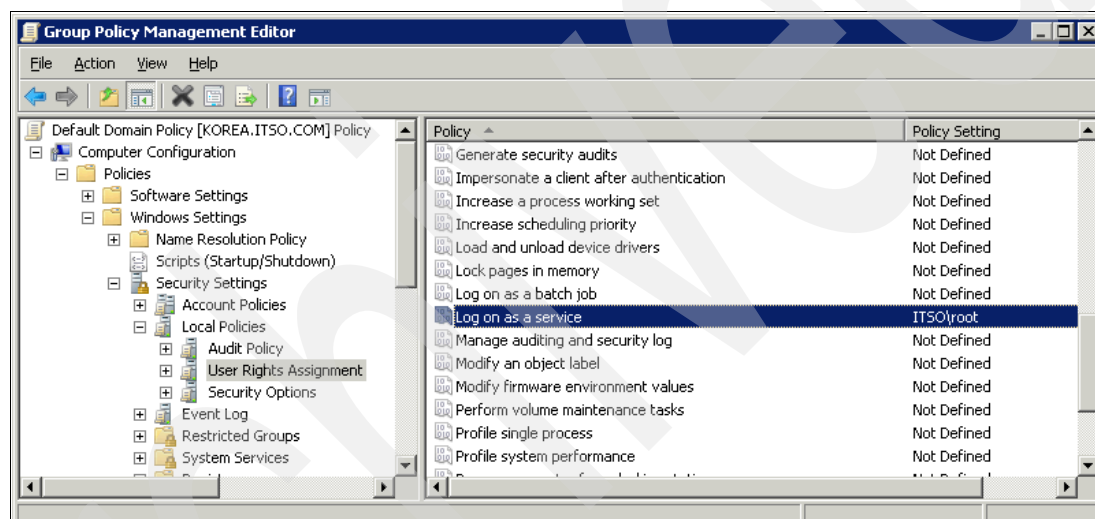


Figure 3-10 Assign log on as a service

Install Windows Server 2008 R2 on Node2

Install Windows Server 2008 R2 on Node2:

1. After the OS is installed, configure a host name and IP, and add a node to Windows domain. This step requires rebooting.
2. Install all the latest critical security patches by using Windows Update.
Select **Start** → **All Programs** → **Windows Update**.
3. Turn off the local firewall as shown in Figure 3-11 on page 103.

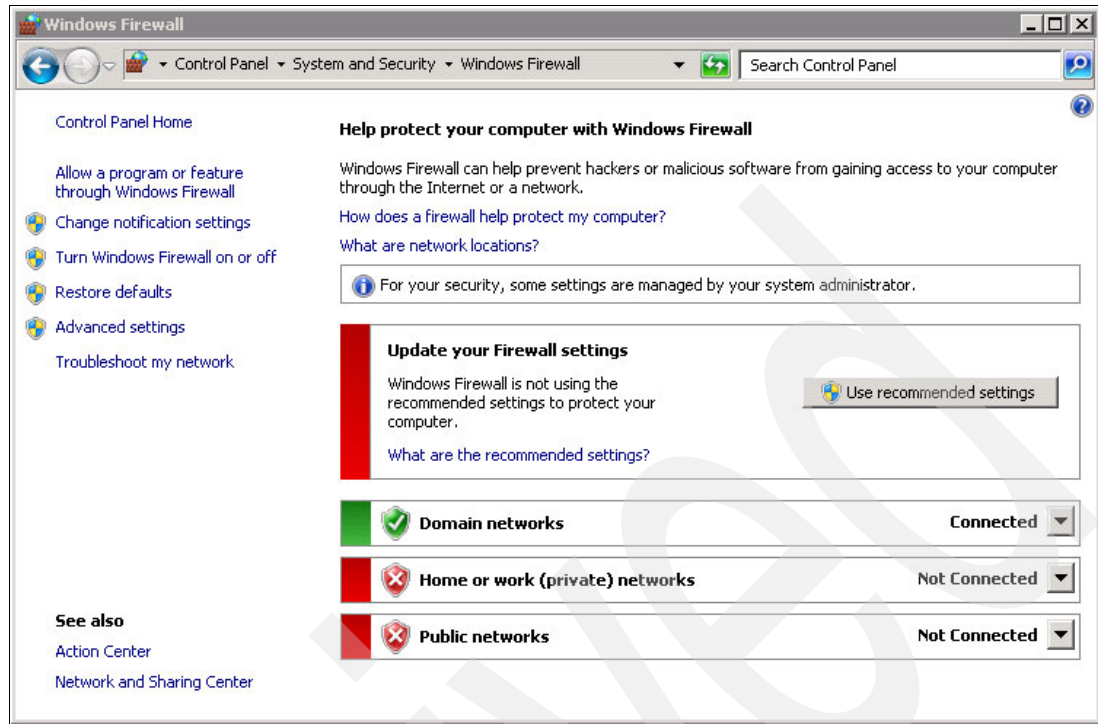


Figure 3-11 Turning off the local firewall

4. In the Control Panel, select **User Accounts**. In the User Accounts window, give other users access to this computer, as shown in Figure 3-12.

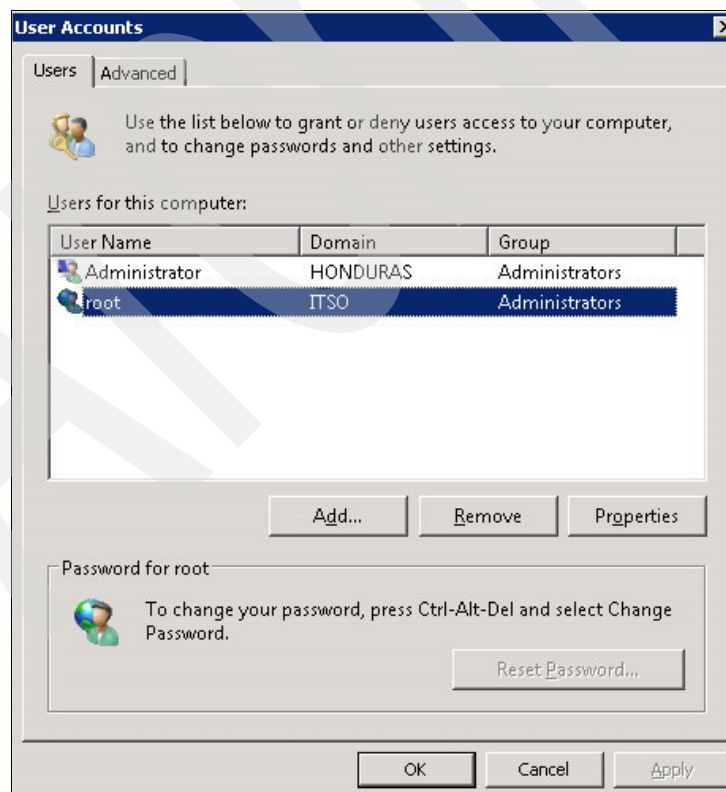


Figure 3-12 Give other users access to this computer

5. Add the ITS0\root account to the local administrators group (Figure 3-13):
 - a. Select **Start** → **Administrative Tools** → **Computer Management** → **Local Users and Groups** → **Groups**.
 - b. Double-click **Administrators** and click **Add**.
 - c. Type ITS0\root and click **OK**.

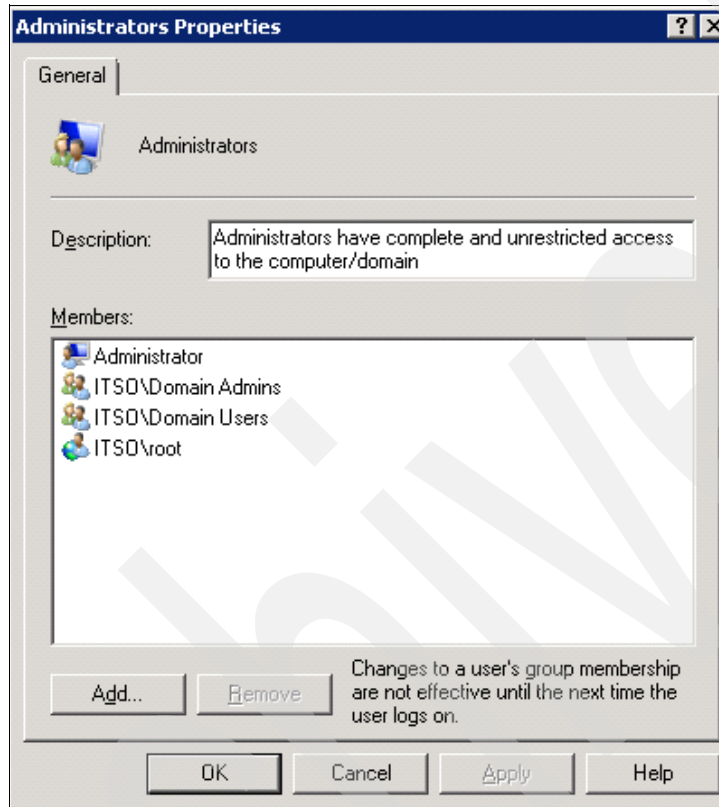


Figure 3-13 Add ITS0\root account to local administrators group

6. Log in as local administrator.
7. Verify the domain policy. See Figure 3-14 on page 105. This policy is set up by the Active Directory server.

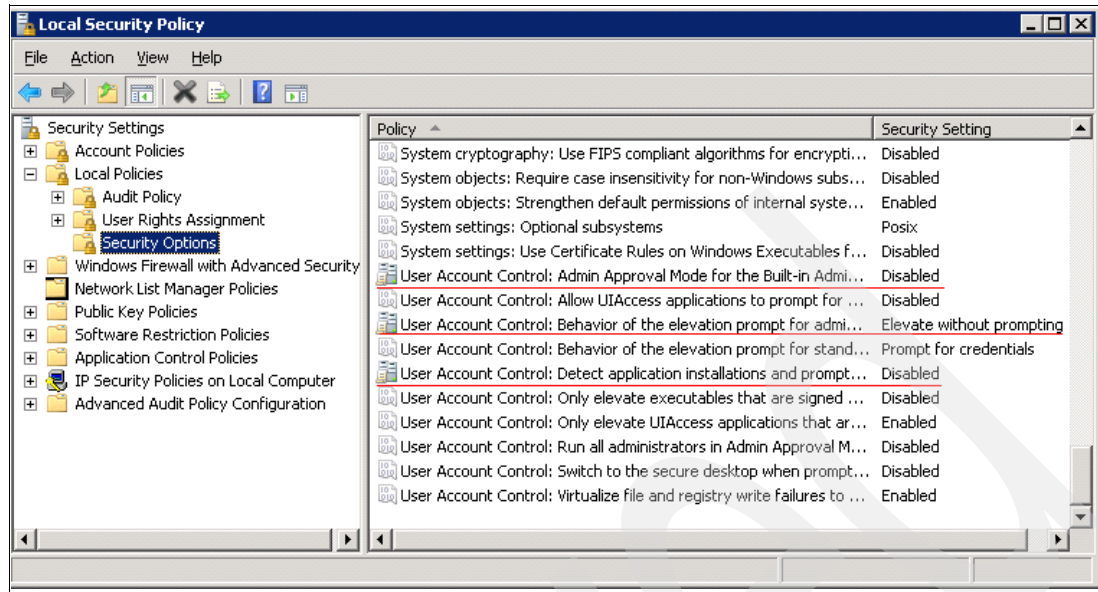


Figure 3-14 Security Options

8. Run the RDAC installation program. This step requires rebooting. This rebooting is to complete the installation of the IBM Storage Multipath driver.
9. Install the feature Subsystem for the UNIX-based applications (SUA) package. Download and install the following SUA fixes, and then reboot:
 - KB977615 (“A hotfix is available for Windows Services for UNIX and for SUA to incorporate a cumulative DST update for December 2009”):
<http://support.microsoft.com/kb/977615>
 - KB981194 (“The input or output stream is redirected incorrectly in a Korn shell”):
<http://support.microsoft.com/kb/981194>
10. Download and run the following executables files from the web addresses listed:
 - v010.0.6030.0 Utilities and SDK for SUA_AMD64.exe
<http://www.microsoft.com/downloads/en/details.aspx?FamilyID=93ff2201-325e-487f-a398-efde5758c47f>
 - v010.0.7063.0 Utilities and SDK for SUA_AMD64.exe
<http://www.microsoft.com/downloads/en/details.aspx?FamilyID=dc03485b-629b-49a6-b5ef-18617d1a9804>

Install the SDK, running as root user. Select all components, and all options. See Figure 3-15. Installation requires rebooting.

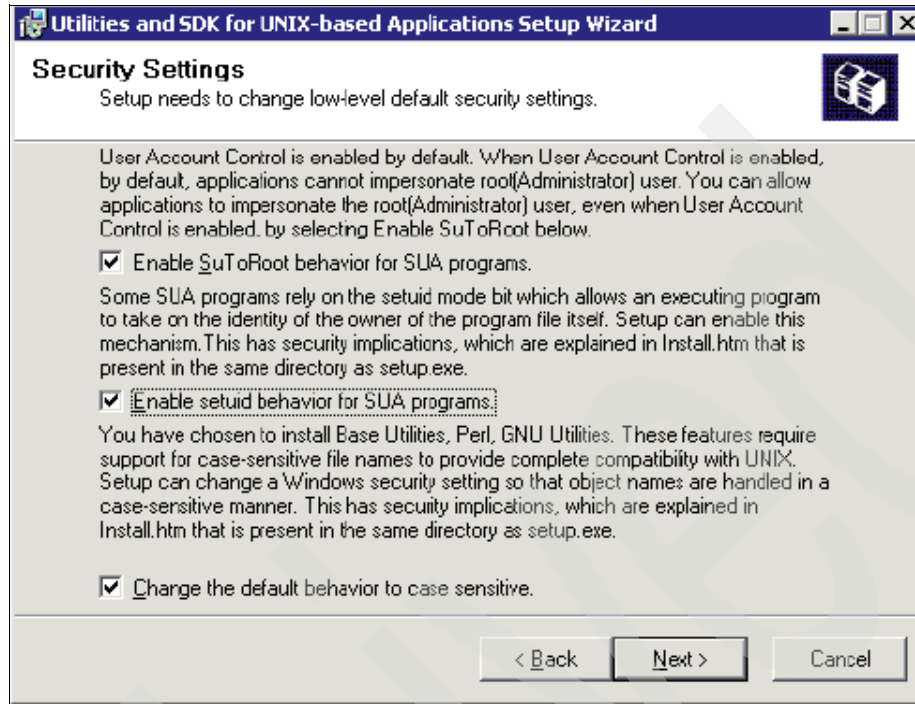
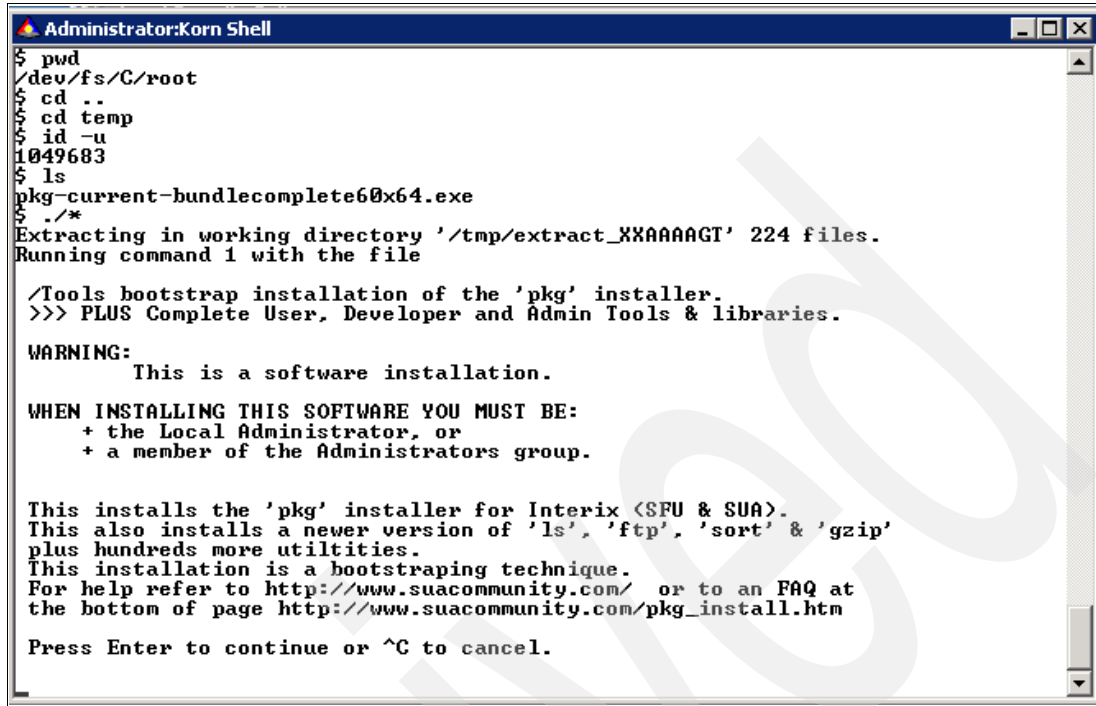


Figure 3-15 Install SDK and utilities

11. At this time, ITS0\root is running as local system administrator group. Install SUA for Vista, Server 2008, Windows 7, and Server 2008 R2 for 64-bit systems, as follows:
 - a. Go to either of the following locations:
 - http://www.suacommunity.com/pkg_install.htm
 - <http://www.interopsystems.com/community/>
 - b. Download and run the following file:
<ftp://ftp.interopsystems.com/pkgs/bundles/pkg-current-bundlecomplete60x64.exe>

Figure 3-16, shows installation and warning message.



```
Administrator:Korn Shell
$ pwd
/dev/fs/C/root
$ cd ..
$ cd temp
$ id -u
1049683
$ ls
pkg-current-bundlecomplete60x64.exe
$ ./*
Extracting in working directory '/tmp/extract_XXXXAAAGT' 224 files.
Running command 1 with the file

/Tools bootstrap installation of the 'pkg' installer.
>>> PLUS Complete User, Developer and Admin Tools & libraries.

WARNING:
  This is a software installation.

WHEN INSTALLING THIS SOFTWARE YOU MUST BE:
  + the Local Administrator, or
  + a member of the Administrators group.

This installs the 'pkg' installer for Interix (SFU & SUA).
This also installs a newer version of 'ls', 'ftp', 'sort' & 'gzip'
plus hundreds more utilities.
This installation is a bootstrapping technique.
For help refer to http://www.suacommunity.com/ or to an FAQ at
the bottom of page http://www.suacommunity.com/pkg\_install.htm

Press Enter to continue or ^C to cancel.
```

Figure 3-16 Installation (showing warning message)

12. During the Interix environment setup, if you receive a warning message like the one in Figure 3-16, check the /var/adm/log/init.log file (shown in Figure 3-17) by issuing the following command:

```
Cat /var/adm/log/init.log
```

```
Starting portmap service
The service name is invalid.

More help is available by typing NET HELPMSG 2185.

Note: it may be alright if net start command failed here
because portmap service may already be running.

Exitstatus = 0 <Success>
```

Figure 3-17 Content of var/adm/log/init.log

13. Remove the daemon start link and SystemWarning file, as shown in Example 3-47.

Example 3-47 Remote Daemon start link

```
$ pwd
/etc
$ find ./ | grep rpc
./init.d/rpc
./rc2.d/S34rpc
./rpc
$ rm ./rc2.d/S34rpc
$ rm ./rc3.d/S40updater
$ cd ..
$ rm SystemWarning
$ pwd
/var/adm
```

14. Set up SSH key generation and sharing.

3.5.5 Install the IBM GPFS Windows version

Install the IBM GPFS Windows version as follows:

1. Log in as itso\root.
2. If you have non-English version of Windows, use the Control Panel to set English for your locale information.
3. Run the Windows installation package by using the following command:

```
C:\> msixexec.exe /package gpfs-3.3.0.0-WindowsServer2008-x64.msi /passive /log
gpfs-install.log AgreeToLicense=yes
```

The installation log shows failure (Example 3-48).

Example 3-48 Creating the software link has failed

```
CAQuietExec: Creating soft link: /usr/local/bin/perl <-- /bin/perl
CAQuietExec: Fri Jun 25 14:21:51 PDT 2010: error: Attempting to link to
non-existing file /usr/local/bin/perl
CAQuietExec: Error 0x80070001: Command line returned an error.
CAQuietExec: Error 0x80070001: CAQuietExec Failed
CustomAction mmwinInstall returned actual error code 1603 (note this may not be
100% accurate if translation happened inside sandbox)
Action ended 14:21:51: InstallFinalize. Return value 3.
DeleteMachineCertificate: Deleting certificate with friendly name: IBM
Action ended 14:21:53: INSTALL. Return value 3.
```

4. Create the link manually (Example 3-49).

Example 3-49 Create the software link manually

```
cd /usr/local/bin
ln -s /bin/perl perl
ls -l perl
```

5. After completion, install GPFS and verify the installation. See Figure 3-18 on page 109.

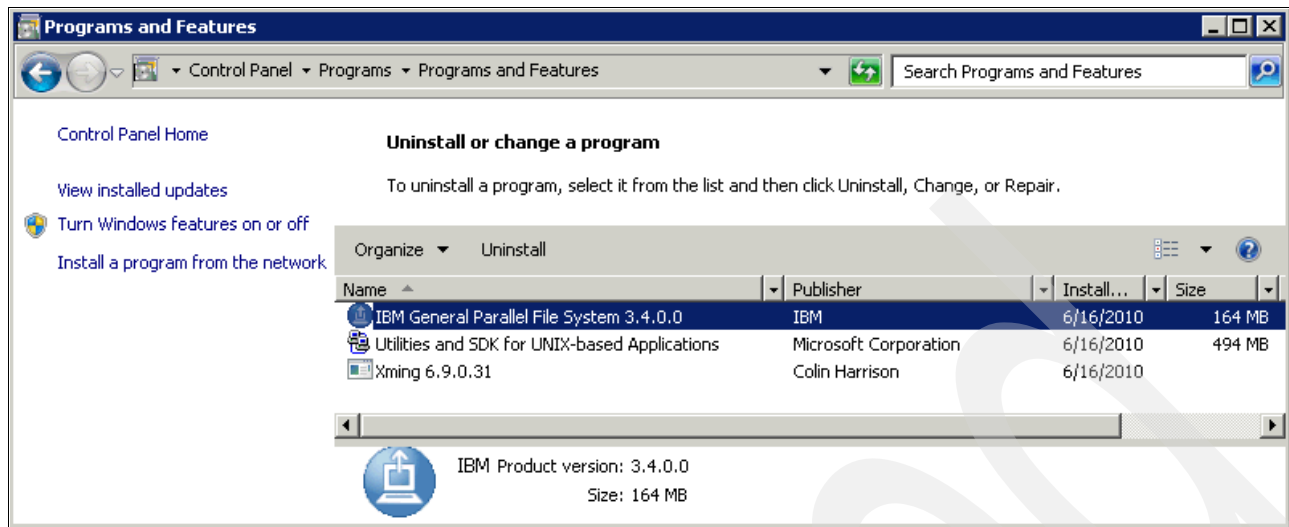


Figure 3-18 Verify GPFS was installed on the node

3.5.6 GPFS Windows cluster setup

This section provides the steps to implement GPFS for Windows cluster configuration.

1. Set up and check IP addresses and host name for Windows servers (Example 3-50).

Example 3-50 Verify IP addresses and host name in /etc/hosts are correct

```
% grep para /etc/hosts
192.168.101.153 paraguay          # Windows node1
10.11.12.153   paraguay-gpfs     # Windows node1
100.110.120.153 paraguay-ib      # Windows node1
```

2. Enable Windows server nodes (Example 3-51).

Example 3-51 Enable Windows server nodes

```
[root@paraguay] mmwinservctl set --account itso/root --password itsoadmin
--remote-shell no
```

```
[root@honduras] ./mmwinservctl enable -N honduras-gpfs,paraguay-gpfs
```

| Node name | Service state | Remote shell | Account name |
|---------------|---------------|--------------|--------------|
| honduras-gpfs | RUNNING | yes | LocalSystem |
| paraguay-gpfs | RUNNING | yes | LocalSystem |

```
[root@honduras]
```

3. Create GPFS cluster on Windows servers (Example 3-52).

Example 3-52 Create Windows cluster using two servers

```
[root@honduras] ./mmcrcluster -N honduras-gpfs:quorum,paraguay-gpfs:quorum -p
honduras-gpfs -s
paraguay-gpfs -r /usr/lpp/mmfs/bin/mmwinrsh -R /usr/lpp/mmfs/bin/mmwinrcp
Mon Jun 28 23:09:08 PDT 2010: 6027-1664 mmcrcluster: Processing node
honduras-gp
fs.itso.com
```

```

Mon Jun 28 23:09:10 PDT 2010: 6027-1664 mmcrcluster: Processing node
paraguay-gp
fs.itso.com

mmcrcluster: Command successfully completed
mmcrcluster: 6027-1254 Warning: Not all nodes have proper GPFS license
designati
ons.
    Use the mmchlicense command to designate licenses as needed.
mmcrcluster: Propagating the cluster configuration data to all affected nodes.
[root@honduras]

```

4. Set up the license for the Windows servers (Example 3-54).

Example 3-53 Set up license

```

[root@honduras] ./mmchlicense server --accept -N honduras-gpfs,paraguay-gpfs

The following nodes will be designated as possessing GPFS server licenses:
    honduras-gpfs.itso.com
    paraguay-gpfs.itso.com
mmchlicense: Command successfully completed
mmchlicense: Propagating the cluster configuration data to all affected nodes.
[root@honduras]

```

5. Check license on cluster nodes (Example 3-54).

Example 3-54 Use mmlslicense to check current licenses on the nodes

```

[root@paraguay] mmlslicense -L
Node name           Required license   Designated license
-----
honduras-gpfs.itso.com    server            server
paraguay-gpfs.itso.com   server            server

Summary information
-----
Number of nodes defined in the cluster:                2
Number of nodes with server license designation:        2
Number of nodes with client license designation:        0
Number of nodes still requiring server license designation: 0
Number of nodes still requiring client license designation: 0
[root@paraguay]

```

6. Start GPFS daemon on all nodes of the cluster and check daemon status (Example 3-55).

Example 3-55 Start GPFS daemon and check daemon status with the mmgetstate command

```

[root@honduras] ./mmstartup -a
Mon Jun 28 23:15:24 PDT 2010: 6027-1642 mmstartup: Starting GPFS ...
[root@honduras] ./mmgetstate -aLs

Node number  Node name           Quorum  Nodes up  Total nodes  GPFS state
Remarks
-----
1            honduras-gpfs         2        2         2           active    quorum

```



```
node
      2      paraguay-gpfs      2      2      2      active      quorum
node
```

Summary information

```
-----
Number of nodes defined in the cluster:      2
Number of local nodes active in the cluster: 2
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 2
Number of quorum nodes active in the cluster: 2
Quorum = 2, Quorum achieved
```

```
[root@honduras]
```

7. Create NSDs for the Windows cluster (Example 3-56).

Example 3-56 check mmcrnsd input file - desc_file

```
[root@paraguay] cat mmcrnsd_input02
2:honduras-gpfs,paraguay-gpfs::::
3:honduras-gpfs,paraguay-gpfs::::
4:honduras-gpfs,paraguay-gpfs::::
5:honduras-gpfs,paraguay-gpfs::::
```

8. Create NSDs by using the **mmcrnsd** command and the **mmcrnsd_input** file (Example 3-57).

Example 3-57 Creating the NSDs

```
[root@paraguay] mmcrnsd -F mmcrnsd_input
mmcrnsd: Processing disk 2
mmcrnsd: Processing disk 3
mmcrnsd: Processing disk 4
mmcrnsd: Processing disk 5
mmcrnsd: Propagating the cluster configuration data to all affected nodes.
[root@paraguay]
```

9. Check NSD input file after the **mmcrnsd** command completes (Example 3-58).

Example 3-58 Verify updates to NSD input file after mmcrnsd completes

```
[root@paraguay] cat mmcrnsd_input_05
# 2:honduras-gpfs::::
gpfs14nsd:honduras-gpfs.itso.com,paraguay-gpfs.itso.com::dataAndMetadata:3001::
# 3:honduras-gpfs::::
gpfs15nsd:paraguay-gpfs.itso.com,honduras-gpfs.itso.com::dataAndMetadata:3002::
# 4:honduras-gpfs::::
gpfs16nsd:honduras-gpfs.itso.com,paraguay-gpfs.itso.com::dataAndMetadata:3001::
# 5:honduras-gpfs::::
gpfs17nsd:paraguay-gpfs.itso.com,honduras-gpfs.itso.com::dataAndMetadata:3002::
[root@paraguay]
```

10. List the currently defined NSD on the cluster (Example 3-59).

Example 3-59 Show define NSD on the cluster

```
[root@paraguay] mmlnsd
```

| File system | Disk name | NSD servers |
|-------------|-----------|---|
| (free disk) | gpfs14nsd | honduras-gpfs.itso.com,paraguay-gpfs.itso.com |
| (free disk) | gpfs15nsd | paraguay-gpfs.itso.com,honduras-gpfs.itso.com |
| (free disk) | gpfs16nsd | honduras-gpfs.itso.com,paraguay-gpfs.itso.com |
| (free disk) | gpfs17nsd | paraguay-gpfs.itso.com,honduras-gpfs.itso.com |

11. Create file system /gpfs1, which is mounted on Windows nodes at G (Example 3-60), and mounted on Linux nodes on /gpfs1.

Example 3-60 The mmcrfs creates /gpfs1 that is mounted at G: on the Windows nodes

```
[root@paraguay] mmcrfs gpfs1 -F mmcrnsd_input -T /gpfs1 -t G
```

GPFS: 6027-531 The following disks of gpfs1 will be formatted on node honduras:

```
gpfs1nsd: size 26214272 KB
gpfs2nsd: size 26214272 KB
gpfs3nsd: size 26214272 KB
gpfs4nsd: size 26214272 KB
```

GPFS: 6027-540 Formatting file system ...

GPFS: 6027-535 Disks up to size 231 GB can be added to storage pool 'system'.

Creating Inode File

Creating Allocation Maps

Clearing Inode Allocation Map

Clearing Block Allocation Map

Formatting Allocation Map for storage pool 'system'

GPFS: 6027-572 Completed creation of file system /dev/gpfs1.

mmcrfs: Propagating the cluster configuration data to all affected nodes.

```
[root@paraguay]
```

12. Mount /gpfs1 on all Windows nodes (Example 3-61).

Example 3-61 Mount /gpfs1 on all node in the Windows cluster and verify it is mounted

```
[root@paraguay] mmmount gpfs1 -a
Wed Jun 30 01:05:29 PDT 2010: 6027-1623 mmmount: Mounting file systems ...
[root@paraguay] mmlsmount all
File system gpfs1 is mounted on 2 nodes.
[root@paraguay] mmlsmount all -L
```

File system gpfs1 is mounted on 2 nodes:

```
10.11.12.154 honduras-gpfs
10.11.12.153 paraguay-gpfs
```

```
[root@paraguay]
```

```
[root@paraguay] df
```

| Filesystem | 512-blocks | Used | Available | Capacity | Type | Mounted on |
|---------------------------------|------------|----------|-----------|----------|------|------------|
| //HarddiskVolume2 | 585727992 | 36753560 | 548974432 | 6% | ntfs | /dev/fs/C |
| //GpfsFileSystems/gpfs1\$gen(1) | 209713152 | 51537920 | 158175232 | 25% | ??? | /dev/fs/G |

```
[root@paraguay]
```

13. Check the cluster configuration (Example 3-62).

Example 3-62 Use mmlscluster to check cluster configuration

```
[root@paraguay] mmlscluster

GPFS cluster information
=====
GPFS cluster name:      honduras-gpfs.itso.com
GPFS cluster id:        723686021965476454
GPFS UID domain:        honduras-gpfs.itso.com
Remote shell command:   /usr/bin/ssh
Remote file copy command: /usr/bin/scp

GPFS cluster configuration servers:
-----
Primary server:  honduras-gpfs.itso.com
Secondary server: paraguay-gpfs.itso.com

Node Daemon node name      IP address      Admin node name
Designation
-----
-
-----
1  honduras-gpfs.itso.com  10.11.12.154    honduras-gpfs.itso.com
quorum
2  paraguay-gpfs.itso.com  10.11.12.153    paraguay-gpfs.itso.com
quorum

[root@paraguay]
```

14. Verify disk on Windows servers are twin-tailed (Example 3-63).

Example 3-63 List disk on Windows servers

```
[root@paraguay] mmwindisk list

Disk Avail Type      Status      Size GPFS Partition ID
-----
0          BASIC  ONLINE      279 GiB
1 YES      BASIC  ONLINE      279 GiB
2          GPFS  ONLINE      25 GiB CF45F4D4-1F6E-425F-B4F2-EF246F01F185
3          GPFS  ONLINE      25 GiB 9693BDE0-D0FF-4562-8446-C90C6B8EEF19
4          GPFS  ONLINE      25 GiB B8015DF3-8F47-4FCD-B615-97AE8D322B90
5 YES      UNALLOC OFFLINE      25 GiB

[root@paraguay]
```

15. Set up tiebreaker disk quorum. Because we currently have a two-node Windows cluster we cannot rely on node quorum so we have to use a tiebreaker disk for quorum. To configure the tiebreaker disk, the GPFS daemon must be shut down on all nodes of the cluster. See Example 3-64 on page 114.

Example 3-64 Shut down GPFS on all node of the cluster

```
[root@paraguay] mmshutdown -a
Wed Jun 30 14:14:09 PDT 2010: 6027-1341 mmshutdown: Starting force unmount of GP
FS file systems
Wed Jun 30 14:14:14 PDT 2010: 6027-1344 mmshutdown: Shutting down GPFS daemons
honduras-gpfs.itso.com: Shutting down!
honduras-gpfs.itso.com: 'shutdown' command about to kill process 1789
paraguay-gpfs.itso.com: Shutting down!
paraguay-gpfs.itso.com: 'shutdown' command about to kill process 695
Wed Jun 30 14:14:34 PDT 2010: 6027-1345 mmshutdown: Finished
$
```

16. Configure tiebreaker quorum using one NSD (Example 3-65).

Example 3-65 Configure GPFS to use tiebreaker disk quorum using 1 NSD

```
$ mmchconfig tiebreakerdisks="gpfs14nsd"
Verifying GPFS is stopped on all nodes ...
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all affected nodes.
[root@paraguay]
```

17. Verify tiebreaker quorum is set up correctly (Example 3-66).

Example 3-66 Use the mmlsconfig and look the tiebreaker disks parameters

```
[root@paraguay] mmlsconfig
Configuration data for cluster honduras-gpfs.itso.com:
-----
clusterName honduras-gpfs.itso.com
clusterId 723686021965502247
autoload no
minReleaseLevel 3.4.0.0
dmapifileHandleSize 32
tiebreakerDisks gpfs14nsd
adminMode central

File systems in cluster honduras-gpfs.itso.com:
-----
/dev/gpfs1
[root@paraguay]
```

18. Restart GPFS daemon on all nodes (Example 3-67).

Example 3-67 Restart GPFS daemon on all nodes

```
[root@paraguay] mmstartup -a
Wed Jun 30 14:18:19 PDT 2010: 6027-1642 mmstartup: Starting GPFS ...
```

19. Check status of the Windows cluster (Example 3-68).

Example 3-68 Check GPFS state on the cluster

```
[root@paraguay] mmgetstate -aLs

Node number  Node name          Quorum  Nodes up  Total nodes  GPFS state  Remarks
-----
----
```

| | | | | | | | |
|------|---|---------------|----|---|---|--------|--------|
| node | 1 | honduras-gpfs | 1* | 2 | 2 | active | quorum |
| node | 2 | paraguay-gpfs | 1* | 2 | 2 | active | quorum |

```

Summary information
-----
Number of nodes defined in the cluster:      2
Number of local nodes active in the cluster: 2
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 2
Number of quorum nodes active in the cluster: 2
Quorum = 1*, Quorum achieved

```

```
[root@paraguay]
```

20. Check file system status on the cluster (Example 3-69).

Example 3-69 Check file system mount status on all node of the cluster

```

[root@paraguay] mmmount all -a
Wed Jun 30 14:19:32 PDT 2010: 6027-1623 mmmount: Mounting file systems ...
[root@paraguay] mmlsmount all -L

File system gpfs1 is mounted on 2 nodes:
  10.11.12.154 honduras-gpfs
  10.11.12.153 paraguay-gpfs
[root@paraguay]

```

3.5.7 Simulating a GPFS failure on the cluster

While a workload is running on both nodes of the cluster that is accessing the GPFS file system, we simulate a GPFS failure on a Windows Server by shutting down the GPFS daemon. The workload on the other node of this two-node cluster should continue to run error-free with full write/read access to files in the file system.

Perform the following steps:

1. Shut down GPFS daemon on Windows server (paraguay). See Example 3-70.

Example 3-70 Use mmshutdown to stop the GPFS daemon on paraguay

```

[root@paraguay] df
Filesystem            512-blocks      Used Available Capacity Type  Mounted on
//HarddiskVolume2    585727992  40973392 544754600      7%  ntfs  /dev/fs/C
//GpfsFileSystems/gpfs1$gen(2)
                        209713152  51320320 158392832     24%  ???  /dev/fs/G
[root@paraguay] mmshutdown
Wed Jun 30 14:23:58 PDT 2010: 6027-1341 mmshutdown: Starting force unmount of GP
FS file systems
Wed Jun 30 14:24:03 PDT 2010: 6027-1344 mmshutdown: Shutting down GPFS daemons
Shutting down!
'shutdown' command about to kill process 1277
Wed Jun 30 14:24:12 PDT 2010: 6027-1345 mmshutdown: Finished
[root@paraguay] hostname
paraguay.itso.com
[root@paraguay]

```

2. Verify that the GPFS daemon is down on paraguay.

After the GPFS daemon is shut down on paraguay, all file systems on that node are unmounted but the file systems remain mounted on the other nodes of the cluster. See Example 3-71.

Example 3-71 mmgetstate and mmlsmount check status of daemon and file systems

```
[root@paraguay] mmlsmount all -L

File system gpfs1 is mounted on 1 nodes:
 10.11.12.154 honduras-gpfs
[root@paraguay]

[root@paraguay] mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|---------------|--------|----------|-------------|------------|---------|
| 1 | honduras-gpfs | 1* | 1 | 2 | active | quorum |
| 2 | paraguay-gpfs | 0 | 0 | 2 | down | quorum |

```
Summary information
-----
Number of nodes defined in the cluster: 2
Number of local nodes active in the cluster: 1
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 2
Number of quorum nodes active in the cluster: 1
Quorum = 1*, Quorum achieved

[root@paraguay]
```

3. Restart the GPFS daemon on paraguay and verify /gpfs1 gets remounted.

After the GPFS daemon has been restarted on paraguay, after GPFS initialization, all file systems with the Automount=yes setting will be automatically remounted on that node. Any application running on the other nodes using the file systems continue to run error-free. See Example 3-72.

Example 3-72 Check GPFS daemon start on paraguay after restarting

```
[root@paraguay] mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|---------------|--------|----------|-------------|------------|---------|
| 1 | honduras-gpfs | 1* | 2 | 2 | active | quorum |
| 2 | paraguay-gpfs | 1* | 2 | 2 | active | quorum |

```
Summary information
-----
Number of nodes defined in the cluster: 2
Number of local nodes active in the cluster: 2
```

```
Number of remote nodes joined in this cluster:    0
Number of quorum nodes defined in the cluster:    2
Number of quorum nodes active in the cluster:    2
Quorum = 1*, Quorum achieved
```

```
[root@paraguay]
```

```
[root@paraguay] df
Filesystem            512-blocks      Used Available Capacity Type  Mounted on
//HarddiskVolume2    585727992  40973488 544754504      7%  ntfs  /dev/fs/C
//GpfsFileSystems/gpfs1$gen(3)
                        209713152  51350528 158362624     24%  ???  /dev/fs/G
[root@paraguay]
```

3.5.8 Adding x86 Linux node to the Windows cluster

The next step is to add a x86_64 Linux client node, running Red Hat, to the Windows cluster, as follows:

1. Identify the Linux node to be added to the cluster (Example 3-73).

Example 3-73 Use uname and hostname to display characteristics of the new node

```
[root@argentina ~]# uname -a
Linux argentina 2.6.18-194.el5 #1 SMP Tue Mar 16 21:52:39 EDT 2010 x86_64
x86_64 x86_64 GNU/Linux
[root@argentina ~]#

[root@paraguay] hostname
honduras.itso.com
[root@paraguay]
```

2. Add new Linux node to the GPFS cluster (Example 3-74).

Example 3-74 Use mmaddnode to add new node to the cluster

```
[root@paraguay] mmaddnode argentina-gpfs
Wed Jun 30 23:31:35 PDT 2010: 6027-1664 mmaddnode: Processing node argentina-gpf
s.itso.com
The authenticity of host 'argentina-gpfs.itso.com (10.11.12.151)' can't be established.
RSA key fingerprint is c4:c2:dd:87:28:34:1a:96:02:da:44:14:c6:24:ed:60.
Are you sure you want to continue connecting (yes/no)? yes
mmaddnode: Command successfully completed
mmaddnode: 6027-1254 Warning: Not all nodes have proper GPFS license designations.
Use the mmchlicense command to designate licenses as needed.
mmaddnode: Propagating the cluster configuration data to all affected nodes.
[root@paraguay]
```

3. Display all node in the GPFS cluster now (Example 3-75).

Example 3-75 Display GPFS cluster information

```
[root@argentina ~]# mmlscluster
```

Warning:

This cluster contains nodes that do not have a proper GPFS license designation. This violates the terms of the GPFS licensing agreement. Use the `mmchlicense` command and assign the appropriate GPFS licenses to each of the nodes in the cluster. For more information about GPFS license designation, see the Concepts, Planning, and Installation Guide.

GPFS cluster information

```
GPFS cluster name:      honduras-gpfs.itso.com
GPFS cluster id:        723686021965502247
GPFS UID domain:        honduras-gpfs.itso.com
Remote shell command:   /usr/bin/ssh
Remote file copy command: /usr/bin/scp
```

GPFS cluster configuration servers:

```
Primary server:  honduras-gpfs.itso.com
Secondary server: paraguay-gpfs.itso.com
```

| Node | Daemon node name | IP address | Admin node name | Designation |
|------|-------------------------|--------------|-------------------------|-------------|
| 1 | honduras-gpfs.itso.com | 10.11.12.154 | honduras-gpfs.itso.com | quorum |
| 2 | paraguay-gpfs.itso.com | 10.11.12.153 | paraguay-gpfs.itso.com | quorum |
| 3 | argentina-gpfs.itso.com | 10.11.12.151 | argentina-gpfs.itso.com | |

```
[root@argentina ~]#
```

4. Set up server license for new Linux node (Example 3-76).

Example 3-76 Add server license for new Linux node with mmchlicense

```
[root@paraguay] mmchlicense server --accept -N argentina-gpfs.itso.com
```

The following nodes will be designated as possessing GPFS server licenses:
argentina-gpfs.itso.com

mmchlicense: Command successfully completed

mmchlicense: Propagating the cluster configuration data to all affected nodes.

```
[root@paraguay]
```


5. Display current licenses for all cluster nodes (Example 3-77).

Example 3-77 Use mmlicense to display all current licenses

```
[root@paraguay] mmlicense -L
```

| Node name | Required license | Designated license |
|-------------------------|------------------|--------------------|
| honduras-gpfs.itso.com | server | server |
| paraguay-gpfs.itso.com | server | server |
| argentina-gpfs.itso.com | client | server |

```
Summary information
```

```
Number of nodes defined in the cluster: 3
Number of nodes with server license designation: 3
Number of nodes with client license designation: 0
Number of nodes still requiring server license designation: 0
Number of nodes still requiring client license designation: 0
[root@paraguay]
```

6. Start GPFS on the new Linux client node (Example 3-78).

Example 3-78 Use mmstartup to start the GPFS daemon on the new node

```
[root@paraguay] mmstartup -N argentina-gpfs
Thu Jul 1 09:44:27 PDT 2010: 6027-1642 mmstartup: Starting GPFS ...
[root@paraguay]
```

7. Check the state of GPFS daemon on all nodes of the cluster. See Example 3-79.

Example 3-79 Use mmgetstate to check the state of the GPFS daemons on all nodes

```
[root@paraguay] mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|----------------|--------|----------|-------------|------------|---------|
| 1 | honduras-gpfs | 1* | 2 | 3 | active | quorum |
| 2 | paraguay-gpfs | 1* | 2 | 3 | active | quorum |
| 3 | argentina-gpfs | 1* | 2 | 3 | active | |

```
Summary information
```

```
Number of nodes defined in the cluster: 3
Number of local nodes active in the cluster: 3
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 2
Number of quorum nodes active in the cluster: 2
Quorum = 1*, Quorum achieved
```

```
[root@paraguay]
```

8. Check the state of file systems from new client node. See Example 3-80.

Example 3-80 Use mmlsmount to display /gpfs1 mount status

```
[root@paraguay] mmlsmount gpfs1 -L

File system gpfs1 is mounted on 3 nodes:
  10.11.12.154    honduras-gpfs
  10.11.12.153    paraguay-gpfs
  10.11.12.151    argentina-gpfs
[root@paraguay]
```

9. Verify the GPFS file system is mounted from argentina. See Example 3-81.

Example 3-81 Use df to display the GPFS file system from the Linux client node

```
[root@argentina ~]# df
Filesystem              1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
  268790176    5561552 249354656   3% /
/dev/sda1              101086      13166    82701  14% /boot
tmpfs                   5115680         0   5115680   0% /dev/shm
/dev/gpfs1             104856576 25723136  79133440  25% /gpfs1

[root@argentina ~]#
```

3.6 DB2 pureScale InfiniBand cluster on AIX

This scenario describes the IBM DB2 pureScale implementation on AIX.

DB2 pureScale is a technology for creating clusters of DB2 servers. DB2 pureScale technology is available on Linux and AIX database servers. It delivers superior reliability, transparency, and scalability for transactional (such as OLTP) workloads. DB2 pureScale is not designed for use with data warehousing. DB2 pureScale provides the Power pureScale feature, which implements a coupling facility (completely in software) and is relies on high-bandwidth low-latency InfiniBand networking.

The main features of DB2 pureScale are as follows:

- ▶ Unlimited capacity
- ▶ Application transparency
- ▶ Continuous availability

3.6.1 Requirements: Hardware, software, network, storage

The hardware requirements are as follows:

- ▶ POWER6® 550 Express Server with Firmware level 3.5.3+
(4) LPARs
- ▶ POWER6 HMC - Version 7 Release 3.5.0+

The software requirements are as follows:

- ▶ POWER6 - AIX Version 6.1 Technology Level (TL) 3. With Service Pack (SP) 3
- ▶ DB2 V98 with pureScale feature

- ▶ DB2 V9.7 fp2 on clients
- ▶ GPFS 3.3.0.7
- ▶ Tivoli SA MP Version 3.1 Fix Pack 7
- ▶ RSCT 2.5.5.2
- ▶ OpenSSH level 4.5.0.5302 or higher
- ▶ XL C/C++ Runtime level is 9.0.0.8 or higher
- ▶ uDAPL (6.1.0.1) for InfiniBand support, and which must have APAR IZ74595:
 - InfiniBand fix UDAPL_IB_SP3F
 - APAR IZ75395

Go to the following web address

<http://www-933.ibm.com/support/fixcentral/>

The network requirements are as follows:

- ▶ At least two networks are required:
 - Ethernet LAN
 - InfiniBand LAN
- ▶ Any of these supported InfiniBand Network switches:
 - IBM 7874-024 - 1U, 24-port 4x DDR InfiniBand Edge Switch
 - IBM 7874-040 - 4U, 48-port 4x DDR InfiniBand Director Switch
 - IBM 7874-120 - 7U, 120-port 4x DDR InfiniBand Director Switch
 - IBM 7874-240 - 14U, 240-port 4x DDR InfiniBand Director Switch

Storage requirements are as follows:

- ▶ Any disk storage supported by GPFS
- ▶ SAN Storage (with direct-attached connections)

3.6.2 GPFS configuration

When DB2 v98 with the pureScale feature is installed on the cluster, a pre-package version on GPFS (3.3.0.7) is installed and configured on all nodes of the cluster. See Table 3-3.

Table 3-3 The pureScale members and IP addresses

| Node type | Role | Name | Mgmt IP | GPFS IP |
|-----------|---------------|------------|-----------------|-----------------|
| AIX LPAR | pureScale CF | brazil-ib | 192.168.101.102 | 100.110.120.102 |
| AIX LPAR | pureScale CF | serbia-ib | 192.168.101.112 | 100.110.120.112 |
| AIX LPAR | pureScale Mem | algeria-ib | 192.168.101.101 | 100.110.120.101 |
| AIX LPAR | pureScale Mem | chile-ib | 192.168.101.111 | 100.110.120.111 |

3.6.3 GPFS cluster diagram

Figure 3-19 shows the DB2 pureScale cluster that is used for this book.

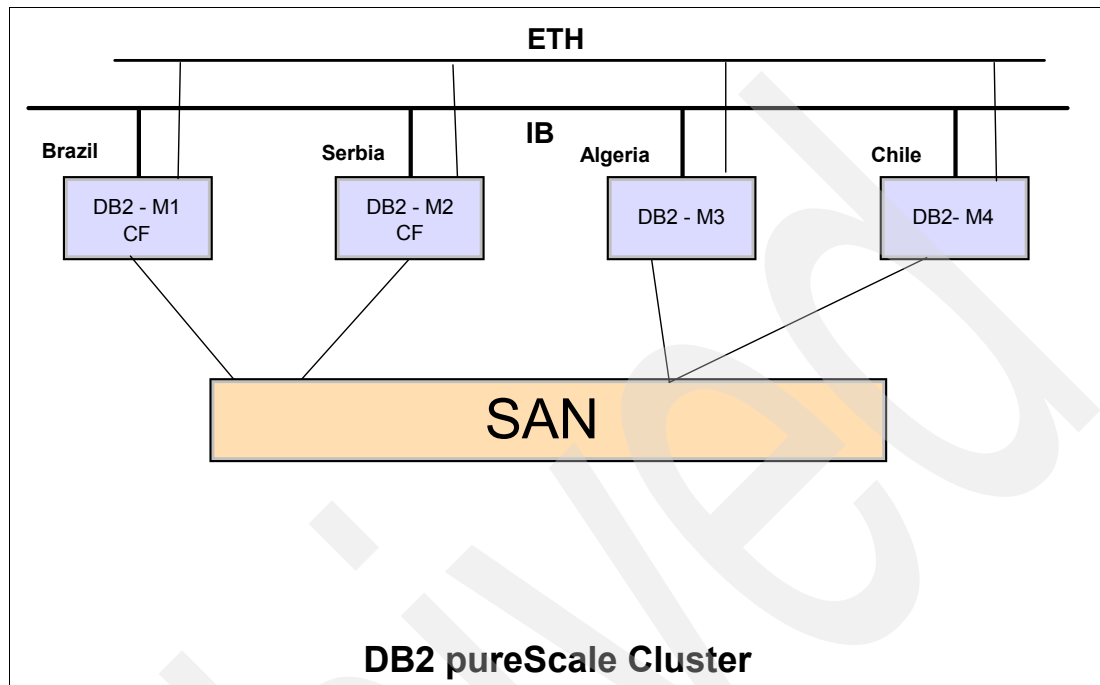


Figure 3-19 DB2 pureScale cluster

3.6.4 Cluster setup and configuration

This section provides the configuration steps to setup the DB2 pureScale cluster.

Install DB2 V98 pureScale

Perform the following steps:

1. Log in as root to the node where DB2 will be installed and go to the directory (use the `cd` command) where the DB2 .tar file was extracted (such as `/DV2V98/ese_dsf/`). Run the following commands to set up the system's display option and to start the DB2 setup.

```
export DISPLAY=localhost:10.0
./db2setup &
```

The Welcome window opens (Figure 3-20 on page 123).

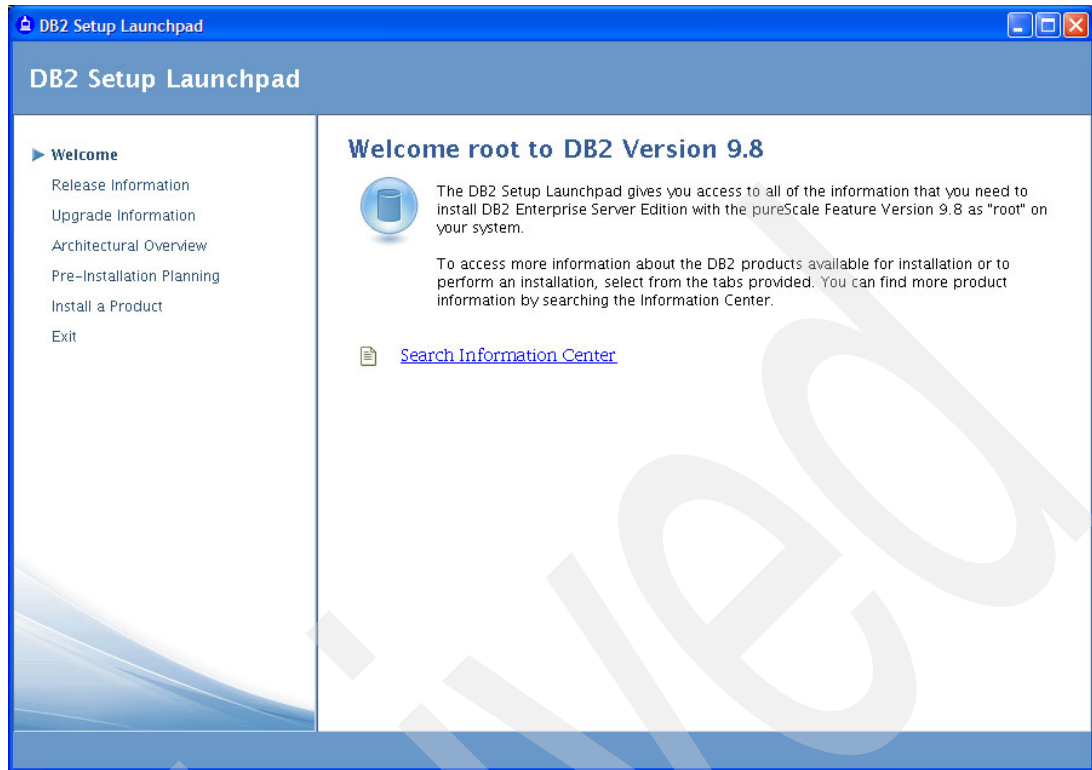


Figure 3-20 DB2 Setup wizard: Welcome window

2. Click **Install the Product**. The next window opens (Figure 3-21).

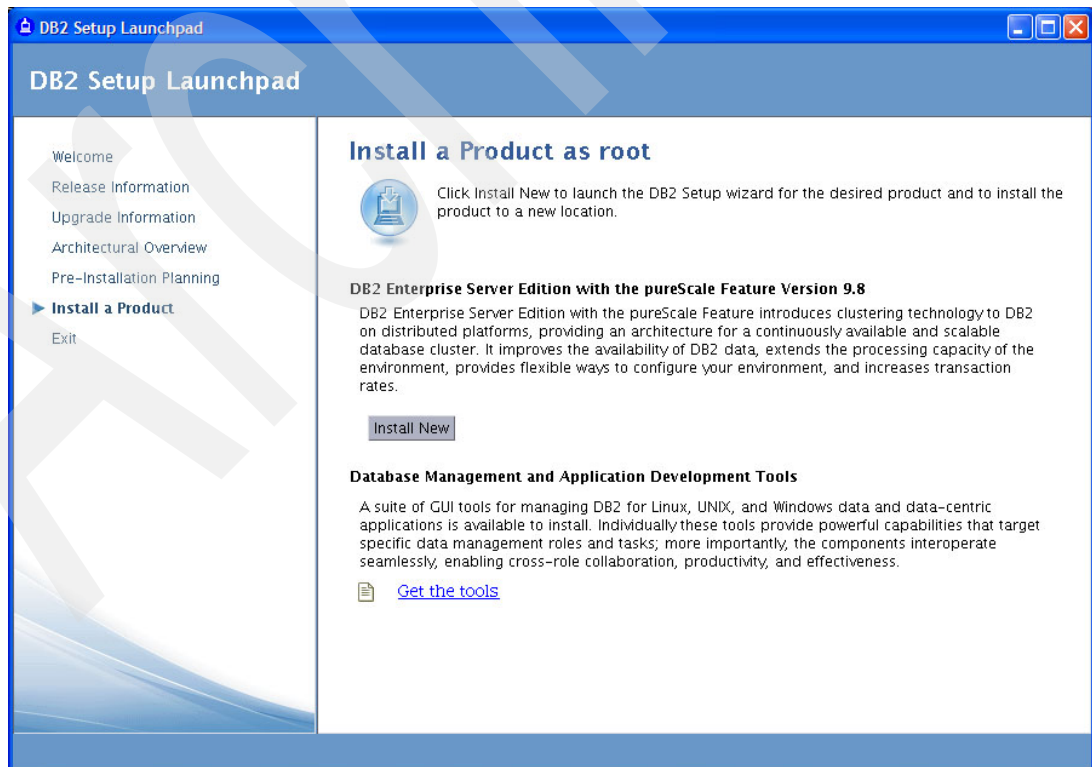


Figure 3-21 DB2 Setup wizard: Install a Product as root

3. Click **Install New**. The next window opens (Figure 3-22).

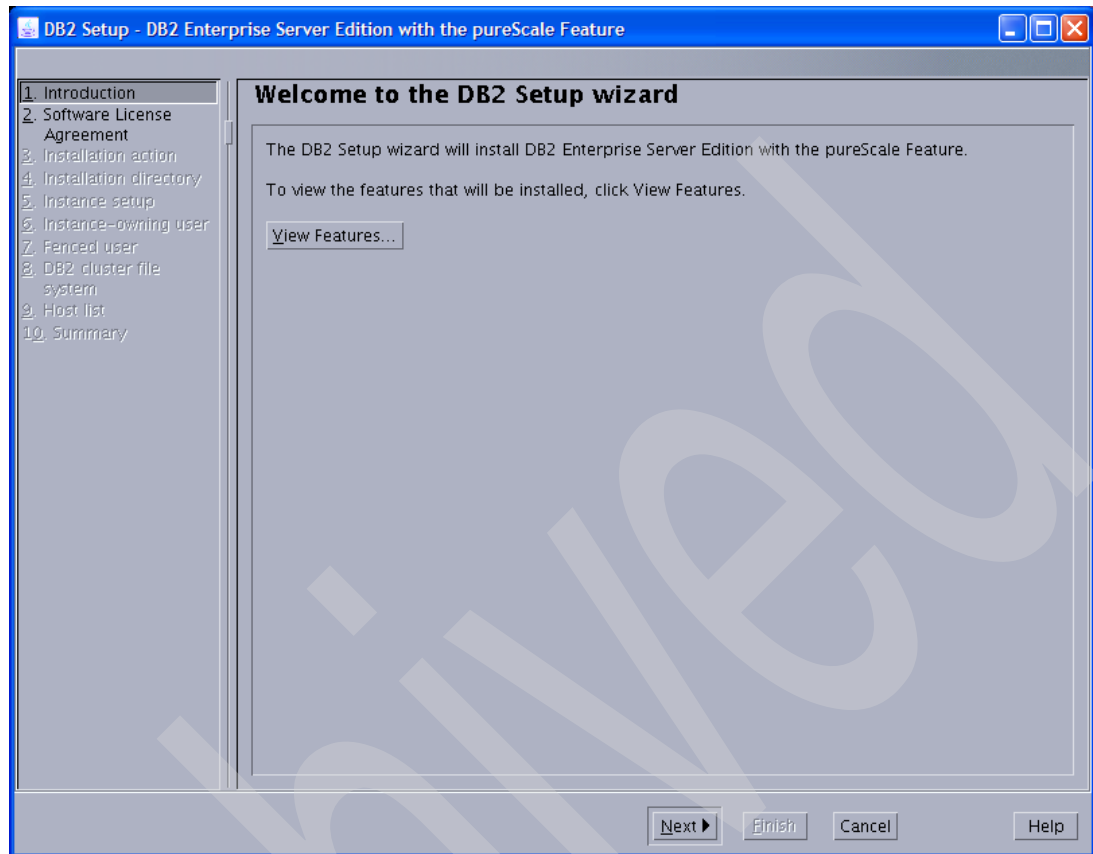


Figure 3-22 DB2 Setup wizard: DB2 ESE installation

4. To view all the DB2 features that will be installed on that node, click **View Features**. The View Features window opens (Figure 3-23 on page 125).

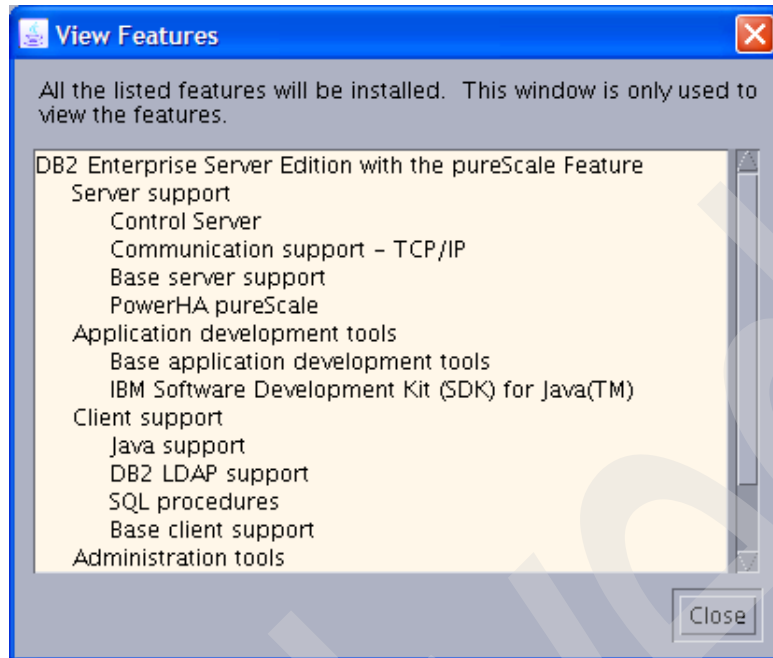


Figure 3-23 DB2 Setup wizard: View DB2 features to be installed

5. Close the window, and then click Next. The Software License Agreement window opens (Figure 3-24).

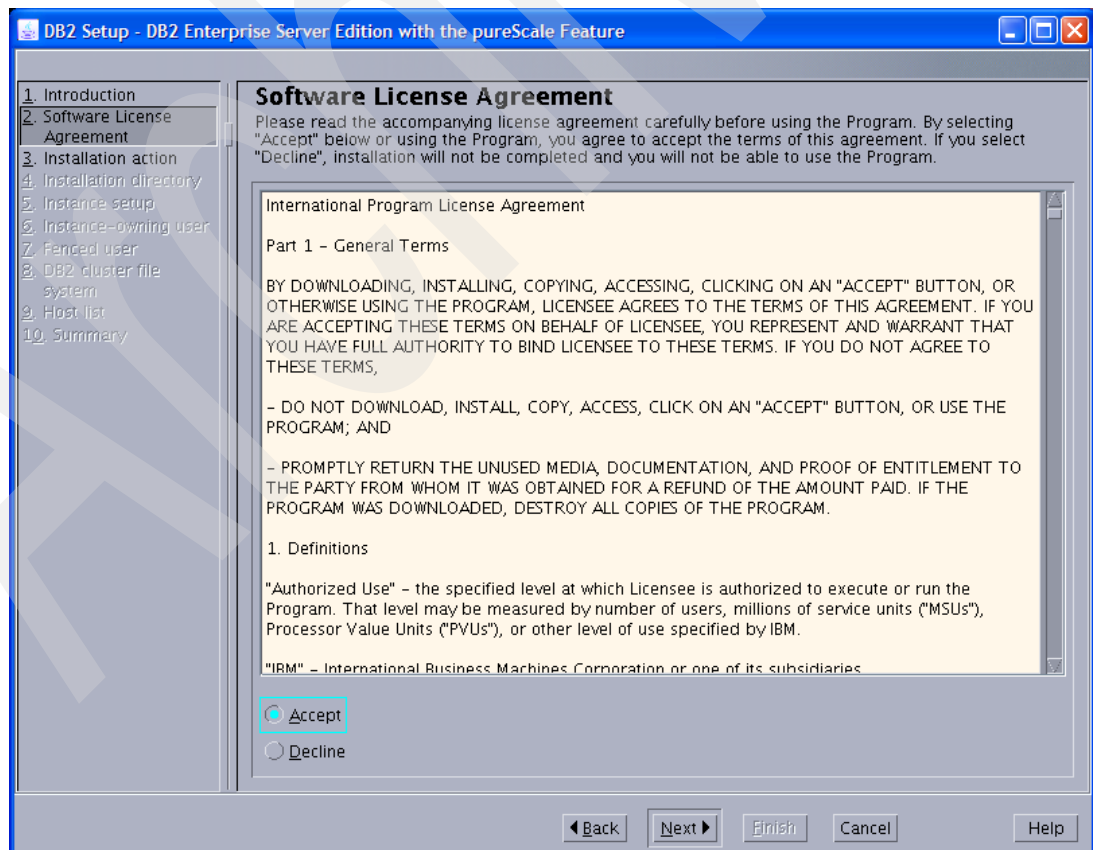


Figure 3-24 DB2 Setup wizard: Accept software license so you can continue

6. Select **Accept** and then click Next to accept the Software License Agreement so that you can continue with the installation.

The next window opens (Figure 3-25).

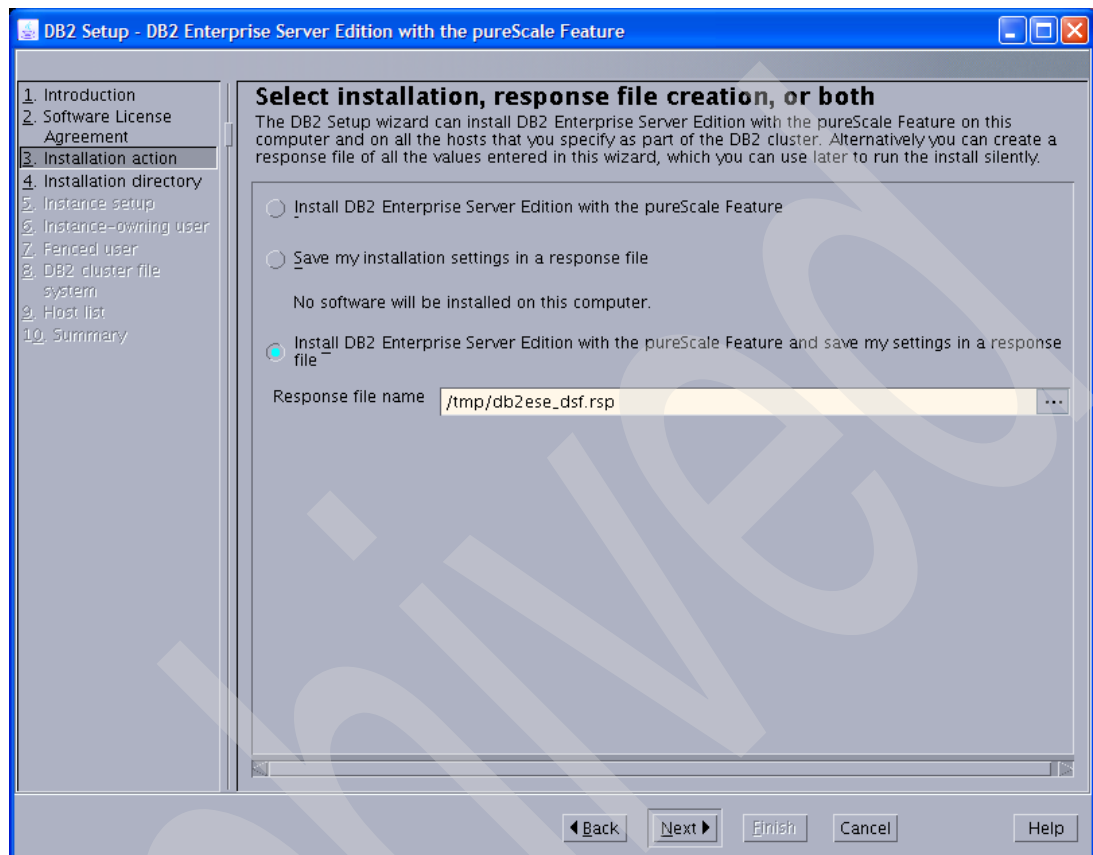


Figure 3-25 DB2 Setup wizard: Set up both Installation and response file creation

7. Select **Install DB2 Enterprise Edition with the pureScale feature and save my settings in a response file** and click **Next**.

The next window opens (Figure 3-26).

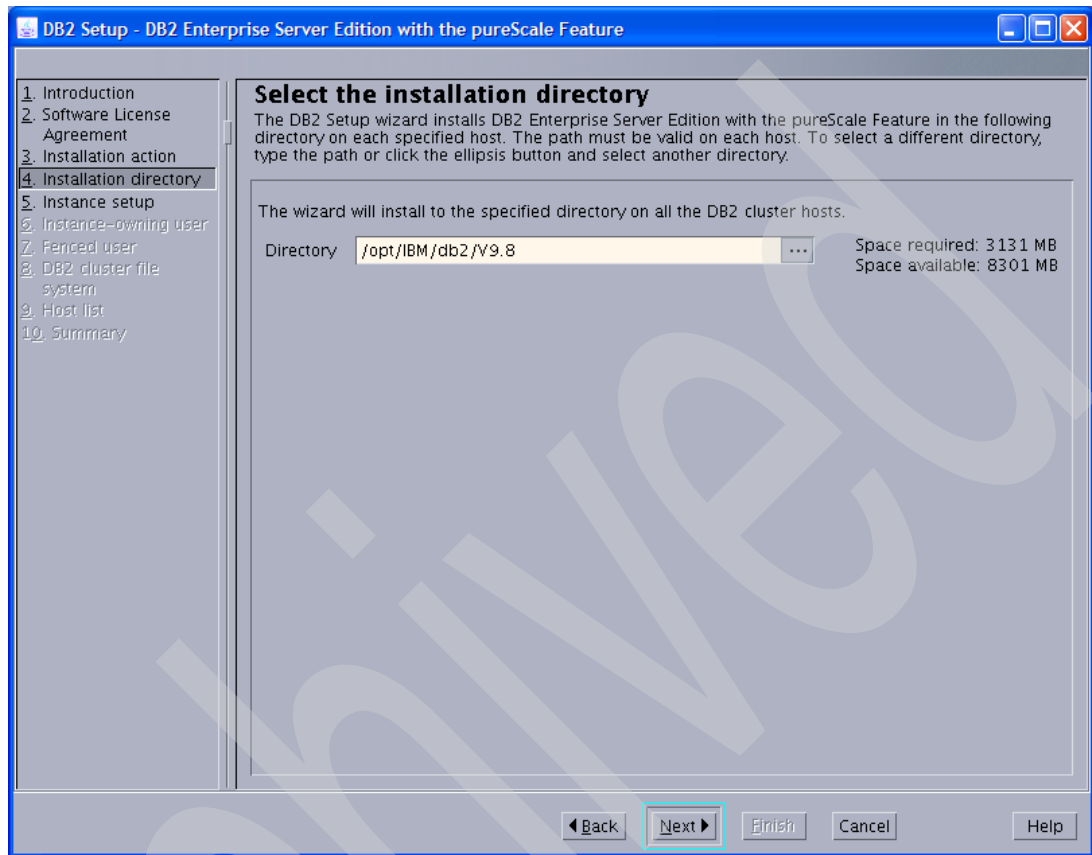


Figure 3-26 DB2 Setup wizard: Defines product installation directory

8. Select the default installation directory, or if you prefer, select a different directory, and then click Next.

The next window opens (Figure 3-27).

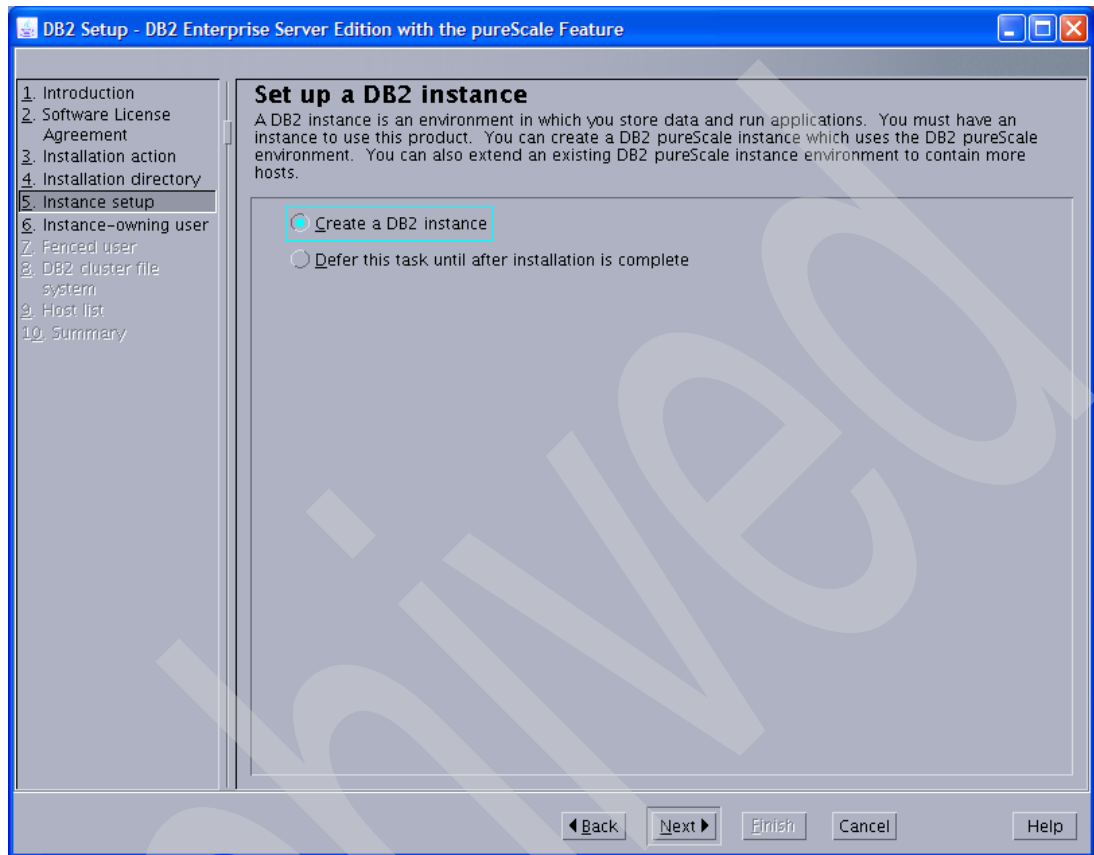


Figure 3-27 DB2 Setup wizard: Set up DB2 instance

9. Select **Create a DB2 Instance** and then click Next to continue.
The next window opens (Figure 3-28).

DB2 Setup - DB2 Enterprise Server Edition with the pureScale Feature

1. Introduction
2. Software License Agreement
3. Installation action
4. Installation directory
5. Instance setup
6. Instance-owning user
7. Fenced user
8. DB2 cluster file system
9. Host list
10. Summary

Set user information for the DB2 instance owner

Specify the instance-owning user information for the DB2 instance. DB2 will use this user to perform instance functions, and will store instance information in the user's home directory. The name of the instance will be the same as the user name.

☒ New user

User name: db2sdin1
UID:
Group name: db2iadm1
GID:
Password:
Confirm password:
Home directory: /home/db2sdin1

☐ Existing user
User name:

☒ Use default UID
☒ Use default GID

Password
• You must specify a value

Back Next Finish Cancel Help

Figure 3-28 DB2 Setup wizard: Set user information for DB2 instance owner

10. Accept the defaults in the User name, Group name, and Home directory fields.
11. Enter a password of your choice.

12. Click Next to continue. The next window opens (Figure 3-29).

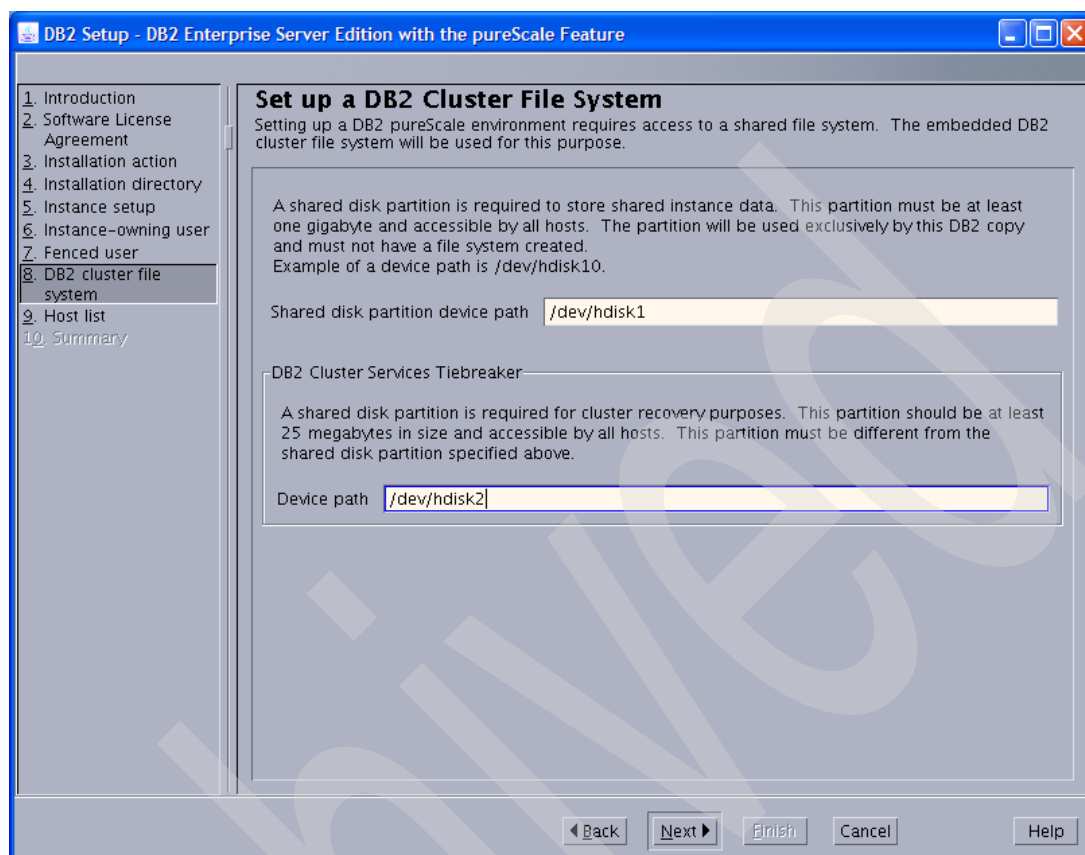


Figure 3-29 DB2 Setup wizard: Configuring DB2 Cluster file system and Tiebreaker disks

13. Set up a DB2 cluster file system and tiebreaker disks. Select separate shared disk partitions. Click Next. The next window opens (Figure 3-30).

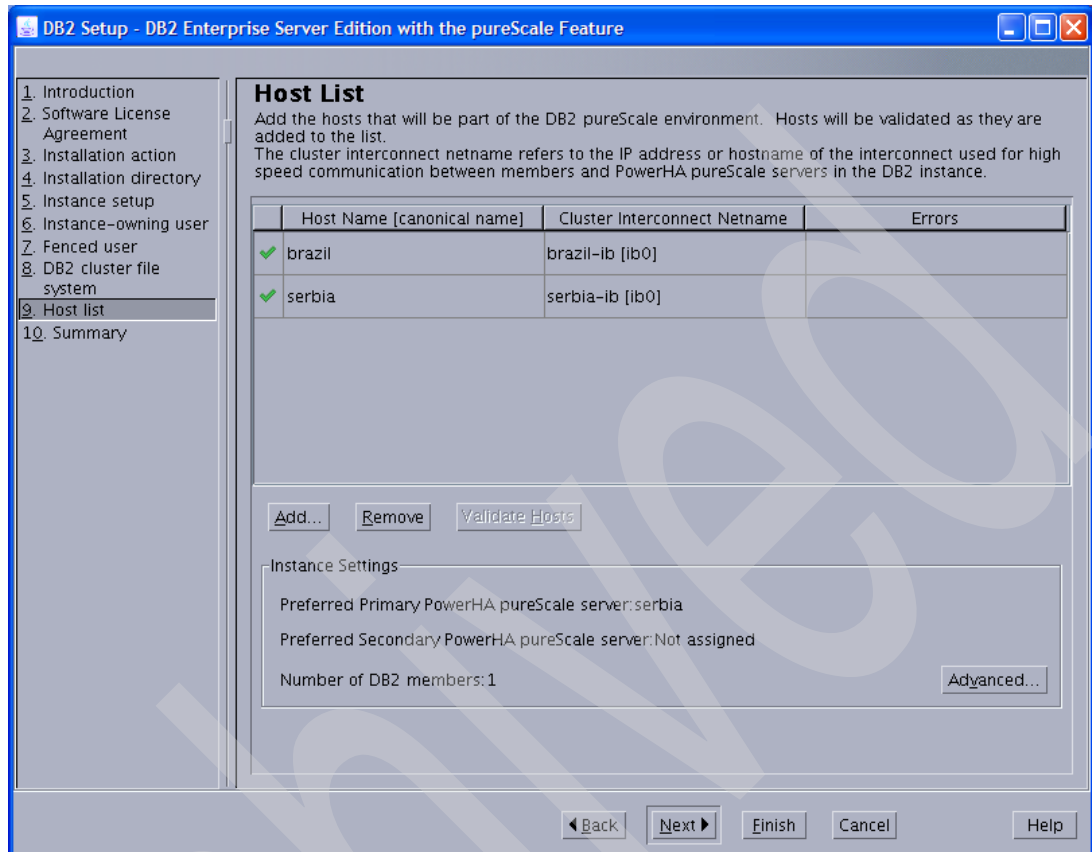


Figure 3-30 DB2 Setup wizard: Successfully added host serbia to pureScale cluster

14. Click **Add** to add all nodes that make up the cluster at this time to the Host List page. The Error panel opens when we try to add host chile (Figure 3-31 on page 132). The reason is that /tmp does not have enough space.

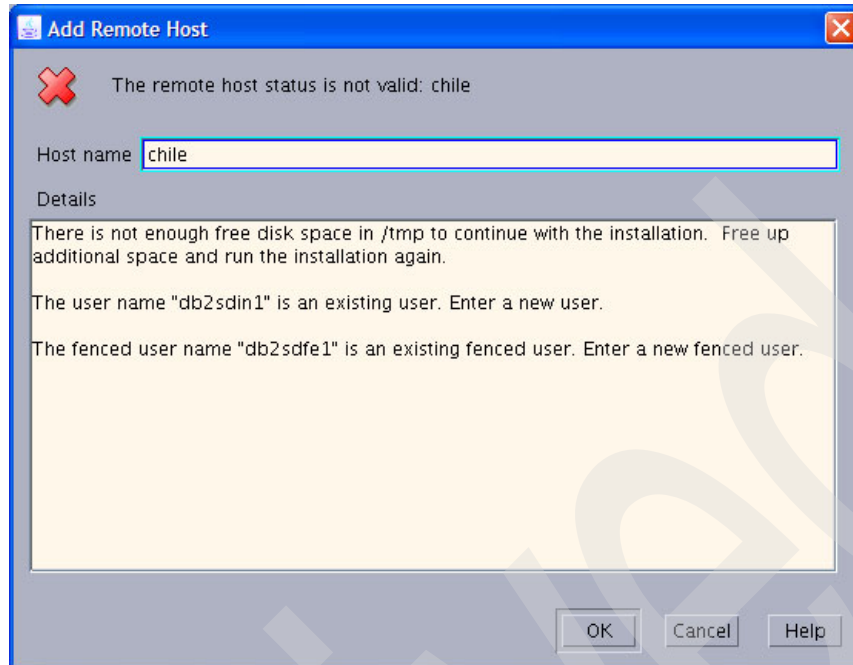


Figure 3-31 Shows errors the DB2 Setup wizard attempted to add host chile to the cluster

Two other messages are also listed:

- The db2sdin1 user already exists on host chile.
- The fenced user db2sdfe1 already exists.

Click **OK** to view message details. The next window opens (Figure 3-32).

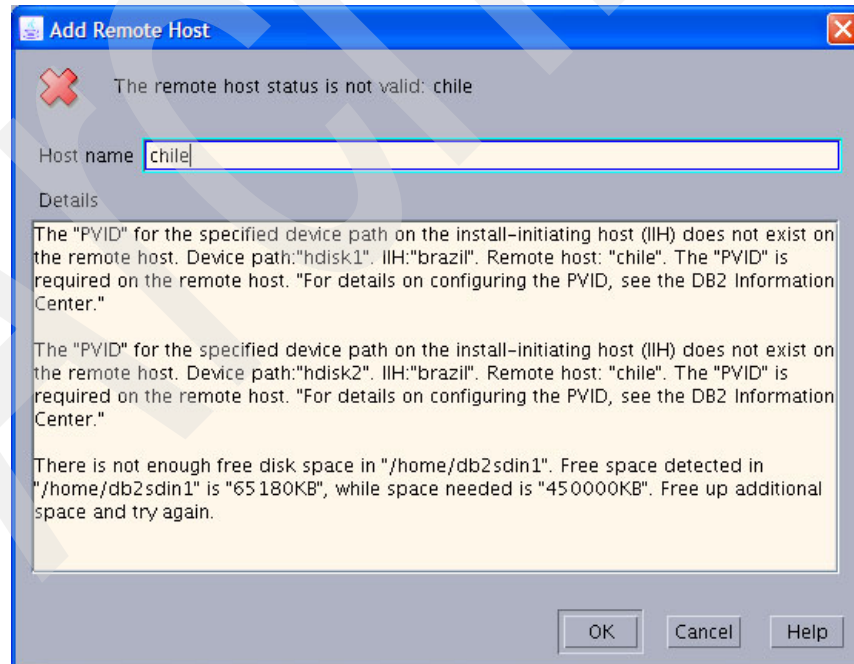


Figure 3-32 Shows an error on host 'chile' because of PVID conflicts

Although we were unable to add chile for the reasons stated, we corrected the problem and chile was successfully added to the cluster. See Figure 3-33.

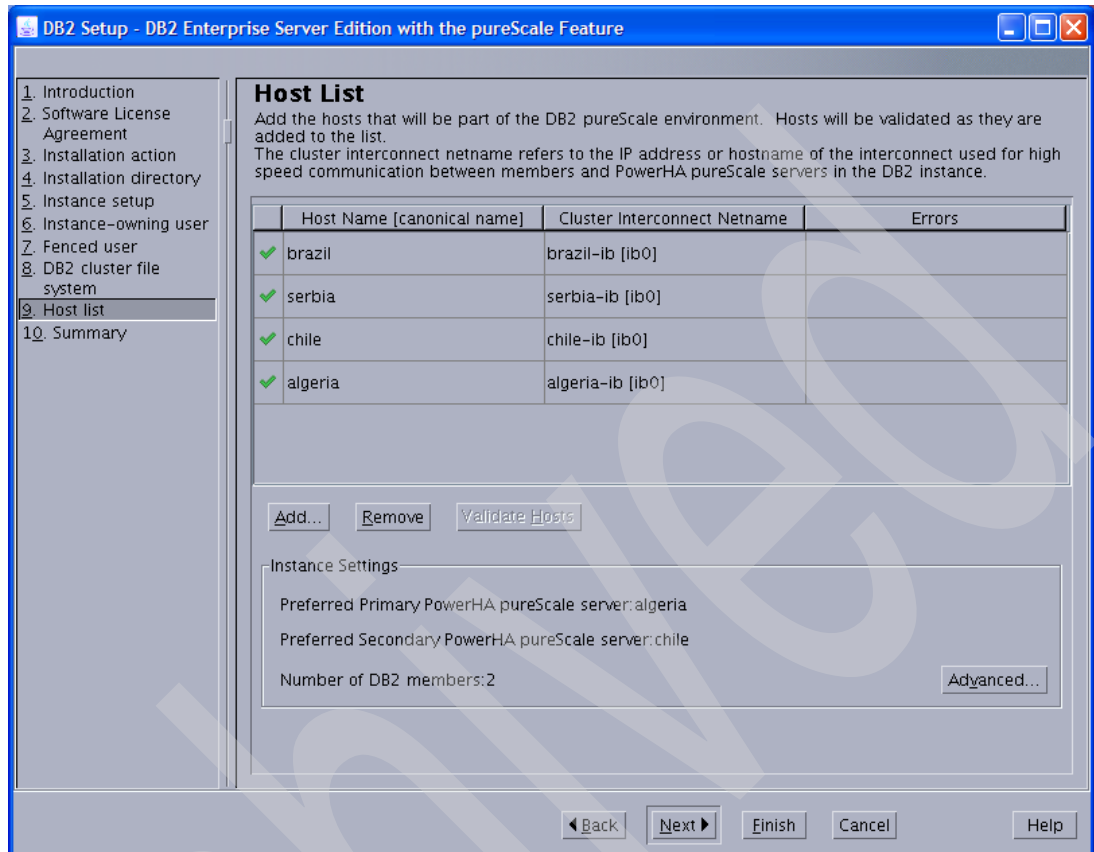


Figure 3-33 Add hosts that will be members of the DB2 pureScale cluster

15. After successfully adding Brazil, Algeria, Serbia and Chile, click **Next** to continue. The next window opens (Figure 3-34).

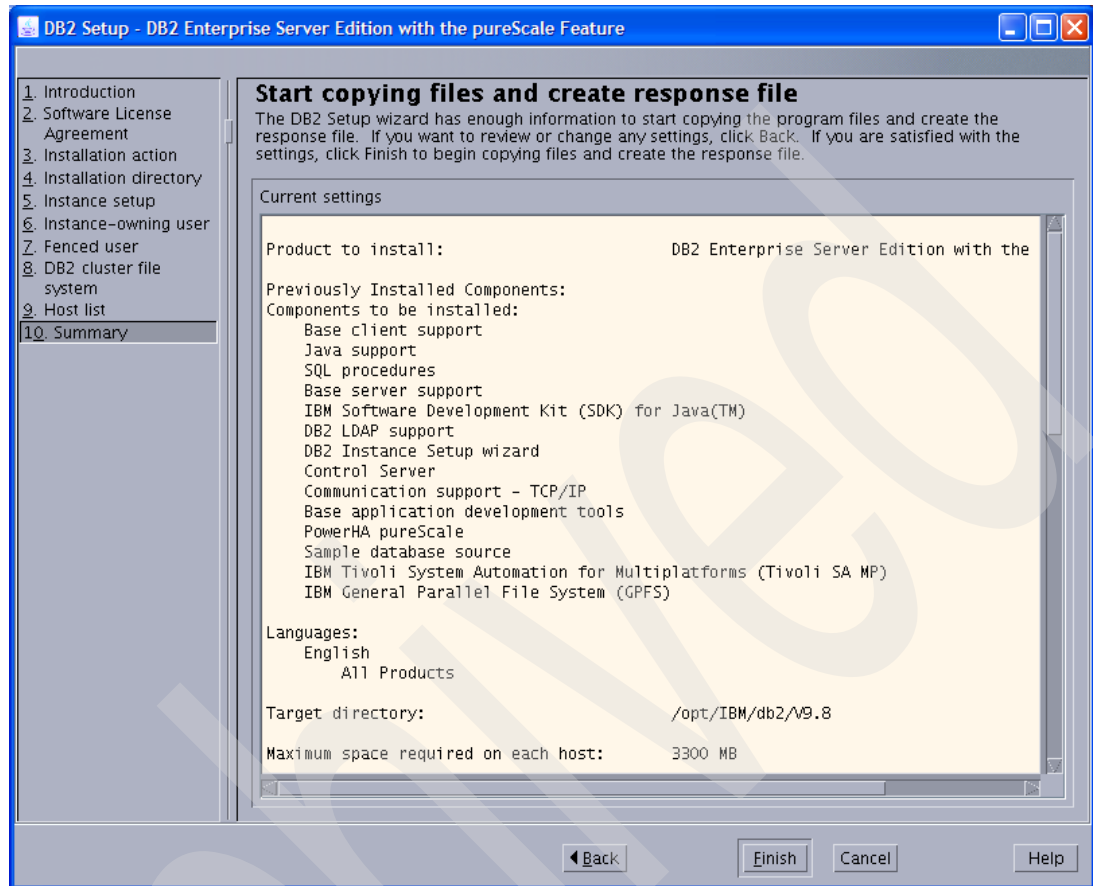


Figure 3-34 DB2 Setup wizard: Copying files to member nodes

16. Click Finish to start copying files and creating response file. The progress window shows progress of the installation (Figure 3-35).

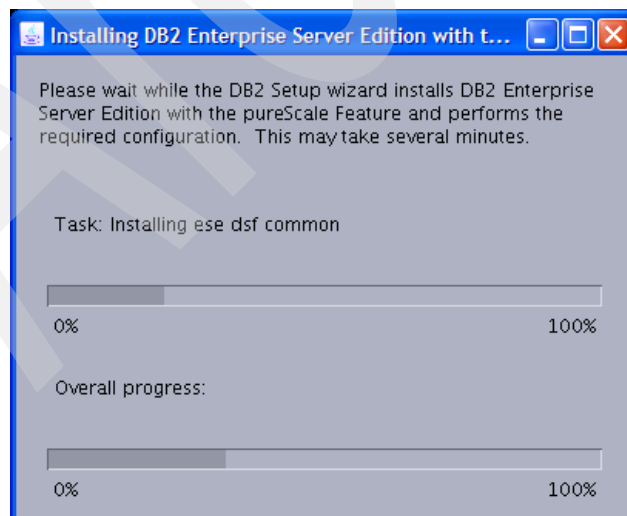


Figure 3-35 DB2 Setup wizard: Installing DB2 ESE w/ pureScale feature on member nodes

The next window opens (Figure 3-36).

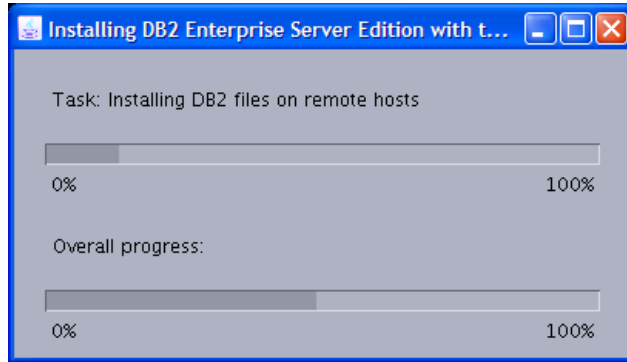


Figure 3-36 DB2 install wizard: installing DB2 files on remote hosts

After completion, the Setup Complete window opens (Figure 3-37).

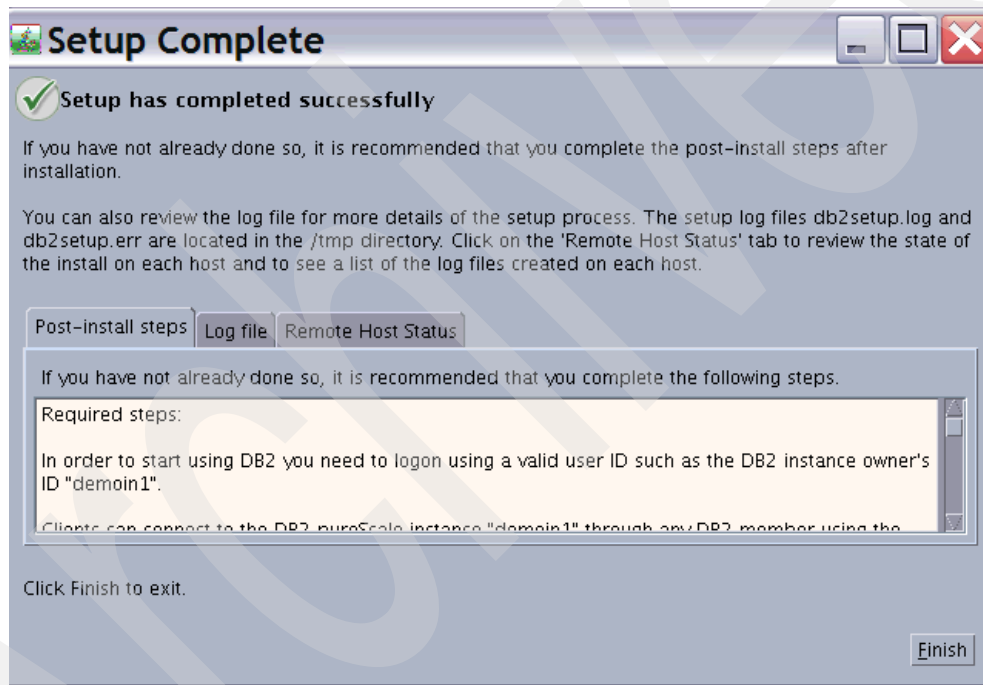


Figure 3-37 Setup complete panel displayed after DB2 Setup wizard completes

Set up uDAPL and InfiniBand on AIX cluster

Perform the following steps:

1. Run the **smitty icm** command, and then select **Add an InfiniBand Communication Manager**. See Example 3-82.

Example 3-82 Add InfiniBand Communication Manager to the cluster

Add an Infiniband Communication Manager

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

| | [Entry Fields] | |
|---|----------------|----|
| Infiniband Communication Manager Device Name | icm | |
| Minimum Request Retries | [1] | +# |
| Maximum Request Retries | [7] | +# |
| Minimum Response Time (msec) | [100] | +# |
| Maximum Response Time (msec) | [4300] | +# |
| Maximum Number of HCA's | [256] | +# |
| Maximum Number of Users | [65000] | +# |
| Maximum Number of Work Requests | [65000] | +# |
| Maximum Number of Service ID's | [1000] | +# |
| Maximum Number of Connections | [65000] | +# |
| Maximum Number of Records Per Request | [64] | +# |
| Maximum Queued Exception Notifications Per User | [1000] | +# |
| Number of MAD buffers per HCA | [64] | +# |

| | | | |
|----------|------------|-----------|----------|
| F1=Help | F2=Refresh | F3=Cancel | F4=List |
| F5=Reset | F6=Command | F7=Edit | F8=Image |
| F9=Shell | F10=Exit | Enter=Do | |

2. Run the **smitty inet** command, and then select **Change / Show Characteristics of a Network Interface** → **ib0**. See Example 3-83.

Example 3-83 Configure InfiniBand interface name, IP address, and netmask

Change / Show an IB Interface

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

| | [Entry Fields] | |
|---|-------------------|----|
| [TOP] | | |
| Network Interface Name | ib0 | |
| INTERNET ADDRESS (dotted decimal) | [100.110.120.112] | |
| Network MASK (hexadecimal or dotted decimal) | [255.255.255.0] | |
| HCA Adapter | [iba0] | + |
| * Adapter's port number | [1] | +# |
| Partition Key | [0xFFFF] | + |
| MTU | [65532] | +# |
| Queue Sizes | [4000] | +# |
| QKey | [0x1E] | + |
| Superpacket | off | + |
| Interface Specific Network Options ('NULL' will unset the option) | | |
| rfc1323 | [1] | |
| [MORE...6] | | |

| | | | |
|---------|------------|-----------|---------|
| F1=Help | F2=Refresh | F3=Cancel | F4=List |
|---------|------------|-----------|---------|

| | | | |
|----------|------------|----------|----------|
| F5=Reset | F6=Command | F7=Edit | F8=Image |
| F9=Shell | F10=Exit | Enter=Do | |

3. Check status of the InfiniBand adapter by using the **ifconfig** command. (Example 3-84).

Example 3-84 Check status of InfiniBand interface with ifconfig

```
# ifconfig -a
en0:
flags=1e080863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST,GROUPRT,64BIT,CHECK
SUM_OFFLOAD(ACTIVE),CHAIN>
    inet 192.168.101.112 netmask 0xfffffc00 broadcast 192.168.103.255
    tcp_sendspace 262144 tcp_recvspace 262144 rfc1323 1
ib0:
flags=e3a0063<UP,BROADCAST,NOTRAILERS,RUNNING,ALLCAST,MULTICAST,LINK0,LINK1,GROUPRT,64BI
T>
    inet 100.110.120.112 netmask 0xfffff00 broadcast 100.110.120.255
    tcp_sendspace 262144 tcp_recvspace 262144 tcp_nodelay 1 rfc1323 1
lo0: flags=e08084b<UP,BROADCAST,LOOPBACK,RUNNING,SIMPLEX,MULTICAST,GROUPRT,64BIT>
    inet 127.0.0.1 netmask 0xff000000 broadcast 127.255.255.255
    inet6 ::1/0
    tcp_sendspace 131072 tcp_recvspace 131072 rfc1323 1
#
```

4. Verify GPFS-DB2 cluster definitions. The DB2 pureScale feature includes DB2 cluster services that uses technology from the following sources:

- IBM General Parallel File System (GPFS)
- IBM Tivoli System Automation for Multiplatforms (Tivoli SA MP)
- IBM Reliable Scalable Clustering Technology (RSCT)

After DB2 pureScale is installed successfully on the cluster, a GPFS cluster is defined and configured on the nodes. Tivoli SA MP and RSCT is also installed and configured. See Example 3-85.

Example 3-85 Use mmlscluster command to display GPFS cluster definition

```
[root on serbia][/] => mmlscluster
```

GPFS cluster information

=====

```
GPFS cluster name:      db2cluster_20100719132846.algeria
GPFS cluster id:        13882457337115216718
GPFS UID domain:        db2cluster_20100719132846.algeria
Remote shell command:   /usr/bin/ssh
Remote file copy command: /usr/bin/scp
```

GPFS cluster configuration servers:

```
Primary server:  algeria
Secondary server: brazil
```

| Node | Daemon node name | IP address | Admin node name | Designation |
|------|------------------|-----------------|-----------------|----------------|
| 1 | algeria | 192.168.101.101 | algeria | quorum-manager |
| 2 | brazil | 192.168.101.102 | brazil | quorum-manager |
| 3 | chile | 192.168.101.111 | chile | quorum-manager |
| 4 | serbia | 192.168.101.112 | serbia | quorum-manager |

```
[root on serbia][/] =>
```

5. List GPFS cluster configuration parameters. Display the GPFS cluster parameters (Example 3-86).

Example 3-86 Use mmlsconfig to display GPFS cluster parameters

```
[root on serbia][/] => mmlsconfig
Configuration data for cluster db2cluster_20100719132846.algeria:
-----
clusterName db2cluster_20100719132846.algeria
clusterId 13882457337115216718
autoload yes
minReleaseLevel 3.3.0.2
dmapifileHandleSize 32
maxFilesToCache 10000
pagepool 256M
verifyGpfsReady yes
assertOnStructureError yes
sharedMemLimit 2047M
failureDetectionTime 35
leaseRecoveryWait 35
tiebreakerDisks gpfs1nsd
adminMode allToAll

File systems in cluster db2cluster_20100719132846.algeria:
-----
/dev/db2fs1
[root on serbia][/] =>
```

6. Display GPFS file system attributes. Display the attributes of the GPFS file (Example 3-87).

Example 3-87 Use the mmlsfs command to display the GPFS file attributes

```
[root on serbia][/] => mmlsfs /dev/db2fs1
flag value          description
-----
-f 32768             Minimum fragment size in bytes
-i 512               Inode size in bytes
-I 32768             Indirect block size in bytes
-m 1                 Default number of metadata replicas
-M 2                 Maximum number of metadata replicas
-r 1                 Default number of data replicas
-R 2                 Maximum number of data replicas
-j cluster           Block allocation type
-D nfs4              File locking semantics in effect
-k all               ACL semantics in effect
-a 1048576            Estimated average file size
-n 255               Estimated number of nodes that will mount file system
-B 1048576            Block size
-Q none              Quotas enforced
  none               Default quotas enabled
-F 262144             Maximum number of inodes
-V 11.05 (3.3.0.2)   File system version
-u yes               Support for large LUNs?
-z no                Is DMAPI enabled?
-L 4194304            Logfile size
-E yes               Exact mtime mount option
```

```

-S no          Suppress atime mount option
-K whenpossible Strict replica allocation option
-P system      Disk storage pools in file system
-d gpfslnsd    Disks in file system
-A yes         Automatic mount option
-o none        Additional mount options
-T /db2sd_20100719132932 Default mount point
[root on serbia][/] =>

```

7. Display the state of the GPFS daemon on all cluster nodes.

Use the **mmgetstate** command to display the state (Example 3-88).

Example 3-88 Display the state of all GPFS daemon on all nodes

```
[root on serbia][/] => mmgetstate -als
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|-----------|--------|----------|-------------|------------|---------|
| ----- | | | | | | |
| - | | | | | | |
| 1 | algeria | 1* | 4 | 4 | active | quorum |
| 2 | brazil | 1* | 4 | 4 | active | quorum |
| 3 | chile | 1* | 4 | 4 | active | quorum |
| 4 | serbia | 1* | 4 | 4 | active | quorum |

Summary information

```

-----
Number of nodes defined in the cluster:      4
Number of local nodes active in the cluster: 4
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 4
Number of quorum nodes active in the cluster: 4
Quorum = 1*, Quorum achieved

```

```
[root on serbia][/] =>
```

8. Verify GPFS file system is mounted on all cluster nodes.

Use the **mmismount** command to verify that the GPFS is mounted (Example 3-89). The GPFS file system is shared among all components of the DB2 pureScale environment.

Example 3-89 Verify the GPFS file system is mounted on all members of the DB2 cluster

```
[root on serbia][/] => mmismount all -L
```

File system db2fs1 is mounted on 4 nodes:

```

192.168.101.101 algeria
192.168.101.102 brazil
192.168.101.111 chile
192.168.101.112 serbia

```

```
[root on serbia][/] =>
```

9. Display DB2 Databases on Local DB Directory.

The local database directory for pureScale cluster resides on the GPFS (Example 3-90). GPFS is used to store the database and instance configuration information, such as logs, metadata, log archives, and backups.

Example 3-90 DB2 database is created on /dev/db2fs1

```
db2 => LIST DATABASE DIRECTORY

System Database Directory

Number of entries in the directory = 1

Database 1 entry:

Database alias           = DTW
Database name            = DTW
Local database directory = /db2sd_20100719132932/db2sdin1
Database release level   = e.00
Comment                  =
Directory entry type      = Indirect
Catalog database partition number = 0
Alternate server hostname =
Alternate server port number =
```

```
db2 =>
```

10. Start DB2 on all node of the pureScale Cluster

Run the **db2start** command (Example 3-91).

Example 3-91 Run db2start to start db2 daemons

```
[root on algeria][/] => su - db2sdin1
$ db2start
```

11. Verify DB2 daemons and processes are running on all cluster nodes. (Example 3-92).

Example 3-92 Check the db2 processes on algeria when db2 is active on that node

```
[root on algeria][SSL_keys/spain_keys] => ps -ef | grep db2
db2sdin1 430170 1327114 169 Jul 29 - 2021:16 db2sysc 0
root 520322 1 0 Jul 29 - 0:00 db2wdog (idle 998) [db2sdin1]
root 557068 708862 0 Jul 29 - 0:00 db2ckpwd (idle 999)
db2sdin1 671802 1327114 0 Jul 29 - 0:25 db2acd 0
db2sdin1 708862 1646610 0 Jul 29 - 0:02 db2sysc (idle 999)
root 856212 708862 0 Jul 29 - 0:00 db2ckpwd (idle 999)
root 999592 1671236 0 Jul 29 - 0:00 db2ckpwd (idle 997)
root 1003538 430170 0 Jul 29 - 0:00 db2ckpwd 0
root 1007820 1671236 0 Jul 29 - 0:00 db2ckpwd (idle 997)
root 1077458 1 0 Jul 29 - 0:00 db2wdog (idle 997) [db2sdin1]
root 1155122 430170 0 Jul 29 - 0:00 db2ckpwd 0
root 1278136 569594 0 16:14:51 pts/0 0:00 grep db2
root 1282208 430170 0 Jul 29 - 0:00 db2ckpwd 0
root 1294442 708862 0 Jul 29 - 0:00 db2ckpwd (idle 999)
root 1327114 1 0 Jul 29 - 0:00 db2wdog 0 [db2sdin1]
db2sdfel 1409052 1327114 0 Jul 29 - 0:00 db2fmp (C) 0
root 1495062 1732642 0 Jul 29 - 0:00 db2ckpwd (idle 998)
```

```

root 1613856 1671236 0 Jul 29 - 0:00 db2ckpwd (idle 997)
root 1646610 1 0 Jul 29 - 0:00 db2wdog (idle 999) [db2sdin1]
db2sdin1 1671236 1077458 0 Jul 29 - 0:02 db2sysc (idle 997)
root 1703992 1732642 0 Jul 29 - 0:00 db2ckpwd (idle 998)
root 1724652 1732642 0 Jul 29 - 0:00 db2ckpwd (idle 998)
db2sdin1 1732642 520322 0 Jul 29 - 0:02 db2sysc (idle 998)
[root on algeria][SSL_keys/spain_keys] =>

```

3.6.5 Use the mmpmon tool to collect statistics from each node

Start the sdtw workload from a DB2 client node and use the GPFS mmpmon tool to collect statistics from each node of the pureScale cluster while the sdtw workload is running.

Example 3-93 collects statistics from node algeria.

Example 3-93 The mmpmon statistics from algeria

```

[root on algeria][SSL_keys/spain_keys] => /gpfs_perf.pl /dev/db2fs1
Starting GPFS performance data gathering on host algeria.
Duration is 60 seconds.

```

```

Fri Jul 30 15:07:14 EDT 2010
mmpmon node 192.168.101.101 name algeria rhist reset status 1
rhist has not been turned on yet
mmpmon node 192.168.101.101 name algeria reset OK
mmpmon node 192.168.101.101 name algeria rhist on OK

```

Gathering Performance data for [/dev/db2fs1] GPFS device.

```

      read      write
MB/sec  MB/sec   fopens fclose   reads writes inodes
Fri Jul 30 15:08:15 EDT 2010

```

*** Histogram and overall statistics ***

```

mmpmon node 192.168.101.101 name algeria rhist s OK read timestamp 1280516895/978675
size range      0 to      255 count      24
  latency range  0.0 to    1.0 count      24
size range    1024 to    2047 count       9
  latency range  0.0 to    1.0 count       8
  latency range  1.1 to   10.0 count       1
size range    2048 to    4095 count       4
  latency range  0.0 to    1.0 count       4
size range    4096 to    8191 count     623
  latency range  0.0 to    1.0 count     616
  latency range  1.1 to   10.0 count       5
  latency range 100.1 to  200.0 count       2
size range    8192 to   16383 count       1
  latency range  30.1 to   100.0 count       1
size range   16384 to   32767 count      35
  latency range  0.0 to    1.0 count      35
mmpmon node 192.168.101.101 name algeria rhist s OK write timestamp 1280516895/978741
size range      256 to    511 count       2
  latency range  0.0 to    1.0 count       2
size range    4096 to    8191 count    32820
  latency range  0.0 to    1.0 count    18360
  latency range  1.1 to   10.0 count    13216
  latency range 10.1 to   30.0 count     231
  latency range  30.1 to   100.0 count     353
  latency range 100.1 to  200.0 count     139

```

```

latency range      200.1 to      400.0 count      348
latency range      400.1 to      800.0 count      173
size range          8192 to     16383 count      200
latency range          0.0 to       1.0 count      178
latency range          1.1 to      10.0 count       18
latency range        30.1 to     100.0 count        2
latency range       100.1 to     200.0 count        1
latency range       200.1 to     400.0 count        1
mmpmon node 192.168.101.101 name algeria fs_io_s OK
cluster:          db2cluster_20100719132846.algeria
filesystem:        db2fs1
disks:              1
timestamp:         1280516895/980263
bytes read:         914393
bytes written:     137274130
opens:              411
closes:             411
reads:              696
writes:             33046
readdir:            4
inode updates:      29

```

Finished.

[root on algeria][SSL_keys/spain_keys] =>

Example 3-94 collects statistics from brazil host.

Example 3-94 The mmpmon gpfs statistics on file system /db2fs1 on brazil

[root on brazil][/] => /gpfs_perf.pl /dev/db2fs1

Starting GPFS performance data gathering on host brazil.

Duration is 60 seconds.

Fri Jul 30 15:14:37 EDT 2010

mmpmon node 192.168.101.102 name brazil rhist reset status 1

rhist has not been turned on yet

mmpmon node 192.168.101.102 name brazil reset OK

mmpmon node 192.168.101.102 name brazil rhist on OK

Gathering Performance data for [/dev/db2fs1] GPFS device.

```

read      write
MB/sec    MB/sec    fopens  fclose    reads writes inodes
Fri Jul 30 15:15:38 EDT 2010

```

*** Histogram and overall statistics ***

mmpmon node 192.168.101.102 name brazil rhist s OK read timestamp
1280517338/627031

```

size range      256 to      511 count      12
latency range    0.0 to      1.0 count      12
size range      4096 to     8191 count     156
latency range    0.0 to      1.0 count     141
latency range    1.1 to     10.0 count      14
latency range   30.1 to    100.0 count       1
size range     16384 to   32767 count      24
latency range    0.0 to      1.0 count      24

```



```

mmpmon node 192.168.101.102 name brazil rhist s OK write timestamp
1280517338/627080
mmpmon node 192.168.101.102 name brazil fs_io_s OK
cluster:      db2cluster_20100719132846.algeria
filesystem:   db2fs1
disks:        1
timestamp:    1280517338/627498
bytes read:   729588
bytes written: 0
opens:        150
closes:       150
reads:        192
writes:       0
readdir:      0
inode updates: 16

```

Finished.
[root on brazil][/] =>

Example 3-95 collects statistics from chile host.

Example 3-95 The mmpmon GPFS statistics from /db2fs1 on chile

```

[root on chile][/] => /gpfs_perf.pl /dev/db2fs1
Starting GPFS performance data gathering on host chile.
Duration is 60 seconds.

```

```

Fri Jul 30 15:16:48 EDT 2010
mmpmon node 192.168.101.111 name chile rhist reset status 1
rhist has not been turned on yet
mmpmon node 192.168.101.111 name chile reset OK
mmpmon node 192.168.101.111 name chile rhist on OK

```

Gathering Performance data for [/dev/db2fs1] GPFS device.

| read | write | | | | | |
|------------------------------|--------|--------|--------|-------|--------|--------|
| MB/sec | MB/sec | fopens | fclose | reads | writes | inodes |
| Fri Jul 30 15:17:49 EDT 2010 | | | | | | |

*** Histogram and overall statistics ***

```

mmpmon node 192.168.101.111 name chile rhist s OK read timestamp 1280517469/787929
size range      0 to      255 count      9
  latency range 0.0 to    1.0 count      9
size range     256 to    511 count      1
  latency range 0.0 to    1.0 count      1
size range    1024 to   2047 count      3
  latency range 0.0 to    1.0 count      3
size range    2048 to   4095 count      2
  latency range 0.0 to    1.0 count      2
size range    4096 to   8191 count     638
  latency range 0.0 to    1.0 count     637
  latency range 30.1 to  100.0 count      1
size range    8192 to  16383 count      3
  latency range 1.1 to   10.0 count      2
  latency range 10.1 to   30.0 count      1
size range   16384 to  32767 count     33
  latency range 0.0 to    1.0 count     33
mmpmon node 192.168.101.111 name chile rhist s OK write timestamp 1280517469/788001

```

```

size range      256 to      511 count      1
  latency range  0.0 to      1.0 count      1
size range      4096 to     8191 count     55275
  latency range  0.0 to      1.0 count     34632
  latency range  1.1 to     10.0 count     19573
  latency range  10.1 to     30.0 count      235
  latency range  30.1 to    100.0 count      241
  latency range  100.1 to   200.0 count      181
  latency range  200.1 to   400.0 count      291
  latency range  400.1 to   800.0 count       23
  latency range  800.1 to  1000.0 count       51
  latency range 1000.1 to     0.0 count       48
size range      8192 to    16383 count      224
  latency range  0.0 to      1.0 count      193
  latency range  1.1 to     10.0 count       30
  latency range  10.1 to     30.0 count       1
mmpmon node 192.168.101.111 name chile fs_io_s OK
cluster:        db2cluster_20100719132846.algeria
filesystem:     db2fs1
disks:          1
timestamp:      1280517469/789156
bytes read:     869556
bytes written:  228790624
opens:          400
closes:         400
reads:          689
writes:         55473
readdir:        0
inode updates:  23

Finished.
[root on chile][/] =>

```

Example 3-96 collects statistics from serbia host.

Example 3-96 mmpmon GPFS stat for /db2fs1 on serbia

```

[root on serbia][/] => /gpfs_perf.pl /dev/db2fs1
Starting GPFS performance data gathering on host serbia.
Duration is 60 seconds.

Fri Jul 30 15:21:35 EDT 2010
mmpmon node 192.168.101.112 name serbia rhist reset status 1
rhist has not been turned on yet
mmpmon node 192.168.101.112 name serbia reset OK
mmpmon node 192.168.101.112 name serbia rhist on OK

Gathering Performance data for [/dev/db2fs1] GPFS device.

  read    write
MB/sec   MB/sec   fopens  fclose   reads writes inodes
Fri Jul 30 15:22:36 EDT 2010

*** Histogram and overall statistics ***

mmpmon node 192.168.101.112 name serbia rhist s OK read timestamp 1280517756/753901
size range      256 to      511 count      6
  latency range  0.0 to      1.0 count      6
size range      4096 to     8191 count     144
  latency range  0.0 to      1.0 count     137

```

```

    latency range      1.1 to      10.0 count      7
    size range         16384 to    32767 count      24
    latency range      0.0 to      1.0 count      24
mmpmon node 192.168.101.112 name serbia rhist s OK write timestamp 1280517756/753966
mmpmon node 192.168.101.112 name serbia fs_io_s OK
cluster:             db2cluster_20100719132846.algeria
filesystem:          db2fs1
disks:               1
timestamp:           1280517756/754363
bytes read:           678696
bytes written:        0
opens:               132
closes:              132
reads:               174
writes:               0
readdir:              0
inode updates:        7

Finished.
[root on serbia][/] =>

```

3.6.6 Simulating a GPFS failure on member 0 (algeria)

To simulate a GPFS failure on a member node, we shut down the GPFS daemon on algeria and verify that the remaining members continue to process db2 transactions and queries. See Example 3-97.

Example 3-97 Use mmshutdown to stop GPFS daemon on algeria

```

[root on algeria][/] => mmshutdown
Mon Aug  2 03:39:39 EDT 2010: mmshutdown: Starting force unmount of GPFS file systems
forced unmount of /db2sd_20100719132932
Mon Aug  2 03:39:44 EDT 2010: mmshutdown: Shutting down GPFS daemons
Shutting down!
'shutdown' command about to kill process 901192
Mon Aug  2 03:39:52 EDT 2010: mmshutdown: Finished
[root on algeria][/] =>

[root on algeria][/] => ps -ef | grep db2
  root   520322      1   0   Jul 29   -   0:00 db2wdog (idle 998) [db2sdin1]
  root   557068  708862   0   Jul 29   -   0:00 db2ckpwd (idle 999)
db2sdin1 708862 1646610   0   Jul 29   -   0:06 db2sysc (idle 999)
  root   856212  708862   0   Jul 29   -   0:00 db2ckpwd (idle 999)
  root   999592 1671236   0   Jul 29   -   0:00 db2ckpwd (idle 997)
  root  1007820 1671236   0   Jul 29   -   0:00 db2ckpwd (idle 997)
  root  1077458      1   0   Jul 29   -   0:00 db2wdog (idle 997) [db2sdin1]
  root  1294442  708862   0   Jul 29   -   0:00 db2ckpwd (idle 999)
  root  1495062 1732642   0   Jul 29   -   0:00 db2ckpwd (idle 998)
  root  1613856 1671236   0   Jul 29   -   0:00 db2ckpwd (idle 997)
  root  1646610      1   0   Jul 29   -   0:00 db2wdog (idle 999) [db2sdin1]
db2sdin1 1671236 1077458   0   Jul 29   -   0:06 db2sysc (idle 997)
  root  1675496 1360118   0 03:45:21 pts/1 0:00 grep db2
  root  1703992 1732642   0   Jul 29   -   0:00 db2ckpwd (idle 998)
  root  1724652 1732642   0   Jul 29   -   0:00 db2ckpwd (idle 998)
db2sdin1 1732642  520322   0   Jul 29   -   0:06 db2sysc (idle 998)
[root on algeria][/] =>

```

Verify GPFS daemon is down on algeria

Use the `mmgetstate` command to verify algeria (Example 3-98).

Example 3-98 Use mmgetstate to verify GPFS daemon is down on algeria

```
[root on brazil][/] => mmgetstate -als
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|-----------|--------|----------|-------------|------------|-------------|
| 1 | algeria | 0 | 0 | 4 | down | quorum node |
| 2 | brazil | 1* | 3 | 4 | active | quorum node |
| 3 | chile | 1* | 3 | 4 | active | quorum node |
| 4 | serbia | 1* | 3 | 4 | active | quorum node |

Summary information

Number of nodes defined in the cluster: 4
Number of local nodes active in the cluster: 3
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 4
Number of quorum nodes active in the cluster: 3
Quorum = 1*, Quorum achieved

```
[root on brazil][/] =>
```

```
[root on brazil][/] => mmlsmount all_local -L
```

File system db2fs1 is mounted on 3 nodes:

| | |
|------------------------|-----------------------------------|
| 192.168.101.112 serbia | db2cluster_20100719132846.algeria |
| 192.168.101.102 brazil | db2cluster_20100719132846.algeria |
| 192.168.101.111 chile | db2cluster_20100719132846.algeria |

```
[root on brazil][/] =>
```

Restart GPFS daemon on algeria

Restart GPFS daemon on algeria and verify that GPFS and db2 processes are started correctly on that node (Example 3-99).

Example 3-99 Check the state of the GPFS daemon on all nodes of the cluster

```
[root on brazil][/] => mmgetstate -als
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|-----------|--------|----------|-------------|------------|-------------|
| 1 | algeria | 1* | 4 | 4 | active | quorum node |
| 2 | brazil | 1* | 4 | 4 | active | quorum node |
| 3 | chile | 1* | 4 | 4 | active | quorum node |
| 4 | serbia | 1* | 4 | 4 | active | quorum node |

Summary information

Number of nodes defined in the cluster: 4
Number of local nodes active in the cluster: 4
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 4
Number of quorum nodes active in the cluster: 4
Quorum = 1*, Quorum achieved

```
[root on brazil][/] => mmlsmount all_local -L
```

```
File system db2fs1 is mounted on 4 nodes:
192.168.101.112 serbia          db2cluster_20100719132846.algeria
192.168.101.102 brazil         db2cluster_20100719132846.algeria
192.168.101.111 chile          db2cluster_20100719132846.algeria
192.168.101.101 algeria        db2cluster_20100719132846.algeria
[root on brazil][/] =>
```

3.6.7 Troubleshooting DB2 pureScale issues

DB2 cluster services control the state of Tivoli SA MP, RSCT, and GPFS resources with defined policies. When a failure occurs in any of the cluster resources, DB2 cluster services try to isolate the problem and maintain the availability of the DB2 cluster.

The following commands can be used to troubleshoot pureScale issues:

- ▶ The **db2cluster** command is used to administer and manage the DB2 pureScale cluster and file system.
- ▶ The **lssam** command lists defined resource groups and their members (Example 3-100).

Example 3-100 lssam output

```
[root on algeria][/] => lssam
Online IBM.ResourceGroup:ca_db2sdin1_0-rg Nominal=Online
  '- Online IBM.Application:ca_db2sdin1_0-rs
    |- Online IBM.Application:ca_db2sdin1_0-rs:brazil
    '- Online IBM.Application:ca_db2sdin1_0-rs:serbia
Online IBM.ResourceGroup:db2_db2sdin1_0-rg Nominal=Online
  '- Online IBM.Application:db2_db2sdin1_0-rs
    |- Online IBM.Application:db2_db2sdin1_0-rs:algeria
    '- Offline IBM.Application:db2_db2sdin1_0-rs:chile
Online IBM.ResourceGroup:db2_db2sdin1_1-rg Nominal=Online
  '- Online IBM.Application:db2_db2sdin1_1-rs
    |- Offline IBM.Application:db2_db2sdin1_1-rs:algeria
    '- Online IBM.Application:db2_db2sdin1_1-rs:chile
Online IBM.ResourceGroup:db2mnt-db2sd_20100719132932-rg Nominal=Online
  '- Online IBM.Application:db2mnt-db2sd_20100719132932-rs
    |- Online IBM.Application:db2mnt-db2sd_20100719132932-rs:algeria
    |- Online IBM.Application:db2mnt-db2sd_20100719132932-rs:brazil
    |- Online IBM.Application:db2mnt-db2sd_20100719132932-rs:chile
    '- Online IBM.Application:db2mnt-db2sd_20100719132932-rs:serbia
Online IBM.ResourceGroup:idle_db2sdin1_997_algeria-rg Nominal=Online
  '- Online IBM.Application:idle_db2sdin1_997_algeria-rs
    '- Online IBM.Application:idle_db2sdin1_997_algeria-rs:algeria
Online IBM.ResourceGroup:idle_db2sdin1_997_chile-rg Nominal=Online
  '- Online IBM.Application:idle_db2sdin1_997_chile-rs
    '- Online IBM.Application:idle_db2sdin1_997_chile-rs:chile
Online IBM.ResourceGroup:idle_db2sdin1_998_algeria-rg Nominal=Online
  '- Online IBM.Application:idle_db2sdin1_998_algeria-rs
    '- Online IBM.Application:idle_db2sdin1_998_algeria-rs:algeria
Online IBM.ResourceGroup:idle_db2sdin1_998_chile-rg Nominal=Online
  '- Online IBM.Application:idle_db2sdin1_998_chile-rs
    '- Online IBM.Application:idle_db2sdin1_998_chile-rs:chile
Online IBM.ResourceGroup:idle_db2sdin1_999_algeria-rg Nominal=Online
  '- Online IBM.Application:idle_db2sdin1_999_algeria-rs
    '- Online IBM.Application:idle_db2sdin1_999_algeria-rs:algeria
Online IBM.ResourceGroup:idle_db2sdin1_999_chile-rg Nominal=Online
  '- Online IBM.Application:idle_db2sdin1_999_chile-rs
    '- Online IBM.Application:idle_db2sdin1_999_chile-rs:chile
Online IBM.ResourceGroup:primary_db2sdin1_900-rg Nominal=Online
```

```

'- Online IBM.Application:primary_db2sdin1_900-rs
  |- Online IBM.Application:primary_db2sdin1_900-rs:brazil
  '- Offline IBM.Application:primary_db2sdin1_900-rs:serbia
Online IBM.Equivalency:ca_db2sdin1_0-rg_group-equ
  |- Online IBM.PeerNode:brazil:brazil
  '- Online IBM.PeerNode:serbia:serbia
Online IBM.Equivalency:cacontrol_db2sdin1_equ
  |- Online IBM.Application:cacontrol_db2sdin1_128_brazil:brazil
  '- Online IBM.Application:cacontrol_db2sdin1_129_serbia:serbia
Online IBM.Equivalency:db2_db2sdin1_0-rg_group-equ
  |- Online IBM.PeerNode:algeria:algeria
  '- Online IBM.PeerNode:chile:chile
Online IBM.Equivalency:db2_db2sdin1_1-rg_group-equ
  |- Online IBM.PeerNode:chile:chile
  '- Online IBM.PeerNode:algeria:algeria
Online IBM.Equivalency:db2_private_network_db2sdin1_0
  |- Online IBM.NetworkInterface:ib0:algeria
  |- Online IBM.NetworkInterface:ib0:chile
  |- Online IBM.NetworkInterface:ib0:brazil
  '- Online IBM.NetworkInterface:ib0:serbia
Online IBM.Equivalency:db2_public_network_db2sdin1_0
  |- Online IBM.NetworkInterface:en0:algeria
  '- Online IBM.NetworkInterface:en0:chile
Online IBM.Equivalency:db2mnt-db2sd_20100719132932-rg_group-equ
  |- Online IBM.PeerNode:serbia:serbia
  |- Online IBM.PeerNode:brazil:brazil
  |- Online IBM.PeerNode:algeria:algeria
  '- Online IBM.PeerNode:chile:chile
Online IBM.Equivalency:idle_db2sdin1_997_algeria-rg_group-equ
  '- Online IBM.PeerNode:algeria:algeria
Online IBM.Equivalency:idle_db2sdin1_997_chile-rg_group-equ
  '- Online IBM.PeerNode:chile:chile
Online IBM.Equivalency:idle_db2sdin1_998_algeria-rg_group-equ
  '- Online IBM.PeerNode:algeria:algeria
Online IBM.Equivalency:idle_db2sdin1_998_chile-rg_group-equ
  '- Online IBM.PeerNode:chile:chile
Online IBM.Equivalency:idle_db2sdin1_999_algeria-rg_group-equ
  '- Online IBM.PeerNode:algeria:algeria
Online IBM.Equivalency:idle_db2sdin1_999_chile-rg_group-equ
  '- Online IBM.PeerNode:chile:chile
Online IBM.Equivalency:instancehost_db2sdin1-equ
  |- Online IBM.Application:instancehost_db2sdin1_brazil:brazil
  |- Online IBM.Application:instancehost_db2sdin1_chile:chile
  |- Online IBM.Application:instancehost_db2sdin1_algeria:algeria
  '- Online IBM.Application:instancehost_db2sdin1_serbia:serbia
Online IBM.Equivalency:primary_db2sdin1_900-rg_group-equ
  |- Online IBM.PeerNode:serbia:serbia
  '- Online IBM.PeerNode:brazil:brazil
[root on algeria][/] =>

```

3.7 Multi-cluster configuration

This scenario describes how to set up a GPFS multi-cluster configuration consisting of two separate local GPFS clusters. The local InfiniBand cluster consists of pSeries LPARs running AIX 6.1; the local Gigabit Ethernet cluster consists of System x nodes. We mount the GPFS file systems that are owned by the local InfiniBand cluster over a wide area network on the nodes of the GigE GPFS cluster. This scenario illustrates the steps to cross-mount GPFS file systems and create a GPFS multi-cluster environment.

3.7.1 Requirements: Hardware, software, network, storage

The hardware requirements are as follows:

- ▶ x86_64 Nodes
- ▶ pLPAR

The software requirements are as follows:

- ▶ AIX
- ▶ Red Hat Enterprise Linux 5.5
- ▶ GPFS 3.4

The network requirements are as follows:

- ▶ InfiniBand LAN
- ▶ GigE LAN
- ▶ GigE simulated WAN

SAN storage is used.

3.7.2 GPFS configuration

This multi-cluster scenario simulates two local GPFS clusters, one located in the Bronx and the other in Brooklyn. See Table 3-4 and Table 3-5.

Table 3-4 Bronx GPFS cluster: Node names, admin IP, and GPFS IP addresses

| Node name | Admin IP | GPFS IP |
|-----------|-----------------|-----------------|
| algeria | 192.168.101.101 | 192.168.101.101 |
| brazil | 192.168.101.102 | 192.168.101.102 |
| chile | 192.168.101.111 | 192.168.101.111 |
| serbia | 192.168.101.112 | 192.168.101.112 |

Table 3-5 Brooklyn GPFS cluster: Node names, admin IP, and GPFS IP addresses

| Node name | Admin IP | GPFS IP |
|----------------|--------------|--------------|
| spain-gpfs | 10.11.12.103 | 10.11.12.103 |
| greece-gpfs | 10.11.12.113 | 10.11.12.113 |
| nigeria-gpfs | 10.11.12.225 | 10.11.12.225 |
| australia-gpfs | 10.11.12.152 | 10.11.12.152 |

3.7.3 GPFS multi-cluster diagram

Figure 3-38 shows a diagram of the multi-cluster.

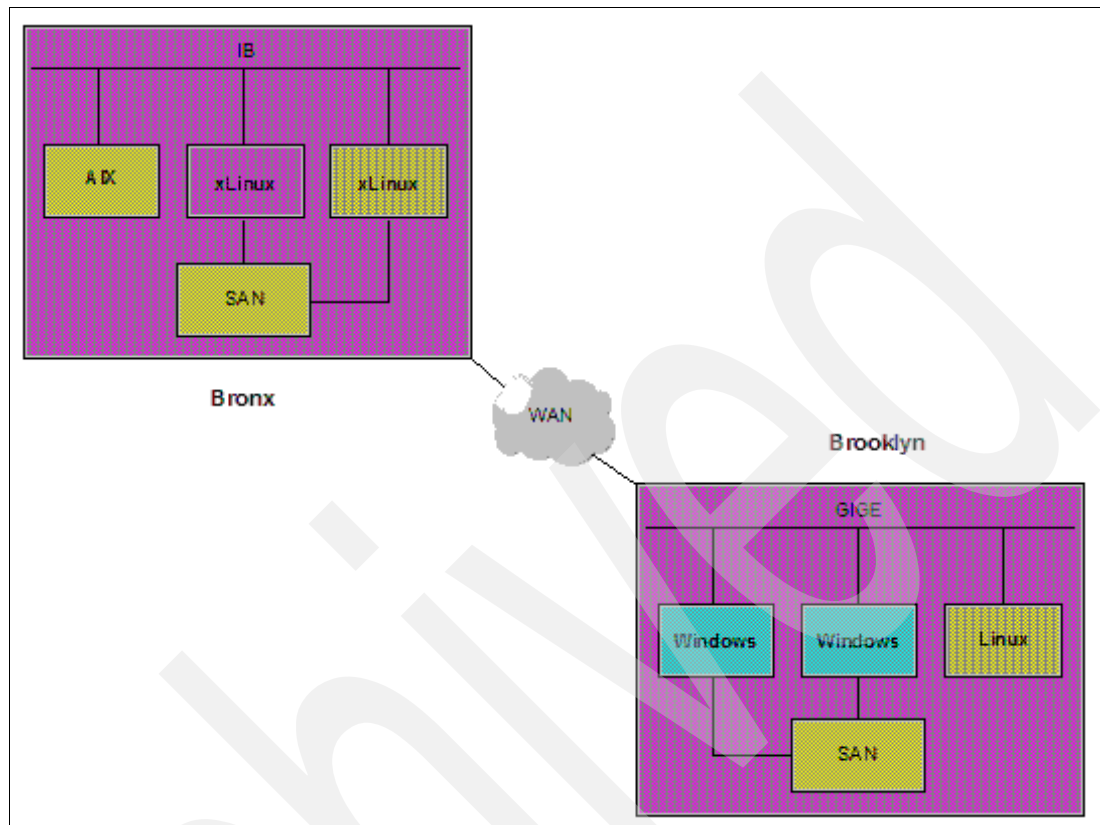


Figure 3-38 Multi-cluster diagram

3.7.4 Multi-cluster setup and configuration

Create two separate GPFS clusters with NSD and file systems as described in this section. Verify the current version of Secure Sockets Layer (SSL) is installed on the nodes of both clusters.

Note: The GPFS daemon handles all inter-cluster communication with the SSL.

Set up passwordless SSH communications between the nodes of both clusters.

The file systems that are owned by the InfiniBand cluster will be mounted on the nodes of the GigE cluster for processing data.

3.7.5 Multi-cluster scenario

The multi-cluster scenario describes a procedure to set up remote file system access on the *Brooklyn cluster* to file systems that are owned and managed by the *Bronx cluster*. The process involves the generation and exchange of SSL authorization keys between the two clusters. Next, the administrator of the GPFS cluster that owns the file system must authorize the remote clusters that want to access the file system. Then, the administrator of the GPFS

cluster that will mount or access the remote file system must define to GPFS the remote cluster and the remote file systems whose access is desired.

We must check the level of GPFS code on both cluster prior to setting up the cross-mount environment to determine which clusters are affected by cross-mounting restrictions.

Note: GPFS 3.4.0.0 format file systems owned by the Brooklyn cluster are not mountable on the Bronx cluster.

The steps are as follows:

1. Show the cluster definition for the local the Bronx cluster. Use the `mm1scluster` command to show the cluster definition and parameters. See Example 3-101.

Example 3-101 Show GPFS Bronx cluster definition and parameters

```
[root on algeria][/] => mm1scluster
```

GPFS cluster information

=====

```
GPFS cluster name:      db2cluster_20100719132846.algeria
GPFS cluster id:        13882457337115216718
GPFS UID domain:        db2cluster_20100719132846.algeria
Remote shell command:   /usr/bin/ssh
Remote file copy command: /usr/bin/scp
```

GPFS cluster configuration servers:

```
Primary server:  algeria
Secondary server: brazil
```

| Node | Daemon node name | IP address | Admin node name | Designation |
|------|------------------|-----------------|-----------------|----------------|
| 1 | algeria | 192.168.101.101 | algeria | quorum-manager |
| 2 | brazil | 192.168.101.102 | brazil | quorum-manager |
| 3 | chile | 192.168.101.111 | chile | quorum-manager |
| 4 | serbia | 192.168.101.112 | serbia | quorum-manager |

```
[root on algeria][/] =>
```

```
[root on algeria][/] => mm1sconfig
```

Configuration data for cluster db2cluster_20100719132846.algeria:

```
clusterName db2cluster_20100719132846.algeria
clusterId 13882457337115216718
autoload yes
minReleaseLevel 3.3.0.2
dmapiFileHandleSize 32
maxFilesToCache 10000
pagepool 256M
verifyGpfsReady yes
assertOnStructureError yes
sharedMemLimit 2047M
failureDetectionTime 35
leaseRecoveryWait 35
tiebreakerDisks gpfs1nsd
adminMode allToAll
```

File systems in cluster db2cluster_20100719132846.algeria:

```
/dev/db2fs1
[root on algeria][/] =>
```

```
[root on algeria][/] => df
Filesystem      512-blocks    Free %Used    Iused %Iused Mounted on
/dev/hd4         655360      269176   59%     13227   30% /
/dev/hd2        5308416     120608   98%     42961   69% /usr
/dev/hd9var      524288       91688   83%      6768   38% /var
/dev/hd3        13762560    6129384   56%      1496    1% /tmp
/dev/hd1        2031616    1278544   38%       386    1% /home
/dev/hd11admin   262144      261416    1%         5    1% /admin
/proc            -           -     -         -     - /proc
/dev/hd10opt     8388608    1652768   81%     18155    9% /opt
/dev/livedump    524288      523552    1%         4    1% /var/adm/ras/livedump
/dev/db2fs1     52428800   32622592   38%      4590    2% /db2sd_20100719132932
[root on algeria][/] =>
```

2. Show the cluster definition for local Brooklyn cluster. Use the **mm1scluster** command to show cluster definition and parameters. See Example 3-102.

Example 3-102 Show GPFS Brooklyn cluster definition and parameters

```
[root@spain ~]# mm1scluster
```

GPFS cluster information

```
=====
```

```
GPFS cluster name:      GPFS-InfiniBand.spain-gpfs
GPFS cluster id:        723685802921743777
GPFS UID domain:        GPFS-InfiniBand.spain-gpfs
Remote shell command:    /usr/bin/ssh
Remote file copy command: /usr/bin/scp
```

GPFS cluster configuration servers:

```
-----
```

```
Primary server:    spain-gpfs
Secondary server:  greece-gpfs
```

| Node | Daemon node name | IP address | Admin node name | Designation |
|------|------------------|--------------|-----------------|----------------|
| 1 | spain-gpfs | 10.11.12.103 | spain-gpfs | quorum-manager |
| 2 | greece-gpfs | 10.11.12.113 | greece-gpfs | quorum-manager |
| 3 | nigeria-gpfs | 10.11.12.225 | nigeria-gpfs | |
| 4 | australia-gpfs | 10.11.12.152 | australia-gpfs | |

```
[root@spain ~]#
```

```
[root@spain ~]# mm1sconfig
```

```
Configuration data for cluster GPFS-InfiniBand.spain-gpfs:
```

```
-----
```

```
clusterName GPFS-InfiniBand.spain-gpfs
clusterId 723685802921743777
autoload no
minReleaseLevel 3.4.0.0
dmapiFileHandleSize 32
verbsRdma enable
[spain-gpfs,greece-gpfs,nigeria-gpfs]
verbsPorts ehca0/1
[common]
adminMode central
```

File systems in cluster GPFS-InfiniBand.spain-gpfs:

```

-----
/dev/gpfs-ib
[root@spain ~]#
[root@spain ~]# df
Filesystem                1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
10188948    3923780    5739244    41% /
/dev/sda2          101082      17840      78023    19% /boot
tmpfs              4296128         0    4296128     0% /dev/shm
192.168.100.41:/nimrepo/bundle
53084160    37312768    15771392    71% /nim
/dev/gpfs-ib      104857600    183808    104673792     1% /gpfs-ib
[root@spain ~]#

```

3. Shut down the GPFS daemons on both clusters. Run the **mmshutdown -a** command on each local cluster to shut down the GPFS daemons so that a GPFS multi-cluster can be configured. See Example 3-103.

Example 3-103 Use mmchutdown -a to shut down to all GPFS daemons on the Bronx cluster

```

[root@spain ~]# mmshutdown -a
Wed Jul 28 06:47:40 EDT 2010: mmshutdown: Starting force unmount of GPFS file systems
Wed Jul 28 06:47:45 EDT 2010: mmshutdown: Shutting down GPFS daemons
australia-gpfs: Shutting down!
spain-gpfs: Shutting down!
greece-gpfs: Shutting down!
nigeria-gpfs: Shutting down!
australia-gpfs: 'shutdown' command about to kill process 4524
australia-gpfs: Unloading modules from /lib/modules/2.6.18-194.el5/extra
australia-gpfs: Unloading module mmfs26
spain-gpfs: 'shutdown' command about to kill process 32103
spain-gpfs: Unloading modules from /lib/modules/2.6.18-194.el5/extra
spain-gpfs: Unloading module mmfs26
australia-gpfs: Unloading module mmfslinux
australia-gpfs: Unloading module tracedev
greece-gpfs: 'shutdown' command about to kill process 20520
greece-gpfs: Unloading modules from /lib/modules/2.6.18-194.el5/extra
greece-gpfs: Unloading module mmfs26
spain-gpfs: Unloading module mmfslinux
spain-gpfs: Unloading module tracedev
nigeria-gpfs: 'shutdown' command about to kill process 17217
nigeria-gpfs: Unloading modules from /lib/modules/2.6.18-194.el5/extra
nigeria-gpfs: Unloading module mmfs26
greece-gpfs: Unloading module mmfslinux
greece-gpfs: Unloading module tracedev
nigeria-gpfs: Unloading module mmfslinux
nigeria-gpfs: Unloading module tracedev
australia-gpfs: mmfsenv: Module mmfslinux is still in use.
australia-gpfs: Unmount all GPFS file systems and issue the command:
australia-gpfs: mmfsadm cleanup
australia-gpfs: ERROR: Module mmfslinux is in use
australia-gpfs: mmfsenv: Error unloading module mmfslinux.
australia-gpfs: ERROR: Module tracedev is in use by mmfslinux
australia-gpfs: mmfsenv: Error unloading module tracedev.
Wed Jul 28 06:47:53 EDT 2010: mmshutdown: Finished
[root@spain ~]#

```

4. Generate a secure key (pair) for communications between clusters. The administrator must generate a pair of SSL keys (public/private) for secure communication between the

two clusters. Issue the **mmauth genkey new** command from any node in the Brooklyn cluster. See Example 3-104.

Note: The **mmauth** command is used to manage secure access to GPFS file systems between clusters.

Example 3-104 Generate ssl key on the Brooklyn cluster

```
[root@spain ~]# mmauth genkey new
Generating RSA private key, 512 bit long modulus
.....+++++++
.....+++++++
e is 65537 (0x10001)
writing RSA key
mmauth: Command successfully completed
mmauth: Propagating the cluster configuration data to all affected nodes.
[root@spain ~]#
```

5. Verify SSL keys are created properly. After the **mmauth genkey new** command completes successfully, the SSL keys (private/public) are stored at the `/var/mmfs/ssl` location on each node of the cluster. See Example 3-105.

Example 3-105 Show ssl keys are in the proper location on Brooklyn cluster

```
[root@spain ~]# ls -ltr /var/mmfs/ssl
total 48
-rw-r--r-- 1 root root 198 Jul 28 06:52 openssl.conf
lrwxrwxrwx 1 root root 34 Jul 28 06:52 id_rsa.pub ->
/var/mmfs/ssl/id_rsa_committed.pub
-rw-r--r-- 1 root root 869 Jul 28 06:52 id_rsa_committed.pub
-rw-r--r-- 1 root root 489 Jul 28 06:52 id_rsa_committed.cert
lrwxrwxrwx 1 root root 7 Jul 28 06:52 id_rsa_committed -> id_rsa1
-rw----- 1 root root 497 Jul 28 06:52 id_rsa1
drwx----- 2 root root 4096 Jul 28 06:52 stage
[root@spain ~]#
```

6. Add security to the GPFS communication network. Issue the **mmchconfig cipherList=AUTHONLY** command on any node of the Brooklyn cluster to enable authorization for network connections. See Example 3-106.

Notes:

- ▶ If **cipherList=DEFAULT** is specified, GPFS does not authenticate or check authorization for network connections.
- ▶ If **cipherList=AUTHONLY** is specified, GPFS does authentication and check authorization for network connections.

Example 3-106 Enable authorization for network connections

```
[root@spain ~]# mmchconfig cipherList=AUTHONLY
Verifying GPFS is stopped on all nodes ...
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root@spain ~]#
```

7. Copy Brooklyn SSL keys to a location accessible by both clusters. Use the `scp` command to copy the public key file to a NFS file system that mounted on the nodes of both the Bronx and Brooklyn clusters. See Example 3-107.

Example 3-107 copy ssl keys to NFS mounted file system accessible by booth clusters

```
[root on algeria][SSL_keys] => scp -p spain:/var/mmfs/ssl/id_*

[root on algeria][SSL_keys/spain_keys] => ls -ltr
total 40
-rw-r--r-- 1 root system 869 Jul 28 06:52 id_rsa_committed.pub
-rw-r--r-- 1 root system 489 Jul 28 06:52 id_rsa_committed.cert
-rw----- 1 root system 497 Jul 28 06:52 id_rsa_committed
-rw----- 1 root system 497 Jul 28 06:52 id_rsa1
-rw-r--r-- 1 root system 869 Jul 28 06:52 id_rsa.pub
[root on algeria][SSL_keys/spain_keys] =>
```

8. Generate secure communication (SSL) keys on the Bronx cluster. See Example 3-108.

Example 3-108 Generate SSL keys on the Bronx cluster

```
[root on algeria][SSL_keys/spain_keys] => mmauth genkey new
Generating RSA private key, 512 bit long modulus
.....+++++++
.....+++++++
e is 65537 (0x10001)
writing RSA key
mmauth: Command successfully completed
mmauth: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root on algeria][SSL_keys/spain_keys] =>
```

9. Verify that the SSL keys are created correctly on Bronx cluster. See Example 3-109.

Example 3-109 Check for SSL keys on the nodes of the Bronx cluster

```
[root on algeria][SSL_keys/spain_keys] => ls -ltr /var/mmfs/ssl
total 40
-rw-r--r-- 1 root system 205 Jul 28 08:05 openssl.conf
-rw-r--r-- 1 root system 898 Jul 28 08:06 id_rsa_committed.pub
-rw-r--r-- 1 root system 509 Jul 28 08:06 id_rsa_committed.cert
lrwxrwxrwx 1 root system 7 Jul 28 08:06 id_rsa_committed -> id_rsa1
-rw----- 1 root system 497 Jul 28 08:06 id_rsa1
lrwxrwxrwx 1 root system 34 Jul 28 08:06 id_rsa.pub ->
/var/mmfs/ssl/id_rsa_committed.pub
drwx----- 2 root system 256 Jul 28 08:06 stage
-rw-r--r-- 1 root system 383 Jul 28 08:56 authorized_keys
[root on algeria][SSL_keys/spain_keys] =>
```

3.7.6 Add security at the GPFS communication network on Bronx cluster

This section demonstrates how to set up security with the GPFS communication network of the bronx cluster.

The steps are as follows:

1. Start configuring security for the Bronx cluster. Use the `mmchconfig cipherList=AUTHONLY` command to set up security on the GPFS network. See Example 3-110.

Example 3-110 Add security to the network on the Bronx cluster with the cipherList

```
[root on algeria][ /SSL_keys/spain_keys] => mmchconfig cipherList=AUTHONLY
Verifying GPFS is stopped on all nodes ...
```

```
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
              affected nodes. This is an asynchronous process.
[root on algeria][ /SSL_keys/spain_keys] =>
[root on algeria][ /SSL_keys/spain_keys] =>
```

2. Copy SSL keys from the Brooklyn cluster to the shared NFS file system. See Example 3-111.

Example 3-111 Copy SSL keys from Bronx cluster to NFS mounted file system, shared by both

```
[root@spain algeria_keys]# scp -p algeria:/var/mmfs/ssl/id_* .

[root@spain algeria_keys]# ls -ltr
total 40
-rw-r--r-- 1 root root 898 Jul 28 08:06 id_rsa.pub
-rw-r--r-- 1 root root 898 Jul 28 08:06 id_rsa_committed.pub
-rw-r--r-- 1 root root 509 Jul 28 08:06 id_rsa_committed.cert
-rw----- 1 root root 497 Jul 28 08:06 id_rsa_committed
-rw----- 1 root root 497 Jul 28 08:06 id_rsa1
[root@spain algeria_keys]#
```

3. Change Bronx cluster configuration to specify openssllibname. For GPFS initialization to complete without errors, specify the PATH to the openssllibraries (library) with the **mmchconfig** command. See Example 3-112.

Example 3-112 Define PATH to openssllibraries for clean GPFS initialization

```
[root on brazil][ /] => /libssl.a(libssl64.so.0.9.8)" <
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
              affected nodes. This is an asynchronous process.
[root on brazil][ /] => mmlsconfig
Configuration data for cluster db2cluster_20100719132846.algeria:
-----
clusterName db2cluster_20100719132846.algeria
clusterId 13882457337115216718
autoload yes
minReleaseLevel 3.3.0.2
dmapiFileHandleSize 32
maxFilesToCache 10000
pagepool 256M
verifyGpfsReady yes
assertOnStructureError yes
sharedMemLimit 2047M
failureDetectionTime 35
leaseRecoveryWait 35
tiebreakerDisks gpfs1nsd
cipherList AUTHONLY
openssllibname /usr/lib/libssl.a(libssl64.so.0.9.8)
adminMode allToAll
```

```
File systems in cluster db2cluster_20100719132846.algeria:
-----
/dev/db2fs1
[root on brazil][/] =>
```

4. Start GPFS daemon of both clusters. Use the **mmstartup -a** command on one node of each cluster to start all GPFS daemons on that cluster. See Example 3-113.

Example 3-113 Restart GPFS daemons on both clusters

```
[root@spain algeria_keys]# mmstartup -a
Wed Jul 28 09:11:47 EDT 2010: mmstartup: Starting GPFS ...
[root@spain algeria_keys]#

[root on algeria][SSL_keys/spain_keys] => mmstartup -a
Wed Jul 28 09:12:08 EDT 2010: mmstartup: Starting GPFS ...
[root on algeria][SSL_keys/spain_keys] =>
```

3.7.7 Verify GPFS daemon is active on all nodes of both clusters

This section shows how to verify if the GPFS daemon is active in all the nodes of both clusters.

The steps are as follows:

1. Check the status of all GPFS daemons. Use the **mmgetstate -aLs** command to check the status of all GPFS daemons on a specific cluster. See Example 3-114.

Example 3-114 Use mmgetstate to verify the GPFS daemons are active on all nodes

```
[root@spain algeria_keys]# mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|----------------|--------|----------|-------------|------------|-------------|
| 1 | spain-gpfs | 2 | 2 | 4 | active | quorum node |
| 2 | greece-gpfs | 2 | 2 | 4 | active | quorum node |
| 3 | nigeria-gpfs | 2 | 2 | 4 | active | |
| 4 | australia-gpfs | 2 | 2 | 4 | active | |

```

Summary information
-----
Number of nodes defined in the cluster:      4
Number of local nodes active in the cluster: 4
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 2
Number of quorum nodes active in the cluster: 2
Quorum = 2, Quorum achieved

[root@spain algeria_keys]#

[root@spain algeria_keys]# mmlsmount all -L

File system gpfs-ib is mounted on 4 nodes:
10.11.12.103    spain-gpfs
10.11.12.113    greece-gpfs
10.11.12.225    nigeria-gpfs
10.11.12.152    australia-gpfs
[root@spain algeria_keys]#
```

```
[root on algeria][SSL_keys/spain_keys] => mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|-----------|--------|----------|-------------|------------|-------------|
| 1 | algeria | 1* | 4 | 4 | active | quorum node |
| 2 | brazil | 1* | 4 | 4 | active | quorum node |
| 3 | chile | 1* | 4 | 4 | active | quorum node |
| 4 | serbia | 1* | 4 | 4 | active | quorum node |

Summary information

```
-----
Number of nodes defined in the cluster:      4
Number of local nodes active in the cluster: 4
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 4
Number of quorum nodes active in the cluster: 4
Quorum = 1*, Quorum achieved
```

```
[root on algeria][SSL_keys/spain_keys] =>
```

```
[root on algeria][SSL_keys/spain_keys] => mmismount all -L
```

File system db2fs1 is mounted on 4 nodes:

```
192.168.101.101 algeria
192.168.101.102 brazil
192.168.101.111 chile
192.168.101.112 serbia
```

```
[root on algeria][SSL_keys/spain_keys] =>
```

This portion of this scenario is intended to show the error messages you might get when you try to mount a higher level formatted GPFS file system on a cluster with a lower code level.

2. Authorize the Bronx cluster access to the Brooklyn file systems.

On the Brooklyn cluster, issue the **mmauth add** command. This command authorizes the Bronx cluster to mount file systems that are owned by the Brooklyn cluster by using the key file received from the administrator of the Bronx cluster to grant secure access to the file systems the Brooklyn cluster owns. See Example 3-115.

Example 3-115 Authorize Bronx cluster to access the file systems owned by the Brooklyn cluster

```
[root@spain algeria_keys]# mmauth add db2cluster_20100719132846.algeria -k
id_rsa.pub
mmauth: Command successfully completed
mmauth: Propagating the cluster configuration data to all affected nodes.
[root@spain algeria_keys]#
```

3. Grant Bronx cluster access to Brooklyn cluster file systems. On the Brooklyn cluster, the system administrator issues the **mmauth grant** command to authorize the Bronx cluster to mount specific file systems owned by the Brooklyn cluster. See Example 3-116.

Example 3-116 Grant Bronx cluster access to file systems owned by Brooklyn cluster

```
[root@spain algeria_keys]# mmauth grant db2cluster_20100719132846.algeria -f /dev/gpfs-ib

mmauth: Granting cluster db2cluster_20100719132846.algeria access to file system gpfs-ib:
access type rw; root credentials will not be remapped.

mmauth: Command successfully completed
mmauth: Propagating the cluster configuration data to all affected nodes.
```



```
[root@spain algeria_keys]#
```

4. Add the remote cluster definition to the Bronx cluster. On the Bronx cluster, the system administrator must now define (identify) the remote cluster (cluster_name, contact nodes and public key) for the Brooklyn cluster. See Example 3-117.

Note: This step permits the cluster that wants to mount the file system a means to locate the serving cluster and ultimately mount its file systems.

Example 3-117 Add remote cluster definition to Bronx cluster

```
[root on algeria][ /SSL_keys/spain_keys] => mmremoteccluster update
GPFS-InfiniBand.spain-gpfs -n spain-gpfs,greece-gpfs,>
mmremoteccluster: Command successfully completed
mmremoteccluster: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root on algeria][ /SSL_keys/spain_keys] =>
```

5. Show all remote cluster definitions on Bronx cluster. Use the **mmremoteccluster show all** command. See Example 3-118.

Example 3-118 Display remote cluster definitions on Bronx cluster

```
[root on algeria][ /SSL_keys/spain_keys] => mmremoteccluster show all
Cluster name:  GPFS-InfiniBand.spain-gpfs
Contact nodes: spain
SHA digest:    ea82f38027c6c88bdfef1bfabd85d50a4e3e047da
File systems:  (none defined)

[root on algeria][ /SSL_keys/spain_keys] =>
```

6. Register the file systems from Brooklyn cluster to be mounted. On the Bronx cluster, the system administrator issues one or more **mmremotefs** commands to identify the file systems in the GigE that are to be accessed by nodes in cluster2. See Example 3-119.

Example 3-119 On Bronx cluster register the Brooklyn cluster file systems

```
[root on algeria][ /SSL_keys/spain_keys] => mmremotefs add gpfs-ib -f /dev/gpfs-ib -C
GPFS-InfiniBand.spain-gpfs -T /spain-gpfs-ib
mmremotefs: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root on algeria][ /SSL_keys/spain_keys] =>
```

7. Display all remote file systems known to the Bronx cluster. Use the **mmremotefs show all** command to display remote file systems. See Example 3-120.

Example 3-120 Display all remote file systems defined on the Bronx cluster

```
[root on algeria][ /SSL_keys/spain_keys] => mmremotefs show all
```

| Local Name | Remote Name | Cluster name | Mount Point | Mount Options | Automount | Drive |
|------------|-------------|----------------------------|----------------|---------------|-----------|-------|
| gpfs-ib | gpfs-ib | GPFS-InfiniBand.spain-gpfs | /spain-gpfs-ib | rw | no | |
| - | | | | | | |

8. Mount remote file system from the Brooklyn cluster. The **mount** command *is expected to fail* because the level of GPFS on the Bronx cluster is not compatible and is lower than the GPFS level on the Brooklyn cluster. See Example 3-121.

Example 3-121 On Bronx cluster mount remote file system owned Brooklyn cluster

```
[root on algeria][/SSL_keys/spain_keys] => mount /spain-gpfs-ib
Cannot mount /dev/gpfs-ib on /spain-gpfs-ib: Host is down
[root on algeria][/SSL_keys/spain_keys] =>
```

9. Authorize the Brooklyn cluster access to the Bronx file systems. On the Bronx cluster, the system administrator issues the **mmauth add** command to perform the following tasks (Example 3-122):
- Authorize the Brooklyn cluster to mount file systems that are owned by the Bronx cluster by using the key file that is received from the administrator of the Bronx cluster.
 - Grant secure access to the file systems it owns.

Example 3-122 Authorize Brooklyn cluster nodes to access file systems owned by Bronx cluster

```
[root on algeria][/SSL_keys/spain_keys] => mmauth update
GPFS-InfiniBand.spain-gpfs      -k id_rsa.pub
mmauth: Command successfully completed
mmauth: Propagating the cluster configuration data to all
      affected nodes. This is an asynchronous process.
[root on algeria][/SSL_keys/spain_keys] =>
```

10. Grant Brooklyn cluster access to the Bronx cluster file systems owned. See Example 3-123.

Example 3-123 Grant Brooklyn nodes access to the file systems owned by the Bronx cluster

```
[root on algeria][/SSL_keys/spain_keys] => mmauth grant
GPFS-InfiniBand.spain-gpfs  -f /dev/db2fs1

mmauth: Granting cluster GPFS-InfiniBand.spain-gpfs access to file system
db2fs1:
      access type rw; root credentials will not be remapped.

mmauth: Command successfully completed
mmauth: Propagating the cluster configuration data to all
      affected nodes. This is an asynchronous process.
[root on algeria][/SSL_keys/spain_keys] =>
```

11. Add the remote cluster definitions to the Brooklyn cluster. See Example 3-124.

Example 3-124 Add remote cluster definitions to the Brooklyn cluster

```
[root@spain algeria_keys]# mmremoteccluster add db2cluster_20100719132846.algeria -n
algeria,brazil,chile
mmremoteccluster: Command successfully completed
mmremoteccluster: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root@spain algeria_keys]#
```

12. Define remote cluster that owns the file systems. On the Brooklyn cluster, the system administrator now must define the cluster name, contact nodes, and public key for Bronx. See Example 3-125.

Example 3-125 Define remote cluster that owns the Bronx file systems

```
[root@spain algeria_keys]# mmremoteccluster update db2cluster_20100719132846.algeria -n
algeria,brazil,chile -k id_rsa.pub
mmremoteccluster: Command successfully completed
mmremoteccluster: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root@spain algeria_keys]#
```

13. Show all remote cluster definitions on the Brooklyn cluster. Use the `mmremoteccluster` command to display the definitions. See Example 3-126.

Example 3-126 Display all remote cluster definitions on the Brooklyn cluster

```
[root@spain algeria_keys]# mmremoteccluster show all
Cluster name: db2cluster_20100719132846.algeria
Contact nodes: algeria,brazil,chile
SHA digest: 0d46bb1204c4bca8107ab0c68b5adfc82c5c4d56
File systems: (none defined)

[root@spain algeria_keys]#
```

14. Register the file systems from Bronx cluster to be mounted. On the Brooklyn cluster, the system administrator must issue one or more `mmremotefs` commands to identify the file systems in the Bronx cluster that are to be accessed by nodes in the Brooklyn cluster. See Example 3-127.

Example 3-127 Add remote file systems to the Brooklyn cluster

```
[root@spain algeria_keys]# mmremotefs add /dev/IB-db2fs1 -f /dev/db2fs1 -C
db2cluster_20100719132846.algeria -T /IB-db2fs1
mmremotefs: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root@spain algeria_keys]#
```

15. Show all remote file systems defined on the Brooklyn cluster. Use the `mmremotefs` command to display the systems. See Example 3-128.

Example 3-128 Show all remote file systems defined to the Brooklyn cluster

```
[root@spain algeria_keys]# mmremotefs show all
```

| Local Name | Remote Name | Cluster name | Mount Point | Mount Options | Automount | Drive | Priority |
|------------|-------------|-----------------------------------|-------------|---------------|-----------|-------|----------|
| IB-db2fs1 | db2fs1 | db2cluster_20100719132846.algeria | /IB-db2fs1 | rw | | no | |
| - | 0 | | | | | | |

```
[root@spain algeria_keys]#
```

3.8 Disaster recovery using GPFS replication

This scenario shows a disaster recovery solution using GPFS synchronous metadata and data replication between two geographically separate sites. This solution does not rely on any specific disk subsystem support.

3.8.1 Requirements: Hardware, software, network, storage

The hardware requirements are as follows:

- ▶ AIX LPAR
- ▶ Linux LPAR

The software requirements are as follows:

- ▶ AIX
- ▶ Red Hat Enterprise Linux 5.5

The network is GigE.

SAN storage is used.

3.8.2 GPFS configuration

Table 3-6 and Table 3-7 show the configuration.

Table abbreviations:

- ▶ Poughkeepsie site is abbreviated to POK.
- ▶ Kingston site is abbreviated to KGN.
- ▶ Buffalo site (tiebreaker) is abbreviated to BUF.

Table 3-6 Site: Node names, OSI images, and IP addresses

| Site | Node name | OSI | Role | Mgmt IP | GPFS IP |
|------|--------------|-------------|-------------|-----------------|--------------|
| POK | mexico-gpfs | AIX 6.1 TL5 | quorum-CCDS | 192.168.101.121 | 10.11.12.121 |
| POK | japan-gpfs | AIX 6.1 TL5 | quorum | 192.168.101.122 | 10.11.12.122 |
| KGN | france-gpfs | AIX 6.1 TL5 | quorum-CCDS | 192.168.101.123 | 10.11.12.123 |
| KGN | ghana-gpfs | AIX 6.1 TL5 | quorum | 192.168.101.124 | 10.11.12.124 |
| BUF | england-gpfs | RH 5.5 | tiebreaker | 192.168.101.143 | 10.11.12.125 |

Table 3-7 Site: Node names, raw disks, and NSD names

| Site number | Site | Node name | Raw disk | NSD name |
|-------------|------|--------------|----------------------|----------|
| 1 | POK | mexico-gpfs | hdisk1,hdisk2,hdisk3 | mexico |
| | POK | japan-gpfs | hdisk1,hdisk2,hdisk3 | japan |
| 2 | KGN | france-gpfs | hdisk1,hdisk2,hdisk3 | france |
| | KGN | ghana-gpfs | hdisk1,hdisk2,hdisk3 | ghana |
| 3 | BUF | england-gpfs | hdisk1 | england |

The POK and BUF sites share the SAN storage and holds the primary and secondary configuration data servers respectively. The BUF site holds a (tiebreaker) quorum node and a disk for file system descriptor only.

Notes:

- ▶ All SAN-attached disks are accessible from all nodes at sites POK and KGN.
- ▶ The NSD definitions on the disk at site BUF are defined on an internal disk that is accessible only from the BUF site.
- ▶ The tiebreaker site (BUF) serves to maintain node and file system descriptor quorum after either site POK or KGN fails.

All disks at the POK site are assigned to a failure group and all disk at the KGN site are assigned to another failure group. The disk at the Buffalo site is assigned to a third failure group.

A single GPFS cluster is defined across two geographic sites (Poughkeepsie and Kingston).

A single replicated GPFS file system is created from disks at the three sites.

The `mmfsctl` command (Example 3-129) is used in this scenario to manage the disaster recovery cluster:

- ▶ For minority takeover in active-active replicated configurations
- ▶ To force GPFS to exclude the specified disks or failure groups from the file system descriptor quorum

Example 3-129 The mmfsctl command syntax

```
mmfsctl Device {exclude | include} {-d DiskList | -F DiskFile | -G FailureGroup}
```

3.8.3 GPFS configuration diagram

Figure 3-39 on page 164 shows a diagram of the configuration. Table 3-8 lists the meanings for the letter designations in the figure.

Table 3-8 Letter designation and meanings for Figure 3-39 on page 164

| Letter | Role |
|--------|-------------------------------------|
| Q | Quorum |
| NQ | Nonquorum |
| P | Primary configuration data server |
| S | Secondary configuration data server |

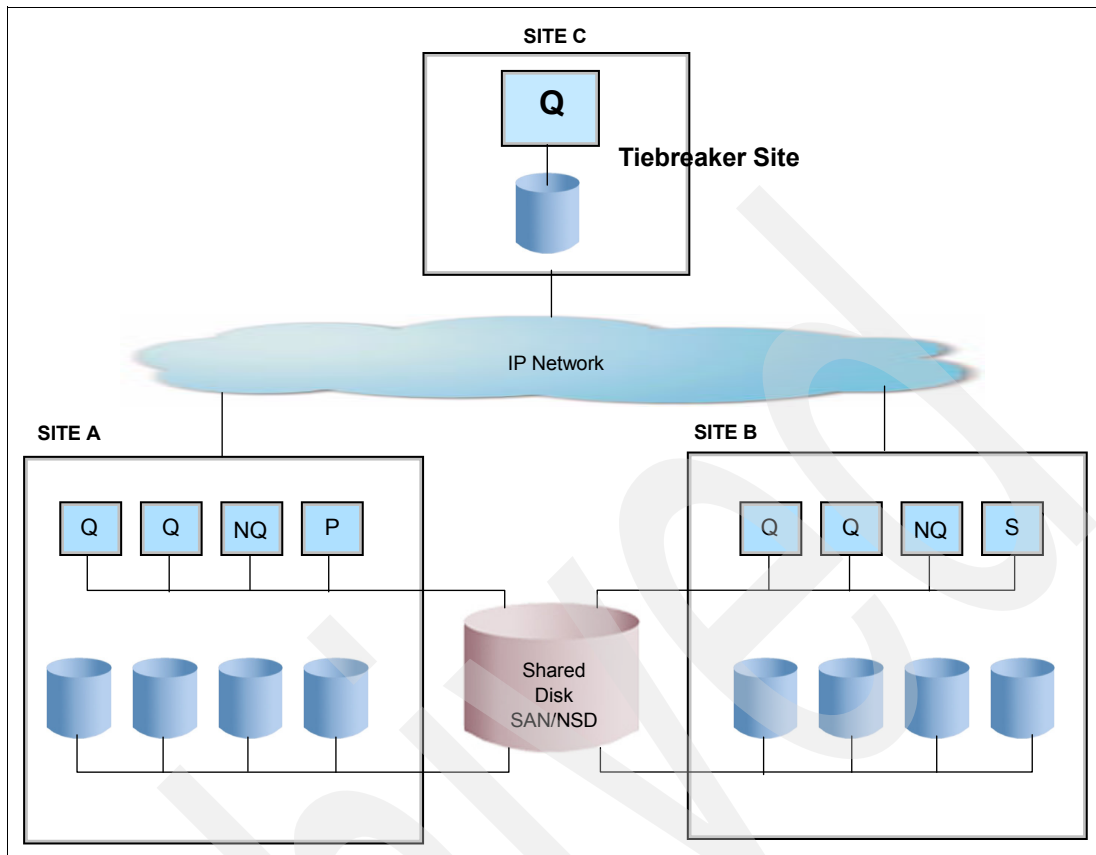


Figure 3-39 Three-site GPFS DR cluster and associated shared disks

3.8.4 Set up and configure GPFS DR cluster

The following section provides details about how to set up and configure GPFS disaster recovery cluster.

The steps are as follows:

1. Set up node description file for creating GPFS cluster. Create a node description file for defining the GPFS cluster across all three sites. See Example 3-130.

Example 3-130 Create a node description file for use by mmcrcluster command

```
mexico-gpfs:quorum-manager
japan-gpfs:quorum-manager
france-gpfs:quorum-manager
ghana-gpfs:quorum-manager
england-gpfs:quorum-client
```

Note: You must define the node on BUF site as a quorum and client, not a manager

2. Create the GPFS cluster. Issue the **mmcrcluster** command with the node description file (in Example 3-130) to create the GPFS cluster. See Example 3-131 on page 165.

Example 3-131 Use mmcrcluster with the previous input file to create the GPFS cluster

```
[root on mexico] [/tmp] => cat mmcrcluster.sh

/usr/lpp/mmfs/bin/mmcrcluster -N /tmp/Node_Desc_File1 -p mexico -s france -r /bin/ssh -R
/bin/scp
[root on mexico] [/tmp] =>

[root on mexico] [/tmp] => mmcrcluster.sh
Tue Jul 13 15:08:53 EDT 2010: mmcrcluster: Processing node mexico
Tue Jul 13 15:08:53 EDT 2010: mmcrcluster: Processing node japan
Tue Jul 13 15:08:55 EDT 2010: mmcrcluster: Processing node france
Tue Jul 13 15:08:57 EDT 2010: mmcrcluster: Processing node ghana
Tue Jul 13 15:08:58 EDT 2010: mmcrcluster: Processing node england
mmcrcluster: Command successfully completed
mmcrcluster: Warning: Not all nodes have proper GPFS license designations.
    Use the mmchlicense command to designate licenses as needed.
mmcrcluster: Propagating the cluster configuration data to all
    affected nodes. This is an asynchronous process.
[root on mexico] [/tmp] =>
```

3. List the node of the GPFS cluster. Display the GPFS cluster definition with the **mmfslcluster** command. See Example 3-132.

Example 3-132 Use mmfslcluster command to display details of the cluster

```
[root on mexico] [/tmp] => mmfslcluster
```

```
=====
Warning:
  This cluster contains nodes that do not have a proper GPFS license
  designation. This violates the terms of the GPFS licensing agreement.
  Use the mmchlicense command and assign the appropriate GPFS licenses
  to each of the nodes in the cluster. For more information about GPFS
  license designation, see the Concepts, Planning, and Installation Guide.
=====
```

GPFS cluster information

=====

```
GPFS cluster name:      mexico
GPFS cluster id:        13882458522525678021
GPFS UID domain:        mexico
Remote shell command:   /bin/ssh
Remote file copy command: /bin/scp
```

GPFS cluster configuration servers:

```
Primary server:  mexico
Secondary server: france
```

| Node | Daemon node name | IP address | Admin node name | Designation |
|------|------------------|-----------------|-----------------|----------------|
| 1 | mexico | 192.168.102.121 | mexico | quorum-manager |
| 2 | japan | 192.168.102.122 | japan | quorum-manager |
| 3 | france | 192.168.102.123 | france | quorum-manager |
| 4 | ghana | 192.168.102.124 | ghana | quorum-manager |
| 5 | england | 192.168.101.143 | england | quorum |

```
[root on mexico] [/tmp] =>
```

4. List GPFS server licenses on the cluster. Show server licenses of all nodes in the cluster by using the `mmllslicense` command. See Example 3-133.

Example 3-133 Use mmllslicense to display licenses defined for each node in the cluster

```
[root on mexico][ /tmp] => mmllslicense -L
```

| Node name | Required license | Designated license |
|-----------|------------------|--------------------|
| mexico | server | none * |
| japan | server | none * |
| france | server | none * |
| ghana | server | none * |
| england | server | none * |

```
Summary information
-----
Number of nodes defined in the cluster: 5
Number of nodes with server license designation: 0
Number of nodes with client license designation: 0
Number of nodes still requiring server license designation: 5
Number of nodes still requiring client license designation: 0
[root on mexico][ /tmp] =>
```

5. Change the server licenses to the nodes in the cluster. See Example 3-134.

Example 3-134 Use mmchlicense to assign server licenses

```
[root on mexico][ /tmp] => mmchlicense server --accept -N all
```

The following nodes will be designated as possessing GPFS server licenses:

```
mexico
japan
france
ghana
england
```

```
mmchlicense: Command successfully completed
mmchlicense: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root on mexico][ /tmp] =>
```

6. Check the server licenses of all nodes in the cluster. See Example 3-135.

Example 3-135 Use mmllslicense to display the current licenses of all nodes in the cluster

```
[root on mexico][ /tmp] => mmllslicense -L
```

| Node name | Required license | Designated license |
|-----------|------------------|--------------------|
| mexico | server | server |
| japan | server | server |
| france | server | server |
| ghana | server | server |
| england | server | server |

```
Summary information
-----
Number of nodes defined in the cluster: 5
Number of nodes with server license designation: 5
```

```
Number of nodes with client license designation:      0
Number of nodes still requiring server license designation:  0
Number of nodes still requiring client license designation:  0
[root on mexico][/]tmp =>
```

7. Set up GPFS cluster configuration parameters by using the **mmchconfig** command. See Example 3-136.

Example 3-136 Use mmchconfig command to set up specific GPFS cluster parameters

```
[root on mexico][/] => mmchconfig maxblocksize=4M
Verifying GPFS is stopped on all nodes ...
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root on mexico][/] =>
```

```
[root on mexico][/] => mmlsconfig
Configuration data for cluster mexico:
```

```
-----
clusterName mexico
clusterId 13882458522525678021
autoload yes
minReleaseLevel 3.4.0.0
dmapifileHandleSize 32
[england-gpfs]
unmountOnDiskFail yes
[common]
maxblocksize 4M
adminMode central
```

```
File systems in cluster mexico:
```

```
-----
/dev/gpfs1
[root on mexico][/] =>
```

8. Define NSDs for the GPFS cluster. Set up an NSD descriptor file to contain disks from all three sites (POK, KGN and BUF). Disks at sites POK and KGN are assigned to failure group 3001 and 3002 respectively. The single disk at the BUF site is assigned to failure group 3003. See Example 3-137.

Example 3-137 Create an NSD descriptor file to be used as input to the mmcrnsd command

```
hdisk1:mexico-gpfs:japan-gpfs:dataAndMetadata:3001
hdisk2:japan-gpfs:mexico-gpfs:dataAndMetadata:3001
hdisk3:mexico-gpfs:japan-gpfs:dataAndMetadata:3001
hdisk4:france-gpfs:ghana-gpfs:dataAndMetadata:3002
hdisk5:ghana-gpfs:france-gpfs:dataAndMetadata:3002
hdisk6:france-gpfs:ghana-gpfs:dataAndMetadata:3002
hdisk:england-gpfs::descOnly:3003
```

9. Create NSDs on the GPFS cluster. Use the **mmcrnsd** command to create NSDs. See Example 3-138 on page 168.

Example 3-138 Create NSDs for this cluster with the mmcrnsd command

```
[root on mexico][ /tmp] => mmcrnsd -F NSD_Desc_File1
mmcrnsd: Processing disk hdisk1
mmcrnsd: Processing disk hdisk2
mmcrnsd: Processing disk hdisk3
mmcrnsd: Processing disk hdisk4
mmcrnsd: Processing disk hdisk5
mmcrnsd: Processing disk hdisk6
mmcrnsd: Processing disk sdf
mmcrnsd: Propagating the cluster configuration data to all
        affected nodes. This is an asynchronous process.
[root on mexico][ /tmp] =>
```

10. Check NSD on the cluster. Display defined NSDs for this cluster by using the `mm1snsd` command. See Example 3-139.

Example 3-139 Use mmlsnsd to display all NSDs on the cluster

```
[root on mexico][ /tmp] => mmlsnsd
```

| File system | Disk name | NSD servers |
|-------------|-----------|------------------------|
| (free disk) | gpfs1nsd | mexico-gpfs,japan-gpfs |
| (free disk) | gpfs2nsd | japan-gpfs,mexico-gpfs |
| (free disk) | gpfs3nsd | mexico-gpfs,japan-gpfs |
| (free disk) | gpfs4nsd | france-gpfs,ghana-gpfs |
| (free disk) | gpfs5nsd | ghana-gpfs,france-gpfs |
| (free disk) | gpfs6nsd | france-gpfs,ghana-gpfs |
| (free disk) | gpfs7nsd | england-gpfs |

```
[root on mexico][ /tmp] =>
```

11. Create site-wide GPFS cluster. Define a GPFS cluster over three geographically separate sites. Two sites are production sites and the third site is a tiebreaker site. File systems will be mounted on all three nodes and will be accessed concurrently from both production sites. The disk subsystem is shared between the production sites and the disk on the tiebreaker site will be a local internal disk.
12. Create site-wide GPFS file system. Use the `mmcrfs` command to create the file system. See Example 3-140.

Example 3-140 Use mmcrfs to create site-wide file system

```
mmcrfs /gpfs/gpfsSH1 gpfs1 -F NSD_Desc_File4 -r 2 -R 2 -m 2 -M 2 -A yes
```

13. View the output from `mmcrfs` command. Example 3-141 shows the output from the `mmcrfs` command.

Example 3-141 Output from mmcrfs command

```
[root on mexico][ /tmp] => mmcrfs_gpfs1.sh
```

The following disks of gpfs1 will be formatted on node mexico:

```
gpfs1nsd: size 26214400 KB
gpfs2nsd: size 26214400 KB
gpfs3nsd: size 26214400 KB
gpfs4nsd: size 26214400 KB
gpfs5nsd: size 26214400 KB
```

```

gpfs6nsd: size 26214400 KB
gpfs7nsd: size 5242880 KB
Formatting file system ...
Disks up to size 409 GB can be added to storage pool 'system'.
Creating Inode File
Creating Allocation Maps
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool 'system'
Completed creation of file system /dev/gpfs1.
mmcrfs: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root on mexico][/tmp] =>

```

14. Verify the file system defined to the GPFS cluster. Use the **mm1sconfig** command to list the system. See Example 3-142.

Example 3-142 Use mm1sconfig to display GPFS file system on this cluster

```

[root on mexico][/tmp] => mm1sconfig
Configuration data for cluster mexico:
-----
clusterName mexico
clusterId 13882458522525678021
autoload no
minReleaseLevel 3.4.0.0
dmapiFileHandleSize 32
adminMode central

File systems in cluster mexico:
-----
/dev/gpfs1
[root on mexico][/tmp] =>

```

15. Mount all GPFS file systems on the cluster. Use the **mmmount** command to mount the systems. See Example 3-143.

Example 3-143 Use mmmount all -a to mount all defined file systems on all nodes of the cluster

```

[root on mexico][/tmp] => mmmount all -a
Thu Jul 15 18:22:15 EDT 2010: mmmount: Mounting file systems ...
[root on mexico][/tmp] => mm1smount all
File system gpfs1 is mounted on 5 nodes.
[root on mexico][/tmp] => mm1smount all -L

File system gpfs1 is mounted on 5 nodes:
10.11.12.221    mexico-gpfs
10.11.12.222    japan-gpfs
10.11.12.223    france-gpfs
10.11.12.224    ghana-gpfs
10.11.12.143    england-gpfs
[root on mexico][/tmp] =>

[root on mexico][/tmp] => df | grep gpfs
/dev/gpfs1      314572800 313642496    1%    4070    3% /gpfs/gpfsSH1
[root on mexico][/tmp] =>

```

16. Verify that the GPFS daemon is active on all cluster nodes. Use the **mmgetstate** command to get the status. See Example 3-144.

Example 3-144 Use mmgetstate to check the status of all GPFS daemons

```
[root@england ~]# mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|--------------|--------|----------|-------------|------------|-------------|
| 1 | mexico-gpfs | 3 | 5 | 5 | active | quorum node |
| 2 | japan-gpfs | 3 | 5 | 5 | active | quorum node |
| 3 | france-gpfs | 3 | 5 | 5 | active | quorum node |
| 4 | ghana-gpfs | 3 | 5 | 5 | active | quorum node |
| 5 | england-gpfs | 3 | 5 | 5 | active | quorum node |

Summary information

```
-----
Number of nodes defined in the cluster:      5
Number of local nodes active in the cluster:  5
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 5
Number of quorum nodes active in the cluster: 5
Quorum = 3, Quorum achieved
```

```
[root@england ~]#
```

17. Check the state of all disks in the file system. Verify that all disks that are defined to the GPFS file system are in the “up” availability state. See Example 3-145.

Example 3-145 Check the state of all disk in the file systems

```
[root@england ~]# mmlsdisk gpfs1
```

| disk name | driver type | sector size | failure group | holds metadata | holds data | status | availability | storage pool |
|-----------|-------------|-------------|---------------|----------------|------------|--------|--------------|--------------|
| gpfs1nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs2nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs3nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs4nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs5nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs6nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs7nsd | nsd | 512 | 3003 | no | no | ready | up | system |

```
[root@england ~]#g
```

18. Prevent false disk errors in the SAN configuration from being reported to the file system manager by enabling the **unmountOnDiskFail** option on the tiebreaker node. See Example 3-146.

Example 3-146 Set unmountOnDiskFail=yes with the mmchconfig command

```
[root@england ~]# mmchconfig unmountOnDiskFail=yes england-gpfs
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root@england ~]#
```

19. On each node of the production sites (KGN, POK), start a stress workload to run for an extended period of time.

Fail over after failure of KGN site

To simulate the complete failure of the KGN site, the GPFS daemon on both (france-gpfs and ghana-gpfs) are shut down. Furthermore, we take down the storage subsystem that is associated with NSD at that site (france-gpfs and ghana-gpfs).

Important: After a site failure, the following events occur:

1. GPFS detects and responds to the failure.
2. The failed nodes are marked as “down.”
3. The failed disks are marked as “down.”

All applications on the surviving site (POK) continue to run without errors.

The steps are as follows:

1. Shut down gpfs on france-gpfs and ghana-gpfs. From the command line, run the `/usr/lpp/mmfs/bin/mmshutdown` command on france-gpfs and ghana-gpfs. See Example 3-147.

Example 3-147 Shut down GPFS on france and ghana with the mmshutdown command

```
[root on france][ /tmp] => mmshutdown
Fri Jul 16 00:13:57 EDT 2010: mmshutdown: Starting force unmount of GPFS file systems
forced unmount of /gpfsSH1
Fri Jul 16 00:14:02 EDT 2010: mmshutdown: Shutting down GPFS daemons
Shutting down!
'shutdown' command about to kill process 7798924
Fri Jul 16 00:14:07 EDT 2010: mmshutdown: Finished
[root on france][ /tmp] =>

[root on ghana][ /tmp] => mmshutdown
Fri Jul 16 00:14:02 EDT 2010: mmshutdown: Starting force unmount of GPFS file systems
forced unmount of /gpfsSH1
Fri Jul 16 00:14:07 EDT 2010: mmshutdown: Shutting down GPFS daemons
Shutting down!
'shutdown' command about to kill process 8716302
Master did not clean up; attempting cleanup now
Fri Jul 16 00:15:08.002 2010: mmfsd64 is shutting down.
Fri Jul 16 00:15:08.003 2010: Reason for shutdown: mmfsadm shutdown command timed out
Fri Jul 16 00:15:08 EDT 2010: mmcommon mmfsdown invoked. Subsystem: mmfs Status: down
Fri Jul 16 00:15:08 EDT 2010: mmcommon: Unmounting file systems ...
Fri Jul 16 00:15:12 EDT 2010: mmshutdown: Finished
[root on ghana][ /tmp] =>
```

2. Verify the GPFS daemon is down on france-gpfs and ghana-gpfs. Use the `mmgetstate` command to verify the daemon. See Example 3-148.

Example 3-148 Verify GPFS daemon is down on france-gpfs and ghana-gpfs; use mmgetstate

```
[root on ghana][ /tmp] => mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|--------------|--------|----------|-------------|------------|-------------|
| 1 | mexico-gpfs | 3 | 3 | 5 | active | quorum node |
| 2 | japan-gpfs | 3 | 3 | 5 | active | quorum node |
| 3 | france-gpfs | 0 | 0 | 5 | down | quorum node |
| 4 | ghana-gpfs | 0 | 0 | 5 | down | quorum node |
| 5 | england-gpfs | 3 | 3 | 5 | active | quorum node |

Summary information

```

-----
Number of nodes defined in the cluster:      5
Number of local nodes active in the cluster: 3
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 5
Number of quorum nodes active in the cluster: 3
Quorum = 3, Quorum achieved

```

```
[root on ghana][ /tmp] =>
```

3. Verify that file systems are mounted on the surviving sites. After the KGN site fails, the GPFS file system (/gpfs) must remain mounted and accessible on all nodes of the POK site and the tiebreaker site. Use **mm1smount**. See Example 3-149.

Example 3-149 Verify /gpfs1 is mounted on all surviving nodes

```
[root on ghana][ /tmp] => mm1smount all -L
```

File system gpfs1 is mounted on 3 nodes:

```

10.11.12.221    mexico-gpfs
10.11.12.222    japan-gpfs
10.11.12.143    england-gpfs

```

```
[root on ghana][ /tmp] =>
```

4. Verify disks served by the KGN site are “down.” After the failure of the KGN site, the disk server at that site should be unavailable. Use **mm1sdisk** to verify. See Example 3-150.

Example 3-150 Disks server by nodes at site KGN are in down state

```
[root@england ~]# mm1sdisk gpfs1
```

| disk name | driver type | sector size | failure group | holds metadata | holds data | status | availability | storage pool |
|--------------|----------------|----------------|------------------|-------------------|---------------|--------|--------------|-----------------|
| gpfs1nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs2nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs3nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs4nsd | nsd | 512 | 3002 | yes | yes | ready | down | system |
| gpfs5nsd | nsd | 512 | 3002 | yes | yes | ready | down | system |
| gpfs6nsd | nsd | 512 | 3002 | yes | yes | ready | down | system |
| gpfs7nsd | nsd | 512 | 3003 | no | no | ready | up | system |

```
[root@england ~]#
```

5. Verify workload running on the POK site continues to run error-free. After the failure of the KGN site, the nodes and disk on the POK site remain available, and any work currently running on the nodes at the POK site continue to run uninterrupted. See Example 3-151.

Example 3-151 Workload running at POK site continues to run error-free

```

.02    /gpfsSH1/japan477/samples/uti1
0.77    /gpfsSH1/japan477/samples
130.69 /gpfsSH1/japan477
working on japan at Fri Jul 16 00:39:56 EDT 2010 copying /usr/lpp/mmfs/
du_stress.sh[13]: ech: not found
0.05    /gpfsSH1/japan478/READMEs
0.36    /gpfsSH1/japan478/bin/.links
49.70   /gpfsSH1/japan478/bin/aix32
52.53   /gpfsSH1/japan478/bin/aix64
127.61  /gpfsSH1/japan478/bin
0.12    /gpfsSH1/japan478/data

```

```

0.42 /gpfsSH1/japan478/include
0.48 /gpfsSH1/japan478/lib
0.44 /gpfsSH1/japan478/messages
0.36 /gpfsSH1/japan478/samples/debugtools
0.16 /gpfsSH1/japan478/samples/ilm
0.02 /gpfsSH1/japan478/samples/net
0.02 /gpfsSH1/japan478/samples/perf
0.02 /gpfsSH1/japan478/samples/uidremap
0.02 /gpfsSH1/japan478/samples/util
0.77 /gpfsSH1/japan478/samples
133.05 /gpfsSH1/japan478

```

6. Start the GPFS daemon on all nodes of the KGN site with the **mmstartup** command. See Example 3-152.

Example 3-152 Run mmstartup command to restart the GPFS daemons at site KGN

```

[root on france][~/tmp] => mmstartup
Fri Jul 16 00:46:46 EDT 2010: mmstartup: Starting GPFS ...
[root on france][~/tmp] =>

[root on ghana][~/tmp] => mmstartup
Fri Jul 16 00:46:51 EDT 2010: mmstartup: Starting GPFS ...
[root on ghana][~/tmp] =>

```

7. Verify GPFS daemons are restarted on all nodes of KGN site. Use **mmgetstate** to verify the daemons. See Example 3-153.

Example 3-153 Use mmgetstate to check the state of the GPFS daemon at all sites

```
[root@england ~]# mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|--------------|--------|----------|-------------|------------|-------------|
| 1 | mexico-gpfs | 3 | 5 | 5 | active | quorum node |
| 2 | japan-gpfs | 3 | 5 | 5 | active | quorum node |
| 3 | france-gpfs | 3 | 5 | 5 | active | quorum node |
| 4 | ghana-gpfs | 3 | 5 | 5 | active | quorum node |
| 5 | england-gpfs | 3 | 5 | 5 | active | quorum node |

Summary information

```

Number of nodes defined in the cluster:      5
Number of local nodes active in the cluster:  5
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 5
Number of quorum nodes active in the cluster: 5
Quorum = 3, Quorum achieved

```

```
[root@england ~]#
```

8. Verify status of the disks in the file systems. After the GPFS daemon on the nodes at the KGN site are started and the file system is mounted, the NSD from KGN site remain in the down state. See Example 3-154.

Example 3-154 NSD from site KGN are in the down state after the daemon starts

```
[root@england ~]# mmlsdisk gpfs1
```

| disk name | driver type | sector size | failure group | holds metadata | holds data | status | availability | storage pool |
|--------------|----------------|----------------|------------------|-------------------|---------------|--------|--------------|-----------------|
| <hr/> | | | | | | | | |
| gpfs1nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs2nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs3nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs4nsd | nsd | 512 | 3002 | yes | yes | ready | down | system |
| gpfs5nsd | nsd | 512 | 3002 | yes | yes | ready | down | system |
| gpfs6nsd | nsd | 512 | 3002 | yes | yes | ready | down | system |
| gpfs7nsd | nsd | 512 | 3003 | no | no | ready | up | system |

```
[root@england ~]#
```

9. The NSDs that are served by the nodes at the KGN site must be brought back online (up) manually with the **mmchdisk** command. See Example 3-155.

Example 3-155 mmchdisk command used to bring down disk back up

```
#mmchdisk /dev/gpfs1 start -d "gpfs6nsd;gpfs5nsd;gpfs4nsd"
```

```
[root on ghana][ /tmp] => -d "gpfs6nsd;gpfs5nsd;gpfs4nsd" <
```

```
Scanning file system metadata, phase 1 ...
  59 % complete on Fri Jul 16 00:53:32 2010
 100 % complete on Fri Jul 16 00:53:34 2010
Scan completed successfully.
Scanning file system metadata, phase 2 ...
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
 33.45 % complete on Fri Jul 16 00:53:55 2010 ( 439448 inodes 52408 MB)
 38.52 % complete on Fri Jul 16 00:54:15 2010 ( 508205 inodes 60353 MB)
 42.81 % complete on Fri Jul 16 00:54:36 2010 ( 589497 inodes 67077 MB)
 45.71 % complete on Fri Jul 16 00:54:56 2010 ( 643186 inodes 71617 MB)
 47.11 % complete on Fri Jul 16 00:55:17 2010 ( 667588 inodes 73825 MB)
 48.86 % complete on Fri Jul 16 00:55:38 2010 ( 692211 inodes 76564 MB)
 50.38 % complete on Fri Jul 16 00:55:58 2010 ( 741528 inodes 78944 MB)
100.00 % complete on Fri Jul 16 00:56:14 2010
Scan completed successfully.
[root on ghana][ /tmp] =>
```

10. Verify that all down NSDs at the KGN site are now available. Use the **mm1sdisk** command to verify the status availability is in the up state. See Example 3-156.

Example 3-156 Use mm1sdisk to verify disks in file system /gpfs1 are now up

```
[root on ghana][ /tmp] => mm1sdisk gpfs1
```

| disk name | driver type | sector size | failure group | holds metadata | holds data | status | availability | storage pool |
|-----------|-------------|-------------|---------------|----------------|------------|--------|--------------|--------------|
| gpfs1nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs2nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs3nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs4nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs5nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs6nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs7nsd | nsd | 512 | 3003 | no | no | ready | up | system |

```
[root on ghana][ /tmp] =>
```

11. Rebalance the /gpfs1 file system. Use the **mmrestripefs** command with the **-b** option to rebalance all files for all disks. See Example 3-157.

Example 3-157 mmrestripefs rebalances the file system and restore the replication of all files

```
[root on ghana][ /tmp] => mmrestripefs gpfs1 -b -N all
Scanning file system metadata, phase 1 ...
Scan completed successfully.
Scanning file system metadata, phase 2 ...
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
0.44 % complete on Fri Jul 16 01:01:40 2010 ( 204572 inodes 370 MB)
0.62 % complete on Fri Jul 16 01:02:00 2010 ( 505332 inodes 519 MB)
100.00 % complete on Fri Jul 16 01:02:07 2010
Scan completed successfully.
[root on ghana][ /tmp] =>
```

12. Verify that /gpfs1 is mounted on all nodes at all sites. Use the **mm1smount** command to verify. See Example 3-158.

Example 3-158 The /gpfs1 is mounted on all nodes at all sites after the KGN site recovers

```
[root on ghana][ /tmp] => mm1smount all -L

File system gpfs1 is mounted on 5 nodes:
10.11.12.221 mexico-gpfs
10.11.12.222 japan-gpfs
10.11.12.143 england-gpfs
10.11.12.223 france-gpfs
10.11.12.224 ghana-gpfs
[root on ghana][ /tmp] =>
```

Identify LPAR name that is associated with a node at site

To ensure the correct LPAR is shut down, identify the name of the LPAR that is associated with the nodes at the KGN site.

One way to identify the LPAR associated with a particular node is as follows:

1. Log in to the node on the site (KGN) for which you want to identify its LPAR.
2. Run the **lparstat -i** command to display the partition name and node name. See Example 3-159.
3. Obtain the partition name associated with ghana.

Example 3-159 Use lparstat -i to identify partition name associated with this host name

```
[root on ghana] [/] => lparstat -i
Node Name                               : ghana
Partition Name                           : 550_3_LP04
Partition Number                         : 6
Type                                     : Shared-SMT
Mode                                     : Capped
Entitled Capacity                        : 0.50
Partition Group-ID                       : 32774
Shared Pool ID                           : 0
Online Virtual CPUs                      : 2
Maximum Virtual CPUs                     : 4
Minimum Virtual CPUs                     : 1
Online Memory                            : 4224 MB
Maximum Memory                           : 5248 MB
Minimum Memory                           : 1152 MB
Variable Capacity Weight                  : 0
Minimum Capacity                         : 0.10
Maximum Capacity                         : 4.00
Capacity Increment                       : 0.01
Maximum Physical CPUs in system           : 4
Active Physical CPUs in system            : 4
Active CPUs in Pool                       : 4
Shared Physical CPUs in system            : 4
Maximum Capacity of Pool                  : 400
Entitled Capacity of Pool                 : 350
Unallocated Capacity                      : 0.00
Physical CPU Percentage                   : 25.00%
Unallocated Weight                        : 0
Memory Mode                              : Dedicated
Total I/O Memory Entitlement               : -
Variable Memory Capacity Weight           : -
Memory Pool ID                            : -
Physical Memory in the Pool                : -
Hypervisor Page Size                      : -
Unallocated Variable Memory Capacity Weight: -
Unallocated I/O Memory entitlement         : -
Memory Group ID of LPAR                   : -
Desired Virtual CPUs                      : 2
Desired Memory                            : 4224 MB
Desired Variable Capacity Weight           : 0
Desired Capacity                         : 0.50
Target Memory Expansion Factor             : -
Target Memory Expansion Size              : -
[root on ghana] [/] =>
```

- Shut down LPAR associated with france and ghana. Log in to the HMC console that is associated with france and ghana, and then perform the LPAR shutdown operation. See Figure 3-40.

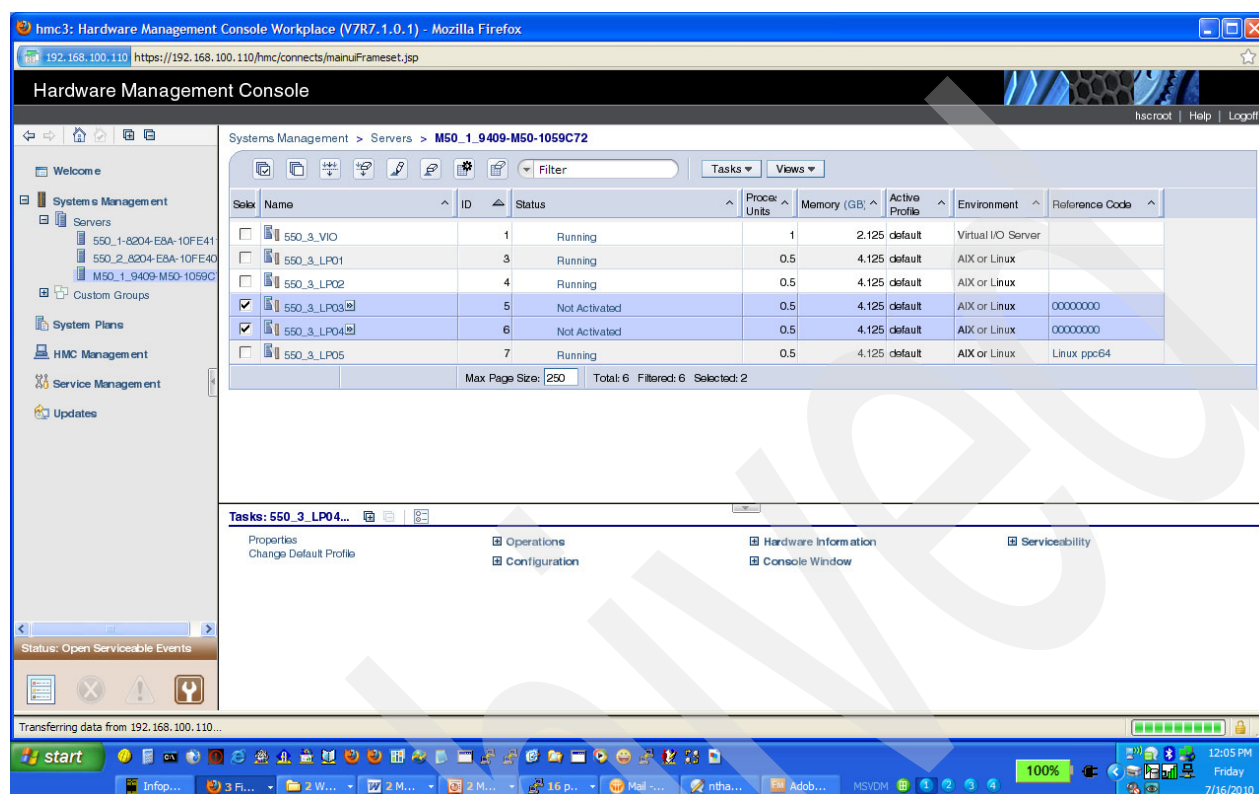


Figure 3-40 HMC console that is used to shut down france and ghana

- Verify france and ghana LPARs are shut down. Use the `mmgetstate` command to verify the status. See Example 3-160.

Example 3-160 Check the state of france and ghana with the mmgetstate command

```
[root@england ~]# mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|--------------|--------|----------|-------------|------------|-------------|
| 1 | mexico-gpfs | 3 | 3 | 5 | active | quorum node |
| 2 | japan-gpfs | 3 | 3 | 5 | active | quorum node |
| 3 | france-gpfs | 0 | 0 | 5 | unknown | quorum node |
| 4 | ghana-gpfs | 0 | 0 | 5 | unknown | quorum node |
| 5 | england-gpfs | 3 | 3 | 5 | active | quorum node |

Summary information

```
-----
Number of nodes defined in the cluster:      5
Number of local nodes active in the cluster:  3
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 5
Number of quorum nodes active in the cluster: 3
Quorum = 3, Quorum achieved
```

```
[root@england ~]#
```

```
[root on mexico][/] => mmgetstate -als
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|--------------|--------|----------|-------------|------------|-------------|
| 1 | mexico-gpfs | 3 | 3 | 5 | active | quorum node |
| 2 | japan-gpfs | 3 | 3 | 5 | active | quorum node |
| 3 | france-gpfs | 0 | 0 | 5 | down | quorum node |
| 4 | ghana-gpfs | 0 | 0 | 5 | down | quorum node |
| 5 | england-gpfs | 3 | 3 | 5 | active | quorum node |

Summary information

```
-----
Number of nodes defined in the cluster:      5
Number of local nodes active in the cluster:  3
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 5
Number of quorum nodes active in the cluster: 3
Quorum = 3, Quorum achieved
```

```
[root on mexico][/] =>
```

6. Verify that /gpfs1 is mounted at POK site after site shutdown. Use **mm1smount** command to determine the status of /gpfs1 on the surviving site. See Example 3-161.

Example 3-161 Use mm1smount to check the mount status of /gpfs1 on the surviving site (POK)

```
[root on japan][/] => mm1smount all -L
```

File system gpfs1 is mounted on 3 nodes:

```
10.11.12.221    mexico-gpfs
10.11.12.222    japan-gpfs
10.11.12.143    england-gpfs
```

```
[root on japan][/] =>
```

7. Verify /gpfs1 is accessible from all nodes of the POK site; verify that the workload, which is running on the nodes of the POK site, is running without errors. See Example 3-162.

Example 3-162 Verify workload running on the nodes of the POK site run error free

```
[root on mexico][/] => du -m /gpfsSH1/japan0
```

```
0.05  /gpfsSH1/japan0/README
0.36  /gpfsSH1/japan0/bin/.links
50.70  /gpfsSH1/japan0/bin/aix32
54.97  /gpfsSH1/japan0/bin/aix64
131.55 /gpfsSH1/japan0/bin
2.25   /gpfsSH1/japan0/samples
138.47 /gpfsSH1/japan0
[root on mexico][/] =>
```

```
[root on japan][/] =>du -m /gpfsSH1/france0
```

```
1166.47 /gpfsSH1/france0/mexico0/bos/inst_root
1374.09 /gpfsSH1/france0/mexico0/bos
0.38   /gpfsSH1/france0/mexico0/samples/perf
0.05   /gpfsSH1/france0/mexico0/samples/uidremap
0.67   /gpfsSH1/france0/mexico0/samples/util
2.41   /gpfsSH1/france0/mexico0/samples
2316.59 /gpfsSH1/france0/mexico0
2316.61 /gpfsSH1/france0
[root on japan][/] =>
```

Important: While the GPFS cluster is in a failover state, no changes to the GPFS configuration are made.

8. Reactivate the LPARs for france and ghana. Log in to the HMC that manages the nodes (france and ghana) and reactivate these LPARs, as shown in Figure 3-41.

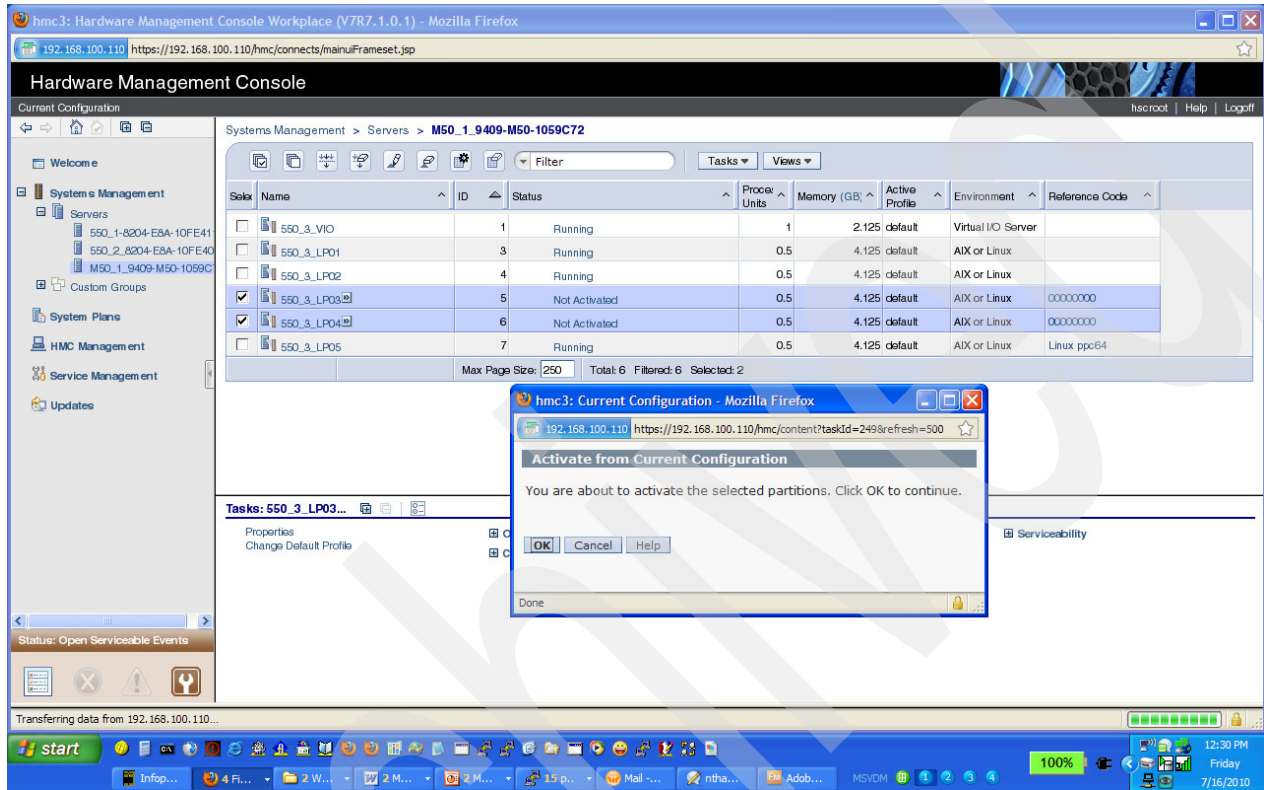


Figure 3-41 HMC console during activation of the france and ghana LPARs

After reactivation of the france and ghana LPARs, the HMC console looks similar to Figure 3-42 on page 180.

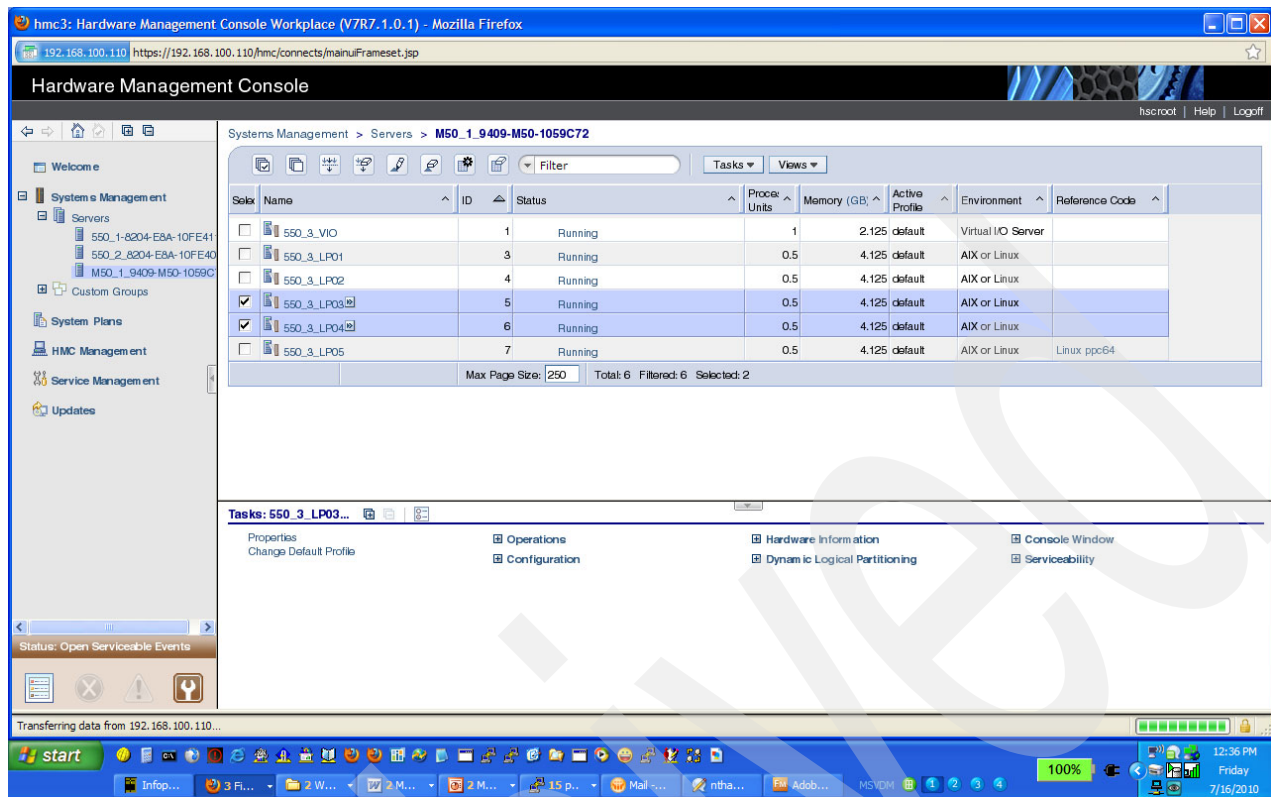


Figure 3-42 HMC console after reactivation of france and ghana LPARs

- Start GPFS on all nodes of the KGN site; after the LPARs for france and ghana are reactivated, start the GPFS daemon on these LPARs. See Example 3-163.

Example 3-163 Manually start the GPFS daemon on all nodes of the KGN site

```
[root on france][/] => mmstartup
Fri Jul 16 13:12:07 EDT 2010: mmstartup: Starting GPFS ...
[root on france][/] =>

[root on ghana][/] => mmstartup
Fri Jul 16 13:12:32 EDT 2010: mmstartup: Starting GPFS ...
[root on ghana][/] =>
```

- Verify GPFS daemon is restarted on all nodes of the KGN site. Use the **mmgetstate** command to verify the active state. See Example 3-164.

Example 3-164 Use mmgetstate to check status of GPFS daemon on all nodes

```
[root on mexico][/] => mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|--------------|--------|----------|-------------|------------|-------------|
| 1 | mexico-gpfs | 3 | 5 | 5 | active | quorum node |
| 2 | japan-gpfs | 3 | 5 | 5 | active | quorum node |
| 3 | france-gpfs | 3 | 5 | 5 | active | quorum node |
| 4 | ghana-gpfs | 3 | 5 | 5 | active | quorum node |
| 5 | england-gpfs | 3 | 5 | 5 | active | quorum node |

Summary information

```

-----
Number of nodes defined in the cluster:      5
Number of local nodes active in the cluster: 5
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 5
Number of quorum nodes active in the cluster: 5
Quorum = 3, Quorum achieved

```

```
[root on mexico] [/] =>
```

11. Verify that /gpfs1 is remounted on all nodes of all sites. Use the `mm1smount` command to verify that the file system is mounted. See Example 3-165.

Example 3-165 Use mm1smount to verify /gpfs1 is mounted on all node of all sites

```
[root on mexico] [/] => mm1smount all -L
```

File system gpfs1 is mounted on 5 nodes:

```

10.11.12.221    mexico-gpfs
10.11.12.222    japan-gpfs
10.11.12.143    england-gpfs
10.11.12.223    france-gpfs
10.11.12.224    ghana-gpfs

```

```
[root on mexico] [/] =>
```

12. Verify the status of all disks in the file systems. Use `mm1sdisk` command to verify the status. See Example 3-166.

Example 3-166 Verify the status of each disk in the file system

```
[root on mexico] [/] => mm1sdisk gpfs1
```

| disk name | driver type | sector size | failure group | holds metadata | holds data | status | availability | storage pool |
|--------------|----------------|----------------|------------------|-------------------|---------------|--------|--------------|-----------------|
| gpfs1nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs2nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs3nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs4nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs5nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs6nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs7nsd | nsd | 512 | 3003 | no | no | ready | up | system |

```
[root on mexico] [/] =>
```

Fail over to surviving site -tiebreaker affected

This section describes how to recover from the failure of two sites in a three-site disaster recovery configuration where the tiebreaker site is also affected. If the tiebreaker site is no longer operational after a disaster, GPFS node and file system quorum is broken. When GPFS quorum is broken, manual intervention is required to resume file-system access. For the surviving site to satisfy its quorum requirements and remain operational, the existing quorum configuration must be modified.

The following quorum changes must be made for the surviving site to continue:

1. Relax node quorum requirements.

To relax node quorum, use the **mmchnode --nonquorum** command, which temporarily changes the designation of each of the failed quorum nodes in the failed sites to nonquorum nodes.

2. Relax file system descriptor quorum.

To relax file descriptor quorum, use the **mmfsctl exclude** command to temporarily eliminate the failed disks from the group of disks from which the GPFS daemon uses to write the file system descriptor file.

3. Power down all nodes at site KGN -france and ghana. Use HMC to identify the LPARs associated with france and ghana. Power down the LPAR associated with france and ghana using shutdown immediately. See Figure 3-43.

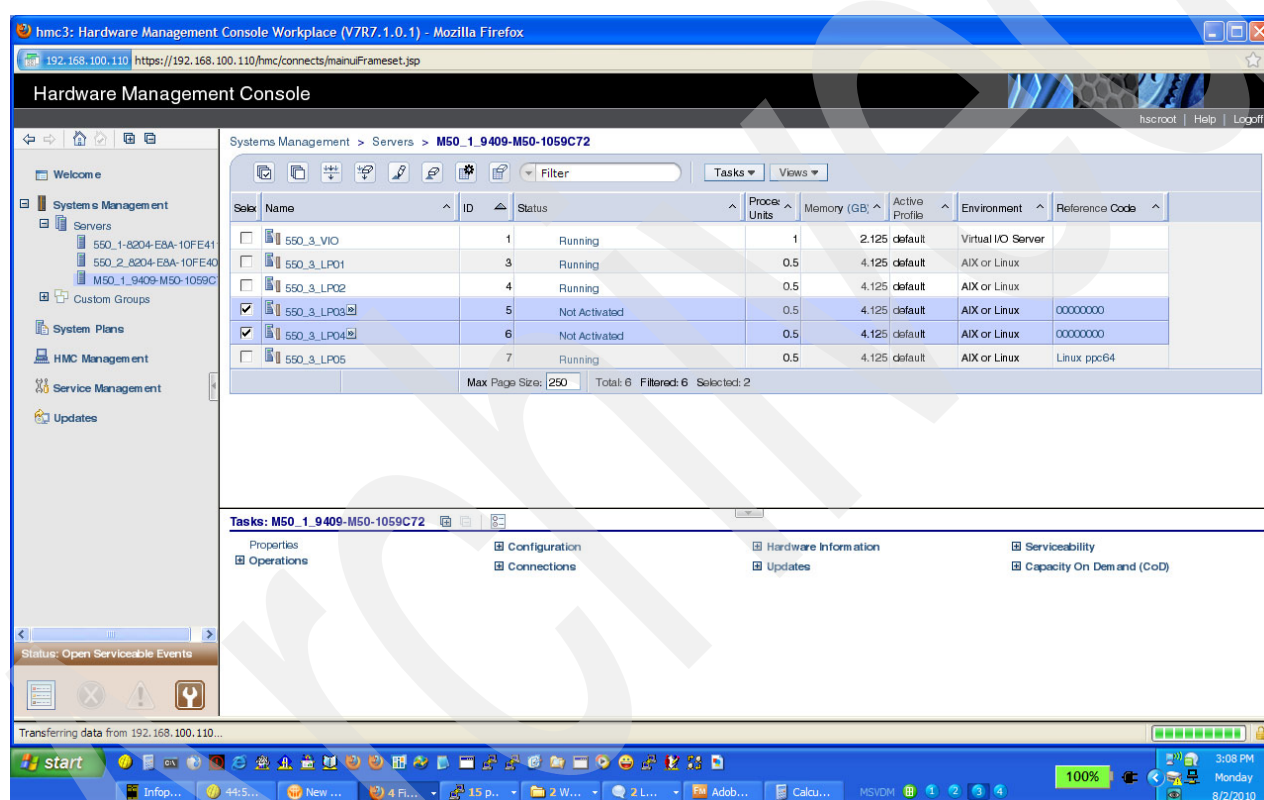


Figure 3-43 HMC console after france and ghana GPFS power-off

4. Verify that all nodes are down at the KGN and BUF sites. After all nodes at the site (KGN) are powered off, GPFS detects the failure and starts recovery. See Example 3-167.

Example 3-167 Use mmgetstate to verify all nodes at site (KGN) are down

```
[root on mexico][/] => mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|--------------|--------|----------|-------------|-------------|-------------|
| 1 | mexico-gpfs | 3 | 0 | 5 | arbitrating | quorum node |
| 2 | japan-gpfs | 3 | 3 | 5 | active | quorum node |
| 3 | france-gpfs | 0 | 0 | 5 | unknown | quorum node |
| 4 | ghana-gpfs | 0 | 0 | 5 | unknown | quorum node |
| 5 | england-gpfs | 0 | 0 | 5 | down | quorum node |

Summary information

```
-----
Number of nodes defined in the cluster:      5
Number of local nodes active in the cluster: 3
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 5
Number of quorum nodes active in the cluster: 3
Quorum = 3, Quorum achieved
```

```
[root on mexico][/] =>
```

5. Check the state of file systems on the cluster. The **mm1smount** command fails because the nodes are down. See Example 3-168.

Example 3-168 The mm1smount fails because all nodes in site KGN and BUF are down

```
[root on mexico][/] => mm1smount all -L
Device not ready.
mm1smount: Command was unable to determine whether file system gpfs1 is mounted.
mm1smount: Command failed. Examine previous error messages to determine cause.
[root on mexico][/]
```

```
[root on mexico][/] => df
Filesystem      512-blocks      Free %Used    Iused %Iused Mounted on
/dev/hd4         655360      95496  86%    14178   52% /
/dev/hd2         6029312     79088  99%    48134   77% /usr
/dev/hd9var      2752512    2079288  25%    11571    5% /var
/dev/hd3         2359296    2050080  14%        125    1% /tmp
/dev/hd1         131072     130360    1%         5    1% /home
/dev/hd11admin   131072     130360    1%         5    1% /admin
/proc            -           -    -         -    - /proc
/dev/hd10opt     524288     119888  78%     8618   39% /opt
/dev/livedump    131072     130392    1%         4    1% /var/adm/ras/livedump
/dev/gpfs1       -           -    -         -    - /gpfsSH1
[root on mexico][/] =>
```

6. Shut down the GPFS daemon on all nodes of the surviving site. Use the **mmshutdown** command to shut down the daemon. See Example 3-169.

Example 3-169 Use mmshutdown to shut down the GPFS daemon on all nodes of surviving site

```
[root on japan][/] => mmshutdown
Tue Aug 3 12:01:39 EDT 2010: mmshutdown: Starting force unmount of GPFS file systems
forced unmount of /gpfsSH1
Tue Aug 3 12:01:44 EDT 2010: mmshutdown: Shutting down GPFS daemons
Shutting down!
'shutdown' command about to kill process 4849724
Tue Aug 3 12:01:49 EDT 2010: mmshutdown: Finished
[root on japan][/] =>
```

7. Change node quorum for france-gpfs and ghana-gpfs. To relax the node quorum requirements, change france-gpfs and ghana-gpfs to *nonquorum* with the **mmchnode --nonquorum** command. See Example 3-170 on page 184.

Example 3-170 Change france-gpfs and ghana-gpfs to nonquorum nodes

```
[root on mexico][/] => mmchnode --nonquorum -N ghana-gpfs,france-gpfs
Tue Aug 3 12:55:16 EDT 2010: mmchnode: Processing node france-gpfs
Tue Aug 3 12:55:16 EDT 2010: mmchnode: Processing node ghana-gpfs
Verifying GPFS is stopped on all affected nodes ...
mmchnode: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root on mexico][/] =>
```

8. Change disk descriptor quorum for site-wide file system. Relax the file system descriptor quorum by using the **mmfsctl** command. See Example 3-171.

Example 3-171 Use mmfsctl to relax file system descriptor quorum

```
[root on japan][/] => mmfsctl gpfs1 exclude -d "gpfs4nsd;gpfs5nsd;gpfs6nsd"
The authenticity of host 'england-gpfs (10.11.12.143)' can't be established.
RSA key fingerprint is 60:e3:5a:ca:a0:c3:c4:46:bb:80:1c:4d:0a:57:97:a5.
Are you sure you want to continue connecting (yes/no)? yes
mmfsctl: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root on japan][/] =>
```

9. Start GPFS on all nodes at the KGN site. Use **mmstartup** to restart daemons at the surviving site. See Example 3-172.

Example 3-172 Restart GPFS daemons at surviving site

```
[root on mexico][/] => mmstartup
Tue Aug 3 13:13:45 EDT 2010: mmstartup: Starting GPFS ...
[root on mexico][/] =>

[root on japan][/] => mmstartup
Tue Aug 3 13:13:38 EDT 2010: mmstartup: Starting GPFS ...
[root on japan][/]
```

10. Check file system and disk on surviving site; verify that /gpfs1 is mounted. See Example 3-173.

Example 3-173 Verify /gpfs1 is mounted on all nodes of the surviving site

File system gpfs1 is mounted on 2 nodes:

```
10.11.12.221    mexico-gpfs
10.11.12.222    japan-gpfs
```

```
[root on mexico][/] =>
```

```
[root on mexico][/] => mmlsdisk gpfs1
```

| disk name | driver type | sector size | failure group | holds metadata | holds data | status | availability | storage pool |
|--------------|----------------|----------------|------------------|-------------------|---------------|--------|--------------|-----------------|
| gpfs1nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs2nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs3nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs4nsd | nsd | 512 | 3002 | yes | yes | ready | down | system |
| gpfs5nsd | nsd | 512 | 3002 | yes | yes | ready | down | system |
| gpfs6nsd | nsd | 512 | 3002 | yes | yes | ready | down | system |
| gpfs7nsd | nsd | 512 | 3003 | no | no | ready | up | system |

```
[root on mexico][/] =>
```

Fail back after temporary outage and configuration change

If the failure was of a temporary nature and the configuration has changed, use the steps in this section to restore the original configuration. Ensure all nodes at the failed site are repaired and any disk problems are corrected:

1. Ensure all nodes contain the latest `mmsdrfs` file. Use `mmchcluster` to check for the most recent levels. See Example 3-174.

Example 3-174 Make sure the mmsdrfs file on each node is at the latest level

```
[root on mexico][/] => mmchcluster -p LATEST
mmchcluster: GPFS cluster configuration servers:
mmchcluster: Primary server: mexico-gpfs
mmchcluster: Secondary server: japan-gpfs
mmchcluster: Propagating the new server information to the rest of the nodes.
mmchcluster: Command successfully completed
[root on mexico][/]
```

2. Move secondary cluster configuration server back to KGN site. See Example 3-175.

Example 3-175 Move the secondary cluster configuration server france-gpfs back to KGN site

```
[root on mexico][/] => mmchcluster -s france-gpfs
mmchcluster: GPFS cluster configuration servers:
mmchcluster: Primary server: mexico-gpfs
mmchcluster: Secondary server: france-gpfs
mmchcluster: Propagating the new server information to the rest of the nodes.
mmchcluster: Command successfully completed
[root on mexico][/] =>
```

3. Restore original node quorum designation. Use the `mmchnode` command. See Example 3-176.

Example 3-176 Change ghana-gpfs and france-gpfs back to quorum nodes

```
[root on mexico][/] => mmchnode --quorum -N ghana-gpfs,france-gpfs
Tue Aug 3 14:28:24 EDT 2010: mmchnode: Processing node france-gpfs
Tue Aug 3 14:28:24 EDT 2010: mmchnode: Processing node ghana-gpfs
mmchnode: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root on mexico][/] =>
```

4. Start GPFS on the all nodes of the KGN site. Use the `mmstartup` command to start the daemons. See Example 3-177.

Example 3-177 Start all of the GPFS daemons at the KGN site

```
[root on france][/] => mmstartup -a
Tue Aug 3 15:22:39 EDT 2010: mmstartup: Starting GPFS ...
[root on france][/] =>
```

5. Restore the file system descriptor quorum. Use the `mmfsctl` command to restore file system descriptor quorum. See Example 3-178.

Example 3-178 Use mmfsctl to restore file system descriptor quorum

```
[root on japan][/] => mmfsctl gpfs1 include -d "gpfs4nsd;gpfs5nsd;gpfs6nsd"
mmfsctl: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root on japan][/] =>
```

6. Bring the disks online by using the `mmchdisk` command. See Example 3-179.

Example 3-179 Use mmchdisk to bring all down disks of the gpfs1 file system back online

```
[root on mexico][/] => mmchdisk gpfs1 start -d "gpfs4nsd;gpfs5nsd;gpfs6nsd"
Scanning file system metadata, phase 1 ...
  59 % complete on Tue Aug  3 15:34:26 2010
 100 % complete on Tue Aug  3 15:34:28 2010
Scan completed successfully.
Scanning file system metadata, phase 2 ...
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
 100.00 % complete on Tue Aug  3 15:34:32 2010
Scan completed successfully.
[root on mexico][/] =>
```

7. Check the state of the GPFS daemon at all sites. Use the `mmgetstate` command to get the state of the daemons. See Example 3-180.

Example 3-180 Verify all GPFS daemons are started all sites

```
[root on ghana][/] => mmgetstate -aLs
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|--------------|--------|----------|-------------|------------|-------------|
| 1 | mexico-gpfs | 3 | 5 | 5 | active | quorum node |
| 2 | japan-gpfs | 3 | 5 | 5 | active | quorum node |
| 3 | france-gpfs | 3 | 5 | 5 | active | quorum node |
| 4 | ghana-gpfs | 3 | 5 | 5 | active | quorum node |
| 5 | england-gpfs | 3 | 5 | 5 | active | quorum node |

Summary information

```
-----
Number of nodes defined in the cluster:      5
Number of local nodes active in the cluster:  5
Number of remote nodes joined in this cluster: 0
Number of quorum nodes defined in the cluster: 5
Number of quorum nodes active in the cluster: 5
Quorum = 3, Quorum achieved
```

```
[root on ghana][/] =>
```

8. Check mount status of `/gpfs1` directory at all sites; verify that the `/gpfs1` file system is mounted on all nodes of all sites. See Example 3-181.

Example 3-181 Verify file system /gpfs1 is mounted on all nodes of all sites

```
File system gpfs1 is mounted on 5 nodes:
```

```
10.11.12.221  mexico-gpfs
10.11.12.222  japan-gpfs
10.11.12.223  france-gpfs
10.11.12.224  ghana-gpfs
10.11.12.143  england-gpfs
```

```
[root on france][/] =>
```

9. Check the state of all disks in the file system to verify their availability. See Example 3-182.

Example 3-182 Verify all disks in file system /gpfs1 are up and available

```
[root on france][/] => mmlsdisk gpfs1
```

| disk name | driver type | sector size | failure group | holds metadata | holds data | status | availability | storage pool |
|--------------|----------------|----------------|------------------|-------------------|---------------|--------|--------------|-----------------|
| <hr/> | | | | | | | | |
| gpfs1nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs2nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs3nsd | nsd | 512 | 3001 | yes | yes | ready | up | system |
| gpfs4nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs5nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs6nsd | nsd | 512 | 3002 | yes | yes | ready | up | system |
| gpfs7nsd | nsd | 512 | 3003 | no | no | ready | up | system |

```
[root on france][/] =>
```

Archived

Management and maintenance

This chapter contains information related to what you can do when the GPFS cluster is already configured and running.

The first section covers migration and upgrades of a running GPFS cluster. The second section covers activities you can do to the cluster configuration, for example expanding your running cluster by adding nodes, disks, and file systems.

The remaining sections explain and provide examples of configuration of more complex GPFS functions, such as access control lists (ACLs), GPFS callback, setting up SNMPD, and SSH key-management when adding a new node or reinstalling an existing one.

This chapter contains the following topics:

- ▶ 4.1, “Migration and update” on page 190
- ▶ 4.2, “Managing a GPFS cluster” on page 200
- ▶ 4.3, “Reducing file system fragmentation: The mmdefragfs command” on page 222
- ▶ 4.4, “Optimizing extended attributes: The fastea option” on page 224
- ▶ 4.5, “Setting up GPFS callback functionality: The callback commands” on page 224
- ▶ 4.6, “Monitoring GPFS configuration status: The SNMPD protocol” on page 226
- ▶ 4.7, “SSH configuration” on page 227

4.1 Migration and update

This section describes tasks to perform when you have to migrate your cluster to a new release or apply fixes.

4.1.1 Migrating GPFS from 3.2 or 3.3 to 3.4

This section describes an example of a rolling migration with a running GPFS cluster from version 3.3 to version 3.4. A version 3.3 cluster configuration is shown in Example 4-1.

A *rolling upgrade* is a method of upgrading the GPFS cluster, updating only one or more nodes at a time. In this way, you can have the production cluster still running on the remaining active nodes without the need to stop the GPFS cluster. You still have to stop applications and the GPFS on the node or nodes on which you are upgrading GPFS.

The complete migration procedures for GPFS are in *General Parallel File System Concepts, Planning, and Installation Guide Version 3 Release 4*, GA76-0413.

For example, GPFS 3.4 does not support AIX 5.2, so you have to upgrade AIX to at least version 5.3 before upgrading GPFS to version 3.4.

When upgrading GPFS, check the latest GPFS frequently asked questions (FAQs). The FAQs provide details about which levels of GPFS are supported by your operating system:

<http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=%2Fcom.ibm.cluster.gpfs.doc%2Fgpfs31%2Fb11ins1118.html>

Example 4-1 Output of mmlsconfig command

```
mmlsconfig
Configuration data for cluster mantest.usa:
-----
clusterName mantest.usa
clusterId 13882457470256956975
autoload no
[usa,germany]
autoload yes
[common]
minReleaseLevel 3.3.0.2
dmapiFileHandleSize 32
tiebreakerDisks nsdhdisk2
adminMode central
```

A two-node cluster configuration with tiebreaker disk, nodes named `usa` and `germany`, is shown in Example 4-2.

Example 4-2 Output of `mmlscluster` command

```
germany:/#mmlscluster
```

```
GPFS cluster information
=====
```

```
GPFS cluster name:      mantest.usa
GPFS cluster id:        13882457470256956975
GPFS UID domain:        mantest.usa
Remote shell command:   /usr/bin/ssh
Remote file copy command: /usr/bin/scp
```

```
GPFS cluster configuration servers:
-----
```

```
Primary server:   usa
Secondary server: germany
```

| Node | Daemon node name | IP address | Admin node name | Designation |
|-------|------------------|-----------------|-----------------|-------------|
| <hr/> | | | | |
| 1 | usa | 192.168.101.132 | usa | quorum |
| 2 | germany | 192.168.101.142 | germany | quorum |

Note that both nodes shown in Example 4-2 are at the same GPFS level:

```
gpfs.base          3.3.0.6  COMMITTED  GPFS File Manager
gpfs.msg.en_US     3.3.0.4  COMMITTED  GPFS Server Messages - U.S.
gpfs.docs.data     3.3.0.1  COMMITTED  GPFS Server Manpages and
```

Now, one node at a time can be migrated to GPFS 3.4. The customer production service does not have to be stopped because other nodes are still up and running the GPFS cluster. Both nodes are currently active and both nodes are mounting the following GPFS file system:

```
/dev/mantestfs 408944640 305897472 26% 4539 3% /mantestfs
```

The first node to be migrated is `germany`.

Perform the following steps before migrating to v3.4:

1. Stop applications on the node and then unmount the file system as follows:

```
germany:/#mmumount /dev/mantestfs
Wed Jun 23 14:23:25 EDT 2010: mmumount: Unmounting file systems ...
```

2. Stop the GPFS daemon, as shown in Example 4-3:

Example 4-3 Stop GPFS daemon

```
germany:/#mmshutdown -N germany
Wed Jun 23 14:23:38 EDT 2010: mmshutdown: Starting force unmount of GPFS file
systems
Wed Jun 23 14:23:43 EDT 2010: mmshutdown: Shutting down GPFS daemons
germany: Shutting down!
germany: 'shutdown' command about to kill process 5898448
Wed Jun 23 14:23:49 EDT 2010: mmshutdown: Finished
```

```
germany:/#mmfsenv -u
/usr/lpp/mmfs/bin/mmfskxload: /usr/lpp/mmfs/bin/aix64/mmfs64 is not loaded.
```

Now that both GPFS cluster nodes are at GPFS 3.3.0.6 level, you may migrate GPFS to version 3.4 as follows:

1. Use the System Management Interface Tool (SMIT) `install_all` command, select the CD-ROM or the device where the software has been copied, and select the three GPFS file sets that SMIT presents, as shown in Example 4-4.

Example 4-4 SMIT showing the software to install

```

+-----+
|                                     SOFTWARE to install                                     |
| [T] Move cursor to desired item and press F7. Use arrow keys to scroll. |
| * ONE OR MORE items can be selected. |
| * Press Enter AFTER making all selections. |
|                                     |
| > gpfs.base                                                                ALL |
|   + 3.4.0.0  GPFS File Manager |
|                                     |
| gpfs.docs                                                                ALL |
|   + 3.4.0.0  GPFS Server Manpages and Documentation |
|                                     |
| gpfs.msg.en_US                                                            ALL |
|   + 3.4.0.0  GPFS Server Messages - U.S. English |
| [M] F1=Help          F2=Refresh          F3=Cancel |
| F1 F7=Select        F8=Image            F10=Exit |
| F5 Enter=Do         /=Find              n=Find Next |
| F9+-----+

```

2. Press F7 and then press Enter to return to the installation menu.

License agreement: Because this product is new, select YES to accept the new license agreement to be able to continue.

3. Press Enter. The installation begins. SMIT indicates OK when the installation completes.
4. Exit SMIT and verify the installed GPFS level on this node. It looks similar to Example 4-5.

Example 4-5 Verify the installed GPFS level

```

germany:~#ls1pp -l | grep gpfs
gpfs.base                3.4.0.0  COMMITTED  GPFS File Manager
gpfs.msg.en_US            3.4.0.0  COMMITTED  GPFS Server Messages - U.S.
gpfs.base                 3.4.0.0  COMMITTED  GPFS File Manager
gpfs.docs.data            3.4.0.0  COMMITTED  GPFS Server Manpages and

```

5. Restart GPFS on the node, as follows:


```

germany:~# mmstartup -N germany
Wed Jun 23 14:39:58 EDT 2010: mmstartup: Starting GPFS ...

```
6. Mount the file system if necessary and restart your applications.
7. After everything is successful on the first node, perform the same procedure on each of the other nodes (second, third, and so on).

After GPFS has been correctly migrated on all nodes, communicate to GPFS that the migration process is finished by using the following command:

```
usa:/#mmchconfig release=LATEST
```

This command is required to enable the latest GPFS functionality provided by the new GPFS version to all nodes in the cluster when all nodes are correctly migrated.

Be aware that this command can show nodes where it is not possible to enable the latest functionality, for example because the node has not been migrated yet or because it is not accessible at the moment. Run the command when all nodes are available; this command does not report errors.

Although all the nodes are active and migrated to GPFS 3.4, the file system still has the same old metadata format and does not support new functionalities. We have two choices here.

- If remote clusters are currently mounting file systems that belong to this 3.4 cluster and the remote cluster contains nodes with GPFS version 3.2 or 3.3.

In this case, issue the `mmchfs -V compat` command, so activated characteristics are compatible also with the versions 3.2 and 3.3. Be sure that nodes with version 3.1 or earlier are no longer able to mount the file system.

- If there are no other clusters that will mount the file system or if remote nodes are already migrated to 3.4, issue the `mmchfs -V full` command. In this way, all new 3.4 functionality can be used on this file system, but nodes with an older GPFS version are no longer able to mount the file system.

After you have only 3.4 nodes, use the following command:

```
usa:/#mmchfs /dev/mantestfs -V full
```

```
You have requested that the file system be upgraded to
version 12.03 (3.4.0.0). This will enable new functionality but will
prevent you from using the file system with earlier releases of GPFS.
Do you want to continue? yes
```

As shown, GPFS indicates everything you have to confirm before continuing.

Note: Windows version has no software upgrade or migration option. Instead, you have to uninstall the previous GPFS version and then install the most recent version on your node.

4.1.2 Applying corrective fixes to GPFS

Applying a corrective fix only to your product level does not completely change the level, for example from 3.3.0.0 to 3.3.0.6, or from 3.4.0.0 to 3.4.1.0.

Download GPFS fixes from the IBM GPFS support web site:

<http://www14.software.ibm.com/webapp/set2/sas/f/gpfs/home.html>

Reviewing the GPFS FAQs for the latest service information is always a good idea:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.custer.gpfs.doc/gpfs_faqs/gpfsclustersfaq.html

After you extract the downloaded file, find the README file. This file indicates which level the downloaded update is for and what you have to do, if anything, before applying the update.

As with migration of the latest level of GPFS, applying fixes is done one node at time, so your cluster can still work. Of course when a node is stopped, be aware of the cluster quorum availability.

You must follow several rules before installing the fixes. Every application that uses GPFS data must be closed on the node where you are applying fixes. If the file system is an NFS-exported file system, the file system must be unexported from NFS, the file system must be unmounted, GPFS must be stopped with the `mmshutdown` command, and you have to be sure that the kernel extensions have been unloaded with the `mmfsenv -u` command. Then, you can proceed to install fixes, using SMIT `update_all` command or the `installp` command on AIX, or the `rpm -Uh` command (upgrade) on Linux.

Note: Windows version has no upgrade option; instead, you have to uninstall the previous version and install the most recent version on your node.

Example 4-6 shows the fixes that are installed on a GPFS 3.3 machine.

Example 4-6 Installing corrective fixes on an AIX node

```
germany:/#lslpp -l | grep gpfs
gpfs.base                 3.3.0.0 COMMITTED GPFS File Manager
gpfs.msg.en_US             3.3.0.0 COMMITTED GPFS Server Messages - U.S.
gpfs.base                  3.3.0.0 COMMITTED GPFS File Manager
gpfs.docs.data             3.3.0.0 COMMITTED GPFS Server Manpages and
```

```
germany:/tmp/33#smitty update_all
```

Update Installed Software to Latest Level (Update All)

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

| [TOP] | [Entry Fields] | |
|---|----------------|---|
| * INPUT device / directory for software | . | |
| * SOFTWARE to update | _update_all | |
| PREVIEW only? (update operation will NOT occur) | no | + |
| COMMIT software updates? | yes | + |
| SAVE replaced files? | no | + |
| AUTOMATICALLY install requisite software? | yes | + |
| EXTEND file systems if space needed? | yes | + |
| VERIFY install and check file sizes? | no | + |
| DETAILED output? | no | + |
| Process multiple volumes? | yes | + |
| ACCEPT new license agreements? | no | + |
| PREVIEW new LICENSE agreements? | no | + |

COMMAND STATUS

```
Command: OK          stdout: yes          stderr: no
```

Before command completion, additional instructions may appear below.

```
[TOP]
geninstall -I "a -cgNqwX -J" -Z -d . -f File 2>&1
```

```
File:
  gpfs.base                3.3.0.6
  gpfs.docs.data           3.3.0.1
  gpfs.msg.en_US           3.3.0.4
```

```
+-----+
                        Pre-installation Verification...
+-----+
Verifying selections...done
[MORE...116]

F1=Help      F2=Refresh    F3=Cancel    F6=Command
F8=Image     F9=Shell      F10=Exit     /=Find
n=Find Next

germany:/#ls1pp -l | grep gpfs
  gpfs.base                3.3.0.6  COMMITTED  GPFS File Manager
  gpfs.msg.en_US           3.3.0.4  COMMITTED  GPFS Server Messages - U.S.
  gpfs.base                3.3.0.6  COMMITTED  GPFS File Manager
  gpfs.docs.data           3.3.0.1  COMMITTED  GPFS Server Manpages and
```

4.1.3 Migrating data to new storage

GPFS enables you to migrate data to new storage while the system is running. A feature of GPFS is to automatically move data within the same file system and between every Network Shared Disk (NSD) that has been configured. The two main GPFS commands used to do this job are the `mmadddisk` command to add the NDS created on new storage to the file system and then `mmde1disk` command to remove disks belonging to the old storage and have data migrated to the new.

When a new disk is assigned to an NSD, a user storage pool can be created. User storage pools are used to group disks with similar properties to have benefit in reliability and price per performance. After the addition of new disks to the file system, you can delete the old disks. During this operation, the data that resides on old disks is moved to new disks, so there is the need to have sufficient free space in the file system to accommodate all the data. If the deleted disk is the last disk available in an user defined storage pool, also the user created storage pool is deleted. When this operation is finished, the file system might be unbalanced, so you might have to run the `mmrestripefs` command to rebalance all files in the file system correctly on the new disks. You can decide to restripe data belonging only to one storage pool and not all the data in the file system by using the `-P` option of the `mmrestripefs` command.

For example, you may migrate data from a storage server to another (which might be bigger and faster). An overview of the steps is as follows:

1. Configure the new storage, so you can correctly see new disks in your operating system.
2. From GPFS, configure a new NSD, based on the new disks.
3. Add the newly created NSD to the old file system. Now, the file system has old and new NSD, so more free space is available.

4. Remove the old NSD from the GPFS file system. GPFS will automatically migrate data to the new NSD.
5. Delete the NSD from the GPFS configuration.
6. Use the OS to remove the old disks.

Details of these steps are described next. Example 4-7 shows that the file system is currently mounted on the cluster.

Example 4-7 Output of mmdf command

```

usa:/#mmdf testmigr
disk      disk size  failure holds   holds      free KB      free KB
name      in KB     group metadata data      in full blocks  in fragments
-----
Disks in storage pool: system (Maximum disk size allowed is 1.3 TB)
nsdhdisk3 26214400    80 yes    yes    22074368 ( 84%)    1512 ( 0%)
nsdhdisk2 125829120   80 yes    yes    106222336 ( 84%)    1312 ( 0%)
-----
(pool total) 152043520                128296704 ( 84%)    2824 ( 0%)
=====
(total)      152043520                128296704 ( 84%)    2824 ( 0%)
=====

Inode Information
-----
Number of used inodes:      4047
Number of free inodes:     145457
Number of allocated inodes: 149504
Maximum number of inodes:  149504

```

Perform the following steps:

1. The file system in Example 4-8 has two NSDs for now. Attach the new storage and configure it in the nodes. Example 4-8 shows two new disks. We proceed to configure the new NSDs on them.

Example 4-8 The mmcrnsd command: NSD creation

```

usa:/#mmcrnsd -F nsd1.conf
mmcrnsd: Processing disk hdisk4
mmcrnsd: Processing disk hdisk5
mmcrnsd: Propagating the cluster configuration data to all
        affected nodes. This is an asynchronous process.

```

```

usa:/#cat nsd1.conf
# hdisk4:usa,germany::dataAndMetadata:40:nsdhdisk4:
nsdhdisk4::dataAndMetadata:40::
# hdisk5:usa,germany::dataAndMetadata:40:nsdhdisk5:
nsdhdisk5::dataAndMetadata:40::

```

```

usa:/#mmfnsd

```

| File system | Disk name | NSD servers |
|-------------|-----------|-------------|
| testmigr | nsdhdisk2 | germany,usa |
| testmigr | nsdhdisk3 | germany,usa |
| (free disk) | nsdhdisk4 | usa,germany |
| (free disk) | nsdhdisk5 | usa,germany |

The nsdhdisk2 and nsdhdisk3 disks now belong to the *testmigr* file system and are inside the old storage; the nsdhdisk4 and nsdhdisk5 disks belong to the new storage.

2. Add the two new NSDs to the *testmigr* file system, as shown in Example 4-9.

Example 4-9 The mmadddisk command: Adding disks to a GPFS file system

```
usa:/#mmadddisk testmigr -F nsd1.conf
The following disks of testmigr will be formatted on node germany:
    nsdhdisk4: size 26214400 KB
    nsdhdisk5: size 26214400 KB
Extending Allocation Map
Checking Allocation Map for storage pool 'system'
Completed adding disks to file system testmigr.
mmadddisk: Propagating the cluster configuration data to all
    affected nodes. This is an asynchronous process.
```

Four NSDs are now on the file system, as shown in Example 4-10.

Example 4-10 The mmdf command: Viewing new NSD associated to a file system

```
usa:/#mmdf testmigr
```

| disk name | disk size in KB | failure holds group metadata | holds data | free KB in full blocks | free KB in fragments |
|---|--------------------|---------------------------------|---------------|---------------------------|-------------------------|
| Disks in storage pool: system (Maximum disk size allowed is 1.3 TB) | | | | | |
| nsdhdisk2 | 125829120 | 80 yes | yes | 106222336 (84%) | 1312 (0%) |
| nsdhdisk3 | 26214400 | 80 yes | yes | 22074368 (84%) | 1512 (0%) |
| nsdhdisk4 | 26214400 | 80 yes | yes | 26212096 (100%) | 248 (0%) |
| nsdhdisk5 | 26214400 | 80 yes | yes | 26212096 (100%) | 248 (0%) |
| (pool total) | 204472320 | | | 180720896 (88%) | 3320 (0%) |
| ===== | | | | | |
| (total) | 204472320 | | | 180720896 (88%) | 3320 (0%) |
| Inode Information | | | | | |
| ----- | | | | | |
| Number of used inodes: | | 4047 | | | |
| Number of free inodes: | | 145457 | | | |
| Number of allocated inodes: | | 149504 | | | |
| Maximum number of inodes: | | 149504 | | | |

3. Remove the old NSD from file system, as shown in Example 4-11.

Example 4-11 The mmdeldisk command: Removing disks from a GPFS file system

```
usa:/#mmdeldisk testmigr "nsdhdisk2;nsdhdisk3"
Deleting disks ...
Scanning system storage pool
Scanning file system metadata, phase 1 ...
Scan completed successfully.
Scanning file system metadata, phase 2 ...
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
1.91 % complete on Wed Jul 7 15:35:00 2010 ( 146491 inodes 443 MB)
8.38 % complete on Wed Jul 7 15:35:31 2010 ( 146491 inodes 1944 MB)
12.69 % complete on Wed Jul 7 15:36:02 2010 ( 146491 inodes 2944 MB)
```

```

17.00 % complete on Wed Jul 7 15:36:36 2010 ( 146498 inodes 3944 MB)
21.32 % complete on Wed Jul 7 15:38:37 2010 ( 146500 inodes 4946 MB)
25.63 % complete on Wed Jul 7 15:39:08 2010 ( 146501 inodes 5946 MB)
31.02 % complete on Wed Jul 7 15:39:40 2010 ( 146503 inodes 7197 MB)
36.41 % complete on Wed Jul 7 15:40:11 2010 ( 146504 inodes 8448 MB)
40.72 % complete on Wed Jul 7 15:40:42 2010 ( 146506 inodes 9449 MB)
45.04 % complete on Wed Jul 7 15:41:14 2010 ( 146507 inodes 10450 MB)
49.35 % complete on Wed Jul 7 15:41:45 2010 ( 146510 inodes 11451 MB)
54.74 % complete on Wed Jul 7 15:42:17 2010 ( 146510 inodes 12701 MB)
59.06 % complete on Wed Jul 7 15:42:48 2010 ( 146512 inodes 13702 MB)
62.29 % complete on Wed Jul 7 15:43:18 2010 ( 146514 inodes 14453 MB)
67.68 % complete on Wed Jul 7 15:43:50 2010 ( 146516 inodes 15704 MB)
72.00 % complete on Wed Jul 7 15:44:10 2010 ( 149504 inodes 16705 MB)
100.00 % complete on Wed Jul 7 15:44:10 2010

```

Scan completed successfully.

Checking Allocation Map for storage pool 'system'

tsdeldisk64 completed.

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

In this way, data has been completely migrated to the new disks, nsdhdisk4 and nsdhdisk5.

4. Verify this information by using the `mmdf` command as shown in Example 4-12.

Example 4-12 The mmdf command output: Data has been migrated to new disks

```

usa:/#mmdf testmigr
disk          disk size  failure holds   holds          free KB          free KB
name          in KB    group metadata data          in full blocks  in fragments
-----
Disks in storage pool: system (Maximum disk size allowed is 1.3 TB)
nsdhdisk4      26214400      80 yes    yes    14366464 ( 55%)    3432 ( 0%)
nsdhdisk5      26214400      80 yes    yes    14368000 ( 55%)    4256 ( 0%)
-----
(pool total)    52428800                                28734464 ( 55%)    7688 ( 0%)
=====
(total)         52428800                                28734464 ( 55%)    7688 ( 0%)
=====

Inode Information
-----
Number of used inodes:      4033
Number of free inodes:     145471
Number of allocated inodes: 149504
Maximum number of inodes:   149504

```

5. Delete the old NSD that is not currently associated to any file system, as shown in Example 4-13.

Example 4-13 The mmdelnsd command: Completely delete old NSD from cluster configuration

```

usa:/#mmdl nsd
File system  Disk name  NSD servers
-----
testmigr     nsdhdisk4  usa,germany
testmigr     nsdhdisk5  usa,germany
(free disk)  nsdhdisk2  germany,usa
(free disk)  nsdhdisk3  germany,usa

usa:/#mmdelnsd "nsdhdisk2;nsdhdisk3"

```



```
mmde1nsd: Processing disk nsdhdisk2
mmde1nsd: Processing disk nsdhdisk3
mmde1nsd: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

4.1.4 Reinstalling a node (mmsrdfs)

In certain situations, reinstalling a node (from the operating system perspective) that is part of a GPFS cluster might be necessary. After reinstallation of the operating system, other nodes in the cluster continue to view this node as being down (**mmgetstate -aL** command) because GPFS has not been installed and configured in the node.

In a situation such as this, you must follow several steps:

1. Reinstall GPFS product.
2. Configure the network so that the node is able to connect correctly to other nodes.

This step implies to also check for **ssh** or **rsh** configuration, depending on what is configured in the cluster.

Now the node is running and is able to connect to other nodes, GPFS is installed, and **ssh** or **rsh** is working correctly.

This example has three nodes in the cluster: **usa**, **germany**, and **slovenia**. The **slovenia** node has been completely reinstalled, from the operating system perspective. Next, the SSH keys, stored under **/root/.ssh** are re-created with the **ssh-keygen -t rsa** command, remote nodes in **authorized_keys** are added, and new **slovenia** keys on **authorized_keys** are put on other nodes. SSH is tested to be sure it is working correctly between nodes. GPFS, however, is still not configured. If you try to start it now, a message indicates that the node does not belong to a cluster; to other nodes, this node appears to be in an *unknown* status.

Example 4-14 shows the current cluster situation.

Example 4-14 Using mmgetstate -aL command to see current cluster node status

```
[root@slovenia var]# mmstartup
mmstartup: This node does not belong to a GPFS cluster.
mmstartup: Command failed. Examine previous error messages to determine cause.

usa:/#mmgetstate -aL
```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|-----------|--------|----------|-------------|------------|-------------|
| 1 | usa | 2 | 2 | 3 | active | quorum node |
| 2 | germany | 2 | 2 | 3 | active | quorum node |
| 4 | slovenia | 0 | 0 | 3 | unknown | quorum node |

The important aspect in this situation is that all GPFS configuration under **/var/mmfs** has been lost. If you have a backup of this directory, you can restore directly from the backup.

If you do not have a backup, you can copy **/var/mmfs/gen/mmsdrfs** from any other node in the cluster. You have to use the **scp** or the **rcp** commands, depending on which one your cluster is using (use the **mm1scluster** command to display the information). If you used the **mmauth** command to add remote clusters, you have also to copy files under the **/var/mmfs/ssl/*** location, which contains the cluster-wide SSL keys.

After completion, be sure that the GPFS configuration is the same on all nodes by issuing the following command:

```
mmchconfig NOOP=yes
```

If this command finishes successfully, you can start GPFS on the node you reinstalled. Although slovenia has been reinstalled, Example 4-15 shows copying the cluster data from the usa node in the existing cluster.

Example 4-15 Using the scp command to copy the GPFS configuration from node usa

```

usa:/#scp /var/mmfs/ssl/* slovenia:/var/mmfs/ssl/
authorized_keys          100% 361    0.4KB/s  00:00
id_rsa.pub               100% 816    0.8KB/s  00:00
id_rsa2                  100% 497    0.5KB/s  00:00
id_rsa_committed         100% 497    0.5KB/s  00:00
id_rsa_committed.cert    100% 449    0.4KB/s  00:00
id_rsa_committed.pub     100% 816    0.8KB/s  00:00
known_cluster.testcluster2.england 100% 691    0.7KB/s  00:00
openssl.conf             100% 183    0.2KB/s  00:00

usa:/#scp /var/mmfs/gen/mmsdrfs slovenia:/var/mmfs/gen
mmsdrfs                  100% 11KB  10.8KB/s  00:00

root@slovenia mmfs]# mmstartup
Fri Jul  9 17:18:14 EDT 2010: mmstartup: Starting GPFS ...
[root@slovenia mmfs]#

usa:/#mmgetstate -aL

```

| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
|-------------|-----------|--------|----------|-------------|------------|-------------|
| 1 | usa | 2 | 3 | 3 | active | quorum node |
| 2 | germany | 2 | 3 | 3 | active | quorum node |
| 4 | slovenia | 2 | 3 | 3 | active | quorum node |

4.2 Managing a GPFS cluster

This section describes and has examples of operations you can perform on your configured cluster.

4.2.1 Adding a node to the cluster

One of the benefits that GPFS provides is that you have a highly scalable cluster. You can start with a small cluster and add nodes to your cluster.

To add a new node to your GPFS cluster, you use the **mmaddnode** command. When adding a node, you may specify the node name, the TCP/IP interface that GPFS will use to contact all nodes, and the node designation:

► manager | client

Indicates whether a node is part of the node pool from which file system managers and token managers can be selected. The default is client.

► quorum | nonquorum

Indicates whether a node is counted as a quorum node. The default is nonquorum.

If you are designating a new node as a quorum node, and adminMode central is in effect for the cluster, GPFS must be down on all nodes in the cluster. Alternatively, you may choose to add the new nodes as nonquorum, and after GPFS has been successfully started on the new nodes, you can change their designation to quorum using the **mmchnode** command.

The following description is of the **mmaddnode** command man page:

```
mmaddnode -N {NodeDesc[,NodeDesc...] | NodeFile}
```

Use the **mmaddnode** command to add nodes to an existing GPFS cluster. On each new node, a mount point directory and character mode device is created for each GPFS file system.

Adhere to the following rules when adding nodes to a GPFS cluster:

- ▶ You may issue the command only from a node that already belongs to the GPFS cluster.
- ▶ Although a node may mount file systems from multiple clusters, the node itself may only be added to a single cluster using the **mmcrcluster** or **mmaddnode** command.
- ▶ The nodes must be available for the command to be successful. If any of the nodes listed are not available when the command is issued, a message listing those nodes is displayed. You must correct the problem on each node and reissue the command to add those nodes.
- ▶ After the nodes are added to the cluster, use the **mmchlicense** command to designate appropriate GPFS licenses to the new nodes.

We use the following procedure to add two nodes to an already working cluster:

1. Be sure that the network is properly configured to have new nodes be able to communicate with the older nodes through TCP/IP. Because this network configuration also includes name resolution, `/etc/hosts` entries must be updated on every nodes or on DNS if you are using one.
2. You must also have SSH working without password-request among all four nodes.
3. On both slovenia and england, run the **ssh-keygen -t rsa** command to have proper SSH configuration files, as shown in Example 4-16.

Example 4-16 Using ssh-keygen command to create new SSH keys

```
[root@slovenia .ssh]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
a9:32:7e:ea:ec:21:f6:67:a0:5e:4d:b8:b1:fb:f6:0b root@slovenia
```

4. The `/root/.ssh/authorized_keys` file must contain the public keys of all four nodes. To fill the `authorized_keys` file, perform the following steps:
 - a. Copy the `id_rsa.pub` public key file on one machine by using FTP.
 - b. Rename it with the source machine name, such as `id_rsa.usa.pub`.
 - c. Run the following command:

```
cat id_rsa.usa.pub >> authorized_keys
```

Repeat the procedure for every node.

When done, use the **ssh *nodename* date** command to test whether the SSH configuration works, without asking for the password.

5. After one `authorized_keys` file has been populated with all nodes, use FTP to copy public keys to the other nodes.

The next example adds two Linux nodes (slovenia and england) to the old two AIX nodes cluster. See the procedure in “Chapter 5. Installing GPFS on Linux nodes” in *General Parallel File System Concepts, Planning, and Installation Guide Version 3 Release 4*, GA76-0413. The following rpms file sets are installed on nodes england and slovenia:

```
[root@england ~]# rpm -qa | grep gpfs
gpfs.libsrc-3.4.0-0
gpfs.gpl-3.4.0-0
gpfs.src-3.4.0-0
gpfs.msg.en_US-3.4.0-0
gpfs.docs-3.4.0-0
gpfs.base-3.4.0-0
```

The step that is most challenging on Linux probably is compiling the GPFS open source portability layer. To be sure it succeeds, you must verify that every Linux prerequisite is already correctly installed. After the portability layer has been compiled on a node, be sure to create an rpm packet with the new binary. Then it is possible to simply install that packet using the rpm command on every other node that has the same Linux level and it is using the same architecture (x86 or power).

Now, two new nodes with GPFS are installed but not configured and the SSH connection is already working between all four nodes.

Add the two new nodes to the cluster:

1. From node usa, add node slovenia:

```
usa:/#mmaddnode-N england:quorum
Thu Jun 24 12:11:05 EDT 2010: mmaddnode: Processing node england
The authenticity of host 'england (192.168.101.143)' can't be established.
RSA key fingerprint is 60:e3:5a:ca:a0:c3:c4:46:bb:80:1c:4d:0a:57:97:a5.
Are you sure you want to continue connecting (yes/no)? yes
mmaddnode: Command successfully completed
mmaddnode: Warning: Not all nodes have proper GPFS license designations.
      Use the mmchlicense command to designate licenses as needed.
mmaddnode: Propagating the cluster configuration data to all
      affected nodes. This is an asynchronous process.
```

2. As suggested by the command output, designate a proper GPFS license for the new configured node:

```
usa:/#mmchlicense server --accept -N england
The following nodes will be designated as possessing GPFS server licenses:
      england
mmchlicense: Command successfully completed
mmchlicense: Propagating the cluster configuration data to all
      affected nodes. This is an asynchronous process.
```

The new node is configured in the cluster and is now a member of the GPFS cluster.

By using the same commands, add the node named `england`. Four nodes are now in the cluster, as shown in Example 4-17.

Example 4-17 Four cluster nodes up and running

| [root@england ~]# mmgetstate -aL | | | | | | |
|----------------------------------|-----------|--------|----------|-------------|------------|-------------|
| Node number | Node name | Quorum | Nodes up | Total nodes | GPFS state | Remarks |
| 1 | usa | 1* | 4 | 4 | active | quorum node |
| 2 | germany | 1* | 4 | 4 | active | quorum node |
| 3 | england | 1* | 4 | 4 | active | quorum node |
| 4 | slovenia | 1* | 4 | 4 | active | quorum node |

4.2.2 Adding a disk to a file system

Adding a disk to a file system can be a regular occurrence if your file systems are rapidly growing in size, and you need more free disk space. GPFS gives you the option to add disks to your file system online, while the cluster is active, and the new empty space can be used as soon as the command completes. When you add a new disk using the `mmaddisk` command, you can decide to use the `-r` balance flag if you want all existing files in your file system to use the new free space. Rebalancing is an I/O-intensive and time consuming operation, and is important only for file systems with large files that are mostly invariant. You have to consider that during the file system rebalancing, GPFS operations might be slower than usual, so be sure to do a rebalancing when there is not high I/O on the file system. You may also add a new disk without rebalancing, and do a complete rebalancing later, with the `mmrestripefs` command.

Perform the following steps to add a new disk to a GPFS file system:

1. Determine whether any already-configured NSDs are available (not yet associated to a file system) by using the `mm1snd -F` command.
2. If there are no available NSD that are already configured in the cluster, you must create a new NSD on a disk device. To create an NSD, you must know the following information:
 - The name of the device of the node on which you ran the `mm1snd -F` command (other nodes in the cluster are automatically discovered)
 - The type of disk usage that this NSD will handle (dataAndMetadata, dataOnly, and metadataOnly descOnly descriptors)
 - Whether assigning one or more NSD servers to this new NSD is necessary (or whether it is only directly attached to the nodes)
 - The failure group on which to configure it (to have GPFS manage replica)
 - Whether the new NSD has to be in a storage pool separate from the system.

Usually you have to define one or more (up to eight) servers to the NSD if data that is contained in this NSD must also be accessible by nodes that do not have direct access to that disk. When a node must read or write data to a disk on which it does not have direct access, the node has to ask, through the network, for a node that is on the server list for that NSD. Everything is done through the network. If no NSD servers are specified, the disk is seen as a *direct attach* disk; if other servers are configured in the cluster and need to access data on that NSD, they cannot use this server.

3. Certain items, such as the NSD server list, can be changed later with the `mmchnsd` command. Other items, such as the NSD name or the storage pool, are impossible to change. Instead, to change the NSD usage or the failure group, you have to use `mmchdisk` command.

In our lab, we add four new disks and allocate them to the Linux boxes in the cluster. You have to compile a plain text file, which will be modified by the `mmcrnsd` command. The file we used in our test is shown in Example 4-18.

Example 4-18 The file in our test

```
germany:/gpfs#cat nsdlinux.conf
dm-2:slovenia,england::dataAndMetadata:40:nsddm2
dm-3:slovenia,england::dataAndMetadata:40:nsddm3
dm-4:slovenia,england::dataAndMetadata:40:nsddm4
dm-5:slovenia,england::dataAndMetadata:40:nsddm5
```

We have four disks, seen through multipath. They are attached to the slovenia and england nodes. Because data on these disks is to be accessed also by the usa and germany nodes, we specify slovenia and england as primary servers. The `mmcrnsd` command is shown in Example 4-19.

Example 4-19 Using `mmcrnsd` command to create new disks

```
germany:/gpfs#mmcrnsd -F nsdlinux.conf
mmcrnsd: Processing disk dm-2
mmcrnsd: Processing disk dm-3
mmcrnsd: Processing disk dm-4
mmcrnsd: Processing disk dm-5
mmcrnsd: Propagating the cluster configuration data to all
        affected nodes. This is an asynchronous process.
germany:/gpfs#mmlnsd
```

| File system | Disk name | NSD servers |
|-------------|-----------|------------------|
| ----- | | |
| mantestfs | nsdhdisk2 | usa,germany |
| mantestfs | nsdhdisk3 | usa,germany |
| mantestfs | nsdhdisk4 | usa,germany |
| mantestfs | nsdhdisk5 | usa,germany |
| (free disk) | nsddm2 | slovenia,england |
| (free disk) | nsddm3 | slovenia,england |
| (free disk) | nsddm4 | slovenia,england |
| (free disk) | nsddm5 | slovenia,england |

- Now the new four NSD disks are ready to be put in a file system with `mmadddisk` command see Example 4-20.

Example 4-20 `mmadddisk` to add NSD in the file system

```
germany:/gpfs#mmadddisk /mantestfs -F nsdlinux.conf
The following disks of mantestfs will be formatted on node usa:
    nsddm2: size 26214400 KB
    nsddm3: size 26214400 KB
    nsddm4: size 26214400 KB
    nsddm5: size 26214400 KB
Extending Allocation Map

Checking Allocation Map for storage pool 'system'
Completed adding disks to file system mantestfs.
mmadddisk: Propagating the cluster configuration data to all
        affected nodes. This is an asynchronous process.
```

The file system has changed, as shown in Example 4-21.

Example 4-21 File system change

| Filesystem | 512-blocks | Free | %Used | Iused | %Iused | Mounted on |
|----------------|------------|-----------|-------|-------|--------|------------|
| Before: | | | | | | |
| /dev/mantestfs | 408944640 | 305897472 | 26% | 4539 | 3% | /mantestfs |
| After: | | | | | | |
| /dev/mantestfs | 618659840 | 515255808 | 17% | 4543 | 3% | /mantestfs |

The new disk is formatted and in use, as shown in Example 4-22.

Example 4-22 New disks in use on the file system

| germany:/gpfs#mmdf /dev/mantestfs | | | | | | |
|---|-----------------|---------------|----------------|------------|------------------------|----------------------|
| disk name | disk size in KB | failure group | holds metadata | holds data | free KB in full blocks | free KB in fragments |
| ----- | | | | | | |
| Disks in storage pool: system (Maximum disk size allowed is 1.1 TB) | | | | | | |
| nsddm5 | 26214400 | 40 | yes | yes | 19777536 (75%) | 488 (0%) |
| nsddm4 | 26214400 | 40 | yes | yes | 19771904 (75%) | 488 (0%) |
| nsddm3 | 26214400 | 40 | yes | yes | 19771392 (75%) | 248 (0%) |
| nsddm2 | 26214400 | 40 | yes | yes | 19774976 (75%) | 248 (0%) |
| nsdhdisk5 | 26214400 | 80 | yes | yes | 19730944 (75%) | 15272 (0%) |
| nsdhdisk4 | 26214400 | 80 | yes | yes | 19732224 (75%) | 12216 (0%) |
| nsdhdisk3 | 26214400 | 80 | yes | yes | 19731712 (75%) | 13584 (0%) |
| nsdhdisk2 | 26214400 | 80 | yes | yes | 19731712 (75%) | 16104 (0%) |
| ----- | | | | | | |
| (pool total) | 209715200 | | | | 158022400 (75%) | 58648 (0%) |
| ===== | | | | | | |
| (total) | 209715200 | | | | 158022400 (75%) | 58648 (0%) |
| ===== | | | | | | |
| Inode Information | | | | | | |
| ----- | | | | | | |
| Number of used inodes: | 4543 | | | | | |
| Number of free inodes: | 196161 | | | | | |
| Number of allocated inodes: | 200704 | | | | | |
| Maximum number of inodes: | 200704 | | | | | |

The **mm_lsattr** command is helpful when you want to verify certain information about files that are in your GPFS file system. It is also helpful for determining whether a single file is correctly replicated, as shown in Example 4-23.

Example 4-23 The mmlsaatr command: Determining whether a single file is replicated correctly

| | |
|-------------------------------------|-----------|
| usa:/testmigr#mmlsattr -L testfile0 | |
| file name: | testfile0 |
| metadata replication: | 2 max 2 |
| data replication: | 2 max 2 |
| immutable: | no |
| appendOnly: | no |
| flags: | |
| storage pool name: | system |
| fileset name: | root |
| snapshot name: | |

4.2.3 GPFS network usage

GPFS can use more than two networks:

- ▶ The administrator network is the network used when issuing a GPFS command.
- ▶ The daemon network is used for all non-admin traffic between all nodes.

By having two networks, you can have a slower network for passing administration-related data, and a faster network for passing customer data.

Both the administrator and the daemon interfaces on the nodes can be modified by using the **mmchconfig** command.

For example in our test cluster, we are using the same interface for both administrator and daemon networks. See Example 4-24.

Example 4-24 Our cluster network configuration

| Node | Daemon node name | IP address | Admin node name | Designation |
|------|------------------|-----------------|-----------------|-------------|
| 1 | usa | 192.168.101.132 | usa | quorum |
| 2 | germany | 192.168.101.142 | germany | quorum |
| 4 | slovenia | 192.168.101.133 | slovenia | quorum |

Now, we want to switch to the daemon network because we purchased a faster network device and we want to separate the two networks. The network for administration remains on 192.168.101; the new daemon network is now on the 10.10.10 subnet.

The steps are as follows:

1. First, the network must be configured. Therefore, new IP addresses must be correctly configured in the operating system and with resolved names.

In our lab, we add all the nodes as new entries in the `/etc/hosts` file.

```
10.10.10.1 germany_daemon
10.10.10.2 usa_daemon
10.10.10.3 slovenia_daemon
```

These are the IP labels for the network. We configured the interfaces on new adapters and they are working correctly.

2. From GPFS, configure the new network.

Prior to this change, GPFS must be stopped on all nodes, if it is active somewhere the command will tell you. So, **mmshutdown -a** command.

When GPFS is down, use the **mmchnode** command to change GPFS network interfaces, as shown in Example 4-25.

Example 4-25 The mmchnode command: Changing GPFS network interfaces

```
usa:/#mmchnode -N usa --daemon-interface=usa_daemon
Tue Jun 29 16:32:03 EDT 2010: mmchnode: Processing node usa
Verifying GPFS is stopped on all nodes ...

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

usa:/#mmchnode -N germany --daemon-interface=germany_daemon
Tue Jun 29 16:32:55 EDT 2010: mmchnode: Processing node germany
```


Verifying GPFS is stopped on all nodes ...

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

```
usa:/#mmchnode -N slovenia --daemon-interface=slovenia_daemon
Tue Jun 29 16:33:40 EDT 2010: mmchnode: Processing node slovenia
Verifying GPFS is stopped on all nodes ...
```

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

Now the GPFS network configuration looks similar to Example 4-26.

Example 4-26 New GPFS network configuration

| Node | Daemon node name | IP address | Admin node name | Designation |
|------|------------------|------------|-----------------|-------------|
| 1 | usa_daemon | 10.10.10.2 | usa | quorum |
| 2 | germany_daemon | 10.10.10.1 | germany | quorum |
| 4 | slovenia_daemon | 10.10.10.3 | slovenia | quorum |

In this way, GPFS administration commands use the 192.168.100 network, and GPFS data uses the 10.10.10 (faster) network.

You may also change the administrator network by using the **mmchconfig** command, as always, but instead of using the **--daemon-interface** option, you must use the **--admin-interface** option.

Be sure to pay attention to rsh and SSH configuration; the new interface must be trusted before configuring it in the cluster.

4.2.4 Adding a remote cluster

GPFS offers the possibility to mount file systems that are owned and managed by another GPFS cluster. This procedure can be performed in three steps:

1. Authorize the remote cluster by using the **mmauth** command.
2. Add the remote cluster by using the **mmremorecluster add** command.
3. Add the remote file system by using the **mmremote fs add** command.

Before starting the procedure, the correct version must be installed on cluster nodes because the procedure is based on OpenSSL.

Example 4-27 shows cluster 1, which will export the `/mounttestfs2` file system.

Example 4-27 Cluster 1 configuration

```
# mmlscluster
```

```
GPFS cluster information
=====
```

```
GPFS cluster name:      testcluster2.england
GPFS cluster id:        13882457517502213483
GPFS UID domain:        testcluster2.england
Remote shell command:   /usr/bin/ssh
Remote file copy command: /usr/bin/scp
```

GPFS cluster configuration servers:

```
-----
Primary server:  england
Secondary server: (none)
```

| Node | Daemon node name | IP address | Admin node name | Designation |
|-----------------|------------------|-----------------|-----------------|-------------|
| 1 | england | 192.168.101.143 | england | quorum |
| ----- | | | | |
| /dev/mantestfs2 | 26214400 | 159744 26054656 | 1% /mountestfs2 | |

Example 4-28 shows cluster 2, which will mount the remote file system (remotefs) from cluster1.

Example 4-28 Cluster 2 configuration

```
germany:/gpfs#mmlscluster
```

```
GPFS cluster information
=====
```

```
GPFS cluster name:      mantest.usa
GPFS cluster id:        13882457470256956975
GPFS UID domain:        mantest.usa
Remote shell command:   /usr/bin/ssh
Remote file copy command: /usr/bin/scp
```

GPFS cluster configuration servers:

```
-----
Primary server:  usa
Secondary server: germany
```

| Node | Daemon node name | IP address | Admin node name | Designation |
|-------|------------------|------------|-----------------|-------------|
| ----- | | | | |
| 1 | usa_daemon | 10.10.10.2 | usa | quorum |
| 2 | germany_daemon | 10.10.10.1 | germany | quorum |
| 4 | slovenia_daemon | 10.10.10.3 | slovenia | quorum |

Perform the following steps:

1. Generate new keys on cluster 1, as shown in Example 4-29.

Example 4-29 Generate new ssh-keys on cluster1

```
-----
# hostname
england
# mmauth genkey new
Generating RSA private key, 512 bit long modulus
...+++++++
.....+++++++
e is 65537 (0x10001)
writing RSA key
mmauth: Command successfully completed

# mmauth update . -l AUTHONLY
Verifying GPFS is stopped on all nodes ...
mmauth: Command successfully completed
-----
```

2. Generate new keys on cluster 2 as shown in Example 4-30.

Example 4-30 Generate new ssh-keys on cluster 2

```
germany:/#mmauth genkey new
Generating RSA private key, 512 bit long modulus
.....+++++++
.....+++++++
e is 65537 (0x10001)
writing RSA key
mmauth: Command successfully completed
```

The mmauth update must be run with the cluster stopped on alla nodes:

```
germany:/#mmauth update . -l AUTHONLY
Verifying GPFS is stopped on all nodes ...
mmauth: Command successfully completed
```

3. On cluster 1, add the authorization for cluster 2, as shown in Example 4-31.

Example 4-31 The mmauth add command: Adding authorization for cluster 2

```
# mmauth add mantest.usa -k id_rsa.pub.usa
mmauth: Command successfully completed
```

At the -k option must be given the keys of cluster 2

4. Grant the authorization of cluster 2 to the file system, as shown in Example 4-32.

Example 4-32 The mmauth grant command: Granting file system authorization

```
# mmauth grant mantest.usa -f /dev/mantestfs2

mmauth: Granting cluster mantest.usa access to file system mantestfs2:
access type rw; root credentials will not be remapped.

mmauth: Command successfully completed
```

Now, cluster 2 is authorized to be added on cluster 1 and to mount the file system. In this way we added cluster 1 to cluster 2 as shown in Example 4-33.

Example 4-33 The mmremotecluster add command: Adding a remote cluster to current cluster

```
germany:/tmp/auth#mmremotecluster add testcluster2.england -n england -k
/tmp/auth/id_rsa.pub.england

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

5. Add the remotefs to cluster 2, as shown in Example 4-34.

Example 4-34 The mmremotefs add command: Adding a remote file system to the current cluster

```
germany:/tmp/auth#mmremotefs add /dev/remotemantestfs2 -f /dev/mantestfs2 -C
testcluster2.England
```

```
-T /remotemantestfs2
```

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

In this way, you can see which remotefs has been configured by using the **mmremotefs** command, as shown Example 4-35.

Example 4-35 Using the mmremotefs command to see configured remote file systems

```
germany:/#mmremotefs show all
Local Name Remote Name Cluster name Mount Point Mount Options Automount Drive Priority
remotemantestfs2 mantestfs2 testcluster2.England /remotemantestfs2 rw no -
0
```

6. Mount the remotefs on cluster 2 by using the standard **mmount** command. See Example 4-36.

Example 4-36 Using the mmount command to mount remote file system

```
germany:/tmp/auth#mmmount /remotemantestfs2
Thu Jul 1 11:48:00 EDT 2010: mmount: Mounting file systems ...
/dev/remotemantestfs2 52428800 52108800 1% 4038 12%
/remotemantestfs2
```

4.2.5 Adding or removing a file system

To add a new file system to a cluster, use the **mmcrfs** command. The command has several options. See Example 4-37.

Example 4-37 The mmcrfs command options

```
germany:/tmp/power#mmcrfs -?
Usage:
mmcrfs Device {"DiskDesc[;DiskDesc...]" | -F DescFile}
[-A {yes | no | automount}] [-B BlockSize] [-D {posix | nfs4}]
[-E {yes | no}] [-j {cluster | scatter}] [-k {posix | nfs4 | all}]
[-K {no | whenpossible | always}] [-L LogFileSize]
[-m DefaultMetadataReplicas] [-M MaxMetadataReplicas]
[-n NumNodes] [-Q {yes | no}] [-r DefaultDataReplicas]
[-R MaxDataReplicas] [-S {yes | no}] [-T Mountpoint]
[-t DriveLetter] [-v {yes | no}] [-z {yes | no}]
[--filesetdf | --nofilesetdf]
[--inode-limit MaxNumInodes[:NumInodesToPreallocate]]
[--mount-priority Priority] [--version VersionString]
```

The following list shows the options required to create a new file system with the **mmcrfs** command, as in Example 4-38:

- Device** A name that you want to give to this new device,
- DiskDesc** The disk descriptors list, or directly the disk descriptions file (-F)
- F DescFile** The file that has been modified by the **mmcrnsd** command.
- T** Sets the mount point, here an example of a newly created file system.

Example 4-38 The mmcrfs command: Creating a completely new file system

```
germany:/gpfs#cat nsd.conf
# hdisk1::dataAndMetadata:80:nsdhdisk2:
nsdhdisk2::dataAndMetadata:80::
# hdisk2::dataAndMetadata:80:nsdhdisk3:
nsdhdisk3::dataAndMetadata:80::
# hdisk3::dataAndMetadata:80:nsdhdisk4:
nsdhdisk4::dataAndMetadata:80::
# hdisk4::dataAndMetadata:80:nsdhdisk5:
nsdhdisk5::dataAndMetadata:80::
```

```
germany:/gpfs#mmcrfs testnewfs -F nsd.conf -T /testnewfs
```

The following disks of testnewfs will be formatted on node usa:

```
nsdhdisk2: size 26214400 KB
nsdhdisk3: size 26214400 KB
nsdhdisk4: size 26214400 KB
nsdhdisk5: size 26214400 KB
```

Formatting file system ...

Disks up to size 1.1 TB can be added to storage pool 'system'.

Creating Inode File

Creating Allocation Maps

Clearing Inode Allocation Map

Clearing Block Allocation Map

Formatting Allocation Map for storage pool 'system'

Completed creation of file system /dev/testnewfs.

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

Notice the output line:

Disks up to size 1.1 TB can be added to storage pool 'system'.

This value is calculated during file system creation time when the **mmcrfs** command is issued, because on each disk there is a portion where GPFS stores its control information and a disk bitmap that shows, in the disk, which area is allocated or not. Therefore, if you try to add a bigger disk to the same storage pool, and the space is not enough, GPFS provides, in advance, the maximum disk size you can add to this storage pool.

Now, we re-create the same file system with the same parameters, but exclude the nsdhdisk2 disk, resulting in smaller disks, as shown Example 4-39 on page 212.

Example 4-39 The mmcrfs command creates a new file system but this time with smaller disks

```
germany:/gpfs#cat nsd.conf
# hdisk2::dataAndMetadata:80:nsdhdisk3:
nsdhdisk3::dataAndMetadata:80::
# hdisk3::dataAndMetadata:80:nsdhdisk4:
nsdhdisk4::dataAndMetadata:80::
# hdisk4::dataAndMetadata:80:nsdhdisk5:
nsdhdisk5::dataAndMetadata:80::

germany:/gpfs#mmcrfs testnewfs -F nsd.conf -T /testnewfs
```

The following disks of testnewfs will be formatted on node usa:

```
nsdhdisk3: size 26214400 KB
nsdhdisk4: size 26214400 KB
nsdhdisk5: size 26214400 KB
```

Formatting file system ...

Disks up to size 415 GB can be added to storage pool 'system'.

Creating Inode File

Creating Allocation Maps

Clearing Inode Allocation Map

Clearing Block Allocation Map

Formatting Allocation Map for storage pool 'system'

Completed creation of file system /dev/testnewfs.

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

Now, the maximum disk size that can be added to disk file system is much smaller, 415 GB, instead of 1.1 TB of the first try.

In this way, the file system contains LUNs only of a necessary capacity (for this moment), although you have the possibility of adding bigger LUNs in the future.

A parameter that can affect GPFS performance is the block size (-B). For information about setting up this parameter, depending on your environment, see Chapter 5, “Performance and tuning” on page 229.

Removing a file system

To remove a file system from your cluster, simply use the **mmdelfs** command. However, first be sure that the file system is unmounted in every node in the cluster. Keep in mind that this step does not delete the disks, but the NSD will be seen as “free” after the file system is deleted. In the event that the disks are damaged or not available, the **-p** parameter must be used. Although GPFS cannot mark them as free, the file system will be removed anyway.

We deleted the previously created /dev/testnewfs file system, as shown in Example 4-40.

Example 4-40 Using mmdelfs command to completely delete a file system

```
germany:/gpfs#mmdelfs /dev/testnewfs
All data on following disks of testnewfs will be destroyed:
nsdhdisk3
nsdhdisk4
nsdhdisk5
Completed deletion of file system /dev/testnewfs.
mmdelfs: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

germany:/gpfs#mmlnsd

| File system | Disk name | NSD servers |
|-------------|-----------|-------------|
| (free disk) | nsdhdisk2 | usa,germany |
| (free disk) | nsdhdisk3 | usa,germany |
| (free disk) | nsdhdisk4 | usa,germany |
| (free disk) | nsdhdisk5 | usa,germany |

4.2.6 Enabling GPFS replication

GPFS provides multiple features to increase data availability of your system. One feature is the file system synchronous replication. You can set the GPFS replication to work on all the file systems or only on some files. Currently the maximum number of copies allowed by GPFS is two. This means that two copies of data and metadata are kept replicated to ensure data availability. You can choose to have only metadata replicated to save disk space and to increase performance because enabled replicas requires additional write times. GPFS replica is based on failure group configuration. When you add new disks to the GPFS file system, you can configure to which failure group the disks belong. You have to configure the same failure group for disks that have in common the same point of failure, for example, the same disk storage system. If you have one storage system in one location and one storage system in another location, you configure the disks belonging to separate storage systems in separate failure groups. In this way, GPFS knows that it has to replicate data and metadata from one storage system to the other. In case of a complete storage system failure, GPFS has all the data correctly replicated on the other storage environment.

This section describes the following replica characteristics:

- ▶ Default metadata
- ▶ Maximum metadata
- ▶ Default data
- ▶ Maximum data

Additional information about the GPFS replication parameters is in the *General Parallel File System Advanced Administration Guide Version 3 Release 4*, SC23-5182.

Default metadata replicas

The default number of copies of metadata for all files in the file system may be specified at file system creation by using the **-m** option on the **mmcrfs** command, or changed at a later time by using the **-m** option on the **mmchfs** command. This value must be equal to or less than **MaxMetadataReplicas**, and cannot exceed the number of failure groups with disks that can store metadata. The allowable values are 1 or 2, with a default of 1.

Maximum metadata replicas

The maximum number of copies of metadata for all files in the file system can be specified at file system creation by using the **-M** option on the **mmcrfs** command. The default is 2. The allowable values are 1 or 2, but it cannot be lower than the value of **DefaultMetadataReplicas**. This value cannot be changed.

Default data replicas

The default replication factor for data blocks may be specified at file system creation by using the **-r** option on the **mmcrfs** command, or changed at a later time by using the **-r** option on the **mmchfs** command. This value must be equal to or less than **MaxDataReplicas**, and the value

cannot exceed the number of failure groups with disks that can store data. The allowable values are 1 and 2, with a default of 1.

If you want to change the data replication factor for the entire file system, the data disk in each storage pool must have a number of failure groups equal to or greater than the replication factor. For example, a failure with error messages occur if you try to change the replication factor for a file system to 2 but the storage pool has only one failure group.

Maximum data replicas

The maximum number of copies of data blocks for a file can be specified at file system creation by using the **-R** option on the **mmcrfs** command. The default is 2. The allowable values are 1 and 2, but cannot be lower than the value of **DefaultDataReplicas**. This value cannot be changed.

File system management examples

In Example 4-41, we start with the **/dev/testmigr** file system that has only two NSDs in it. Example 4-41 shows the configuration.

Example 4-41 File system for test configuration

| disk name | disk size in KB | failure group | holds metadata | holds data | free KB in full blocks | free KB in fragments |
|---|-----------------|---------------|----------------|------------|------------------------|----------------------|
| ----- | | | | | | |
| Disks in storage pool: system (Maximum disk size allowed is 1.3 TB) | | | | | | |
| nsdhdisk4 | 26214400 | 80 | yes | yes | 14369024 (55%) | 736 (0%) |
| nsdhdisk5 | 26214400 | 80 | yes | yes | 14361344 (55%) | 11048 (0%) |
| ----- | | | | | | |
| (pool total) | 52428800 | | | | 28730368 (55%) | 11784 (0%) |
| ===== | | | | | | |
| (total) | 52428800 | | | | 28730368 (55%) | 11784 (0%) |
| ===== | | | | | | |
| Inode Information | | | | | | |
| ----- | | | | | | |
| Number of used inodes: | | | | 4033 | | |
| Number of free inodes: | | | | 145471 | | |
| Number of allocated inodes: | | | | 149504 | | |
| Maximum number of inodes: | | | | 149504 | | |

To more fully secure the data, in this scenario we purchase new storage and activate the GPFS replica, for data and metadata. We configure two new disks on the OS and create two new NSDs on them. New NSDs have a different failure group number of initial disks, they are in a new system. The following file is used to create NSD:

```
# hdisk2:usa,germany::dataAndMetadata:60:nsdhdisk2:
nsdhdisk2::dataAndMetadata:60::
# hdisk3:usa,germany::dataAndMetadata:60:nsdhdisk3:
nsdhdisk3::dataAndMetadata:60::
```

We add new NSDs to the file system by using the **mmaddisk** command, as shown in Example 4-42 on page 215.

Example 4-42 Using mmadddisk command to add disks

```
usa:/# mmadddisk /dev/testmigr -F nsd1.conf
```

The following disks of testmigr will be formatted on node germany:

nsdhdisk2: size 125829120 KB

nsdhdisk3: size 26214400 KB

Extending Allocation Map

Checking Allocation Map for storage pool 'system'

Completed adding disks to file system testmigr.

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

The **mmdf** command output is shown in Example 4-43.

Example 4-43 The mmdf command output to view new disks configured in the file system

```
usa:/# mmdf /dev/testmigr
```

| disk name | disk size in KB | failure holds group metadata | holds data | free KB in full blocks | free KB in fragments |
|---|--------------------|---------------------------------|---------------|---------------------------|-------------------------|
| ----- | | | | | |
| Disks in storage pool: system (Maximum disk size allowed is 1.3 TB) | | | | | |
| nsdhdisk2 | 125829120 | 60 yes | yes | 125826816 (100%) | 248 (0%) |
| nsdhdisk3 | 26214400 | 60 yes | yes | 26212096 (100%) | 248 (0%) |
| nsdhdisk4 | 26214400 | 80 yes | yes | 14369024 (55%) | 736 (0%) |
| nsdhdisk5 | 26214400 | 80 yes | yes | 14361344 (55%) | 11048 (0%) |
| ----- | | | | | |
| (pool total) | 204472320 | | | 180769280 (88%) | 12280 (0%) |
| ===== | | | | | |
| (total) | 204472320 | | | 180769280 (88%) | 12280 (0%) |
| ===== | | | | | |
| Inode Information | | | | | |
| ----- | | | | | |
| Number of used inodes: | 4033 | | | | |
| Number of free inodes: | 145471 | | | | |
| Number of allocated inodes: | 149504 | | | | |
| Maximum number of inodes: | 149504 | | | | |

As you can see, new disks are added, they are free, and data remains on two old disks, nsdhdisk4 and nsdhdisk5. We configure the file system to have data and metadata replicated:

```
usa:/# mmchfs /dev/testmigr -m 2 -r 2
```

Now, configuration of the file system has been changed, the default option for data and metadata is 2, so we have two copies. However, data that was already in the file system is unchanged.

Because we have to force GPFS to properly replicate all files, according to the new configuration, we use the **-R** option with the **mmrestripefs** command, as shown in Example 4-44.

Example 4-44 Using mmrestripefs command to replicate all files

```

usa:/#mmrestripefs /dev/testmigr -R
Scanning file system metadata, phase 1 ...
Scan completed successfully.
Scanning file system metadata, phase 2 ...
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
 2.74 % complete on Wed Jul 7 17:39:36 2010 ( 146491 inodes 638 MB)
 9.20 % complete on Wed Jul 7 17:40:02 2010 ( 146491 inodes 2139 MB)
19.94 % complete on Wed Jul 7 17:40:29 2010 ( 146491 inodes 4639 MB)
32.85 % complete on Wed Jul 7 17:40:54 2010 ( 146498 inodes 7642 MB)
45.76 % complete on Wed Jul 7 17:41:18 2010 ( 146500 inodes 10644 MB)
58.67 % complete on Wed Jul 7 17:41:44 2010 ( 146501 inodes 13647 MB)
71.58 % complete on Wed Jul 7 17:42:11 2010 ( 146503 inodes 16649 MB)
82.33 % complete on Wed Jul 7 17:42:37 2010 ( 146504 inodes 19151 MB)
93.09 % complete on Wed Jul 7 17:43:03 2010 ( 146506 inodes 21653 MB)
93.11 % complete on Wed Jul 7 17:43:29 2010 ( 146507 inodes 24656 MB)
93.13 % complete on Wed Jul 7 17:44:20 2010 ( 146510 inodes 30161 MB)
93.15 % complete on Wed Jul 7 17:45:10 2010 ( 146514 inodes 35665 MB)
100.00 % complete on Wed Jul 7 17:45:55 2010
Scan completed successfully.

```

Example 4-45 shows the output of the file system after the command **mmrestripefs -R** command ends.

Example 4-45 The mmdf output after the mmrestripefs -R command runs

```

usa:/#mmdf /dev/testmigr
disk      disk size  failure holds   holds      free KB      free KB
name      in KB    group metadata data      in full blocks      in fragments
-----
Disks in storage pool: system (Maximum disk size allowed is 1.3 TB)
nsdhdisk2 125829120    60 yes    yes    113977856 ( 91%)    8784 ( 0%)
nsdhdisk3 26214400    60 yes    yes    14361344 ( 55%)    8888 ( 0%)
nsdhdisk4 26214400    80 yes    yes    14386432 ( 55%)    3304 ( 0%)
nsdhdisk5 26214400    80 yes    yes    14340864 ( 55%)    11552 ( 0%)
-----
(pool total) 204472320    157066496 ( 77%)    32528 ( 0%)
=====
(total) 204472320    157066496 ( 77%)    32528 ( 0%)

Inode Information
-----
Number of used inodes:      4036
Number of free inodes:     145468
Number of allocated inodes: 149504
Maximum number of inodes:  149504

```

4.2.7 Exporting or importing a file system

GPFS allows you to move a GPFS file system definition from one cluster to another (an export-import function) or to delete file system definition and save its configuration, to be able to import the file system to a later time.

Two commands are available:

- ▶ The **mmexportfs** command saves the current file system definition in a file and exports it from the current cluster.
- ▶ The **mmimportfs** command imports the file system in a cluster where this file system is not already known.

The primary uses of these commands are to move a file system from one cluster to another or to mount a file system on another cluster, for example for backup purposes.

This section shows how these commands work.

The first cluster is named `mantest.usa` and consists of three nodes: `slovenia`, `usa` and `germany`.

We create two new NSDs, `nsddm2` and `nsddm3`, that are attached only to the node `slovenia`, which is a Linux node. On these two NSDs, we create a new file system named `testexportfs`. Example 4-46 shows the file system configuration.

Example 4-46 The `mmddf`, `mmlsfs`, and `mmlnsd` commands to show file system configuration

```
germany:/gpfs#mmddf /dev/testexportfs
disk      disk size  failure holds    holds    free KB    free KB
name      in KB      group metadata data      in full blocks    in fragments
-----
Disks in storage pool: system (Maximum disk size allowed is 234 GB)
nsddm2    26214400    40 yes    yes    26128384 (100%)    488 ( 0%)
nsddm3    26214400    40 yes    yes    26128896 (100%)    456 ( 0%)
-----
(pool total)    52428800    52257280 (100%)    944 ( 0%)
=====
(total)    52428800    52257280 (100%)    944 ( 0%)

Inode Information
-----
Number of used inodes:    4038
Number of free inodes:    48186
Number of allocated inodes:    52224
Maximum number of inodes:    52224
-----

germany:/gpfs#mmlsfs /dev/testexportfs
flag      value    description
-----
-f        8192    Minimum fragment size in bytes
-i        512    Inode size in bytes
-I        16384    Indirect block size in bytes
-m        1        Default number of metadata replicas
-M        2        Maximum number of metadata replicas
-r        1        Default number of data replicas
-R        2        Maximum number of data replicas
```

| | | |
|------------------|--------------------------|--|
| -j | cluster | Block allocation type |
| -D | nfs4 | File locking semantics in effect |
| -k | all | ACL semantics in effect |
| -n | 32 | Estimated number of nodes that will mount file |
| system | | |
| -B | 262144 | Block size |
| -Q | none | Quotas enforced |
| | none | Default quotas enabled |
| -V | 12.03 (3.4.0.0) | File system version |
| -u | yes | Support for large LUNs? |
| -z | no | Is DMAPI enabled? |
| -L | 4194304 | Logfile size |
| -E | yes | Exact mtime mount option |
| -S | no | Suppress atime mount option |
| -K | whenpossible | Strict replica allocation option |
| --create-time | Mon Jul 12 17:47:37 2010 | File system creation time |
| --fastea | yes | Fast external attributes enabled? |
| --filesetdf | no | Fileset df enabled? |
| --inode-limit | 52224 | Maximum number of inodes |
| -P | system | Disk storage pools in file system |
| -d | nsddm2;nsddm3 | Disks in file system |
| -A | yes | Automatic mount option |
| -o | none | Additional mount options |
| -T | /testexportfs | Default mount point |
| --mount-priority | 0 | Mount priority |

germany:/gpfs#mmlnsd

| File system | Disk name | NSD servers |
|--------------|-----------|-------------|
| testexportfs | nsddm2 | slovenia |
| testexportfs | nsddm3 | slovenia |
| testmigr | nsdhdisk4 | usa,germany |
| testmigr | nsdhdisk5 | usa,germany |
| testmigr | nsdhdisk2 | usa,germany |
| testmigr | nsdhdisk3 | usa,germany |

We are ready to export. The file system must be unmounted on all nodes to be exported, because the definition will be deleted from this cluster. We run the **mmexportfs** command, as shown in Example 4-47.

Example 4-47 Using mmexportfs to export file system definition and remove it from the cluster

```
germany:/gpfs#mmexportfs /dev/testexportfs -o testexportfsConfigEsported
```

```
mmexportfs: Processing file system testexportfs ...
mmexportfs: Propagating the cluster configuration data to all
              affected nodes. This is an asynchronous process.
```

This command creates a file named **testexportfsConfigEsported** that contains the definition of the **/dev/testexportfs** cluster file system. The definition is shown in Example 4-48 on page 219.

Example 4-48 An example of mmexportfs output created file

```
germany:/gpfs#cat testexportfsConfigEsported
%%9999%%:00_VERSION_LINE::1202:3:141::lc:usa:germany:0:/usr/bin/ssh:/usr/bin/scp:1388245747
0256956975:lc2:1277314609::mantest.usa:2:1:2:2:::central:0.0:
%%home%%:10_NODESET_HDR:::3:TCP:AUTHONLY:1191:::1202:1202:AL:::
%%home%%:70_MMFS_CFG:1:# ::::::::::::::::::::
%%home%%:70_MMFS_CFG:2:# WARNING: This is a machine generated file. Do not edit!
::::::::::::::::::
%%home%%:70_MMFS_CFG:3:# Use the mmchconfig command to change configuration parameters.
::::::::::::::::::
%%home%%:70_MMFS_CFG:4:# ::::::::::::::::::::
%%home%%:70_MMFS_CFG:5:clusterName mantest.usa:::
%%home%%:70_MMFS_CFG:6:clusterId 13882457470256956975:::
%%home%%:70_MMFS_CFG:7:autoLoad no:::
%%home%%:70_MMFS_CFG:8:[usa,germany]:::
%%home%%:70_MMFS_CFG:9:autoLoad yes:::
%%home%%:70_MMFS_CFG:10:[common]:::
%%home%%:70_MMFS_CFG:11:minReleaseLevel 1202 3.4.0.0:::
%%home%%:70_MMFS_CFG:12:dmapifileHandleSize 32:::
%%home%%:70_MMFS_CFG:13:cipherList AUTHONLY:::
%%home%%:70_MMFS_CFG:14:[usa,germany]:::
%%home%%:70_MMFS_CFG:15:openssllibname
/usr/lib/libssl.a(libssl64.so.0.9.8):::
%%home%%:70_MMFS_CFG:16:[slovenia]:::
%%home%%:70_MMFS_CFG:17:openssllibname libssl.so.6:::
%%home%%:70_MMFS_CFG:18:[common]:::
%%home%%:30_SG_HDR:testexportfs:152:::0:::
%%home%%:40_SG_ETCFS:testexportfs:1:%2Ftestexportfs:
%%home%%:40_SG_ETCFS:testexportfs:2: dev = /dev/testexportfs
%%home%%:40_SG_ETCFS:testexportfs:3: vfs = mmfs
%%home%%:40_SG_ETCFS:testexportfs:4: nodename = -
%%home%%:40_SG_ETCFS:testexportfs:5: mount = mmfs
%%home%%:40_SG_ETCFS:testexportfs:6: type = mmfs
%%home%%:40_SG_ETCFS:testexportfs:7: account = false
%%home%%:50_SG_MOUNT:testexportfs:rw:mtime:atime:::
%%home%%:60_SG_DISKS:testexportfs:1:nsddm2:52428800:40:dataAndMetadata:6585C0A84C3B8B44:nsd
:slovenia::other::dmm:user:::::slovenia::::
%%home%%:60_SG_DISKS:testexportfs:2:nsddm3:52428800:40:dataAndMetadata:6585C0A84C3B8B45:nsd
:slovenia::other::dmm:user:::::slovenia::::
```

This file system is not known by the mantest.usa cluster. If you want this new file system on the same cluster, you have to import it, as shown in Example 4-49.

Example 4-49 Using mmimportfs command to re-import the file system

```
germany:/gpfs#mmimportfs /dev/testexportfs -i testexportfsConfigEsported

mmimportfs: Processing file system testexportfs ...
mmimportfs: Processing disk nsddm2
mmimportfs: Processing disk nsddm3

mmimportfs: Committing the changes ...

mmimportfs: The following file systems were successfully imported:
testexportfs
mmimportfs: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

Now, you are able to correctly mount the file system on this cluster, as before.

You might want to mount the file system to another cluster. Remember that the remote cluster *must* see all the disks that are part of this file system to be able to access them, and therefore to mount the file system. In our case, we import /dev/testexportfs on the testcluster2.england cluster, which is a single-node cluster. Because the england node is able to access the two disks where the file system is, importing is not a problem.

In this test, we import the file system on another cluster, as shown Example 4-50.

Example 4-50 Importing file system on another cluster

```
# fdisk -l | grep dm
Disk /dev/dm-2: 26.8 GB, 26843545600 bytes
Disk /dev/dm-3: 26.8 GB, 26843545600 bytes

# mmimportfs /dev/testexportfs -i testexportfsConfigEsported

mmimportfs: Processing file system testexportfs ...
mmimportfs: Processing disk nsddm2
mmimportfs: Incorrect node slovenia specified for command.
mmimportfs: Processing disk nsddm3
mmimportfs: Incorrect node slovenia specified for command.

mmimportfs: Committing the changes ...

mmimportfs: The following file systems were successfully imported:
      testexportfs
mmimportfs: The NSD servers for the following disks from file system testexportfs
were reset or not defined:
      nsddm2
      nsddm3
mmimportfs: Use the mmchnsd command to assign NSD servers as needed.
```

Important: Although GPFS correctly imports the file system, some modification to NSD was necessary. The reason is because on the mantest.usacluster, the two NSDs that belong to the /dev/testexportfs file system are configured to have the slovenia node as the NSD server. However, this cluster has only one node, england; the slovenia node is not present. Therefore, GPFS resets the NSD server situation on these two NSDs. Now they appear similar to Example 4-51.

Example 4-51 NSDs on the cluster as they appear after using the mmimportfs command

```
# mmchnsd
```

| File system | Disk name | NSD servers |
|--------------|-----------|---------------------|
| testexportfs | nsddm2 | (directly attached) |
| testexportfs | nsddm3 | (directly attached) |

In our case, the file system import is acceptable because we have only one node in the cluster. However, if additional nodes are in the cluster, you must change the NSD server after the import process by using the **mmchnsd** command.

Now, we can mount the /dev/testexportfs file system on the england node, as shown in Example 4-52.

Example 4-52 Mount the imported file system

```
# hostname
england
# mmmount /dev/testexportfs
Mon Jul 12 18:22:11 EDT 2010: mmmount: Mounting file systems ...
# mmdf /dev/testexportfs
```

| disk name | disk size in KB | failure holds group metadata | holds data | free KB in full blocks | free KB in fragments |
|---|--------------------|---------------------------------|---------------|---------------------------|-------------------------|
| ----- | | | | | |
| Disks in storage pool: system (Maximum disk size allowed is 234 GB) | | | | | |
| nsddm2 | 26214400 | 40 yes | yes | 26128384 (100%) | 488 (0%) |
| nsddm3 | 26214400 | 40 yes | yes | 26128896 (100%) | 456 (0%) |
| ----- | | | | | |
| (pool total) | 52428800 | | | 52257280 (100%) | 944 (0%) |
| ===== | | | | | |
| (total) | 52428800 | | | 52257280 (100%) | 944 (0%) |
| ===== | | | | | |

```
Inode Information
-----
Number of used inodes:      4038
Number of free inodes:     48186
Number of allocated inodes: 52224
Maximum number of inodes:  52224
```

Another important point is that you can directly use the /var/mmfs/gen/mmsdrfs GPFS configuration file as input to the **mmimportfs** command to import a GPFS file system on a cluster. The reason is because the /var/mmfs/gen/mmsdrfs file contains all the cluster files, including the configuration of all file systems. Therefore, if you do not want the file system definition to also be removed from the source cluster, you can move the mmsdrfs file to a temporary directory of the destination cluster and used it with the **mmimportfs** command. Also, in this case, NSD servers are cleaned for every node that is unknown in the destination cluster.

Example 4-53 shows how we import a file system on another cluster by using only the mmsdrfs file.

Example 4-53 Importing a file system using the mmsdrfs file

```
germany: /# scp /var/mmfs/gen/mmsdrfs england:/tmp/mmsdrfs.germany
mmsdrfs
100% 11KB 11.2KB/s 00:00

Node england:
# mmimportfs /dev/testexportfs -i /tmp/mmsdrfs.germany

mmimportfs: Processing file system testexportfs ...
mmimportfs: Processing disk nsddm2
mmimportfs: Incorrect node slovenia specified for command.
mmimportfs: Processing disk nsddm3
mmimportfs: Incorrect node slovenia specified for command.
```

```

mmimportfs: Committing the changes ...

mmimportfs: The following file systems were successfully imported:
    testexportfs
mmimportfs: The NSD servers for the following disks from file system testexportfs
were reset or not defined:
    nsddm2
    nsddm3
mmimportfs: Use the mmchnsd command to assign NSD servers as needed.

# mmmount /dev/testexportfs
Mon Jul 12 18:32:05 EDT 2010: mmmount: Mounting file systems ...
# df | grep /dev/testexportfs
/dev/testexportfs      52428800    171520  52257280    1% /testexportfs

```

Example 4-54 shows an extract of the mmsdrf file, and the /dev/testexportfs configuration.

Example 4-54 An extract of mmsdrfs showing the file system configuration

```

%%home%:40_SG_ETCFS:testexportfs:1:%2Ftestexportfs:
%%home%:40_SG_ETCFS:testexportfs:2:    dev            = /dev/testexportfs
%%home%:40_SG_ETCFS:testexportfs:3:    vfs          = mmfs
%%home%:40_SG_ETCFS:testexportfs:4:    nodename     = -
%%home%:40_SG_ETCFS:testexportfs:5:    mount       = mmfs
%%home%:40_SG_ETCFS:testexportfs:6:    type        = mmfs
%%home%:40_SG_ETCFS:testexportfs:7:    account     = false
%%home%:50_SG_MOUNT:testexportfs::rw:mtime:atime::::::::::::::::::
%%home%:60_SG_DISKS:testexportfs:1:nsddm2:52428800:40:dataAndMetadata:6585C0A84C3B8B44:nsd:slov
enia::other::dmm:user::::::::slovenia::::
%%home%:60_SG_DISKS:testexportfs:2:nsddm3:52428800:40:dataAndMetadata:6585C0A84C3B8B45:nsd:slov
enia::other::dmm:user::::::::slo

```

Both the **mmexportfs** and **mmimportfs** commands support the **a11** option, which exports (or imports) all file systems at the same time. You use the **mmimportfs** command with the **-S** parameter to specify a file with the following format:

```
DiskName:ServerList
```

This format accepts one disk per line, followed by the new server list. In this way, NSD server configuration can be modified *during* the import operation so that you do not have to modify it *after* the import operating with the **mmchnsd** command.

4.3 Reducing file system fragmentation: The **mmdefragfs** command

Fragmentation of files in a file system cannot be avoided. When GPFS has to write a file, a free full block has to be allocated and when the file is closed the last logical block of data is reduced to the actual number of subblocks required, creating in this way a fragmented block. The worst situation is when a file system appears full because no more full free blocks are available, but there might fragments of files with free space after them.

The GPFS **mmdefragfs** command can reduce file system fragmentation. The **mmdefragfs** command moves pieces of fragmented data to another fragmented block where there is free

space ultimately resulting in free full blocks. Use the **mmdefragfs -i** command to query how much your file system is fragmented, helping you decide whether or not to run defragmentation process. The defragmentation process can be run with the file system mounted or unmounted. When it is run on a mounted file system, the process might be slow because of possible data read/write conflicts that occur when your application tries to read or write a block at the same time that the **mmdefragfs** command wants to move it. If the file system is heavily accessed, this situation can happen often. Running time of this command depends on how much data there is in the file system, especially how much data is fragmented, and therefore, how many block GPFS has to relocate.

Example 4-55 shows the **mmdefragfs** command run on file system. First, the **mmdefragfs** command is run with the **-i** parameter, only to query data. Second, the command is run without parameters, performing the actual defragmentation.

Example 4-55 mmdefragfs shows actual file system fragmentation and then defragment it

```

germany:/#mmdf /dev/testnewfs
disk          disk size  failure holds    holds          free KB          free KB
name          in KB     group metadata data          in full blocks  in fragments
-----
Disks in storage pool: system (Maximum disk size allowed is 415 GB)
nsdhdisk3      26214400      80 yes      yes          752128 ( 3%)      16664 ( 0%)
nsdhdisk4      26214400      80 yes      yes          752384 ( 3%)      16120 ( 0%)
nsdhdisk5      26214400      80 yes      yes          752384 ( 3%)      20744 ( 0%)
-----
(pool total)    78643200                                2256896 ( 3%)      53528 ( 0%)
=====
(total)         78643200                                2256896 ( 3%)      53528 ( 0%)

Inode Information
-----
Number of used inodes:      4576
Number of free inodes:      73760
Number of allocated inodes: 78336
Maximum number of inodes:   78336

germany:/#mmdefragfs /dev/testnewfs -i
Checking fragmentation in file system 'testnewfs'...
"testnewfs" 78336 inodes: 4576 allocated / 73760 free

          free subblk    free
disk      disk size  in full  subblk in      %      %
name      in nSubblk  blocks  fragments free blk blk util
-----
nsdhdisk3  3276800    94016    2083    2.869  99.936
nsdhdisk4  3276800    94048    2015    2.870  99.939
nsdhdisk5  3276800    94048    2593    2.870  99.921
-----
(total)    9830400    282112    6691           99.932

germany:/#mmumount /dev/testnewfs
Thu Jul 15 17:40:40 EDT 2010: mmumount: Unmounting file systems ...

germany:/#mmdefragfs /dev/testnewfs

```

Defragmenting file system 'testnewfs'...
 100.00 % complete on Thu Jul 15 17:41:50 2010

| disk name | free subblk in full blocks | | blk freed | free subblk in fragments | | % free blk | | % blk util | |
|--------------|----------------------------------|--------|--------------|--------------------------------|-------|---------------|-------|---------------|-------|
| | before | after | | before | after | before | after | before | after |
| nsdhdisk3 | 94880 | 96448 | 49 | 2053 | 381 | 2.90 | 2.94 | 99.94 | 99.99 |
| nsdhdisk4 | 94976 | 95904 | 29 | 1985 | 863 | 2.90 | 2.93 | 99.94 | 99.97 |
| nsdhdisk5 | 94880 | 96320 | 45 | 2593 | 975 | 2.90 | 2.94 | 99.92 | 99.97 |
| (total) | 284736 | 288672 | 123 | 6631 | 2219 | | | 99.93 | 99.98 |

4.4 Optimizing extended attributes: The fastea option

GPFS version 3.4 introduces features regarding extended attributes for files. GPFS version 3.4 can more quickly read and write extended attributes than previous versions because it has a completely new organization regarding where extended attributes data are written. Versions prior to 3.4 had, at the beginning of the file, an indicator that provided information about whether extended attributes were specified for that file. If extended attributes were set, GPFS had to determine and read where this information was actually written, losing time in the process. In version 3.4, extended attributes are written directly on file inodes, so the reading of these attributes is set much faster.

If you have file systems created with previous version of GPFS, you have to use the **mmigratefs** command to migrate the file system to enable the fastea option:

```
mmigratefs Device [--fastea]
```

The **--fastea** flag means that the file system uses the new method to more quickly access extended file attributes.

To determine whether your file system is using the new faster method, use the following command:

```
mmfsfs --fastea
```

4.5 Setting up GPFS callback functionality: The callback commands

Callback functionality was added in GPFS 3.3 to help you automate certain tasks. GPFS is able to run user scripts when certain events occurs. In this way, customers can create their own executable scripts, which can be called automatically by GPFS after the defined event happens. For example, you may create a script to delete temporary old files when a low disk space condition is met. Use the following commands to configure callback:

| | |
|----------------------|--|
| mmaddcallback | Sets up a new pair-event to monitor executables. |
| mmiscallback | Lists already configured callbacks. |
| mmdelcallback | Deletes incorrect or old callback conditions. |

GPFS can monitor two types of events:

- ▶ Global events trigger a callback on all nodes that are configured in the cluster.
- ▶ Local events trigger a callback script only on the node where the event occurs.

Global events are listed in Table 4-1.

Table 4-1 Global events

| Global event | Description |
|------------------------|--|
| nodeJoin | Triggered when one or more nodes join the cluster. |
| nodeLeave | Triggered when one or more nodes leave the cluster. |
| quorumReached | Triggered when a quorum has been established in the GPFS cluster. |
| quorumLoss | Triggered when a quorum has been lost in the GPFS cluster. |
| quorumNodeJoin | Triggered when one or more quorum nodes join the cluster. |
| quorumNodeLeave | Triggered when one or more quorum nodes leave the cluster. |
| clusterManagerTakeover | Triggered when a new cluster manager node has been elected. This event occurs when a cluster first starts or when the current cluster manager fails or resigns and a new node takes over as cluster manager. |

Local events are listed in Table 4-2.

Table 4-2 Local events

| Local event | Description |
|--------------------------------------|---|
| lowDiskSpace | Triggered when the file system manager detects that disk space is running below the low threshold that is specified in the current policy rule. |
| noDiskSpace | Triggered when the file system manager detects that a disk ran out of space. |
| softQuotaExceeded | Triggered when the file system manager detects that a user or file set quota has been exceeded. |
| preMount, preUnmount, mount, unmount | Triggered when a file system is about to be mounted or unmounted or has been mounted or unmounted successfully. These events are generated for explicit mount or unmount commands, a remount after GPFS recovery, and a forced unmount when GPFS shuts down. |
| preStartup | Triggered after the GPFS daemon completes its internal initialization and joins the cluster, but before the node runs recovery for any VFS (GPFS-specific extensions) mount points that were already mounted, and before the node starts accepting user-initiated sessions. |
| startup | Triggered after a successful GPFS startup and when the node is ready for user initiated sessions. |
| preShutdown | Triggered when GPFS detects a failure and is about to shutdown. |
| shutdown | Triggered when GPFS completed the shutdown. |

After selecting a global or local event that you want GPFS to monitor, and writing your own script, add the callback by using the **mmaddcallback** command as shown in Example 4-56 on page 226.

Example 4-56 *mmaddcallback* command syntax

```
mmaddcallback CallbackIdentifier --command CommandPathname --event
Event[,Event...] [--priority Value] [--async | --sync [--timeout Seconds]
[--onerror Action] [-N {Node[,Node...] | NodeFile | NodeClass}] [--parms
ParameterString ...]
```

The command parameters are explained in *General Parallel File System Administration and Programming Reference Version 3 Release 4*, SA23-2221.

Primarily, you set the following parameters:

- ▶ *CallbackIdentifier*, which is a name that you give to your callback
- ▶ *CommandPathname*, which is the complete path of your script
- ▶ *Event*, which indicates the event for which GPFS has to run your command

For this book, we configured and tested the callback functionality. Our setup and examples of how it works is described in Chapter 3, “Scenarios” on page 69 and Chapter 8, “Information lifecycle management (ILM)” on page 317.

4.6 Monitoring GPFS configuration status: The SNMPD protocol

With GPFS, you may use the SNMPD protocol to monitor the status and the configuration of your cluster. It is possible to have one node, called the *collector* node, on which every SNMPD message is sent. To have this working properly, use the following steps.

1. The Net-SNMP master agent daemon (*snmpd*) must be installed on the collector node. The reason is because the GPFS subagent process connects to the *snmpd*.
2. On the master agent, the *snmpd* must be configured according to *General Parallel File System Advanced Administration Guide Version 3 Release 4*, SC23-5182:

```
master agentx
AgentXSocket tcp:localhost:705
trap2sink managementhost
```

In this configuration, *managementhost* is the host name or IP address of the host to which you want the SNMP traps to be sent.

3. If your GPFS cluster has a large number of nodes or a large number of file systems for which information must be collected, you must increase the timeout and retry parameters for communication between the Net-SNMP master agent and the GPFS subagent to allow time for the volume of information to be transmitted. The *snmpd* configuration file entries for this are as follows:

```
agentXTimeout 60
agentXRetries 10
```

The entries have the following settings:

- *agentXTimeout* is set to 60 seconds for subagent to master agent communication
- *agentXRetries* is set to 10 for the number of communication retries

Other values might be appropriate depending on the number of nodes and file systems in your GPFS cluster.

4. After modifying the configuration file, restart the SNMP daemon.

5. Make the GPFS management information base (MIB) available on the nodes that have a management application installed (such as Tivoli NetView®). The `/usr/lpp/mmfs/data/GPFS-MIB.txt` file contains the GPFS MIB.

Perform the following steps:

- a. Copy the file into your SNMP MIB directory (usually `/usr/share/snmp/mibs`).
- b. Change your `snmpd.conf` file by adding the following line to enable the new MIB:
`mibs +GPFS-MIB`
- c. Stop and restart `snmpd` to have it reload the configuration file.

6. After `snmpd` has been configured with GPFS MIB, the collector node must be declared in GPFS by using the `mmchnode` command, as follows:

- To assign a collector node and start the SNMP agent:

```
mmchnode --snmp-agent -N NodeName
```

- To unassign a collector node and stop the SNMP agent:

```
mmchnode --nosnmp-agent -N NodeName
```

- To determine whether a GPFS SNMP subagent collector node is defined:

```
mmfsccluster |grep snmp
```

- To change the collector node, issue the following two commands:

```
mmchnode --nosnmp-agent -N OldNodeName
```

```
mmchnode --snmp-agent -N NewNodeName
```

4.7 SSH configuration

GPFS recognizes the `adminMode` configuration parameter, which is set with the `mmchconfig` command and can have either of the following values:

- `allToAll`

Indicates that every node in the cluster can be used to issue commands to all other nodes in the cluster. (This value is necessary for clusters that are running GPFS 3.2 or earlier.)

- `central`

Indicates that a subset of nodes can be used by GPFS to issue remote commands to other nodes.

Regarding the SSH configuration, use `allToAll` to be sure that every node is authorized to all other node in the cluster to have everything properly working. If the `adminMode` is set to `central`, having all nodes SSH-authorized is not necessary, but we can have only a small set of designated nodes that can be trusted to run commands on their nodes without the password. Therefore, the SSH configuration is influenced by which method you choose to administer for your cluster.

When the SSH keys are generated for the first time on a node, for example with the `ssh-keygen -t` command for RSA or DSA, we create both private (`id_rsa`) and public (`id_rsa.pub`) keys for this node. To have an unattended SSH or SCP communication between nodes, the keys are created without a passphrase. To use SSH or SCP directly to another node, put the public key for the first node in the `authorized_keys` file, which is on the destination node. In this way, the first node is authorized to run remote commands to the secondary node. If you want all nodes to run remote commands on every other node, all nodes must have an `authorized_keys` file that contains all the public keys from all the nodes.

Therefore, create an `authorized_keys` file with all the public keys and then use FTP, for example, to copy it to all the nodes.

To be sure that everything works correctly, try a remote command between nodes before the cluster is configured. If you want to use the *central* value of the `adminMode` parameter, you have to authorize all the nodes or only a subset of nodes by adding, into the `authorized_key` file, the SSH public keys of the nodes.

Another aspect to consider is that if a node is lost and must be reinstalled, you must re-create the SSH keys so that the node has new public keys. You must also remove the old SSH keys from the `authorized_key` file on the nodes and put newly re-created SSH key on the nodes.

Example 4-57 shows the SSH `authorized_keys` configuration on node `germany`. In this case, inside this file are public keys for all four nodes of the cluster. Therefore, all nodes are authorized to enable remote commands on `germany`.

Example 4-57 An example of `authorized_keys` file

```
germany:/.ssh#cat authorized_keys
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAwGsYm3yySWpkQGBxwEgqXdqTbVqnNqA2igB5sb3p91VGAIoQ0bZ0+T
zf8WmhLSCKhSXk3XFu4eWfONFT0TEqNY+Tz241H++bRh8f7y4jj6zs3waym1Hi6wr5RkPW3e1UWj32EYNY
7AZ1HIYsM3Gjs3Lnc4Ad0RYrgSMru8j8VTWXTXwEZ76NWW10JFXsPhnOVGtMMNtW1/4d4/XAYyR48aLY/2
OZR86dxSIXtMB+6aTbQBa+ZS5KPM1Dv0jhTYAU4DwaR6HjzhRqk+KTS48a0dwE17ctcs7C6P6vFaZsGM06
hQNHrkjIEDzyRspRzTZteJVCN93mRYW4I4TdQBqUDQ== root@england
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAum/YgIESxKwCoYg9VYjbbJKbeVH+u4LrZWt6U5EJJGZSdBy6rJnX
7Qvo2VvnpnON5UThhv4K4bZ2Mj8f5QLJwr03Q+/S0Z5DI fAsZ2NaGWwDFiKSPnjAhgeS3c6uiFIq/L97J7
f0RyoP9cr21SzCh5aphQPeHrofU5PfunEL22mn5do08sTNXFRbfnLH9U1WRx7YWKcTjkQHfiBLRWe1vm08
eOnrJu+AFriBPF9Xugk2ydbREArqUBp2+GmUf0FfbMzBogRXrU7W7RESLH1uywdTeibmsjzoL+RtLeNwb0
QYQ5B2iNvPhWgp/bl4h8cnfYnXYavnGLJrvj/1JkrQ== root@usa
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEA8vdGxBRN3R7i06efXPgL/wmM+LJcNEWKKP00qKW0ItG35WLbFTqj+b
QLjkHUd61Yc4xdqjWNLQ32pFHxLfg7hYwtxpNi6RMupcWLGZzF93btDSAAaPRBPSbmlwfnLswqnW6mDu6G
sG1JY9caqYmKmp0w+dx6kaI4aXS0trpEsDmHic+6TyCffew3qSck2owGxR3L1J+4UKBBa7o/c+MiK1ksPD
JK1FJIKq+QwDn1LfAivKETMJuGoZf9rifhkmbAa3wEjshUCVRjIC80tr9KFpI989sPjC6udLeKATWTG5Jc
cbCS56b72dJ1u57hFBVZqX68HBj0Pi6sQtstVvjYpQ== root@germany
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAyU1UFQ8HYf0mpeGwY58431xsDqEj2Iuc773jdgaz6gPEnyufeExoZ
uTpbnaJg4+VpGmaotCXgD/Ng7+crtBj8DqAP6q807BfIvI61xkCMGiGpWX1A7UYn77u+3CKV1Njx5uANdi
3o0yob9FeeEwHWg/FqvunAYA48LtWhzqwJ040ateCV1M/iH2huDxpgrIr7eZdx3bp1Ngh13yVQ0VfNWpDs
SHHYWLFiHmVUxzqboamAzOPLh3NkkJsHI0fys5up92YwKZKCYBtUs+cCU6nRW/9wHVIVXVB5nQpislpFVI
PmVA+aOLCJJAAodWS4F2ojPqjc0Ya+4ZcbH1cmA7uJQ== root@slovenia
```

Performance and tuning

This chapter describes the main consideration factors that affect GPFS performance, and provides methods to evaluate and optimize GPFS, Linux, and AIX parameters.

This chapter contains the following topics:

- ▶ 5.1, “GPFS architecture” on page 230
- ▶ 5.3, “GPFS parametrization” on page 231
- ▶ 5.4, “Network parametrization” on page 237
- ▶ 5.5, “Parametrization” on page 247
- ▶ 5.6, “Monitoring performance” on page 250

5.1 GPFS architecture

This section introduces key aspects of GPFS architecture and how to evaluate it to properly define the sizing of the GPFS file system.

In GPFS, I/O represents the core part of the tuning; for better organization and easy understanding, it can be split into two types:

► IP network-based I/O

GPFS uses the IP network to perform several critical operations and to provide certain functions:

- Quorum negotiation
- Token management
- File system descriptor synchronization
- Configuration synchronization
- Network Shared Disk (NSD) server, used to provide access to the NSD volumes through the IP network, to the nodes that do not have direct access to the disks

► Disk-based

GPFS, as a file system, uses the disks to store information and, especially on small clusters, to store health-check information, which can be organized into the following categories:

- Data transfer to NSD
- Quorum negotiation when tiebreaker is used

This I/O-based utilization can be described as shown in Figure 5-1.

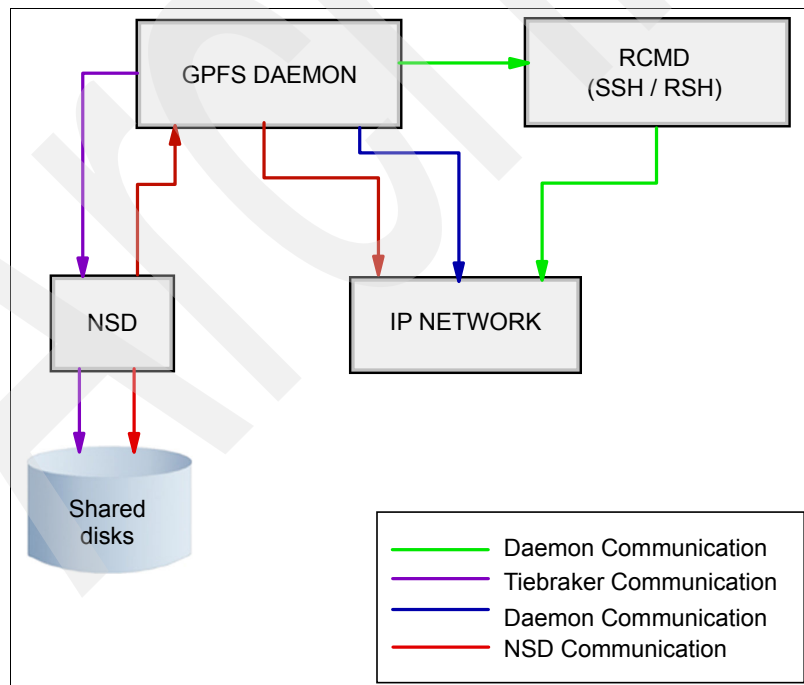


Figure 5-1 GPFS I/O flow

5.2 NSD considerations

When the NSD client/server architecture is being used, GPFS uses the IP network to transfer the NSD information, which basically consists of data and metadata information.

This data and metadata information is transferred using a standard client/server method, which implies that a client talks directly and exclusively to a single server.

To avoid single point of failure (SPOF) situations, GPFS allows you to specify 0 - 8 NSD servers for each NSD volume:

- ▶ 0 servers: All members of the cluster have direct connection to the NSD volume (for example, through a shared storage)
- ▶ 1 - 8 servers: Members that do not have direct connection to the NSD volume will connect to these servers to access the volume

However, when multiple servers are specified to work as NSD servers, the load is not automatically split across them. If the same server list is used to create all volumes, the load will be concentrated into a single server, resulting in a poor resource utilization. As standard configuration, an advisable approach is to divide the server lists in a way to spread the NSD load into as many servers as possible.

For more information about NSD architecture, see Chapter 2, “Infrastructure planning” on page 5.

Important: When using NSD servers, although the configuration allows the definition of a single NSD server per volume, always specify more than one (1) to avoid data corruption or file system failure in case of a NSD server failure (for example, member goes down).

5.3 GPFS parametrization

GPFS customization is organized into the following sets:

- ▶ File system parametrization
Parameters that affects only a specific file system
- ▶ Cluster parametrization
Parameters that affect all file systems into the cluster

This section presents both sets of parameters and how they affect the system performance, resource needs, and system behavior.

5.3.1 File system parametrization

As a key factor to optimize the file system performance, determining the behavior and needs for the applications that will be using the file system is important.

Several basic questions can help with determining that information:

- ▶ How many files will be kept in the file system?
- ▶ What is the average file size?
- ▶ What types of files will be kept in the file system?

- What operations will be performed with the files?
- How many files will be opened in parallel?

The number of files in the file system and the average file size are especially important during file-system creation. This information helps you determine the file system block size, which cannot be changed after the file system has been created.

The block size is the largest contiguous amount of disk space that is allocated to a file and therefore the largest amount of data that can be accessed in a single I/O operation. GPFS supports block sizes of 16 KB, 64 KB, 128 KB, 256 KB, 512 KB, 1024 KB, 2048 KB, and 4096 KB. The block size directly affects the amount of time to retrieve or write information to the disks, especially when considering small-block writes to RAID arrays. Each block size is described in Table 5-1.

Table 5-1 Block sizes

| Block size (KB) | Description |
|------------------|--|
| 16 | This size optimizes the use of disk storage at the expense of large data transfers. |
| 64 | This size offers a compromise if there are a mix of many files of approximately 64 KB or less in size. It makes more efficient use of disk space than 256 KB, and allows faster I/O operations than 16 KB. |
| 128 | This size offers better performance for small file, small random read and write, and metadata-intensive workloads. |
| 256 | This size is the default block size and normally is the best block size for file systems that contain large files accessed in large read and write operations. |
| 1024, 2048, 4096 | These sizes are more efficient if the dominant I/O pattern is sequential access to large files. |

When RAID arrays are used, a larger block size may be more effective and help avoid the penalties involved in small block write operations. For example, in a RAID configuration using four data disks and one parity disk (a 4+P configuration), which uses a 64 KB stripe size, the optimal file system block size would be an integral multiple of 256 KB (4 data disks × 64 KB stripe size = 256 KB). A block size of an integral multiple of 256 KB results in a single data write operation that encompasses the four data disks and a parity-write to the parity disk. If a block size smaller than 256 KB, such as 64 KB, is used, write performance is degraded by the read-modify-write behavior. A 64 KB block size results in a single disk, writing 64 KB, and a subsequent read from the three remaining disks to compute the parity that is then written to the parity disk. The extra read degrades performance.

The choice of block size also affects the performance of certain metadata operations, in particular, block allocation performance. The GPFS block allocation map is stored in blocks, similar to regular files. Consider the following information when the block size is small:

- Storing a given amount of data requires more blocks, resulting in additional work to allocate those blocks.
- One block of allocation map data contains less information.

Important: GPFS divides each block into 32 sub-blocks, which consists of the smallest unit of disk space that can be allocated. For files that are smaller than one block size, the files are stored in *fragments* that are composed by sub-blocks.

It is possible to find the file-system block size through the `-B` parameter of the `mm1sfs` command. The block size is defined by the `-B` parameter of the `mmcrfs` command.

Example 5-1 shows that the file system `/dev/testnewfs` has a block size of 256 KB.

Tip: With the `mm1sfs` command, the block size is always indicated in bytes; with the `mmcrfs` command, the modifiers KB and MB can be used to determine kilobytes and megabytes.

Example 5-1 Determine the actual block size of a file system

| | | |
|--|---------------|-------------------|
| usa:/usr/lpp/mmfs/bin#mm1sfs /dev/testnewfs -B | | |
| flag | value | description |
| ----- | | |
| -B | 262144 | Block size |
| usa:/usr/lpp/mmfs/bin# | | |

Note: For more information about file system creation parameters, see Chapter 4, “Management and maintenance” on page 189. Also see the following resource:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.gpfs33.install.doc/blins_fsblksiz.html

5.3.2 Block allocation methods

During the file system creation, GPFS provides two algorithms: *cluster* and *scatter*. These algorithms determine how GPFS will allocate the blocks on the NSD volumes to store data:

► Cluster

GPFS attempts to allocate blocks in clusters. Blocks that belong to a given file are kept next to each other within each cluster.

This allocation method provides a better performance when writing and reading sequential files only and when the file system fragmentation is kept low. If the fragmentation increases, the performance tends to decrease.

As an attempt to keep the file system fragmentation low, use the `mmdegragfs` command periodically. However, on a large highly used file system, with little free space, this operation can take a long time to complete.

► Scatter

GPFS chooses the location of the blocks randomly.

This allocation method provides more consistent file system performance by averaging out performance variations because of block location.

Initially this method tends to provide a lower performance than cluster, however the performance does not tend to change as the file system fragmentation increases.

The scatter method also tends to always provide a higher file system creation rate than cluster, because of the parallelism of accessing multiple NSD volumes simultaneously.

Scatter is the default method on large implementations.

To determine which block allocation method is being used on the file system, use the `mm1sfs` command, as Example 5-2 shows.

Example 5-2 Determine the block allocation method

| | | |
|--------------------------------|---------|-----------------------|
| usa:/#mm1sfs /dev/testnewfs -j | | |
| flag | value | description |
| ----- | | |
| -j | cluster | Block allocation type |
| usa:/# | | |
| ----- | | |

5.3.3 GPFS general parametrization

After the file system is created, GPFS optimization adjusts the configuration to match the available system resources, and balances these resources with the GPFS configuration to match the most needed operation.

GPFS cache considerations

GPFS creates cache to optimize I/O performance in the file systems. It caches metadata and statistical information. Customizing the cache configuration requires a previous knowledge of the I/O behavior of the application that will use the file system.

GPFS organizes the cache in three parameters:

► pagepool

The GPFS pagepool parameter is used to cache user data and file system metadata.

The pagepool parameter mechanism allows GPFS to implement read and write requests asynchronously.

Increasing the size of pagepool increases the amount of data or metadata that GPFS can cache without requiring synchronous I/O.

The amount of memory that is available for GPFS pagepool on a particular node might be restricted by the operating system and other software that is running on the node.

The optimal size of the pagepool depends on the needs of the application and effective caching of its reaccessed data.

For systems where applications access large files, reuse data, benefit from GPFS prefetching of data, or have a random I/O pattern, increasing the value for pagepool might prove beneficial.

► maxFilesToCache

This parameter is the total number of files that can be cached at one time.

Every entry in the file cache requires some pageable memory to hold the content of the file's inode plus control data structures.

This entry is in addition to any of the file's data and indirect blocks that might be cached in the pagepool.

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

`maxFilesToCache × 3 KB`

For systems where applications use a large number of files, of any size, increasing the value for maxFilesToCache might prove beneficial.

This approach is particularly true for systems where a large number of small files are accessed. Be sure the value is large enough to handle the number of concurrently open files, and allow caching of recently used files.

► **maxStatCache**

This parameter sets aside additional pageable memory to cache attributes of files that are not currently in the regular file cache.

This parameter is useful for improving the performance of both the system and GPFS `stat()` calls for applications with a working set that does not fit in the regular file cache.

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

$\text{maxStatCache} \times 400 \text{ bytes}$

For systems where applications test the existence of files, or the properties of files, without actually opening them (as backup applications do), increasing the value for `maxStatCache` might prove beneficial.

Setting to higher values on user-interactive nodes, and smaller on dedicated compute-nodes, provides a more accurate utilization of the stat cache because, for example, the `ls -l` command performance is mostly a human response issue; most of stat operations are triggered by human interactive programs.

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

The `mmchconfig` command can be used to change the values of `pagepool`, `maxFilesToCache`, and `maxStatCache`.

The `pagepool` parameter is the only one of these parameters that may be changed while the GPFS daemon is running; the other parameters are enabled only after the GPFS daemon is reloaded.

GPFS memory utilization considerations

The memory that is allocated to GPFS for a determined node is used in several key operations within GPFS. As the file system utilization increases, especially during parallel utilization, the GPFS memory dependency increases too.

Although increasing the amount of memory that GPFS uses for caching generally improves the performance of the file system locally, GPFS achieves parallelism by issuing tokens to the members when a file is requested by it. Those tokens are managed by the token manager servers; these tokens are kept in the system memory of these nodes.

When the cache size is increased, the token dependency also increases.

Consider the following information:

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

- Under normal situations, a GPFS token manager roughly handles a maximum of 600000 tokens (this number depends on how many distinct byte-range tokens are used when multiple nodes access the same file).

Be sure to keep the file system cache configuration with the following formula:

$$(\#MGRS-1)*600000*(\#TOKENMEM/256*1024) \geq \#MEMBERS * (MFTC + MSC)$$

The formula has the following definitions:

- **MGRS**

MGRS is the number of members that are token managers in the cluster. If only one node is a token manager, the -1 can be ignored.

- **TOKENMEM**

This value is the amount of memory reserved to manage the tokens.

- **MEMBERS**

This value is the number of nodes that are part of the cluster.

- **MFTC**

This value is the maximum files to cache size.

- **MSC**

This value is the maximum statistical cache size

As indicated previously, if the caching size increases too much, without increasing the amount of token managers and the amount of memory allocated to the token managers, memory can become overcommitted, which decreases system performance.

The `maxFilesToCache` and `maxStatCache` parameters are indirectly affected by the `distributedTokenServer` configuration parameter. The reason is because distributing the tokens across multiple token servers might allow keeping more tokens than if a file system has only one.

GPFS achieves parallelism by issuing tokens to the members when a file is requested. A token allows a node to cache data it has read from disk, because the data cannot be modified elsewhere without revoking the token first. When the file is closed, the token is released.

Direct I/O operations

GPFS allows applications to open the files with the `O_DIRECT` parameter, which is more commonly referred as Direct I/O.

The `O_DIRECT` parameter instructs GPFS to not cache the contents of that specific file; instead all write and read operations are flushed directly into the NSD volumes. Increasing cache sizes when the majority of the file operations require Direct I/O can lead to a decrease in performance.

Use the `mmchattr` command to manually define the GPFS cache policy. Use the `mmfsattr` command to determine the actual file attribute. See Example 5-3 on page 237.

Example 5-3 Define and list direct I/O attributes into a file

```
usa:/testnewfs#mmchattr -D yes /testnewfs/mille135
usa:/testnewfs#mmlsattr -L /testnewfs/mille135
file name:                /testnewfs/mille135
metadata replication: 1 max 2
data replication:        1 max 2
immutable:               no
appendOnly:              no
flags:                  directio
storage pool name:       system
fileset name:            root
snapshot name:
```

If the file has the direct I/O attribute manually defined, it is listed in the `flags` line of the `mmlsattr` command output.

Notes: Applications that perform parallel I/O, mostly database managers, use direct I/O on all its data files. The most common examples are as follows:

- ▶ Oracle Real Application Clusters (RAC)
- ▶ IBM DB2 PureScale

Reference information about parametrization settings

See the following resources for more information about the parametrization settings:

- ▶ http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.gpfs.33.install.doc/bllins_cache.html
- ▶ http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.gpfs.33.basicadm.doc/blladm_direct.html
- ▶ http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.gpfs.33.advancedm.doc/blladv_lgsysumtss.html
- ▶ <http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.cluster.gpfs.doc/gpfs31/bllins1147.html>

5.4 Network parametrization

GPFS uses TCP/IP to transmit data across the network. TCP/IP is a combination of two logical protocols encapsulated into a data link protocol; it is the base protocol used on most of the network environments including the Internet.

In most local area networks (LAN), the data link protocol used is the Ethernet protocol, which is the focus of this section. This section also summarizes the most common factors that can affect TCP/IP over Ethernet performance.

The organization of these protocols can be represented by the OSI model shown in Figure 5-2.

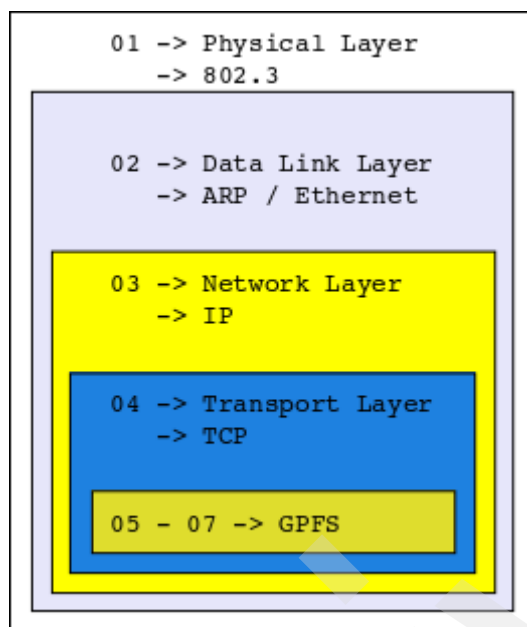


Figure 5-2 OSI model representation

Ethernet protocol

The Ethernet protocol is standardized as IEEE 802.3, and is the most widespread wired LAN technology. Ethernet protocol was in use starting in the 1970s.

The first publicly available version of Ethernet (Type II frame) was based on several devices sharing the same coaxial cable to connect itself to the network, forming a loop; that connection had the following basic characteristics:

- ▶ The network has a maximum shared speed of 10 Mbps.
- ▶ The channel (cable) is used by all nodes to send and receive data, by definition half duplex
- ▶ The devices are identified to the network by signed (using MAC address for signature) broadcast packages
- ▶ The maximum supported frame size is 1518 bytes
- ▶ The maximum payload, data, size per frame is 1500 bytes

The early standard was called Ethernet Type II Frame, as illustrated in Figure 5-3.

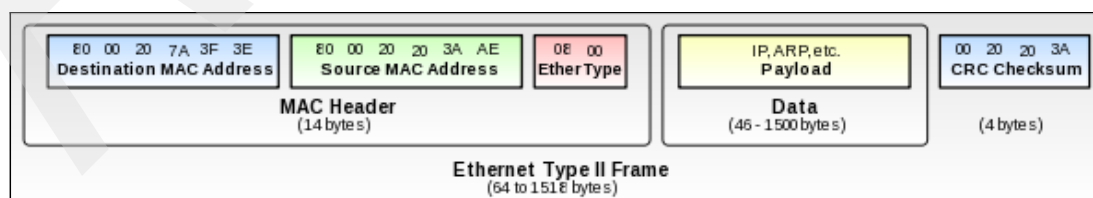


Figure 5-3 Ethernet Type II Frame

As the technology evolved, the Ethernet networks became more resilient and faster. Most actual networks have the following capabilities:

- ▶ Connection speed of 1 Gbps, using twisted-pair cables, providing full-duplex connections
- ▶ Support of frames up to 9042 bytes
- ▶ Support of payload, data, and size per frame of 9000 bytes
- ▶ Support of alternative offload techniques for large frames, called generic segmentation offload (GSO)
- ▶ Support of media negotiation protocol, which negotiates link speed and duplex capabilities between the devices that are physically connected to the network

Whichever way the devices are configured by default, keeping compatibility with 1980s devices results in two common performance constraints:

- ▶ Small frame size used in the network configuration
- ▶ Small transmit and receive buffer size in the system, limiting the number of frames that the application can send to the system kernel without causing a network interrupt request

In the first Ethernet networks, the maximum of 1518 bytes per frame made sense because of the high error rate of the networks. Small processing capabilities and memory available in the systems also prevented the use of bigger transmission buffers.

Note: The payload size is more commonly referred to as maximum transmission unit (MTU). Because the default frame size configuration is set to 1500 bytes, frames of 9000 bytes are commonly referred to as *Jumbo* frames.

Certain devices, such as the Shared Ethernet Adapter (SEA) provided by the System p Virtual I/O Server (VIOS), support 65394 bytes per frame. These frame size configurations are referred to as *Super Jumbo* frames.

In the current implementations, those small frame and buffer sizes instead generate high interrupt request (IRQ) rates in the systems, which slow down the overall data processing capabilities of the systems (Linux and AIX).

On Linux, the MTU and transmit buffers (queue) can be defined with the assistance of the **ip** command, as shown in Example 5-4.

Example 5-4 Adjust and list MTU and transmit queue length on Linux

```
[root@slovenia ~]# ip link set eth0 txqueuelen 8192 mtu 9000
[root@slovenia ~]# ip link show eth0
3: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc pfifo_fast qlen 8192
    link/ether 6a:88:87:a9:c7:03 brd ff:ff:ff:ff:ff:ff
[root@slovenia ~]#
```

Important: All configuration that is defined by the **ip** command is for the current running system only. If the server is rebooted, the **mtu** and **txqueuelen** parameters have to be defined again.

On AIX, the network interfaces are organized under a device hierarchy; *ent* devices (*physical adapter*) is higher in the hierarchy than *en* devices (*network stack*).

The parametrization is defined by the **chattr** command and can be checked by the **lsattr** command, as shown in Example 5-5.

Example 5-5 List and adjust MTU size and transmit queue length on AIX

```
# chdev -l ent0 -a jumbo_frames=yes -a large_send=yes -P
ent0 changed
#chdev -l en0 -a mtu=9000 -P
en0 changed
# lsattr -El en0 -a mtu
mtu 9000 Maximum IP Packet Size for This Device True
# lsattr -El ent0 -a jumbo_frames -a large_send
jumbo_frames yes Transmit jumbo frames True
large_send yes Enable hardware TX TCP resegmentation True
```

Important: Large MTU sizes are not supported as the standard configuration on most network switches. After the server adapter has been changed, the adapter and the switch configuration must be checked too.

Larger transmission buffers can sometimes lead the frame to expire before it reaches its destination, leading to an increase in package retransmissions, slowing down the network.

The adjustment of the offload techniques to Linux systems is done with the **ethtool** command, as shown in Example 5-6.

Example 5-6 Define GSO on Linux

```
[root@slovenia ~]# ethtool -K eth0 gso on
[root@slovenia ~]# ethtool -K eth0 gro on
[root@slovenia ~]# ethtool -k eth0
Offload parameters for eth0:
rx-checksumming: on
tx-checksumming: on
scatter-gather: off
tcp segmentation offload: off
udp fragmentation offload: off
generic segmentation offload: on
generic-receive-offload: on
[root@slovenia ~]#
```

Tip: Certain devices support Time Sharing Option (TSO) / UDP Fragmentation Offload (UFO) and other optimizations. See your network adapter documentation for information about how to enable these options on the adapters.

Certain switches perform periodic speed and duplex negotiations to ensure the link health. Disabling the media speed and duplex, and negotiation by forcing a specific speed and duplex of a controlled healthy network can remove the delays from the periodical negotiation and reduce the initial delay during the initialization of the physical connection between the devices.

The definition of speed and duplex can be achieved by the **ethtool** command on Linux, and **chattr** command in the *ent* adapter on AIX.

Note: See the following resources for more information:

- ▶ <http://www.ibm.com/support/docview.wss?uid=isg3T1011715>
- ▶ http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=/iphb1/iphb1_vios_planning_sea_adapter.htm
- ▶ http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=/iphb1/iphb1_vios_planning_sea_procs.htm
- ▶ <http://publib.boulder.ibm.com/infocenter/iserics/v5r4/index.jsp?topic=/rzahq/rzahqiscsimtucons.htm>
- ▶ http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp?topic=/com.ibm.aix.prftungd/doc/prftungd/adapter_mtu_setting.htm
- ▶ http://publib.boulder.ibm.com/infocenter/tsminfo/v6/index.jsp?topic=/com.ibm.i.tsm.perf.doc/t_network_aix_srv_clnt_mtu_mss.html
- ▶ <http://www.redhat.com/mirrors/LDP/HOWTO/NFS-HOWTO/performance.html>

TCP/IP protocols

After the basic optimizations have been implemented at the link level, the parametrization of the data protocols must be re-evaluated.

The TCP/IP protocols are responsible for encapsulating and controlling delivery of data inside a media protocol. As with Ethernet, the TCP/IP has as its default configuration reduced package and buffer sizes to maintain compatibility with slower, older, devices.

The main objective when optimizing the performance of TCP/IP networks is to increase its utilization to the nearest possible of its full capacity. This task is done by increasing the TCP package sizes, which enables the physical devices to continuously transmit and receive packages.

However, when increasing the package sizes, it is important to not pass the limit of data that can be transmitted without reorganizing the packages. This limit defines the maximum value to be attributed to the TCP send and receive buffers (window size).

The limit can be calculated by the following formula:

$(\text{media link speed}) * (\text{round trip time}) / 8 = (\text{TCP window size in bytes})$

The formula contains the following values:

- ▶ Media link speed

Determine the media link speed with the **entstat** command on AIX (Example 5-7) and with the **ethtool** command on Linux (Example 5-8).

Example 5-7 Determine the media link speed on AIX with the entstat command

```
# entstat -d ent0 | grep 'Media Speed Running'
Media Speed Running: 1000 Mbps Full Duplex
#
```

Example 5-8 Determine the media link speed on Linux with the ethtool command

```
[root@slovenia ~]# ethtool eth0 | grep Speed
Speed: 1000Mb/s
[root@slovenia ~]#
```

Because the media link speed is represented in bits, the value must be converted to bytes, which is achieved by dividing the result by 8.

► Round trip time

This factor represents the amount of time that the package needs to reach the other device. For this formula, the value is specified in seconds and can be evaluated with the assistance of the ICMP protocol using the **ping** command, shown in Example 5-9.

Example 5-9 Determine round trip with ping command

```
usa:/#ping -c 10 germany
PING germany (192.168.101.142): 56 data bytes
64 bytes from 192.168.101.142: icmp_seq=0 ttl=255 time=0 ms
64 bytes from 192.168.101.142: icmp_seq=1 ttl=255 time=1 ms
64 bytes from 192.168.101.142: icmp_seq=2 ttl=255 time=1 ms
64 bytes from 192.168.101.142: icmp_seq=3 ttl=255 time=1 ms
64 bytes from 192.168.101.142: icmp_seq=4 ttl=255 time=1 ms
64 bytes from 192.168.101.142: icmp_seq=5 ttl=255 time=1 ms
64 bytes from 192.168.101.142: icmp_seq=6 ttl=255 time=1 ms
64 bytes from 192.168.101.142: icmp_seq=7 ttl=255 time=1 ms
64 bytes from 192.168.101.142: icmp_seq=8 ttl=255 time=1 ms
64 bytes from 192.168.101.142: icmp_seq=9 ttl=255 time=1 ms

--- germany ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max = 0/1/1 ms
usa:/#
```

► TCP window size

Example 5-10 shows the equation to determine the recommended TCP window size that matches the values presented by Example 5-7 on page 241, Example 5-8 on page 241, and Example 5-9.

Example 5-10 Calculate the TCP window size to a 1 Gbps network with 1 ms round trip time

$((1 \times 1000^3) \times 0.01) / 8 = 1250000$ Bytes

The adjustment of the TCP window size on AIX is achieved by the **no** command, as shown in Example 5-11.

Example 5-11 Define TCP window size on AIX with NO

```
usa:/#no -p -o sb_max=1250000 -o tcp_recvspace=1250000 -o tcp_sendspace=1250000
Setting sb_max to 1250000
Setting sb_max to 1250000 in nextboot file
Setting tcp_recvspace to 1250000
Setting tcp_recvspace to 1250000 in nextboot file
Setting tcp_sendspace to 1250000
Setting tcp_sendspace to 1250000 in nextboot file
Change to tunable tcp_recvspace,tcp_sendspace, will only be effective for future
connections
usa:/#
```

On Linux, instead of static values, a range of three values is defined:

- ▶ Minimum package size
Default configuration is set to 4096 bytes, and is a reasonable size.
- ▶ Default package size
Calculated by the equation that is shown in Example 5-10 on page 242.
- ▶ Maximum package size
Four times larger than the default package size is reasonable.

The definition of the package range size can be defined with the **sysctl** command, as shown in Example 5-12.

Example 5-12 Define TCP window range size on Linux with SYSCTL

```
[root@slovenia ~]# sysctl -w net.ipv4.tcp_rmem="4096 1250000 5000000"
net.ipv4.tcp_rmem = 4096 1250000 5000000
[root@slovenia ~]# sysctl -w net.ipv4.tcp_wmem="4096 1250000 5000000"
net.ipv4.tcp_wmem = 4096 1250000 5000000
```

On a large network, one expected consequence of optimizing the network frame configuration is the increase of overall utilization, which increases the round trip time (RTT), changing the values of the equation that are used to determine the TCP window size.

The TCP protocol implements a set of sub protocols and routines to automatically adjust the parametrization to match the network utilization. Those protocols are as follows:

- ▶ TCP Large Window Extensions (RFC1323)
Allows the system to automatically adjust the window size.
- ▶ TCP Selective Acknowledgments Option (RFC2018)
Allows the TCP receiver to inform the sender of exactly which data is missing and must be retransmitted.
- ▶ Path MTU (PMTU)
Allows the host to determine the largest possible MTU for the network segment.

On Linux, these protocols are enabled on the default installation; on AIX, RFC1323 and RFC2018 must be enabled by using the **no** command, as shown in Example 5-14 on page 244.

Path MTU (PMTU) is enabled by default on Linux and AIX. To verify the parametrization on Linux nodes, the **sysctl** command can be used as shown in Example 5-13.

Example 5-13 Show network runtime parameters on Linux with sysctl

```
[root@slovenia ~]# sysctl -a | grep ^net | tail
net.core.xfrm_aevent_etime = 10
net.core.optmem_max = 20480
net.core.message_burst = 10
net.core.message_cost = 5
net.core.netdev_max_backlog = 1000
net.core.dev_weight = 64
net.core.rmem_default = 129024
net.core.wmem_default = 129024
net.core.rmem_max = 131071
net.core.wmem_max = 131071
```

```
[root@slovenia ~]#
```

On AIX, `rfc1323` and `rfc2018` parameters are defined by using the `no` command, as shown in Example 5-14.

Example 5-14 Define SACK and rfc1323 on AIX

```
usa:/#no -p -o rfc1323=1 -o sack=1
Setting rfc1323 to 1
Setting rfc1323 to 1 in nextboot file
Setting sack to 1
Setting sack to 1 in nextboot file
Change to tunable rfc1323,sack, will only be effective for future connections
usa:/#
```

Another important `no` command parameter that is important to be checked is the `ipqmaxlen` parameter. This parameter is an IP specific queue that defines the number of IP packages that can be received and queued for processing.

The default configuration `ipqmaxlen` is set to 100. However, in most NSD servers, increasing this queue size can improve the network performance by reducing the interrupt requests. On the servers for this book, `ipqmaxlen` was defined as 512, as shown in Example 5-15.

Example 5-15 Define ipqmaxlen on AIX

```
usa:/#no -r -o ipqmaxlen=512
Setting ipqmaxlen to 512 in nextboot file
Warning: changes will take effect only at next reboot
usa:/#
```

Linux has automatic tuning capabilities that are enabled by default in the kernel, and that automatically adjust the maximum TCP window size to match the server capabilities. The kernel parameter is called `ipv4.tcp_moderate_rcvbuf`. This tuning feature automatically affects the system call `setsockopt()`, defining a limit to the `SO_SNDBUF` and `SO_RCVBUF` parameters. This parametrization is shown in Example 5-16.

Example 5-16 Determine Linux kernel autotune configuration with sysctl

```
[root@slovenia ~]# sysctl -n net.ipv4.tcp_moderate_rcvbuf
1
[root@slovenia ~]#
```

However, the `net.core.wmem_max` and `net.core.rmem_max` parameters are the actual limiters for `SO_SNDBUF` and `SO_RCVBUF`. Customizing these values automatically disables the `net.ipv4.tcp_moderate_rcvbuf` parameter. Be sure to manually set it to the same value as the manually defined maximum value of `net.ipv4.tcp_rmem` and `net.ipv4.tcp_wmem` parameters. Determine the running configuration as shown in Example 5-17.

Example 5-17 Determine the current maximum TCP window size on Linux

```
[root@slovenia ~]# sysctl -n net.core.rmem_max
131071
[root@slovenia ~]# sysctl -n net.core.wmem_max
131071
[root@slovenia ~]#
```

As additional consideration, both Linux and AIX implement the Explicit Congestion Notification TCP protocol (RFC3168); which requires hardware support in the network level (switches and routers).

In the environment that we built for this book, no significant affect was observed by enabling or not enabling this protocol, which is disabled by default.

Carefully consider the use of this protocol, especially because it includes more control data on the TCP/IP packages, reducing the space available for the actual data that needs to be transferred.

Note: See the following resources for more information:

- ▶ <http://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt>
- ▶ http://publib.boulder.ibm.com/infocenter/aix/v6r1/topic/com.ibm.aix.prftungd/doc/prftungd/tcp_workload_tuning.htm
- ▶ <http://tools.ietf.org/html/rfc3168>
- ▶ http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.gpfs321.install.doc/blins_netperf.html

Network utilization statistics

After any customization into the Ethernet configuration is set, an important task is to monitor the network connection to be sure the parametrization is not causing any transmission error in the network.

Linux

On Linux, the network device utilization statistics can be determined by using the **ip** command, as shown in Example 5-18.

Example 5-18 Network utilization statistics on Linux with ip command

```
[root@slovenia ~]# ip -s -s link show eth0
3: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc pfifo_fast qlen 1000
    link/ether 6a:88:87:a9:c7:03 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped  overrun  mcast
       714557137  2781210    0        0         0         0
    RX errors: length  crc      frame   fifo    missed
               0        0        0        0        0
    TX: bytes  packets  errors  dropped  carrier  collsns
       532559731  736590    0        0         0         0
    TX errors: aborted  fifo    window  heartbeat
               0        0        0         0
```

When larger packages are defined in the configuration, package retransmission can cause even more slowing down of the network. In addition, on busy networks, some packages are discarded because the reception queue is filling.

On Linux, if the package-drop statistics (observed on Example 5-18) start to increase, the `net.core.netdev_max_backlog` queue size can help to control the package drop and re-transmissions.

In the network that we defined for this book, the backlog queue size was adjusted from 1000 to 2500, to prevent this type of problem, as shown in Example 5-19.

Example 5-19 Adjust the network device backlog size on Linux

```
[root@slovenia ~]# sysctl -n net.core.netdev_max_backlog
1000
[root@slovenia ~]# sysctl -w net.core.netdev_max_backlog=2500
net.core.netdev_max_backlog = 2500
[root@slovenia ~]#
```

AIX

On AIX, the network device utilization statistics can be determined by using the **entstat** command, as shown in Example 5-20.

Example 5-20 Network device utilization statistics on AIX with entstat command

```
# entstat -d ent0
Aix entstat estatistics:
-----
ETHERNET STATISTICS (ent0) :
Device Type: 2-Port 10/100/1000 Base-TX PCI-X Adapter (14108902)
Hardware Address: 00:02:55:2f:a5:9e
Elapsed Time: 34 days 16 hours 39 minutes 25 seconds

Transmit Statistics:                      Receive Statistics:
-----
Packets: 30427309                        Packets: 87887411
Bytes: 107770387256                     Bytes: 61379890500
Interrupts: 0                           Interrupts: 72885627
Transmit Errors: 0                       Receive Errors: 0
Packets Dropped: 0                      Packets Dropped: 0
Bad Packets: 0

Max Packets on S/W Transmit Queue: 1151
S/W Transmit Queue Overflow: 0
Current S/W+H/W Transmit Queue Length: 1

Broadcast Packets: 9601                  Broadcast Packets: 7420665
Multicast Packets: 26868                 Multicast Packets: 303234
No Carrier Sense: 0                     CRC Errors: 0
DMA Underrun: 0                         DMA Overrun: 0
Lost CTS Errors: 0                      Alignment Errors: 0
Max Collision Errors: 0                  No Resource Errors: 0
Late Collision Errors: 0                 Receive Collision Errors: 0
Deferred: 44                            Packet Too Short Errors: 0
SQE Test: 0                             Packet Too Long Errors: 0
Timeout Errors: 0                       Packets Discarded by Adapter: 0
Single Collision Count: 0                Receiver Start Count: 0
Multiple Collision Count: 0
Current HW Transmit Queue Length: 1

General Statistics:
-----
No mbuf Errors: 0
Adapter Reset Count: 0
Adapter Data Rate: 2000
```


Driver Flags: Up Broadcast Running
Simplex 64BitSupport ChecksumOffload
PrivateSegment LargeSend DataRateSet

2-Port 10/100/1000 Base-TX PCI-X Adapter (14108902) Specific Statistics:

Link Status : Up
Media Speed Selected: Auto negotiation
Media Speed Running: 1000 Mbps Full Duplex
PCI Mode: PCI-X (100-133)
PCI Bus Width: 64-bit
Latency Timer: 144
Cache Line Size: 128
Jumbo Frames: Disabled
TCP Segmentation Offload: Enabled
TCP Segmentation Offload Packets Transmitted: 2918041
TCP Segmentation Offload Packet Errors: 0
Transmit and Receive Flow Control Status: Enabled
XON Flow Control Packets Transmitted: 0
XON Flow Control Packets Received: 44
XOFF Flow Control Packets Transmitted: 0
XOFF Flow Control Packets Received: 44
Transmit and Receive Flow Control Threshold (High): 49152
Transmit and Receive Flow Control Threshold (Low): 24576
Transmit and Receive Storage Allocation (TX/RX): 8/56
#

5.5 Parametrization

GPFS, as a parallel file system, focuses disk I/O parametrization on controlling the amount of the following items:

- ▶ Parallel I/O tasks that are performed by each cluster member
- ▶ Bandwidth that is used by the GPFS to perform I/O operations on each cluster member

5.5.1 Controlling the bandwidth

GPFS has the option `maxMBpS` in its configuration that allows control of the maximum bandwidth that is used to perform I/O operations. Be sure to set this parameter to the maximum bandwidth available to access the storage device.

The servers in our testing laboratory were connected to an IBM DS4700 storage device, which does not provide an active-active (multibus) multipath disk access to the 2 Gbps SAN switch, thereby reducing the maximum connection speed to 256 MBps.

Even if the real disk speed does not reach 256 MBps, GPFS tries to reach the maximum speed available.

Consider the following information when evaluating the maxMBps parameter sizing:

- ▶ GPFS always tries to provide a uniform performance from all the cluster members. If the SAN connection speed is not uniform across all directly attached cluster members, the overall cluster performance will have bottlenecks caused by the slower members.
- ▶ Be sure not to pass the server I/O limit speed when sizing the maxMBps parameter to avoid I/O congestion to the server. Apply the following formula in most cases:

$$(\text{disk adapter speed}) * (\text{number of active connected disk adapters}) / 8 = \text{maxMBps}$$

We used the following values for this book:

$$(2 \text{ Gbps}) * (1 \text{ adapter}) / 8 = 256 \text{ MBps}$$

The customizing of the maxMBps parameter can be performed by the `mmchconfig` command, as shown in Example 5-21.

Example 5-21 Define maxMBps

```
# mmchconfig maxMBps=256
```

Note: Defining the maxMBps parameter to lower values can also help when the intention is to limit it to below the system capacity.

When maxMBps parameter is changed, the GPFS must be restarted to activate the configuration change.

5.5.2 Controlling GPFS parallelism

GPFS achieves parallelism in the cluster by creating several threads in each member to handle the I/O requests. These threads are of two types:

- ▶ `prefetchThreads` parameter
Handles the sequential data reading, read-ahead, and write-behind operations. Under the default configuration, this parameter is defined to open no more than 72 prefetchThreads threads in parallel.
- ▶ `worker1Threads` parameter
Controls the maximum number of parallel file operations that are performed concurrently, including random I/O requests. Under the default configuration, this setting is defined to open no more than 48 worker1Threads threads in parallel.

When customizing the number of threads be sure to have previously identified the type of operation that is performed by the applications that use the file system.

Increasing the number of prefetchThreads can have the following result:

- ▶ Improved response time of reading and write large sequential amount of data, such as video stream files.

Increasing the number of worker1Threads can have the following results:

- ▶ Increased file system capacity to create and remove small files in parallel
- ▶ Improved write and read performance for non-sequential files, such as database files

When customizing the thread parametrization, consider the following formula:

`worker1Threads + prefetchThreads <= 550 threads` on a 64-bit system
`worker1Threads + prefetchThreads <= 164 threads` on a 32-bit system

5.5.3 Linux considerations

The disk I/O operation queues (read and write) are managed in a system kernel by an *elevator algorithm*.

With the Linux kernel disk subsystem, you may select which algorithm is used. Under the 2.6.x kernel, the following algorithms are available:

- ▶ Completely Fair Queuing (CFQ)

This algorithm is designed for mid-size to large-size systems that have several processing units (cores). It keeps the I/O requests balanced across all processing units.

The algorithm is general purpose and is the default choice for almost all systems.

- ▶ Deadline

This algorithm tries to provide the minimum possible I/O latency to perform the request. It performs an aggressive I/O reorganization and delivers it in a round-robin fashion.

- ▶ NOOP

This algorithm provides a simple first in first out (FIFO) queue management. It assumes that all optimization is performed on the host bus adapter (HBA) or on the storage level.

The algorithm is suited to virtualized storage devices, such as the following images:

- VMware disk images (vmdsk)
- KVM and XEN disk images (qcow, vdi, bochs)

- ▶ Anticipatory (AS)

This algorithm introduces a controlled delay before dispatching the I/O request to aggregate sequential reads and writes as an attempt to reduce seek operations during the I/O request.

The algorithm is suited to the following items:

- Disks that store files that are always continuously read and write
- Locally attached disks
- Desktop environments

For a GPFS NSD server, or direct-attached cluster member, CFQ and Deadline are the most common choices and help deliver higher throughput as possible.

However, no significant performance improvement is usually detected on small and mid-size systems, such as those used during development of this book.

The elevator algorithm can be selected by sending to the kernel the elevator parameter during boot time, as shown in Example 5-22.

Example 5-22 Choose the elevator algorithm

```
[root@slovenia ~]# grep -v ^# /boot/grub/menu.lst
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux Server (2.6.18-194.el5)
    root (hd0,0)
    kernel /vmlinuz-2.6.18-194.el5 ro root=/dev/rootvg/rootlv elevator=cfq
    initrd /initrd-2.6.18-194.el5.img
[root@slovenia ~]#
```

5.5.4 Storage device considerations

Most storage devices provide read-ahead caches that help with sequential reads and writes. However, because of the parallelism characteristics of GPFS, the storage caches tend not to help with any performance improvement, and only overload the storage device processing unit.

When assigning LUNs from SAN-connected storage that use RAID disk arrays, be sure not to assign two LUNs from the same disk array to the GPFS file system to avoid increasing I/O latencies because of seek operations during data send or retrieve.

5.6 Monitoring performance

GPFS provides several tools and information to help monitor and identify performance bottlenecks.

The most common useful indicator of performance bottlenecks to the GPFS is the waiter information, accessible through the **mmdiag** and **mmfsadm** commands, as shown in Example 5-23.

Example 5-23 Determine waiters into GPFS through mmdiag and mmfsadm

```
[root@slovenia bin]# ./mmfsadm dump waiters
[root@slovenia bin]# ./mmdiag --waiters
=== mmdiag: waiters ===

[root@slovenia bin]#
```

Because of small utilization of the systems for this book, no waiters are observed. However through the identification of the waiters, you can determine which resources that GPFS is waiting to deliver, the determined request, or even identify which cluster member is slowing down the system performance.

An example output of the **mmdiag --waiters** command from the man page is shown in Example 5-24.

Example 5-24 Waiters, through the mmdiag command

```
=== mmdiag: waiters ===
0x11DA520 waiting 0.001147000 seconds, InodePrefetchWorker:
for I/O completion
0x2AAAAAB02830 waiting 0.002152000 seconds, InodePrefetchWorker:
for I/O completion
0x2AAAAB103990 waiting 0.000593000 seconds, InodePrefetchWorker:
for I/O completion
0x11F51E0 waiting 0.000612000 seconds, InodePrefetchWorker:
for I/O completion
0x11EDE60 waiting 0.005736500 seconds, InodePrefetchWorker:
on ThMutex 0x100073ABC8 (0xFFFFC2000073ABC8)
(CacheReplacementListMutex)
```

A direct approach to determine the file system I/O statistics is through the **mmpmon** command. This command can display two types of I/O statistics:

- ▶ **fs_io_s**

Displays I/O statistics for each mounted file system.

Through the **fs_io_s** type, you may determine the file system I/O behavior, which helps with the proper sizing of the GPFS cache and thread configuration

- ▶ **io_s**

Displays I/O statistics for the entire node.

The type of data to be shown is sent to the **mmpmon** command through a command file. In the command file, all statistics that should be shown must be listed. See Example 5-25.

Example 5-25 Command file for mmpmon

```
[root@slovenia bin]# cat /tmp/mmpmon_parm_file
fs_io_s
[root@slovenia bin]#
```

The **mmpmon** command gathers information at specified time intervals.

Example 5-26 shows two snapshots of the file system I/O utilization, every 5000 milliseconds (5 seconds).

Example 5-26 Collect file system statistics with mmpmon

```
[root@slovenia bin]# ./mmpmon -i /tmp/mmpmon_parm_file -d 5000 -r 2
mmpmon node 192.168.101.133 name slovenia fs_io_s OK
cluster:      itso.linux
filesystem:    gpfsfs1
disks:         4
timestamp:     1281968756/936815
bytes read:    0
bytes written: 0
opens:         2106
closes:        2106
reads:         0
writes:        0
readdir:       4271
inode updates: 1767
mmpmon node 192.168.101.133 name slovenia fs_io_s OK
cluster:      itso.linux
filesystem:    gpfsfs1
disks:         4
timestamp:     1281968761/937306
bytes read:    0
bytes written: 0
opens:         2106
closes:        2106
reads:         0
writes:        0
readdir:       4271
inode updates: 1767
[root@slovenia bin]#
```

Archived

Problem determination

This chapter covers General Parallel File System (GPFS) problem determination and troubleshooting and is intended as a guide to the approach to adopt when attempting to resolve a problem on a GPFS cluster or file system.

This chapter is not intended to replace the comprehensive documentation available for GPFS, *General Parallel File System Problem Determination Guide Version 3 Release 4*, GA76-0415. Instead, consider it as a useful supplement to guide you through the problem determination methodology.

A GPFS problem can manifest itself in many ways, and often the root cause of the problem is not obvious. This chapter describes an approach to problem determination that can guide you through the initial steps in problem determination and to help you narrow and identify the component or subsystem that is experiencing a problem.

In addition to helping you determine the likely cause of the problem, this chapter describes several of the most common problems that can occur with GPFS and, where possible, describes the actions you can take to resolve them.

This chapter contains the following topics:

- ▶ 6.1, “Problem determination process” on page 254
- ▶ 6.2, “GPFS debug commands” on page 258
- ▶ 6.3, “GPFS problem scenarios” on page 266

6.1 Problem determination process

This section describes the problem determination process.

6.1.1 Defining the problem

The first step in problem resolution is to clearly define the problem. It is important when trying to solve a problem to understand exactly what the users of the system perceive the problem to be. A clear definition of the problem is useful in two ways:

- ▶ It can give you a hint as to the cause of the problem.
- ▶ It allows you to demonstrate that the problem has been resolved.

Consider, for example, the situation where an application fails because it is unable to access a file. The problem might be as a result of the /gpfs1 file system not being mounted or being temporarily unavailable. The person solving the problem might fix this and demonstrate that the problem has been fixed by using the `mmdf` command to show that the /gpfs1 file system is mounted and then showing that the application runs successfully.

This example can also be used to illustrate another difficulty with problem determination. Problems can be hidden by other problems. When you fix or try to fix the most visible problem, another one might become apparent. The problems that are revealed during the problem determination process might be related to the one that was initially reported: multiple problems with the same symptoms. In some cases, you might discover problems that are completely unrelated to the one that was initially reported.

In this example, simply remounting the file system might not solve the problem being experienced by the user. The file system might have been unmounted because of a GPFS daemon (`mmfsd`) event or failure or an unavailable file system resource (an unavailable NSD), so that the file system cannot be remounted until the underlining problem has been fixed. To further complicate matters, the GPFS daemon or file system failing event might be the result of problems in other system components such as disk failures or network failures.

In this example, a better way of proving that the problem has been resolved is to verify the health of the critical components needed by the application (such as GPFS cluster and file system), and then restart the application and verify that it starts and runs correctly.

6.1.2 Gathering information from the user

The best way of understanding a problem from the user's perspective is to ask the user questions. Talking to the user can help you deduce where the problem might be, and the time frame in which the user expects it to be resolved. A user's expectations might be beyond the scope of the system or the application it is running. Ask the following questions when gathering information from the user during problem determination:

- ▶ What is the problem?

Ask the user to explain what the problem is and how it affects the user. Depending on the situation and nature of the problem, this question can be supplemented by either of the following questions:

- What is the system doing?
- What is the system not doing?

After you have determined the symptoms, try to establish the history of the problem.

- ▶ How did you first notice the problem?

- ▶ Did you do anything different that made you notice the problem?
- ▶ When did it happen?
- ▶ Does it always happen at the same time, for example, when the same job or application is run?
- ▶ Does the same problem occur elsewhere?
- ▶ Is only one node experiencing the problem or are multiple nodes experiencing the same problem?
- ▶ Have any changes been made recently?
This question refers to any type of change made to the system, ranging from adding new hardware or software, to configuration changes to existing software.
- ▶ If a change has been made recently, were all of the prerequisites met before the change was made?

Software problems most often occur when changes have been made to the system, and either the prerequisites have not been met, for example, system firmware is not at the minimum required level, or instructions were not followed properly.

Other changes, such as the addition of hardware can bring their own problems, such as cables incorrectly assembled, contacts bent, or addressing that is misconfigured.

The “How did you first notice the problem?” question might not help you directly, but it is useful in getting the person to talk about the problem. After starting to talk, the person invariably tells you things that can enable you to build a picture to help you to decide the starting point for problem resolution.

If the problem occurs on more than one node, look for similarities and differences between the situations.

6.1.3 Gathering application, GPFS, and system information

The process of defining the problem provides you with the external symptoms and hopefully a direction to investigate.

This section focuses on gathering information from the critical components of GPFS to help you more quickly identify the source of the problem. The critical components are divided, based on what the application and what GPFS needs to run successfully. The critical components are as follows:

- ▶ Application
- ▶ GPFS
- ▶ Operating System
 - Network
 - Disk

The idea is to first gather all of the data that might be needed and then examine it to determine the source of the problem.

Gathering application information

Gathering application-specific diagnostic data (if available) can help you perform initial problem determination of the application to determine whether the application is the source of the problem. If the source of the problem is not within the application, examine the GPFS or operating system data.

Gathering GPFS information

If you want to quickly gather the state of the GPFS cluster, for example when contacting IBM service, you can use the **gpfs.snap** command.

Considerations when gathering GPFS information

Consider the following items when running GPFS snap:

- From which nodes should I collect data?

Which nodes are experiencing the problem? If you do not know, the best approach is often to collect data from all nodes. In a large cluster, a sufficient way might be to start by collecting data from the manager nodes and the node experiencing the problem.

- The **mmdsh** command

When collecting data from multiple nodes you can use the **mmdsh** command. This command uses the administration communication method that is defined in the cluster to send any command out to the nodes.

The **mmdsh** command requires passwordless remote-shell communications between the nodes on which you want to collect data.

The syntax of the **mmdsh** command to be used is as follows:

```
/usr/lpp/mmfs/bin/mmdsh -N [all | working collective file] command-name
```

The *working collective file* is a file that consists of node names, one per line. One way to create, test, and use the working collective is as follows:

- a. Determine whether a working collective is needed.

GPFS 3.2.1.12 and later have an *all* nodeclass that is defined and can be used in conjunction with the **mmdsh** command to send the command to all the nodes in the cluster. This technique can be tested with the following command, using the operating system **date** command:

```
/usr/lpp/mmfs/bin/mmdsh -N all date
```

- b. If nodes in your cluster are known to be down when a problem is occurring, you can eliminate the timeout that **mmdsh** command experiences and reduce the time needed to collect the data needed. The following commands can be executed at the time of a problem to create a working collective (nodelist) that can be used in the same way that **-N all** is used in the previous example:

```
mmfsadm dump cfgmgr|awk '/ up / {print $3} /\(cluster/ {exit} /Current  
clock/ {exit} ' > nodelist
```

- c. Use the following command to test the collective:

```
/usr/lpp/mmfs/bin/mmdsh -F nodelist date
```

- Creating a GPFS dump directory

When GPFS encounters an internal problem, certain state information is saved in the GPFS dump directory for later analysis by IBM service.

One suggestion is that you create a directory for the placement of the problem determination information. Do not place the information in a GPFS file system, because that information might not be available if GPFS fails.

The default location of the dump directory is `/tmp/mmfs`, which must be created if it does not already exist. Administrators who want to use a separate directory for GPFS dumps can change the directory by issuing the following command:

```
mmchconfig dataStructureDump=/name_of_some_other_big_file_system
```

- ▶ Gathering data live versus gathering data after the problem has occurred

When determining GPFS problems, identifying when data must be gathered is important.

Gathering data live implies that you are currently experiencing a GPFS problem or that problem determination can occur only with data that is captured while the problem is occurring.

Gathering data after the failure and historical data (such as `mmfs.logs`, system error reports, and so on) can be gathered anytime after the problem occurred.

GPFS problems by categories

To help streamline the GPFS problem determination process, categorizing where the problem was encountered can help. The following categories have been created to break possible problems into topic areas:

- ▶ Installation of GPFS code
- ▶ Cluster creation and maintenance
- ▶ Cluster startup
- ▶ File system creation, maintenance, and startup
- ▶ Cluster and file system is active and operational

Gathering system information

Gathering system information helps to determine how the cluster is configured, the errors that are being produced, and the overall state of the cluster.

When looking at a GPFS cluster, consider that the scope of a system typically spans a cluster of computers that are connected through an interconnect (such as Ethernet) and have shared disks. Shared disk access is often through a storage area network (SAN) or a RAID array to provide direct disk access from multiple nodes. This approach means that collecting system data from any one of the nodes within the cluster (or clusters) might be necessary, from the network subsystem and from the disk subsystem.

The `gpfs.snap` command, mentioned previously, can collect initial local system data for any of the supported platforms (AIX, Linux, and Windows). This data is used in conjunction with the collected GPFS-specific data to help determine whether the problem is within GPFS or outside of GPFS.

If the problem is not within GPFS, you likely will have to refer to the platform-specific documentation to obtain additional system-data-collection procedures and to continue with system-problem determination.

6.2 GPFS debug commands

This section describes the commands that are used in collecting documentation when trying to perform problem determination regarding the issue with the GPFS cluster or file system. It lists the commands that a system administrator can use in an attempt narrow the root cause of a problem and the commands that IBM Service will run to help determine the problem.

6.2.1 Data collection commands

The following commands are used for data collection:

- ▶ **mmdiag**
- ▶ **mmfsadm**
- ▶ **mmtracectl**

mmdiag

This command was introduced in GPFS 3.4. It was created to give the system administrator an interface to query the mmfs daemon and gather useful information such as memory pool usage, I/O history, pending Remote Procedure Calls (RPCs), current trace levels, and more without the potential of causing an interruption in the file systems or clusters availability.

The options to the **mmdiag** command (as shown in Example 6-1) simplify the output of the **mmfsadm dump** options used in previous versions of GPFS and can be incorporated in scripts that are used to monitor cluster health.

Example 6-1 Options for mmdiag command

| | |
|---------------|---|
| mmdiag usage: | |
| --help | Display this help message |
| --all | Display everything |
| --version | Display information about the running GPFS build |
| --waiters | Display mmfsd threads waiting for events |
| --threads | Display mmfsd thread stats and the list of active threads |
| --memory | Display information about mmfsd memory usage |
| --network | Display information about mmfsd network connections |
| --config | Display configuration parameters and their settings |
| --trace | Display current trace status and trace levels |
| --iohist | Display recent IO history |
| --tokenmgr | Display information about token management |
| --stats | Display some general GPFS stats |

mmfsadm

The **mmfsadm** command and several of its options can be important during initial problem determination and when gathering documentation for submission to IBM Service.

The **mmfsadm** command is used to query the mmfs daemon without the use of locking. The command does this so that it can collect the data even when there are errors that have blocked other commands. In certain cases, the locking can cause either the mmfs daemon or the node to fail.

If you are running GPFS 3.4 or newer it is best to use the **mmdiag** command instead of the **mmfsadm** command for general queries of the mmfs daemon.

mmfsadm dump waiters

Waiters are a normal part of workload on a GPFS cluster. They represent work that the mmfs daemon must perform at the time the **mmfsadm dump waiters** command was executed. Under normal conditions, waiters seen in one iteration of the command have been completed before you run it again. However, for example, if the GPFS file system is hanging or performance is slow, the same waiters gain time upon multiple queries.

Gathering waiters from all nodes is the first step in problem determination path.

The ability to gather waiters from the cluster and make decisions based on those waiters is an important step in determining what might be happening on the cluster.

If you have a small number of nodes and no remote clusters, you can gather waiters from all of the nodes.

Gathering data from hundreds or thousands of nodes can be difficult as the network plays a larger role and can cause command timeouts and failures. However, to get a clear picture of what the problem might be, this approach is necessary.

If file systems are remotely mounted, all remote nodes in any remote cluster are part of the local cluster and the commands that are used to gather data on your local cluster must be gathered on those remote clusters also. This approach means logging into one of the nodes in the remote cluster and issuing the same command on those nodes as in the local cluster. If access to the remote cluster is not possible, instead gather the data that can be gathered locally for analysis.

Gathering the waiters

To understand where the problem might be, waiters must be gathered from all nodes in the cluster: all nodes in the local cluster and, if applicable, nodes in remote clusters.

From a node that can communicate through RSH or SSH with all nodes, use the command in Example 6-2 to capture waiters, redirecting to a file to allow for later investigation.

Example 6-2 Command to gather waiters from all nodes

```
/usr/lpp/mmfs/bin/mmdsh -F [all | wcoll-file] mmfsadm dump waiters >
/tmp/mmfs/allwaiters.$(date +%m%d%H%M)
```

On clusters with GPFS-3.4 nodes, you can use the **mmdia** **--waiters** command to replace **mmfsadm dump waiters** in Example 6-2.

This command can be executed a number of times to show movement of waiters, which might help in determining the cause of current issue.

mmfsadm dump all

The **dump all** option of the **mmfsadm** command is used when detailed information about the internal state of the daemon is needed for problem determination. The nodes on which the **dump all** option is run is usually determined by the nodes that have waiters at the time a problem is occurring. See Example 6-3.

Example 6-3 Command to gather detailed mmfs daemon state

```
/usr/lpp/mmfs/bin/mmfsadm dump all > /tmp/$(hostname -s).dumpall
```

See 6.3.3, “Scenario 2: Analyzing waiters” on page 269 for the procedure to analyze gathered waiters.

mmfsadm dump kthreads

Sometimes, you have to determine what the mmfsd kernel threads are doing. The **dump kthreads** option as shown in Example 6-4, lists all the kernel threads that are running at the time the command is executed. Use this output if a kernel thread does not appear to be responding. The thread and its associated traceback is displayed.

This output can be very large and time consuming to collect and as a result is not part of the general information that is requested when a reported problem occurs.

Example 6-4 Command to gather mmfs kern threads

```
/usr/lpp/mmfs/bin/mmfsadm dump kthreads > $(hostname -s).dumpkthreads
```

mmfs traces

GPFS tracing uses various tracing facilities based on the operating system. In AIX, it is based on the AIX trace facility, in Linux it uses embedded trace subsystem, and on Windows it uses the Windows ETL subsystem. The amount of detail that the traces produce varies, depending on the trace levels that are set. The majority of problems use default tracing levels. You can use the **mmdiag --trace** command to check the current levels.

When needed, the use of tracing in addition to the **mmfsadm dump waiters** and **mmfsadm dump all**, can assist in the problem determination process.

The two commands for taking traces in the GPFS environment are **mmtrace** and **mmtracectl**. Both result in traces being taken but are used at separate times for different reasons and purposes.

The **mmtrace** command is used when gathering traces on a limited number of nodes, and when the problem can be readily re-created. Although the **mmtrace** command has several parameters, the most common way to invoke it without options. See Example 6-5 for the syntax.

Example 6-5 The mmtrace syntax usage

```
mmtracectl [start | stop] [noformat | formatall] [tail | head | cont] [dispatch]
           [trace={io | all | def | same}]
           [trace="trace_class level [trace_class level ...]"]
```

The **mmtracectl** command is best used when tracing must span a recycle operation, whether it is daemon or node. There are more options with **mmtracectl** that allow you to take traces on multiple nodes, based on the actions of one. See Example 6-6 for syntax for **mmtracectl**. See the *General Parallel File System Administration and Programming Reference Version 3 Release 4, SA23-2221* for all **mmtracectl** parameters, and their explanation and uses.

Example 6-6 The mmtracectl syntax usage

```
mmtracectl {--start | --stop | --off | --set}
           [--trace={io | all | def | "Class Level [Class Level ...]"}]
           [--trace-recycle={off | local | global | globalOnShutdown}]
           [--aix-trace-buffer-size=BufferSize]
           [--tracedev-buffer-size=BufferSize]
           [--trace-file-size=FileSize] [--trace-dispatch={yes | no}]
           [-N {Node[,Node...]} | NodeFile | NodeClass}]
```

Each trace method for both AIX and Linux use the buffers TRCFILESIZE and TRCBUFSIZE to determine the size of buffers that are used to hold the trace before it is stopped and formatted. These buffers are cyclical and can be overwritten quickly on a node that has high usage. As a result, the default values for these parameters must be modified by exporting, prior to invoking the **mmtrace** command. The defaults are as follows:

- ▶ TRCFILESIZE defaults to 64 MB on both AIX and Linux.
- ▶ TRCBUFSIZE defaults to 32 MB on AIX and 64 KB on Linux.

Setting TRCFILESIZE to 128 MB on both AIX or Linux and the TRCBUFSIZE to 64 MB on AIX and 1 MB on Linux were found in our testing to be good values to initially use. It may be necessary to increase these parameters higher. On AIX, TRCBUFSIZE must minimally be one half of TRCFILESIZE. On Linux, TRCBUFSIZE can be a maximum of 1 MB.

Setting up tracing by using mmtrace

The following steps show how to set up tracing by using **mmtrace** on AIX and Linux.

1. Export the TRCFILESIZE and TRCBUFSIZE buffers:
 - On AIX:


```
export TRCFILESIZE=128000000; export TRCBUFSIZE=64000000
```
 - On Linux:


```
export TRCFILESIZE=128000000; export TRCBUFSIZE=1000000
```
2. Start the trace:


```
/usr/lpp/mmfs/bin/mmtrace
```
3. Check that **mmtrace** is running:


```
ps -ef | grep trace
```

 - On AIX, the command output is as follows:


```
root 13842      1  0 06:21:02 pts/1  0:00 /bin/trace -a -l -L 128000000 -T
64000000 -j 005,006,00A,306,307,308,309 -o /tmp/mmfs/trcfile.usa
```
 - On Linux, the command output is as follows:


```
root      19129      1  0 Jul13 ?          00:00:00 /usr/lpp/mmfs/bin/lxtrace on
/tmp/mmfs/lxtrace.trc.slovenia 256000000 1000000
```
4. Check tracing levels to ensure they are at the intended levels. Most tracing is done with the default tracing value of 4 if not otherwise requested by IBM Service. Use the following command:


```
/usr/lpp/mmfs/bin/mmdiag --trace
```

Example 6-7 shows a portion of the **mmdiag --trace** output.

Example 6-7 Output of mmdiag command

```
=== mmdiag: trace ===
trace levels: (tracing is on)
      alloc : 4  (disk space allocation)
      allocmgr : 4  (allocation manager)
      basic : 4  ('basic' classes)
      brl : 4  (byte range locks)
      cleanup : 4  (cleanup routines)
      cmd : 4  (TS commands)
      defrag : 4  (defragmentation)
      dentryexit : 4  (daemon routine entry/exit)
      disk : 4  (physical disk I/O)
```

```
dmapi : 4 (data management)
ds : 4 (data shipping)
errlog : 4 (error logging)
fs : 4 (file system)
fsck : 3 (online multinode fsck)
```

5. Re-create the failure.

6. Stop the trace:

```
/usr/lpp/mmfs/bin/mmtrace stop
```

– On AIX, the command output is as follows:

```
mmtrace: move /tmp/mmfs/trcfile.usa /tmp/mmfs/trcfile.100713.06.27.37.usa
mmtrace: formatting /tmp/mmfs/trcfile.100713.06.27.37.usa to
/tmp/mmfs/trcrpt.100713.06.27.37.usa
```

– On Linux, the command output is as follows:

```
mmtrace: move /tmp/mmfs/lxtrace.trc.slovenia
/tmp/mmfs/trcfile.100713.06.30.34.slovenia
mmtrace: formatting /tmp/mmfs/trcfile.100713.06.30.34.slovenia to
/tmp/mmfs/trcrpt.100713.06.30.34.slovenia
```

7) Submit the information and any additional requested documentation to IBM Service for review.

Setting up tracing by using mmtracectl

The following steps show how to set up tracing by using **mmtracectl** on AIX and Linux.

1. Start tracing by setting TRCFILESIZE and TRCBUFSIZE buffers to increased values and also to recycle trace on all nodes when an event occurs on one node.

– On AIX, the command is as follows:

```
/usr/lpp/mmfs/bin/mmtracectl --start --trace-recycle=global
--trace-file-size=128000000 --aix-trace-buffer-size=64000000
```

– On Linux, the command is as follows:

```
/usr/lpp/mmfs/bin/mmtracectl --start --trace-recycle=global
--trace-file-size=128000000 --tracedev-buffer-size=1000000
```

2. Check that the trace is running:

```
ps -ef | grep trace
```

– On AIX, the **ps** command output is as follows:

```
root 13842      1  0 06:21:02 pts/1    0:00 /bin/trace -a -l -L 128000000 -T
64000000 -j 005,006,00A,306,307,308,309 -o /tmp/mmfs/trcfile.usa
```

– On Linux, the **ps** command output is as follows:

```
root      19129      1  0 Jul13 ?          00:00:00 /usr/lpp/mmfs/bin/lxtrace on
/tmp/mmfs/lxtrace.trc.slovenia 256000000 1000000
```

3. Check tracing levels to ensure that they are at the intended levels. Most tracing is done with the default tracing value of 4, if not otherwise requested by IBM Service.

```
/usr/lpp/mmfs/bin/mmdiag --trace
```

Example 6-8 on page 263 shows a portion of the **mmdiag --trace** output.

Example 6-8 Output of mmdiag command

```
=== mmdiag: trace ===
  trace levels: (tracing is on)
    alloc : 4  (disk space allocation)
    allocmgr : 4  (allocation manager)
    basic : 4  ('basic' classes)
    brl : 4  (byte range locks)
    cleanup : 4  (cleanup routines)
    cmd : 4  (TS commands)
    defrag : 4  (defragmentation)
    dentryexit : 4  (daemon routine entry/exit)
    disk : 4  (physical disk I/O)
    dmapi : 4  (data management)
    ds : 4  (data shipping)
    errlog : 4  (error logging)
    fs : 4  (file system)
    fsck : 3  (online multinode fsck)
```

4. Wait for the failure event to occur.
5. After the failure event, the mmfs daemon traces are in the `/tmp/mmfs` location on each node. Gather the `trcrpt` file (or files) information along with the `gpfs.snap` file (or files) data for submission to IBM Service.

Using the `gpfs.snap` script (which parameters on which nodes)

The `gpfs.snap` script gathers all the general documentation that is needed for IBM Service to investigate most issues. The script creates a `.tar` file in the `/tmp/gpfs.snapOut` location. Depending on the options that are used to invoke it, the amount and scope of the documentation that is gathered can vary. Although having a `gpfs.snap` script output from your cluster is important, gathering waiters and dumping all information should take precedence.

Because the size of the cluster is not known, taking one *master* snapshot from one node and a *non-master* snapshot from all the other nodes is necessary. The node that is chosen on which to run the master snap must be able to communicate (with the `ssh` or `rsh`) to all other nodes in the cluster. Note the following information:

- ▶ The `gpfs.snap` script (with no options) gathers a *master snap*. This script gathers information about the node on which it was invoked and also gathers information about the cluster as a whole. Part of that cluster information is a file that consists of merged mmfs log files from all the nodes.
- ▶ The `gpfs.snap -z` script (with option) generates a *non-master snap* that consists of data gathered from only the node on which it was executed.

See *General Parallel File System Problem Determination Guide Version 3 Release 4*, GA76-0415 for a full syntax and use of the `gpfs.snap` script.

6.2.2 Data analysis commands and scripts

This section describes common commands and scripts used for analyzing collected data.

The AWK scripts listed in this section are in the `/usr/lpp/mmfs/samples/debugtools` directory. They are used to help analyze the mmfs `trcrpts` (`trsum.awk`) file or the `/var/log/messages` on Linux (`fsstructlx.awk`) or the output from `errpt -a` command on AIX (`fsstruct.awk`).

trsum.awk

This .awk script takes a trcrpt data file that has been gathered by using either the **mmtrace** or **mmtracectl** method and divides the trace into stanzas that help you with a visual understanding of what occurred over the time period that the trace was gathered.

The syntax for trsum.awk is as follows:

```
trsum.awk [detail=0] trcrpt-file > somefile
```

In the syntax, detail=0 indicates to omit the detail section of the trsum and only generate the histogram sections.

The stanzas are in the usual trsum output as shown in Table 6-1.

Table 6-1 The trsum stanza outputs

| Stanza | Details shown |
|-------------------------------|---|
| Detail trace section | Shows system call entry, exit, and the elapsed time. |
| Elapsed time | Duration of the trace based on the size of the trace buffers and node activity. |
| Operations stats | GPFS operation that is performed and total time, count, and average time to perform those operations. |
| User thread stats | Per thread account of time broken out into time waiting on GPFS and the application. |
| Total App-read/write | Histograms based application reads/writes information. |
| Max concurrent App-read/write | Maximum reads/writes that the application had concurrent at any one time. |
| Total I/Os | Histogram based on I/O times giving total count and individual read/write counts. |
| Per disk I/Os | Histogram based on I/O times giving total count and individual read/write counts. |
| Max concurrent I/Os | Maximum I/Os running concurrently. |
| Disk utilization | Per disk utilization over the duration of the trace showing I/Os handled. |
| Number of sectors | Histogram showing the number of sectors (in 512 byte increments) written/read. |
| I/O throughput | Histogram showing M/sec of reads/writes from first I/O to last. |
| RPC responsiveness | Histogram of any RPC that was called during the trace. |

fsstruct.awk for AIX and fsstructlx.awk for Linux

The purpose of these scripts is to summarize events either from **errpt -a** from AIX or /var/log/messages from Linux. The scripts can either gather mmfs specific or general events and create a one-line time-stamped entry for each.

The output generated by these .awk scripts can help identify where an mmfs problem started. For instance, an unmount of a file system may have started as an NSD failure on nodeA and propagated to the other nodes as an error E_NO_MGR (rc=212). Being able to identify which node was first to report a problem can help point to what the real issue is.

Because of where mmfs errors are reported in various locations, if the cluster contains mixed operating systems, the use of working collectives are needed when collecting data to take this into account.

Although these scripts can be run on a single node, they are really meant to take these error messages log files from multiple or all nodes in the cluster and condense all the entries to allow sorting.

For example, the following commands collect error log information coming from multiple nodes:

► On AIX:

```
mmddsh -N [all|aix.wcoll] "errpt -a" > errpt.out
```

► On Linux:

```
mmddsh -N [all | linux.wcoll] 'cat /var/log/messages' > messages.out
```

To create an mmfs-only summary, use the following command:

```
/usr/lpp/mmfs/samples/debugtools/fsstruct[1x].awk errpt.out | sort -k 1,1 > all.gpfs
```

To get a summary from both the operating system and GPFS from the same file, use the following command:

```
/usr/lpp/mmfs/samples/debugtools/fsstruct[1x].awk all=1 errpt.out | sort -k 1,1 > all.err
```

6.2.3 mmfs logs

When there is problem with GPFS or a GPFS file system, one of the first places to look is in the mmfs logs that are located in `/var/adm/ras`. Although looking on the node where the problem was noticed is a good start, because GPFS is a parallel file system the problem might involve or have involved another node or nodes. Gathering logs from all nodes is often necessary to have a more thorough view of the problem.

The following command can be used to gather logs from a specific date. This example is similar to what can be found in the `/usr/lpp/mmfs/samples/gatherlogs.samples.sh` script. For example, for a problem that was seen on June 24, use the following command:

```
mmddsh -N all "grep -h \"Jun 24\" /var/adm/ras/mmfs.log.2*" | sort -k 4,5 > allGPFS.logs
```

Example 6-9 shows a daemon assert that was generated on a node and found by this method.

Example 6-9 An assert found in the logs on node usa

```
usa: Tue Jul 27 08:16:47.357 2010: Signal 6 at location 0x900000000712650 in process 7798934,
link reg 0xFFFFFFFFFFFFFFFF.
usa: Tue Jul 27 08:16:47.365 2010:  r0: 0xFFFFFFFFFFFFFFFF      r1: 0x0000000110FC34A0
usa: Tue Jul 27 08:16:47.377 2010:  r2: 0xFFFFFFFFFFFFFFFF      r3: 0x0000000000000000
usa: Tue Jul 27 08:16:47.385 2010:  r4: 0xFFFFFFFFFFFFFFFF      r5: 0xFFFFFFFFFFFFFFFF
usa: Tue Jul 27 08:16:47.393 2010:  r6: 0xFFFFFFFFFFFFFFFF      r7: 0xFFFFFFFFFFFFFFFF
usa: Tue Jul 27 08:16:47.401 2010:  r8: 0xFFFFFFFFFFFFFFFF      r9: 0xFFFFFFFFFFFFFFFF
usa: Tue Jul 27 08:16:47.409 2010: r10: 0xFFFFFFFFFFFFFFFF     r11: 0xFFFFFFFFFFFFFFFF
usa: Tue Jul 27 08:16:47.417 2010: r12: 0xFFFFFFFFFFFFFFFF     r13: 0x0000000110FD2800
usa: Tue Jul 27 08:16:47.425 2010: r14: 0x0000000110090C58      r15: 0x0000000000000020
```

```

usa: Tue Jul 27 08:16:47.433 2010: r16: 0x0000000110D44E70 r17: 0x0000000000000000
usa: Tue Jul 27 08:16:47.441 2010: r18: 0x0000000110000DA0 r19: 0x0000000000000000
usa: Tue Jul 27 08:16:47.449 2010: r20: 0x0000000000000002 r21: 0x0000000110BFF98D
usa: Tue Jul 27 08:16:47.457 2010: r22: 0x000000000000000E r23: 0xFFFFFFFFFFFFFFFF
usa: Tue Jul 27 08:16:47.465 2010: r24: 0x0000000000000000 r25: 0x0000000110FC379B
usa: Tue Jul 27 08:16:47.473 2010: r26: 0x0000000110FC3790 r27: 0x0000000000000001
usa: Tue Jul 27 08:16:47.481 2010: r28: 0x0000000110FCAA78 r29: 0x0000000000000006
usa: Tue Jul 27 08:16:47.493 2010: r30: 0x0000000000000006 r31: 0x0000000000005354
usa: Tue Jul 27 08:16:47.509 2010: iar: 0x0900000000712650 msr: 0xA00000000000D032
usa: Tue Jul 27 08:16:47.521 2010: cr: 0x0000000000002000 link: 0xFFFFFFFFFFFFFFFF
usa: Tue Jul 27 08:16:47.529 2010: ctr: 0xFFFFFFFF006FA300 xer: 0x00000000FFFFFFFF
usa: Tue Jul 27 08:16:47.537 2010: exad: 0x0000000110F528C8
usa: Tue Jul 27 08:16:47.545 2010: 0x900000000712650 pthread_kill() + 0xB0
usa: Tue Jul 27 08:16:47.553 2010: 0x900000000711EC8 _p_raise() + 0x48
usa: Tue Jul 27 08:16:47.561 2010: 0x90000000002BD2C raise() + 0x4C
usa: Tue Jul 27 08:16:47.569 2010: 0x900000000088504 abort() + 0xC4
usa: Tue Jul 27 08:16:47.577 2010: 0x900000000088344 __assert_c99() + 0x2E4
usa: Tue Jul 27 08:16:47.585 2010: 0x100005C54 logAssertFailed() + 0x244
usa: Tue Jul 27 08:16:47.593 2010: 0x10002CD0C runTSTest(int,const char*,int&) + 0xFD8
usa: Tue Jul 27 08:16:47.601 2010: 0x100035E94 runTSDebug(int,int,char**) + 0x380
usa: Tue Jul 27 08:16:47.609 2010: 0x100426B14 RunClientCmd(MessageHeader*,IpAddr,unsigned
short,int,int,StripeGroup*,RpcContext*) + 0x8F8
usa: Tue Jul 27 08:16:47.617 2010: 0x100428418 HandleCmdMsg(void*) + 0x67C
usa: Tue Jul 27 08:16:47.625 2010: 0x10003B94C Thread::callBody(Thread*) + 0xDC
usa: Tue Jul 27 08:16:47.633 2010: 0x100003014 Thread::callBodyWrapper(Thread*) + 0xB0
usa: Tue Jul 27 08:16:47.641 2010: 0x9000000006FAD50 _pthread_body() + 0xF0
usa: Tue Jul 27 08:16:47.649 2010: 0xFFFFFFFFFFFFFFFFC

```

6.3 GPFS problem scenarios

Given the data collection and tools described in previous sections, this section provides the steps to use when collecting documentation for common GPFS problem scenarios.

6.3.1 Considerations

This section provides details about the areas to consider when challenges occur in your environment.

File system not mounting

Several reasons exist for a file system not mounting. Depending on the scope of the problem can determine where the focus of the problem determination begins.

Reasons for not mounting are as follows:

- ▶ GPFS quorum (Quorum is defined as one plus half of the explicitly defined quorum nodes in the GPFS cluster.)
- ▶ Node rNode recovery
- ▶ Down NSDs
- ▶ DMAPi is enabled

If the file system not mounting is cluster-wide, such as after rebooting a cluster, focus on determining whether the cluster quorum is met. If the quorum semantics are broken, GPFS performs recovery in an attempt to achieve quorum again.

If the file system that is not mounting is limited to one or a few nodes in the cluster, the problem determination process should begin by looking at the mmfs logs and waiters from all the nodes.

Quorum

Issues with quorum state at cluster startup time are noticed as the node or nodes are not joining the cluster or a file system is not mounting. GPFS quorum must be maintained within the cluster for GPFS to remain active. Use the commands as follows to check the quorum states:

- ▶ Use the following command to determine how many and which nodes are quorum nodes:
`mmfsccluster | grep quorum`
- ▶ Use the following command to determine state (down, arbitrating, or active) of these nodes:
`mmgetstate -a`

The formula for quorum is as follows:

$(\text{\#quorumNodes}/2) + 1$

If quorum is not met, the cluster does not form and the file systems does not mount.

GPFS file system hangs

The general data collection steps when a GPFS file system hang occurs are as follows:

1. Gather waiters from all nodes (see 6.2.1, “Data collection commands” on page 258).
2. From all node with waiters, use `mmfsadm dump all`.
3. If any remote clusters exists, perform steps 1 and 2 from those clusters.
4. Use `gpfs.snap` from one node; use `gpfs.snap -z` from all other nodes.

GPFS related command hangs

The general data collection steps when file-system-related commands or scripts hang (such as `ls`, `cp`, `mv`, `mkdir`, and so on) are as follows:

1. Check for long waiters. If waiters exist, treat the problem as a possible hang.
2. If no waiters exist, continue and re-create the capturing of the trace.
3. Export the appropriate TRCFILESIZE and TRCBUFSIZE buffers.
4. Start the mmfs tracing with the `mmtrace` command.
5. Re-create the error.
6. Stop mmfs tracing.
7. Gather the data with the `mmfsadm dump all` command.

GPFS administration command hangs

The general data error collection steps when a GPFS administration command hangs are as follows:

1. Run the script `/tmp/mmtrace.out`
2. Recreate the capturing of the trace.
3. Set the export `DEBUG=1`
4. Recreate the error again
5. Exit the trace
6. Run the `gpfs.snap` script

6.3.2 Scenario 1: Upgrading GPFS

When upgrading, consider what software you have upgraded, for example, upgrading GPFS from one PTF to another and going from one AIX Technical Level or Linux Service Pack to the next. In upgrades such as these, be sure to check the `mmfs.logs`, and `errpt -a` or `/var/log/messages` for issues with the OS, network, or disk subsystem that can affect the GPFS cluster. Pursue those issues either first or in parallel with the GPFS efforts.

A typical problem after upgrading GPFS from an earlier version is that the `mmfsd` daemon fails to start. The `/var/adm/mmfs.logs.latest` file indicates the error shown in Example 6-10.

Example 6-10 Checking the `mmfs.log` file for errors

```
Removing old /var/adm/ras/mmfs.log.* files:
Loading kernel extension from /usr/lpp/mmfs/bin . . .
GPFS: 6027-506 /usr/lpp/mmfs/bin/mmfskxload: /usr/lpp/mmfs/bin/aix64/mmfs64 is
already loaded at 66063360.
Sun Jul  4 02:54:15.924 2010: GPFS: 6027-1548 Error: daemon and kernel extension
do not match.
```

The problem is that the new code, either a PTF or version upgrade, was applied and the kernel extension from the previous PTF or version upgrade did not unload. The solution is to unload the old kernel extension so that the new one can be loaded.

This step can be accomplished in two ways:

- ▶ The best and most complete way is to reboot the node or nodes that are receiving this error. This way ensures that everything is loaded correctly. See Example 6-11.

Example 6-11 Rebooting the node to load the kernel extension

```
Sun Jul  4 04:03:12 MEDT 2010: Node rebooted. Starting mmautoload...
Sun Jul  4 04:03:24 MEDT 2010: runmmfs starting
Removing old /var/adm/ras/mmfs.log.* files:
Loading kernel extension from /usr/lpp/mmfs/bin . . .
GPFS: 6027-500 /usr/lpp/mmfs/bin/aix64/mmfs64 loaded and configured.
Sun Jul  4 04:03:25.353 2010: GPFS: 6027-310 mmfsd64 initializing. {Version:
3.3.0.6   Built: May 27 2010 16:43:37} ...
```

- ▶ The second way is to run the `mmfsenv -u` command to unload the kernel extension. This command attempts to unload the kernel extension. For instance, if the command is successful on AIX, the message stating that the kernel extension has unloaded is displayed (Example 6-12). If unsuccessful, the kernel is still loaded (Example 6-13).

Example 6-12 Unloading the kernel extension

```
> /usr/lpp/mmfs/bin/mmfsenv -u
/usr/lpp/mmfs/bin/mmfskxunload: module /usr/lpp/mmfs/bin/aix32/mmfs unloaded.
```

Example 6-13 The kernel is still loaded

```
> mmfsenv -u
sysconfig(SYS_CFGKMOD-term): Device busy
```

If the kernel extension cannot be unloaded, rebooting is mandatory.

6.3.3 Scenario 2: Analyzing waiters

This section gives a basic technique for analyzing waiters that can help the problem determination process in finding the cause or causes of a file system hang or performance issue that is occurring or has occurred on the GPFS cluster. The intention is to provide a high level overview at what the significant waiters are and what information can be gleaned from them.

Note: Long running waiters are usually indicative of a slow disk, a slow network, or a slow CPU.

The presumption is that waiters have been gathered from all nodes, including nodes in the local cluster and any remote clusters that remotely mount file systems.

The process here as a general practice is one that first removes waiters that are considered *secondary* to any problem that is occurring. After these waiters are removed, a hierarchy can be applied to the remaining waiters. This hierarchy is a list of waiter types that is traversed from top to bottom, looking for a match. After a match is found, these waiter types can point to areas such as node recovery, I/O, networking, or tuning to investigate. They can also reveal a GPFS deadlock or simply a heavily used GPFS file system.

The following list provides information from the GPFS waiters that might help with identifying the waiters that can be ignored and removed from the analysis. Typically, the text follows the *reason* portion of a waiter. Example 6-14 shows where the reason portion is found in a typical GPFS waiter.

Example 6-14 Showing the waiter information

```
0x40001D5B6C0 waiting 5628.211637314 seconds, Create handler: on ThCond
0x18002F07AD0 (0xD000000002F07AD0) (Lk0bj), reason 'change_lock_shark waiting to
set acquirePending flag'
```

- Make a copy of the original waiters file to use for this process.

Remove waiters from the waiters file that have the following reasons:

- change_lock_shark waiting to set acquirePending flag
- waiting for <lock-type> lock where <lock-type> can be LX, RF, RS, RO, XW, WW
- waiting because of local byte range lock conflict
- waiting for fetch-n-lock to complete
- RPC wait for tmMsgTellAcquire1 on node xxx.xxx.xxx
- waiting because of local byte range lock conflict
- wait for SubToken to become stable
- waiting until pit work is complete

- Based on the following ordered list, organize the remaining waiters into the categories listed, looking for matches to the given text of each waiters text

a. Node Recovery

Look for the following waiters:

- GroupProtocolDriverThread
- RPC wait for ccMsgGroupLeave
- RPC wait for sgmMsgExeTMPhase
- MMFS group recovery phase
- waiting for stripe group takeover

These types of waiters point to recovery being done in the cluster. If a node or nodes have left or are joining the cluster, RPCs exist that all nodes must respond to. If one (or more) nodes do not respond, activity on the cluster will hang until responses are received.

RPCs are interprocess communications that allow mmfs daemons to transfer data to and from each other by using multiple threads and one or more processes.

Perform the following action:

- i. Node recovery is driven by the Cluster Manager node. Use the **mm1smgr** command to locate the Cluster Manager.
 - ii. Use the **mmdiag --network** (GPFS-3.4) or **mmfsadm dump tscomm** command on the Cluster Manager to locate the nodes that have “Pending messages.”
 - iii. Check these nodes for network and resource or performance issues.
 - iv. If a problem is to be opened with IBM Service, gather the default documentation that is outlined in “Scenario 1: Upgrading GPFS” on page 268.
- b. Local Disk I/O

Look for the following waiters:

- NSD I/O worker: for I/O completion on disk
- Mount handler: for open disk device

These waiters appear on NSD server nodes and can indicate an issue in the disk subsystem depending on the their length.

Perform the following action:

- i. Use system tools such as **iostat** or **topas** to monitor the output for any abnormal disk activity for the disks and review the system error logs for AIX, Linux or Windows along with the event logs for disk subsystem and SAN for possible causes.
 - ii. If a problem is to be opened with IBM Service, gather the default documentation that is outlined in “Scenario 1: Upgrading GPFS” on page 268.
- c. Network I/O

Look for the following waiters:

- RPC wait for NSD I/O completion on
- RPC wait for getData on node <xxx.xxx.xxx>
- waiting for exclusive use of connection

These waiters point to an issue in the network between the client node and the NSD server that they point to. These waiters may be a secondary issue if there are also long waiters waiting on Local Disk I/O. Investigate Local Disk I/O waiters first.

Perform the following action:

- i. Check the network with commands such as **netstat -D** or **netstat -s**, looking for packet drops or retransmit errors in the TCP section. All network switches are between the client and the NSD server node.
 - ii. Look at the system error logs for AIX, Linux, or Windows for network-related errors.
 - iii. If a problem is to be opened with IBM Service, gather the default data that is outlined in 6.3.2, “Scenario 1: Upgrading GPFS” on page 268.
- d. Token Revoke

Look for the following waiter:

- RPC wait for tmMsgRevoke on node <xxx.xxx.xxx>

This waiter happens when one node is waiting on a resource that is held by another. There may be multiple nodes that have waiters waiting on *tmMsgRevokes*. Many times one node can be identified as the source of the contention as the *tmMsgRevoke* point to one node or one node points to the next until the source is found.

Perform the following action:

- i. Gather the default documentation that is outlined in 6.3.2, “Scenario 1: Upgrading GPFS” on page 268.
 - ii. Determine the node that is holding the resources deadlock by examining the *tmMsgRevoke message(s)* and the IP address they point to. If one node can clearly be identified as the source, issue the **mmfsadm cleanup** command on that node. This step causes the mmfs daemon to recycle.
- e. Other waiters
- Look for the following waiters:
- stealEngine loop wrapping
 - In kernel waiting to quiesce

These waiters do not have the same cause in common and is why they are placed in the “Other waiters” category.

Perform the following action:

- i. The “stealEngine loop wrapping” waiter usually is present if the page pool is too small and there are not enough buffers in the page pool to work with. Increase the page pool with the **mmchconfig** command (using the **-l** or **-i** option can alleviate this issue).
 - ii. The “In kernel waiting to quiesce” waiter usually results from the **mmcrsnapshot**, **mmdeisnapshot**, **mmdeifileset**, or **mmfsctl suspend** commands.
 - ii. If a problem is to be opened with IBM Service, gather the default documentation that is outlined in 6.3.2, “Scenario 1: Upgrading GPFS” on page 268 along.
- f. Long waiters
- Action: Gather the documentation that is listed in the “Gathering the waiters” on page 259.
- Start the investigation with the category that is highest on the ordered list.
- See 6.3.5, “Scenario 4: GPFS file system hang” on page 273 for an example of applying the rules given a hung file system.

6.3.4 Scenario 3: Application failure

You reach this scenario because initial problem determination was performed on the application involved and you determined that the eliminated problem is not in the GPFS cluster. Perform the following steps:

1. Gather the needed GPFS documentation:
 - Collect the mmfs logs from all the nodes in the GPFS cluster (see “mmfs logs” on page 265) on or before the time of failure.
 - Collect system data (**errpt -a** from AIX, `/var/log/messages` from Linux and Windows) from all nodes and use the `fsstruct.awk` or `fsstructlx.awk` scripts.
2. Review the gathered mmfs logs and system logs to determine if there may be reason for the application failure.

The next several examples show what might be found in those searches:

- ▶ Example 6-15 and Example 6-16 show what are found in the mmfs logs.
- ▶ Example 6-17 shows what is found after combining and sorting the /var/log/messages from all the nodes.

Example 6-15 shows the mmfs logs, which indicate a disk failure on gpfs1nsd that is part of the gpfsfs file system.

Example 6-15 The mmfs logs show a disk failure

```
germany: Sun Jul  4 18:45:12.582 2010: GPFS: 6027-680 Disk failure.  Volume
gpfsfs. rc = 22. Physical volume gpfs1nsd.
germany: Sun Jul  4 18:45:13.301 2010: GPFS: 6027-680 Disk failure.  Volume
gpfsfs. rc = 22. Physical volume gpfs1nsd.
```

Other examples of problems that can cause an application failure are expulsion either for an expired lease (Example 6-16) or connection drop (Example 6-17) and a deadman switch timer (Example 6-18).

Example 6-16 Expelled because of expired lease

On the node that is being expelled:

On the Cluster manager node (mm1smgr):

```
Tue Jul 6 21:48:54.407 2010: Node 192.168.101.142 (germany in mantest.usa) is
being expelled due to expired lease.
```

Example 6-17 Expelled because of connection reset by peer

On the node detecting connection reset(slovenia):

```
Tue Jul 6 08:21:25.046 2010: Close connection to 192.168.101.142 germany
(Connection reset by peer)
Tue Jul 6 08:22:03.698 2010: This node will be expelled from cluster mantest.usa
due to expel msg from 192.168.101.142 (germany)
```

On the Cluster manager node (mm1smgr ..):

```
Tue Jul 6 08:22:03.691 2010: Expel 192.168.101.133 (slovenia) request from
192.168.101.142 (germany). Expelling 192.168.101.133 (slovenia)
```

Example 6-18 Deadman switch timer

```
07/13@17:30:01 slovenia /usr/sbin/cron[13356]: (root) CMD ([ -x /usr/lib64/sa/sa1
] && exec /usr/lib64/sa/sa1 -S ALL 1 1)
07/13@17:35:01 slovenia /usr/sbin/cron[13853]: (root) CMD ([ -x /usr/lib64/sa/sa1
] && exec /usr/lib64/sa/sa1 -S ALL 1 1)
07/13@17:39:43 solvenia kernel: GPFS Deadman Switch timer [0] has expired; IOs in
progress: 0
07/13@17:40:01 slovenia/usr/sbin/cron[14139]: (root) CMD ([ -x /usr/lib64/sa/sa1 ]
&& exec /usr/lib64/sa/sa1 -S ALL 1 1)
07/13@17:42:14 slovenia nfsd[14666]: nfssvc: Setting version failed: errno 16
(Device or resource busy)
07/13@17:42:14 slovenia sm-notify[14667]: Already notifying clients; Exiting!
```

6.3.5 Scenario 4: GPFS file system hang

The goal of this scenario is to determine the node or nodes that are causing the hang situation and gather all the documentation possible.

Example 1

A node in a local cluster remotely mounts its file system from a remote cluster. The external symptom is a file system hang. Perform the following steps:

1. View the waiters from all the nodes. See Example 6-19.

Example 6-19 Viewing the waiters

```
usa: 0x108395A0 waiting 491.816423025 seconds, SharedHashTab fetch handler: on ThCond
0x1077FEB0 (0x1077FEB0) (MsgRecord), reason 'RPC wait' for tmMsgTellAcquire1 on node
192.168.101.142
usa: 0x10819CA0 waiting 6751.419359689 seconds, Writebehind worker: on ThCond 0x400025491B0
(0x400025491B0) (MsgRecord), reason 'RPC wait' for NSD I/O completion
usa: 0x40000918260 waiting 6752.659368250 seconds, Writebehind worker: on ThCond 0x1086C5A0
(0x1086C5A0) (MsgRecord), reason 'RPC wait' for NSD I/O completion
usa: 0x40000916FB0 waiting 6750.199351265 seconds, Writebehind worker: on ThCond usa:
0x107B6950 waiting 6750.309352025 seconds, Writebehind worker: on ThCond 0x400010ADB60
(0x400010ADB60) (MsgRecord), reason 'RPC wait' for NSD I/O completion
usa: 0x1065A450 waiting 6748.259337872 seconds, Sync handler: on ThCond 0x180030BE068
(0xD0000000030BE068) (LkObj), reason 'waiting for WW lock'
germany:0x11181C630 waiting 6008.507415856 seconds, NSD I/O Worker: on ThCond 0x11709EB60
(0x11709EB60) (MsgRecord), reason 'RPC wait' for getData onnode 192.168.101.132
```

2. Using the rules Example 6-19, first remove the *secondary* waiters from all the waiters that are gathered. These excluded waiters are shown in Example 6-20.

Example 6-20 Removing the secondary waiters

```
usa: 0x108395A0 waiting 491.816423025 seconds, SharedHashTab fetch handler: on ThCond
0x1077FEB0 (0x1077FEB0) (MsgRecord), reason 'RPC wait' for tmMsgTellAcquire1 on node
192.168.101.142
usa: 0x1065A450 waiting 6748.259337872 seconds, Sync handler: on ThCond 0x180030BE068
(0xD0000000030BE068) (LkObj), reason 'waiting for WW lock'
```

3. Using the hierarchy from the list, look for each type of waiter to help classify where the problem might be. The 'RPC wait' for NSD I/O completion waiter on usa server node says that an I/O is waiting in the network for its read or write processing to complete. The 'RPC wait' for getData waiter on the germany server node says that it is waiting to get data from node 192.168.101.132, which is the usa node. These nodes are waiting for each other and is the result cause of the hang condition. See Example 6-21.

Example 6-21 Finding using the waiter the reason of the hang condition

```
usa: 0x10819CA0 waiting 6751.419359689 seconds, Writebehind worker: on ThCond 0x400025491B0
(0x400025491B0) (MsgRecord), reason 'RPC wait' for NSD I/O completion
usa: 0x40000918260 waiting 6752.659368250 seconds, Writebehind worker: on ThCond 0x1086C5A0
(0x1086C5A0) (MsgRecord), reason 'RPC wait' for NSD I/O completion
usa: 0x40000916FB0 waiting 6750.199351265 seconds, Writebehind worker: on ThCond usa:
0x107B6950 waiting 6750.309352025 seconds, Writebehind worker: on ThCond 0x400010ADB60
(0x400010ADB60) (MsgRecord), reason 'RPC wait' for NSD I/O completion
germany:0x11181C630 waiting 6008.507415856 seconds, NSD I/O Worker: on ThCond 0x11709EB60
(0x11709EB60) (MsgRecord), reason 'RPC wait' for getData onnode 192.168.101.132
```

4. In hang conditions where RPC waits are occurring, the starting point for further investigation is the dump tscomm stanza from the **dump all** command that has been gathered from each of the nodes that have waiters.

Start with usa node. The dump tscomm stanza from usa (part of it is in Example 6-22) shows that the dest of the *pending* nsdMsgWrite is 192.168.101.142, which correlates to the germany node in the remote cluster.

Example 6-22 The dump tscomm stanza output

```
===== dump tscomm =====
```

Pending messages:

```
msg_id 581893, service 16.1, msg_type 1 'nsdMsgWrite', n_dest 1, n_pending 1
this 0x1081F2F0, n_xhold 1, ccP 0x1800276BBB0 cbFn 0x0
sent by 'Writebehind worker' (0x1086B180)
dest 192.168.101.142 status pending , err 0, reply len 0
msg_id 581895, service 16.1, msg_type 1 'nsdMsgWrite', n_dest 1, n_pending 1
this 0x400010B6BB0, n_xhold 1, ccP 0x1800276BBB0 cbFn 0x0
sent by 'Writebehind worker' (0x10842890)
dest 192.168.101.142 status pending , err 0, reply len 0
msg_id 581385, service 16.1, msg_type 1 'nsdMsgWrite', n_dest 1, n_pending 1
this 0x400025482A0, n_xhold 1, ccP 0x1800276BBB0 cbFn 0x0
sent by 'Writebehind worker' (0x10824210)
dest 192.168.101.142 status pending , err 0, reply len 0
msg_id 581386, service 16.1, msg_type 1 'nsdMsgWrite', n_dest 1, n_pending 1
this 0x400010AD3D0, n_xhold 1, ccP 0x1800276BBB0 cbFn 0x0
sent by 'Writebehind worker' (0x10875B90)
dest 192.168.101.142 status pending , err 0, reply len 0
```

Because all pending messages point to 192.168.101.142 (germany), the investigation now moves to the node germany in the remote cluster. The dump tscomm from germany is shown in Example 6-23.

Example 6-23 Pending messages from dump tscomm

```
===== dump tscomm =====
```

Pending messages:

```
msg_id 16372342, service 13.1, msg_type 1 'tmMsgRevoke', n_dest 1, n_pending 1
this 0x1170CA750, n_xhold 0, ccP 0xF1000003E0D97A18 cbFn 0x11007A998
sent by 'tmServerSideRevokeThread' (0x110299310)
dest 192.168.101.132 status pending , err 0, reply len 0
msg_id 16372343, service 13.1, msg_type 1 'tmMsgRevoke', n_dest 1, n_pending 1
this 0x11702C970, n_xhold 0, ccP 0xF1000003E0D97A18 cbFn 0x11007A998
sent by 'tmServerSideRevokeThread' (0x110299310)
dest 192.168.101.132 status pending , err 0, reply len 0
msg_id 16335429, service 0.0, msg_type 252 'getData', n_dest 1, n_pending 1
this 0x11709E9F0, n_xhold 1, ccP 0xF1000003E0D97A18 cbFn 0x0
sent by 'NSD I/O Worker' (0x11181C630) dest 192.168.101.132 status pending ,
err 0, reply len 2097152
```

The example shows that all pending messages are pointing to 192.168.101.132 (usa), which confirms that the issue appears to be in the network but does not identify root cause.

5. Check the error logs (**errpt -a** or `/var/log/messages`), **netstat** outputs, and switch logs for possible causes. Re-creating and running **iptrace** or **tcpdump**, as needed, might be necessary.

This hang condition was cleared by recycling the mmfs daemon on one of the nodes. Because it was only a client node, **mmshutdown** and **mmstartup** was performed on usa and the hang was resolved.

Example 2

Further evidence that this issue was network related can found by first identifying the socket to which 192.168.101.132 is attached. This information is in the **tscomm** stanza, in the Connections between TS nodes portion (Example 6-24). In this case, it is socket 14.

Example 6-24 Connection table from dump tscomm

Connections between nodes:

| node | destination | status | err b | sock | sent | rcvd | sseq | rseq | retry | ver | ostype |
|--------|-----------------|-----------|-------|------|-------|-------|-------|-------|-------|------|--------|
| <cln1> | 192.168.101.132 | connected | 0 - | 14 | 27362 | 27363 | 27350 | 27363 | 0 | 1202 | AIX /B |

By using the **dump tscomm** stanza again (Example 6-25), look in the Message receiver state portion for the Receiver threads that reference the socket identified previously (socket 14) and the **msg_id**. This information reveals that this socket had not received all the bytes of data it was expecting.

Example 6-25 Message receiver state from dump tscomm

Message receiver state:
 Receiver 553325
 sock 14: reading data, expecting 2097152 bytes, received 1924393, type reply, msg_id 16335429

6.3.6 Scenario 5: File system unmounting

The starting point for analyzing unmounts is the mmfs logs on the node where the unmount occurred. This node however, might not be the original source where the initial unmount was manifested. The mmfs logs from all the nodes must be gathered and investigated to determine which node initially unmounted and what caused the initial error.

Example 1

While attempting to use **dd** command to write to the file system on node slovenia, a Stale NFS file handle message is received before completion of the command. See Example 6-26.

Example 6-26 Initial file system error

```
[root@slovenia ras]# dd if=/dev/zero of=/testmigr/test.ouptut bs=256K count=10000
dd: writing `testmigr/test.ouptut': Stale NFS file handle
817+0 records in
816+0 records out
213909504 bytes (214 MB) copied, 2.99825 seconds, 71.3 MB/s
```

Perform the following steps:

1. Look first at the mmfs logs on slovenia, the cause for the **unmount** is that too many disks are unavailable. See Example 6-27.

Example 6-27 Portion of mmfs.log.latest from slovenia

```
Tue Jul 13 09:56:53.389 2010: File System testmigr unmounted by the system with
return code 217 reason code 0
Tue Jul 13 09:56:53.390 2010: Too many disks are unavailable.
Tue Jul 13 09:56:53.391 2010: File system manager takeover failed.
Tue Jul 13 09:56:53.390 2010: Too many disks are unavailable.
Tue Jul 13 09:56:53 EDT 2010: mmcommon preunmount invoked. File system:
testmigr Reason: SGPanic
```

2. Because the problem that is revealed in the mmfs logs says that too many disks are unavailable, determine the status of the NSDs in the file system, with **mmlsdisk**. Example 6-28 shows that two disks are down.

Example 6-28 mmlsdisk out from /testmigr file system

```
[root@slovenia ras]# mmlsdisk testmigr
```

| disk name | driver type | sector size | failure group | holds metadata | holds data | status | availability | storage pool |
|--------------|----------------|----------------|------------------|-------------------|---------------|--------|--------------|-----------------|
| nsdhdisk2 | nsd | 512 | 60 | yes | yes | ready | up | system |
| nsdhdisk3 | nsd | 512 | 60 | yes | yes | ready | down | system |
| nsdhdisk4 | nsd | 512 | 80 | yes | yes | ready | down | system |
| nsdhdisk5 | nsd | 512 | 80 | yes | yes | ready | up | system |

Evidence that further investigation is needed is that the node slovenia has no direct or local access to the NSDs in the /dev/testmigr file system. See Example 6-29.

Example 6-29 mmlsnd output showing slovenia as a NSD client

```
[root@slovenia ras]# mmlsnd -m -f testmigr
```

| Disk name | NSD volume ID | Device | Node name | Remarks |
|-----------|------------------|--------------|-----------|-------------|
| nsdhdisk2 | COA865844C34E4FC | /dev/hdisk1 | germany | server node |
| nsdhdisk2 | COA865844C34E4FC | /dev/hdisk7 | usa | server node |
| nsdhdisk3 | COA865844C34E4FE | /dev/hdisk2 | germany | server node |
| nsdhdisk3 | COA865844C34E4FE | /dev/hdisk8 | usa | server node |
| nsdhdisk4 | COA865844C34CCD0 | /dev/hdisk3 | germany | server node |
| nsdhdisk4 | COA865844C34CCD0 | /dev/hdisk9 | usa | server node |
| nsdhdisk5 | COA865844C34CCD2 | /dev/hdisk4 | germany | server node |
| nsdhdisk5 | COA865844C34CCD2 | /dev/hdisk10 | usa | server node |

3. Gather mmfs logs from the node. See 6.2.3, “mmfs logs” on page 265.

Collecting logs from all nodes in the cluster reveals that **unmount** started on node germany and propagated to all nodes in the cluster because of a disk problem on node germany. See Example 6-30.

Example 6-30 mmfs logs from time of unmount

```
germany: Tue Jul 13 09:56:00.819 2010: File System testmigr unmounted by the system with
return code 217 reason code 0
germany: Tue Jul 13 09:56:00.828 2010: Too many disks are unavailable.
germany: Tue Jul 13 09:56:00.836 2010: File system manager takeover failed.
germany: Tue Jul 13 09:56:00.852 2010: Too many disks are unavailable.
```

```

germany: Tue Jul 13 09:56:01 EDT 2010: mmcommon preunmount invoked. File system: testmigr
Reason: SGPanic
usa: Tue Jul 13 09:56:51 EDT 2010: mmcommon preunmount invoked. File system: testmigr
Reason: SGPanic
usa: Tue Jul 13 09:56:51.437 2010: Recovery Log I/O Failed, Unmounting file system
testmigr
usa: Tue Jul 13 09:56:51.451 2010: Too many disks are unavailable.
usa: Tue Jul 13 09:56:51.459 2010: File System testmigr unmounted by the system with
return code 218 reason code 0
usa: Tue Jul 13 09:56:51.467 2010: Too many disks are unavailable.
usa: Tue Jul 13 09:56:51.489 2010: Node 192.168.101.132 (usa) resigned as manager for
testmigr.
usa: Tue Jul 13 09:56:51.507 2010: Too many disks are unavailable.
usa: Tue Jul 13 09:56:51.516 2010: Node 192.168.101.142 (germany) appointed as manager for
testmigr.
usa: Tue Jul 13 09:56:51.582 2010: Node 192.168.101.142 (germany) resigned as manager for
testmigr.
usa: Tue Jul 13 09:56:51.599 2010: Too many disks are unavailable.
usa: Tue Jul 13 09:56:51.609 2010: Node 192.168.101.133 (slovenia) appointed as manager
for testmigr.
slovenia: Tue Jul 13 09:56:53 EDT 2010: mmcommon preunmount invoked. File system:
testmigr Reason: SGPanic
slovenia: Tue Jul 13 09:56:53.389 2010: File System testmigr unmounted by the system with
return code 217 reason code 0
slovenia: Tue Jul 13 09:56:53.390 2010: Too many disks are unavailable.
slovenia: Tue Jul 13 09:56:53.390 2010: Too many disks are unavailable.
slovenia: Tue Jul 13 09:56:53.391 2010: File system manager takeover failed.

```

4. Investigation of what the disk issue might be points to node germany. Review system logs (**errpt -a**) and any relevant SAN or disk subsystem logs for errors. In this case, **errpt -a** showed that node germany received SC_DISK_SDDAPPCM_ER and SC_DISK_ERR7 on nsdhdisk2, and nsdhdisk4.

Example 2

In this example, the unmount occurs during heavy system activity while running a backup of the file system. The backup fails with insufficient memory (ENOMEM) messages and the file system unmounts.

Perform the following steps:

1. Look at the mmfs logs on usa at roughly the reported time of the failure. Example 6-31 shows Signal 11 (SIGSEGV) and then Signal 6 (SIGABORT). Signal 11 is a memory violation, which indicates that GPFS tried get or use memory and was denied. Signal 6 is the mmfsd, generating the assert.

Example 6-31 The mmfs logs from usa showing mmfs daemon assert

```

Mon Jul 12 14:29:46.912 2010: Signal 11 at location 0x40050B10 in process 884, link reg
0xFFFFFFFFFFFFFFFF.
Mon Jul 12 14:29:46.913 2010: rax    0x0000000000000000    rbx    0x0000000000000038
Mon Jul 12 14:29:46.912 2010: rcx    0x0000001010FA35D8    rdx    0x0000000000000000
Mon Jul 12 14:29:46.913 2010: rsp    0x00007FFF12A7DE58    rbp    0x0000000000000000
Mon Jul 12 14:29:46.912 2010: rsi    0x0000000000000089    rdi    0x0000000000000000
Mon Jul 12 14:29:46.913 2010: r8     0x00007FFF12A7DE14    r9     0x0000001010FA3618
Mon Jul 12 14:29:46.912 2010: r10    0x0000001010FA3618    r11    0x0000000000000001
Mon Jul 12 14:29:46.913 2010: r12    0x0000000000000089    r13    0x000000101120DD34
Mon Jul 12 14:29:46.912 2010: r14    0x00000010112D6A08    r15    0x000000000000000C
Mon Jul 12 14:29:46.913 2010: rip    0x0000000040050B10    eflags 0x0000000000010202
Mon Jul 12 14:29:46.912 2010: cs:gsfs 0x0000000000000033    err    0x0000000000000006
Mon Jul 12 14:29:46.913 2010: trapno 0x000000000000000E

```

```

Mon Jul 12 14:29:47.511 2010: Traceback:
Mon Jul 12 14:29:47.513 2010: 0:0000000040050B10 BaseCacheObj::BaseCacheObj().40050AF0 + 20
Mon Jul 12 14:29:47.514 2010: 1:0000000040050BCD CacheObj::CacheObj(MutexName).40050BB0 + 1D
Mon Jul 12 14:29:47.513 2010: 2:00000000400A2032 BufferDesc::BufferDesc() + 12
Mon Jul 12 14:29:47.514 2010: 3:00000000400A0D64 BufferHashTab::new_entry() + 34
Mon Jul 12 14:29:47.513 2010: 4:0000000040051373 SharedHashTab::lookupAndHold(CacheObj**, ObjKey
const&, StripeGroup*, int) + 303
Mon Jul 12 14:29:47.514 2010: 5:0000000040049B43 HandleMBHashLookup(MBHashLookupParms*) + A3
Mon Jul 12 14:29:47.513 2010: 6:0000000040048BCE Mailbox::msgHandlerBody(void*) + 27E
Mon Jul 12 14:29:47.514 2010: 7:000000004003EEEE0 Thread::callBody(Thread*) + A0
Mon Jul 12 14:29:47.513 2010: 8:000000004003BAF5 Thread::callBodyWrapper(Thread*) + A5
Mon Jul 12 14:29:47.514 2010: 9:00002AB47A8DD143 start_thread + 93
Mon Jul 12 14:29:47.553 2010: Signal 6 at location 0x2AB47AEB81F1 in process 884, link reg
0xFFFFFFFFFFFFFFFF.
Mon Jul 12 14:29:47.554 2010: mmfsd is shutting down.

```

Based on *Traceback* from the assert and the *insufficient memory* error from that application, the file system unmount points the investigation toward possible issues with `maxFilesToCache` and `maxStatCache` tunable parameters.

The `mmlsconfig` output shows that the `mFTC` and `mSC` values have been changed from the default of 1000 and 4000 respectively to 2000000 for each, as shown in Example 6-32.

Example 6-32 The mFTC and mSC values from the mmlsconfig command output

```

Configuration data for cluster mantest.usa:
-----
clusterName mantest.usa
clusterId 13882457470256956975
autoload no
autoload yes
maxFilesToCache 2000000
maxStatCache 2000000
minReleaseLevel 3.4.0.0
dmapiFileHandleSize 32
cipherList AUTHONLY
openssllibname /usr/lib/libssl.a(libssl64.so.0.9.8)
adminMode central

```

2. To determine whether the values for `mFTC` and `mSC` are tuned correctly for this cluster, determine whether the `mmfs` daemon can allocate all the `mFTC` and `mSC` that was requested. Look at the “`dump fs`” stanza of `dump all` command that was previously collected or execute a `mmfsadm dump fs` command on any node and look for the `UMALLOC limits` heading. The `fileCacheLimit` and `statCacheLimit` lines are of interest. See Example 6-33.

Example 6-33 Checking for the tunable parameters fileCacheLimit and statCacheLimit

```

UMALLOC limits:
bufferDescLimit      101855 desired    262144
fileCacheLimit        777097 desired    2000000
statCacheLimit        777097 desired    2000000
diskAddrBufLimit     155419 desired    400000

```

The `fileCacheLimit` and `statCacheLimit` lines show that the `mmfs` daemon can allocate only 777097 of the 2000000 `mFTC` and `mSC` that are requested. This indication means that the `mFTC` and `mSC` have been over committed for this cluster and are in need of tuning.

See Chapter 5, “Performance and tuning” on page 229 for how to tune these values.

IBM Power Systems virtualization and GPFS

This chapter introduces the implementation of the IBM General Parallel File System (GPFS) in a virtualized environment.

The chapter provides basic concepts about the IBM Power Systems PowerVM™ virtualization, how to expand and monitor its functionality, and guidelines about how GPFS is implemented to take advantages of the Power Systems virtualization capabilities.

This chapter contains the following topics:

- ▶ 7.1, “IBM Power Systems (System p)” on page 280
- ▶ 7.2, “Shared Ethernet Adapter and Host Ethernet Adapter” on page 306

7.1 IBM Power Systems (System p)

The IBM Power Systems (System p) is a RISC-based server line. It implements a full 64-bit multicore PowerPC® architecture with advanced virtualization capabilities, and is capable of executing AIX, Linux, and applications that are based on Linux for x86.

This section introduces the main factors that can affect GPFS functionality when implemented in conjunction with IBM Power Systems.

7.1.1 Introduction

On IBM Power Systems, virtualization is managed by the IBM PowerVM, which consists of two parts:

- ▶ The *hypervisor* manages the memory, the processors, the slots, and the Host Ethernet Adapters (HEA) allocation.
- ▶ The *Virtual I/O Server (VIOS)* manages virtualized end devices: disks, tapes, and network sharing methods such Shared Ethernet Adapters (SEA) and N_Port ID Virtualization (NPIV). This end-device virtualization is possible through the virtual slots that are provided, and managed by the VIOS, which can be a SCSI, Fibre Channel, and Ethernet adapters.

When VIOS is in place, it can provide fully virtualized servers without any direct assignment of physical resources.

The hypervisor does not detect most of the end adapters that are installed in the slots, rather it only manages the slot allocation. This approach simplifies the system management and improves the stability and scalability, because no drivers have to be installed on hypervisor itself.

One of the few exceptions for this detection rule is the host channel adapters (HCAs) that are used for InfiniBand. However, the drivers for all HCAs that are supported by Power Systems are already included in hypervisor.

IBM PowerVM is illustrated in Figure 7-1 on page 281.

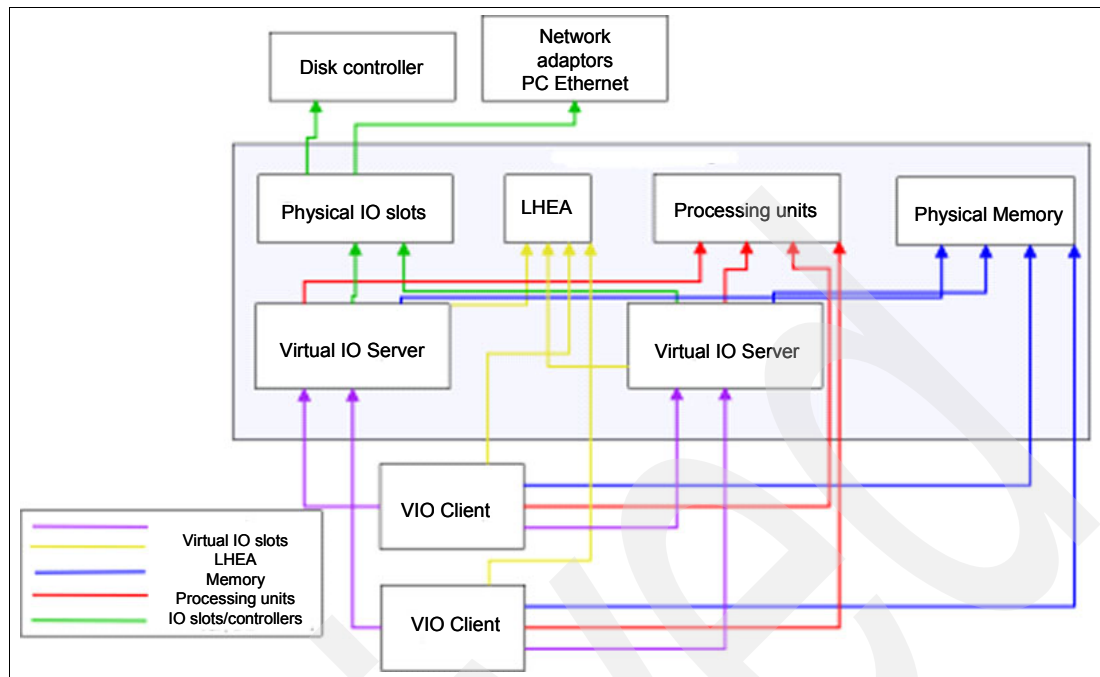


Figure 7-1 PowerVM operation flow

The following sections demonstrate how to configure IBM PowerVM to take advantages of the performance, scalability, and stability while implementing GPFS within the Power Systems virtualized environment.

The following software has been implemented:

- ▶ Virtual I/O Server 2.1.3.10-FP23
- ▶ AIX 6.1 TL05
- ▶ Red Hat Enterprise Linux 5.5

7.1.2 Virtual I/O Server (VIOS)

The VIOS is a specialized operating system that runs under a special type of logical partition (LPAR).

It provides a robust virtual storage and virtual network solution; when two or more Virtual I/O Servers are placed on the system, it also provides high availability capabilities between the them.

Note: In this section, the assumption is that the VIOS is already installed; only the topics that directly affect GPFS are covered. For a nice implementation of VIOS, a good starting point is to review the following resources:

- ▶ *IBM PowerVM Virtualization Managing and Monitoring*, SG24-7590
- ▶ *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940
- ▶ http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iph8/hwparent_systemp.htm
- ▶ http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=/iphb1/iphb1_vios_planning_vscsi_sizing.htm

Also, consider that VIOS is a dedicated partition with a specialized OS that is responsible to process I/O; any other tool installed in the VIOS requires more resources, and in some cases can disrupt its functionality.

To improve the system reliability, the systems mentioned on this chapter have been configured to use two Virtual I/O Servers for each managed system; all disk and network resources are virtualized.

In the first stage, only SEA and virtual SCSI (VSCSI) are being used; in the second stage, NPIV is defined to the GPFS volumes and HEA is configured to demonstrate its functionality.

The first stage is illustrated in Figure 7-2.

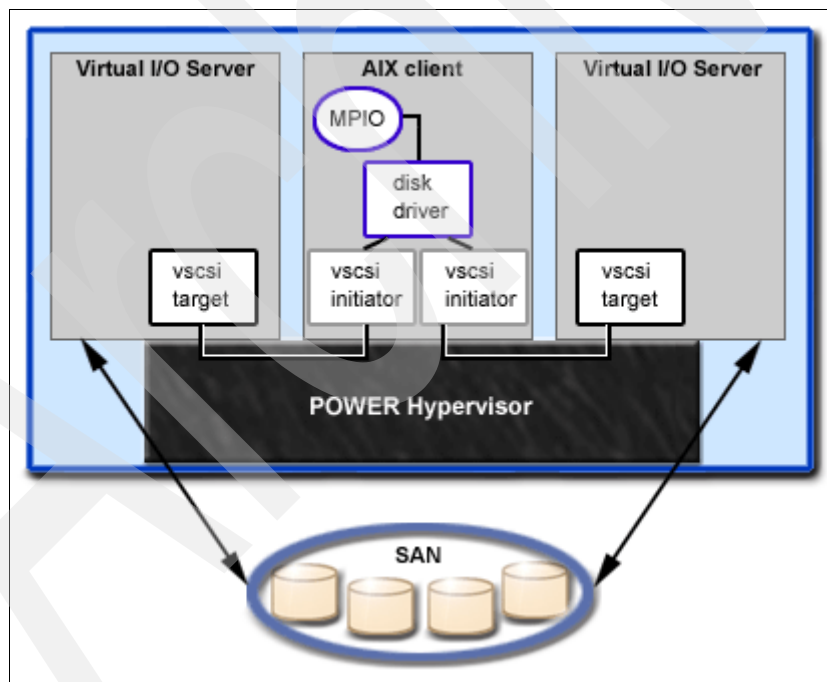


Figure 7-2 Dual VIOS setup

Because the number of processors and amount of memory for this setup are limited, the Virtual I/O Servers are set with the minimum amount of resources needed to define a working environment.

When you set up a production environment, a good approach regarding processor allocation is to define the LPAR as uncapped and to adjust the entitled (desirable) processor capacity according to the utilization that is monitored through the **lparstat** command.

7.1.3 VSCSI and NPIV

VIOS provides two methods for delivering virtualized storage to the virtual machines (LPARs):

- ▶ Virtual SCSI target adapters (VSCSI)
- ▶ Virtual Fibre Channel adapters (NPIV)

7.1.4 Virtual SCSI target adapters (VSCSI)

This implementation is the most common and has been in use since the earlier Power5 servers. With VSCSI, the VIOS becomes a storage provider. All disks are assigned to the VIOS, which provides the allocation to its clients through its own SCSI target initiator protocol.

Its functionality, when more than one VIOS is defined in the environment, is illustrated in Figure 7-3.

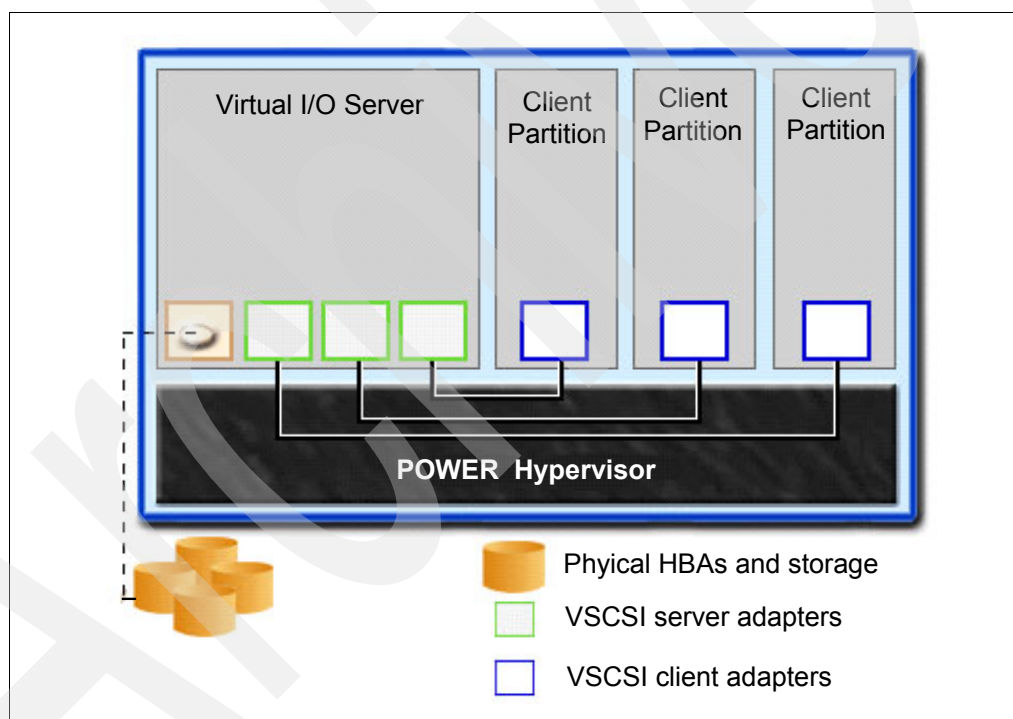


Figure 7-3 VSCSI connection flow

VSCSI allows the construction of high-density environments by concentrating the disk management to its own disk management system.

To manage the device allocation to the VSCSI clients, the VIOS creates a virtual SCSI target adapter in its device tree, this device is called vhost as shown in Example 7-1 on page 284.

Example 7-1 VSCSI server adapter

```
$ lsdev -dev vhost2
name          status      description
vhost2        Available  Virtual SCSI Server Adapter
```

On the client partition, a virtual SCSI initiator device is created, the device is identified as an `ibmvscsi` adapter into Linux, as shown in Example 7-2; and as a virtual SCSI client adapter on AIX, as shown in Example 7-3.

On Linux, the device identification can be done in several ways. The most common ways are checking the `scsi_host` class for the host adapters and checking the interrupts table. Both methods are demonstrated on Example 7-2.

Example 7-2 VSCSI Client adapter on Linux

```
[root@slovenia ~]# cat /sys/class/scsi_host/host{0,1}/proc_name
ibmvscsi
ibmvscsi
[root@slovenia ~]# grep ibmvscsi /proc/interrupts
18:    408803      2485    1515    1527  XICS      Level    ibmvscsi
21:      355       336     331     280  XICS      Level    ibmvscsi
```

On AIX, the device identification is easier to accomplish, because the VSCSI devices appear in the standard device tree (`lsdev`). A more objective way to identify the VSCSI adapters is to list only the VSCSI branch in the tree, as shown in Example 7-3.

Example 7-3 VSCSI client adapter on AIX

```
usa:/# lsparent -C -k vscsi
vscsi0 Available  Virtual SCSI Client Adapter
vscsi1 Available  Virtual SCSI Client Adapter
usa:/#
```

The device creation itself is done through the system Hardware Management Console (HMC) or the Integrated Virtualization Manager (IVM) during LPAR creation time, by creating and assigning the virtual slots to the LPARs.

Note: The steps to create the VIOS slots are in the management console documentation. A good place to start is by reviewing the virtual I/O planning documentation. The VIOS documentation is on the IBM Systems Hardware information website:

<http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphat/iphblconfigure1parp6.htm>

GPFS can load the storage subsystem, specially if the node is an NSD server. Consider the following information:

- ▶ The number of processing units that are used to process I/O requests might be twice as many as are used on Direct attached standard disks.
- ▶ If the system has multiple Virtual I/O Servers, a single LUN cannot be accessed by multiple adapters at the same time.
- ▶ The virtual SCSI multipath implementation does not provide load balance, only failover capabilities.

- ▶ If logical volumes are being used as back-end devices, they cannot be shared across multiple Virtual I/O Servers.
- ▶ Multiple LPARs may share the same Fibre Channel adapter to access the Storage Area Network (SAN).

VIOS configuration

As previously explained, VSCSI implementation works on a client-server model. All physical drive operations are managed by VIOS, and the volumes are exported using VSCSI protocol.

Because GPFS works by accessing the disk in parallel through multiple servers, it is required to properly set up the access control attributes, defined in the SAN disks that will be used by GPFS. The parameter consists of disabling two resources available in the SAN disks, the `reserve_policy` and the `PR_key_value`. This technique can be accomplished with the **chdev** command, as shown in Example 7-4.

Example 7-4 Persistent Reserve (PR) setup

```
$ chdev -dev hdisk15 -attr reserve_policy=no_reserve PR_key_value=none
hdisk15 changed
$ lsdev -dev hdisk15 -attr | grep -E 'reserve_policy|PR_key_value'
PR_key_value      none
Reserve Key                               True
reserve_policy    no_reserve
```

The main purpose of this parametrization is to avoid Persistent Reserve (PR) problems in the system.

Despite GPFS being able to handle PR SCSI commands since version 3.2, the VSCSI does not implement these instructions. Therefore this must be disabled in the GPFS configuration and on the SAN disks that are assigned to the VIOS.

Note: For more information about Persistent Reserve, see the following resources:

- ▶ http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.gpfs33.diagnostics.doc/bl1pdg_persresrv.html
- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/p7hb1/iphb1setnoreserve.htm>
- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/p7hb1/iphb1configurereserve.htm>

After the parametrization of the disks are defined, it must be allocated to the Virtual I/O Client (VIOC). The allocation into VIOS is done through the **mkvdev** command. To list the allocation map of a specific vhost adapter, the **lsmapi** command can be used.

To facilitate the identification of the adapter after it has been mapped to the VIOC, make a list of the allocation map before, as shown in Example 7-5 on page 286, so that you can compare it with the results after. Use the **-vadapter** parameter to limit the output to only a specific adapter; to list the allocation map for all the VSCSI adapters, use the **-a11** parameter.

Example 7-5 *lsmap before map disk*

```

$ lsmap -vadapter vhost3
SVSA                      Physloc                      Client Partition ID
-----
vhost3                    U9117.MMA.101F170-V1-C12    0x00000004

VTD                        lnx_rootvg_vtd
Status                     Available
LUN                       0x8200000000000000
Backing device             lnx_rootvg
Physloc

```

As shown in Example 7-5, the vhost3 has only one adapter allocated to it and this adapter is identified by the *Backing device* attribute.

Important: The device lnx_rootvg shown in Example 7-5 is a logical volume created on the VIOS to install the operating system itself.

Logical volumes must be used as backing devices for the GPFS disks, because the disks cannot be shared outside the managed system (physical server) itself.

To allocate the SAN disk to the VIOC, the **mkvdev** command is used as shown Example 7-6.

The following mandatory parameters are used to create the map:

- vdev** Identifies the block device to be allocated to the VIOC.
- vadapter** Determines the vhost adapter that will receive the disk.
- dev** Determines the Virtual Target Device (VTD) that will be assigned to this map.

Example 7-6 *Allocating disk with mkvdev*

```

$ mkvdev -vdev hdisk11 -vadapter vhost3 -dev lnx_hdisk1
lnx_hdisk1 Available

```

Tip: Physical Volumes (PV) ID that is assigned on VIOS is transparently passed to the VIOC, so that disk identification across multiple VIOS and multiple volumes running on an AIX VIOC is easier.

This technique can be accomplished by issuing the **chdev** command, for example as follows:

```

$ chdev -dev hdisk11 -attr pv=yes -perm
hdisk11 changed
$

```

Linux is not able to read AIX disk labels nor AIX PVIDs, however the **fdisk** and **cfdisk** commands are able to identify whether the disk has an AIX label on it.

After the allocation is completed, it can be verified with the **lsmap** command. as shown in Example 7-7 on page 287.

Example 7-7 The vhost with LUN mapped

```
$ lsmmap -vadapter vhost3
```

| SVSA | Physloc | Client Partition ID |
|--------|--------------------------|---------------------|
| ----- | ----- | ----- |
| vhost3 | U9117.MMA.101F170-V1-C12 | 0x00000004 |

| | |
|-----------------------|--|
| VTD | lnx_hdisk1 |
| Status | Available |
| LUN | 0x8300000000000000 |
| Backing device | hdisk11 |
| Physloc | U789D.001.DQDYKYW-P1-C1-T1-W202200A0B811A662-L5000000000000 |

| | |
|----------------|--------------------|
| VTD | lnx_rootvg_vtd |
| Status | Available |
| LUN | 0x8200000000000000 |
| Backing device | lnx_rootvg |
| Physloc | |

| | |
|----------------|--------------------|
| VTD | vhost3_cdrom |
| Status | Available |
| LUN | 0x8100000000000000 |
| Backing device | |
| Physloc | |

Important: The same operation must be replicated to all Virtual I/O Servers that are installed on the system, and the LUN number must match on all disks. If the LUN number does not match, the multipath driver might not work correctly, which can lead to data corruption problems.

When the device is not needed anymore, or has to be replaced, the allocation can be removed with the **rmvdev** command, as shown in Example 7-8. The **-vtd** parameter is used to remove the allocation map. It determines which virtual target device will be removed from the allocation map.

Example 7-8 rmvdev

```
$ rmvdev -vtd lnx_hdisk1
```

Note: For more information about the commands that are used to manage the VIOS mapping, see the following VIOS and command information:

- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/p7hb1/iphb1kickoff.htm>
- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphcg/mkvdev.htm>
- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphcg/rmvdev.htm>
- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphcg/lsmmap.htm>

Virtual I/O Client setup and AIX

As previously mentioned, under AIX the VSCSI adapter is identified by the name *vscsi* followed by the adapter number. Example 7-9 on page 288 shows how to verify this information.

When the server has multiple VSCSI adapters provided by multiple VSCSI servers, balancing the I/O load between the VIOS becomes important. During maintenance of the VIOS, determining which VSCSI adapter will be affected also becomes important.

This identification is accomplished by a trace that is done by referencing the slot numbers in the VIOS and VIOC on the system console (HMC).

In AIX, the slot allocation can be determined by issuing the **lsslot** command. The slot information is also shown when the **lscfg** command is issued. Both methods are demonstrated in Example 7-9; the slot names are highlighted.

Example 7-9 Identifying VSCSI slots into AIX

```

usa:/#lsslot -c slot | grep vscsi
U9117.MMA.101F170-V3-C2 Virtual I/O Slot vscsi0
U9117.MMA.101F170-V3-C5 Virtual I/O Slot vscsi1
usa:/#lscfg -l vscsi*
  vscsi1          U9117.MMA.101F170-V3-C5-T1 Virtual SCSI Client Adapter
  vscsi0          U9117.MMA.101F170-V3-C2-T1 Virtual SCSI Client Adapter
usa:/#

```

With the slots identified in the AIX client, the slot information must be collected in the VIOS, which can be accomplished by the **lsmap** command as shown in Example 7-10.

Example 7-10 Identifying VSCSI slots into VIOS

```

$ lsmap -vadapter vhost2 | grep vhost2
vhost2          U9117.MMA.101F170-V1-C11          0x00000003
$

```

Note: On System p, the virtual slots follow a naming convention:

<type>.<model>.<serial>.<lpar id>.<slot number>.<port>

This naming convention has the following definitions:

- ▶ *type:* Machine type
- ▶ *model:* Model of the server, attached to the machine type
- ▶ *serial:* Serial number of the server
- ▶ *lpar id:* ID of the LPAR itself
- ▶ *slot number:* Number assigned to the slot when the LPAR was created
- ▶ *port:* Port number of the specific device (Each virtual slots has only one port.)

With the collected information, the connection between the VIOS and VIOC can be determined by using the **lshwres** command, as shown in Example 7-11.

Example 7-11 VSCSI slot allocation on HMC using lshwres

```

hscroot@hmc4:~> lshwres -r virtualio -m 570_1-9117-MMA-101F170 --subtype scsi
--filter lpar_ids=3 -F "lpar_name lpar_id slot_num remote_lpar_id remote_lpar_name
remote_slot_num"
570_1_AIX 3 5 2 570_1_VIO_2 11
570_1_AIX 3 2 1 570_1_VIO_1 11
hscroot@hmc4:~>

```

In Example 7-11 on page 288, the VSCSI slots that are created on usa (570_1_AIX) are distributed in the the following way:

- ▶ Virtual slot 5 on 570_1_AIX is connected to virtual slot 11 on 570_1_VIO_2.
- ▶ Virtual slot 2 on 570_1_AIX is connected to virtual slot 11 on 570_1_VIO_1.

When configuring multipath by using VSCSI adapters, it is important to determine the configuration of each adapter properly into the VIOC, so that if a failure of a VIOS occurs, the service is not disrupted.

The following parameters have to be defined for each VSCSI adapter that will be used in the multipath configuration:

▶ **vscsi_path_to**

Values other than zero, define the periodicity, in seconds, of the heartbeats that are sent to the VIOS. If the VIOS does not send the heartbeat within the designated time interval, the controller understands it as failure and the I/O requests are re-transmitted.

By default, this parameter is disabled (defined as 0), and can be enabled only when multipath I/O is used in the system.

On the servers within our environment, this value is defined as 30.

▶ **vscsi_err_recov**

With the **fc_error_recov** parameter on the physical Fibre Channel adapters, when set to **fast_fail**, sends a **FAST_FAIL** event to the VIOS when the device is identified as failed, and resends all I/O requests to alternate paths.

The default configuration is **delayed_fail** and must be changed only when multipath I/O is used in the system.

On the servers within our environment, the value of the **vscsi_err_recov** is defined as **fast_fail**.

The configuration of the VSCSI adapter is performed by the **chdev** command, as shown in Example 7-12.

Example 7-12 Configuring fast_fail on the VSCSI adapter on AIX

```
usa:/# chdev -l vscsi0 -a vscsi_err_recov=fast_fail -a vscsi_path_to=30
vscsi0 changed
```

To determine whether the configuration is in effect, the use the **lsattr** command, as shown in Example 7-13.

Example 7-13 Determining VSCSI adapter configuration on AIX

```
usa:/# lsattr -El vscsi0
vscsi_err_recov fast_fail N/A True
vscsi_path_to 30
```

Notes: If the adapter is already being used, the parameters cannot be changed online.

For the servers within our environment, the configuration is changed on the database only and the servers are rebooted.

This step can be done as shown in the following example:

```
usa:/# chdev -l vscsi0 -a vscsi_err_recov=fast_fail -a vscsi_path_to=30 -P
vscsi0 changed
```

For more information about this parameterization, see the following resources:

- ▶ http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=iphb1/iphb1_vios_disks.htm
- ▶ <http://www.ibm.com/support/docview.wss?uid=isg1IZ28554>
- ▶ <http://www.ibm.com/support/docview.wss?uid=isg1IZ67858>

The disks are identified as standard hdisks, as shown in Example 7-14.

Example 7-14 Listing VSCSI disks on AIX

```
usa:/# lsdev -Cc disk
hdisk0 Available Virtual SCSI Disk Drive
hdisk1 Available Virtual SCSI Disk Drive
hdisk2 Available Virtual SCSI Disk Drive
hdisk3 Available Virtual SCSI Disk Drive
hdisk4 Available Virtual SCSI Disk Drive
hdisk5 Available Virtual SCSI Disk Drive
usa:/#
```

To determine whether the disks attributes are from both VSCSI adapters, use the **lspath** command, as shown Example 7-15.

Example 7-15 List virtual disk paths with lspath

```
usa:/# lspath -Hl hdisk2
status name parent

Enabled hdisk2 vscsi0
Enabled hdisk2 vscsi1
usa:/#
```

To recognize new volumes on the server, use the **cfgmgr** command after the allocation under the VIOS is completed.

When the disk is configured in AIX, the disk includes the standard configuration, as shown in Example 7-16.

Example 7-16 VSCSI disk with default configuration under AIX

```
usa:/# lsattr -El hdisk0 -a hcheck_cmd -a hcheck_interval -a hcheck_mode
hcheck_interval 0 Health Check Interval True
hcheck_mode nonactive Health Check Mode True
usa:/#
```

As a best practice, all disks exported through VIOS should have the `hcheck_interval` and `hcheck_mode` parameters customized to the following settings:

- ▶ Set `hcheck_interval` to 60 seconds
- ▶ Set `hcheck_mode` as `nonactive` (which it is already set to)

This configuration can be performed using the `chdev` command, as shown Example 7-17.

Example 7-17 Configuring `hcheck` parameters in VSCSI disk under AIX*

```
usa:/# chdev -l hdisk0 -a hcheck_mode=nonactive -a hcheck_interval=60 -P
hdisk0 changed
usa:/# lsattr -El hdisk0 -a hcheck_interval -a hcheck_mode
hcheck_interval 60          Health Check Interval True
hcheck_mode      nonactive Health Check Mode      True
```

Note: If the device is being used, as in the previous example, the server has to be rebooted so that the configuration can take effect.

The AIX MPIO implementation for VIOS disks does not implement a load balance algorithm, such as `round_robin`, it implements only fail over.

By default, when multiple VSCSI are in place, all have the same weight (priority), because the only actively transferring data is the VSCSI with the lower virtual slot number.

Using the server USA as an example, by default, the only VSCSI adapter that is transferring I/O is the `vscsi0`, which is confirmed in Example 7-9 on page 288. To optimize the throughput, various priorities can be set on the disks to spread the I/O load among the VIOS, as shown in Figure 7-4 on page 292.

Figure 7-4 on page 292 shows the following information:

- ▶ The `hdisk1` and `hdisk3` use `vio570-1` as the active path to reach the storage; `vio570-11` is kept as standby.
- ▶ The `hdisk2` and `hdisk4` use `vio570-11` as the active path to reach the storage, `vio570-1` is kept as standby.

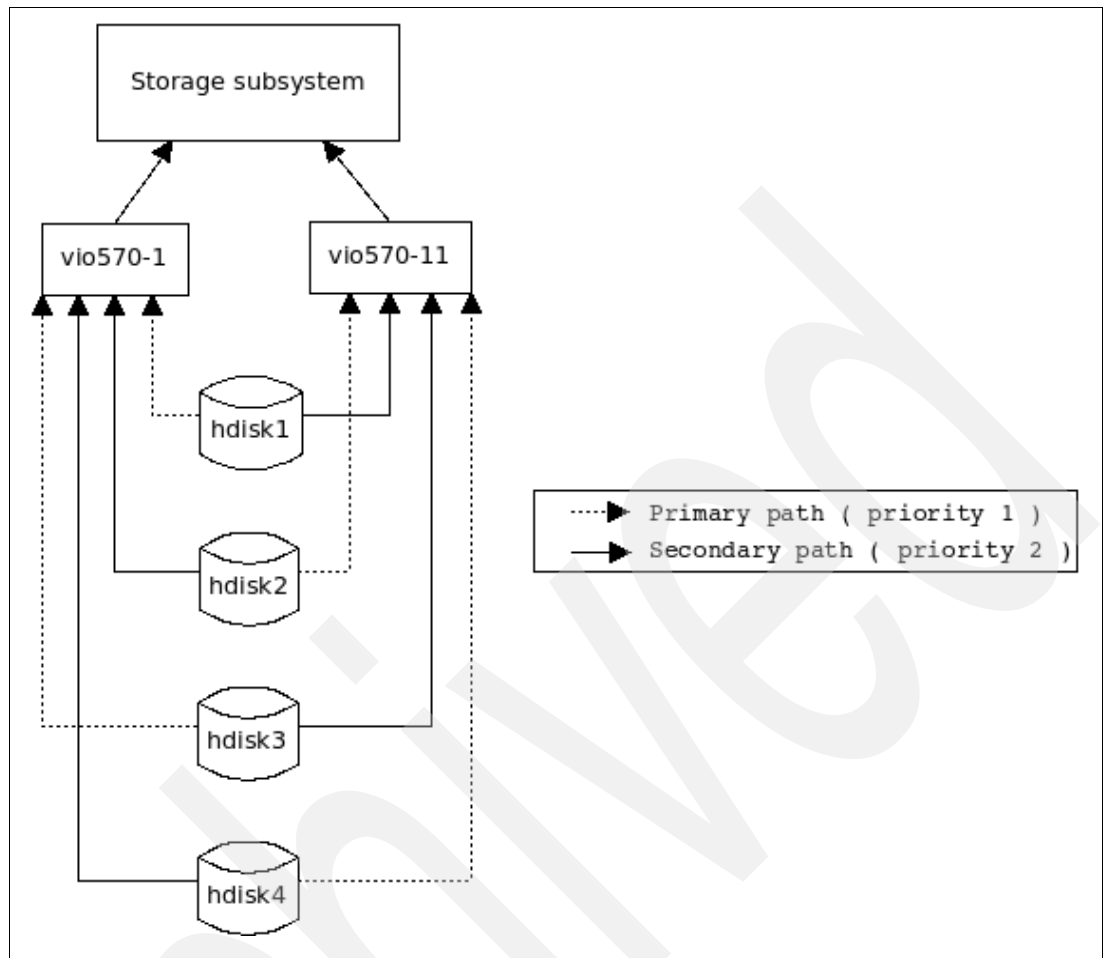


Figure 7-4 Split disk load across multiple VIOS

The path priority must be determined on each path and on each disk. The command **lspath** can assist with accomplishing this task, as shown in Example 7-18.

Example 7-18 Determine path priority on VSCSI hdisk using *lspath*

```
usa:/# lspath -l hdisk5 -p vscsi0 -D -a priority
hdisk5 vscsi0 priority 1 Priority True
usa:/#
```

Tip: If listing the priority of all paths on a determined hdisk is necessary, scripting can make the task easier. The following example was used during the writing of this book:

```
#!/usr/bin/awk -f
BEGIN {
  if ( ARGV == 2 )
  {
    while (("lspath -l " ARGV[1] | getline) > 0)
    {
      "lspath -l "$2" -p "$3" -D -a priority" | getline V1;
      print $2,$3,V1;
    }
  }
}
```

The script gives the following output:

```
usa:/bin# bin/disk.awk hdisk5
hdisk5 vscsi0 priority 1 Priority True
hdisk5 vscsi1 priority 1 Priority True
usa:/bin#
```

To change the priority of the paths, the **chpath** command is used, as shown in Example 7-19

Example 7-19 Change path priority on a VSCSI disk under AIX

```
usa:/# chpath -l hdisk5 -p vscsi0 -a priority=2
path Changed
usa:/#
```

The AIX virtual disk driver (VDASD) also allows parametrization of the maximum transfer rate at disk level. This customization can be set through the **chdev** command, and the default rate is set to 0x40000.

To view the current configuration, use the **lsattr** command, as shown in Example 7-20.

Example 7-20 Show the transfer rate of a VSCSI disk under AIX

```
usa:/# lsattr -El hdisk5 -a max_transfer
max_transfer 0x40000 Maximum TRANSFER Size True
usa:/#
```

Determine the range of values that can be used; see Example 7-21.

Example 7-21 Listing the range of speed that can be used on VSCSI disk under AIX

```
usa:/# lsattr -l hdisk5 -Ra max_transfer
0x20000
0x40000
0x80000
0x100000
0x200000
0x400000
0x800000
0x1000000
usa:/#
```

To set this customization, the disk must be placed in a *Defined* state, previous to the execution, as shown in Example 7-22.

Attention: Increasing the transfer rate also increases the VIOS processor utilization. This type of customization can overload the disk subsystem and the storage subsystem. Changes to the max_transfer rate parameter should first be discussed with your storage provider or the IBM technical personnel.

Example 7-22 Change the speed of a VSCSI disk under AIX

```
usa:/#rmdev -l hdisk5
hdisk5 Defined
usa:/#chdev -l hdisk5 -a max_transfer=0x80000
hdisk5 Changed
usa:/#mkdev -l hdisk5
hdisk5 Available
usa:/#
```

Note: For more information, see the following resources:

- ▶ *IBM Midrange System Storage Implementation and Best Practices Guide*, SG24-6363
- ▶ http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=/iphb1/iphb1_vios_mpio.htm
- ▶ <http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp?topic=/com.ibm.aix.baseadm/doc/baseadmdita/devpathctrlmodatts.htm>
- ▶ <http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp?topic=/com.ibm.aix.cmds/doc/aixcmds1/chpath.htm>

Virtual I/O client setup on Linux

In the same way as AIX, the VSCSI adapter under Linux is organized by slot numbering. However, instead of organizing the VSCSI devices into a device tree, the VSCSI adapters are organized under the scsi_host class of sysfs file system.

The device identification can be done with the **systool** command, shown in Example 7-23.

Example 7-23 VSCSI adapter under Linux

```
[root@slovenia ~]# systool -v -c scsi_host host0
Class = "scsi_host"

Class Device = "host0"
Class Device path = "/sys/class/scsi_host/host0"
cmd_per_lun      = "16"
config           =
host_busy        = "0"
mad_version      = "1"
os_type          = "3"
partition_name   = "570_1_VIO_1"
partition_number = "1"
proc_name        = "ibmvscsi"
scan             = <store method only>
sg_tablesize     = "255"
srp_version      = "16.a"
state            = "running"
```



```

uevent                = <store method only>
unchecked_isa_dma     = "0"
unique_id             = "0"
vhost_loc             = "U9117.MMA.101F170-V4-C2-T1"
vhost_name            = "vhost3"

```

```

Device = "host0"
Device path = "/sys/devices/vio/30000002/host0"
uevent                = <store method only>

```

Tip: The VSCSI driver that is available on Linux provides more detailed information about the VIOS itself, resulting in easier administration on large virtualized environments.

Module configuration

The Linux VSCSI driver includes a set of configurations that are optimal for non-disk shared environments.

On shared disk environments, the `client_reserve` parameter must be set to **0** (zero), under the module configuration in the `/etc/modprobe.conf` file, as shown in Example 7-24.

Example 7-24 The `ibmvscsi` module configuration for multiple VIOS environment

```

[root@slovenia ~]# grep "^options ibmvscsic" /etc/modprobe.conf
options ibmvscsic client_reserve=0
[root@slovenia ~]#

```

After the module is reloaded with the `client_reserve` parameter set to a multiple VIOS environment, the configuration can be verified, as shown in Example 7-25.

Example 7-25 List `ibmvscsi` configuration on Linux using `systool`

```

[root@slovenia ~]# systool -v -m ibmvscsic
Module = "ibmvscsic"

Attributes:
  refcnt          = "11"
  srcversion      = "4976AFA11E415F206B2E25A"
  version        = "1.5.9"

Parameters:
  client_reserve  = "0"
  fast_fail      = "1"
  init_timeout   = "300"
  max_channel    = "3"
  max_id         = "64"
  max_requests   = "100"

Sections:
  .bss           = "0xd0000000001dd9f0"
  .data.rel.local = "0xd0000000001d8f58"
  .data.rel      = "0xd0000000001d8cb0"
  .data          = "0xd0000000001d8a60"
  .exit.text     = "0xd0000000001d3b14"
  .gnu.linkonce.this_module = "0xd0000000001d9100"
  .module_sig    = "0xd0000000001dd9f0"

```

```

.opd                = "0xd000000001dd6a8"
.rodata.str1.8      = "0xd000000001d46e0"
.rodata             = "0xd000000001d4678"
.strtab            = "0xd000000001d7a48"
.stubs             = "0xd000000001d3b98"
.symtab            = "0xd000000001d6398"
.text              = "0xd000000001d0000"
.toc               = "0xd000000001dd9d8"
.toc1              = "0xd000000001dd380"
__bug_table        = "0xd000000001d5533"
__param            = "0xd000000001d8bc0"
__versions         = "0xd000000001d55d8"
_init.text         = "0xd000000001d3b48"

```

```
[root@slovenia ~]#
```

Notes: If VSCSI is being used on boot disks, as with the setup for the systems, the Linux initial RAM disk (initrd) must be re-created, and then the system must be rebooted with the the new initrd. To re-create the initrd, use the `mkinitrd` command, as follows:

```
[root@slovenia ~]# mkinitrd -f /boot/initrd-$(uname -r).img $(uname -r)
```

On larger setups, the module parameters `max_id` and `max_requests` might also have to be changed.

Evaluate this carefully, mainly because changing this parameters can cause congestion with the Remote Direct Memory Access (RDMA) traffic between the LPARs, and increase processor utilization with the VIOS.

Consider alternative setups, such as NPIV, which tend to consume fewer system resources.

After the VSCSI adapter is configured, verify the existence of the disks existence by using `lscfg` command, as shown in Example 7-26.

Example 7-26 List VSCSI disks under Linux

```

[root@slovenia ~]# lscfg | grep sd
+ sdh          U9117.MMA.101F170-V4-C5-T1-B0-T4-L0
+ sdg          U9117.MMA.101F170-V4-C5-T1-B0-T3-L0
+ sdf          U9117.MMA.101F170-V4-C5-T1-B0-T2-L0
+ sde          U9117.MMA.101F170-V4-C5-T1-B0-T1-L0
+ sdd          U9117.MMA.101F170-V4-C2-T1-B0-T5-L0
+ sdc          U9117.MMA.101F170-V4-C2-T1-B0-T4-L0
+ sdb          U9117.MMA.101F170-V4-C2-T1-B0-T3-L0
+ sda          U9117.MMA.101F170-V4-C2-T1-B0-T2-L0
[root@slovenia ~]#

```

On Linux, the default disk configuration already defines reasonable parameters for GPFS operations. The disk parametrization can be verified, as shown Example 7-27 on page 297.

Example 7-27 VSCSI disk configuration under Linux

```
[root@slovenia slaves]# systool -c block sda -v
Class = "block"

Class Device = "sda"
Class Device path = "/sys/block/sda"
  dev          = "8:0"
  range        = "16"
  removable    = "0"
  size         = "20971520"
  stat         = "      5370      16246      306856      32291      5483      13637
152956      175917      0      35438      208209"
  uevent       = "<store method only>"

Device = "0:0:2:0"
Device path = "/sys/devices/vio/30000002/host0/target0:0:2/0:0:2:0"
  delete       = "<store method only>"
  device_blocked = "0"
  dh_state     = "detached"
  iocounterbits = "32"
  iodone_cnt   = "0x2aa9"
  ioerr_cnt    = "0x3"
  iorequest_cnt = "0x2aa9"
  model        = "VDASD"
  queue_depth  = "16"
  queue_type   = "none"
  rescan       = "<store method only>"
  rev          = "0001"
  scsi_level   = "4"
  state        = "running"
  timeout      = "60"
  type         = "0"
  uevent       = "<store method only>"
  vendor       = "AIX"
```

Note: On Linux, you may determine the disk existence in several ways:

- ▶ Determining the disk existence on `/proc/partitions`.
- ▶ Using the `systool` command to list all devices under the `block` class or `scsi_disk` class.
- ▶ Determining which disks were found by the `fdisk` command.

As multipath software, Linux uses the standard `dm-mpio` kernel driver, which provides fail-over and load-balancing (also called `multibus`) capabilities. The configuration used on the servers for this project is shown in Example 7-28.

Example 7-28 the dm-mpio configuration for VSCSI adapters on Linux (/etc/multipath.conf)

```
[root@slovenia ~]# egrep -v "^#|^$" /etc/multipath.conf
defaults {
  polling_interval 30
  failback         immediate
  no_path_retry    5
  rr_min_io        100
  path_checker     tur
  user_friendly_names yes
```

```

}
devices {
    device {
        vendor            "IBM"
        product           "1820N00"
        path_grouping_policy group_by_prio
        prio_callout       "/sbin/mpath_prio_alua /dev/%n"
        getuid_callout     "/sbin/scsi_id -g -u -s /block/%n"
        path_checker       tur
        polling_interval   30
        path_selector      "round-robin 0"
        rr_min_io          100
        rr_weight           priorities
        failback           immediate
        no_path_retry      queue
    }
}
[root@slovenia ~]#

```

Note: Any disk tuning or customization on Linux must be defined for each path (disk). The mpath disk is only a kernel pseudo-device used by dm-mpio to provide the multipath capabilities.

Disk maintenance

The VSCSI for Linux driver allows dynamic operations on the virtualized disks. Basic operations are as follows:

- Adding a new disk:
 - a. Scan all SCSI host adapters where the disk is connected (Example 7-29).

Example 7-29 Scan scsi host adapter on Linux to add a new disk

```

[root@slovenia ~]# echo "- - -" > /sys/class/scsi_host/host0/scan
[root@slovenia ~]# echo "- - -" > /sys/class/scsi_host/host1/scan

```

- b. Update the dm-mpio table with the new disk (Example 7-30).

Example 7-30 Update dm-mpio device table to add a new disk

```

[root@slovenia ~]# multipath -r

```

- Removing a disk:
 - a. Determine which paths belongs to the disk (Example 7-31). The example shows that the disks sdi and sdj are paths to the mpath5 mpath device.

Example 7-31 List which paths are part of a disk into dm-mpio, Linux

```

[root@slovenia slaves]# multipath -ll mpath5
mpath5 (3600a0b80001146320000f1a54c18bd40) dm-6 AIX,VDASD
[size=25G][features=1 queue_if_no_path][hwhandler=0][rw]
\_ round-robin 0 [prio=2][enabled]
\_ 0:0:6:0 sdi 8:128 [active][ready]
\_ 1:0:5:0 sdj 8:144 [active][ready]
[root@slovenia slaves]#

```

- b. Remove the disk from dm-mpio table as shown in Example 7-32 on page 299.

Example 7-32 Remove a disk from dm-mpio table on Linux

```
[root@slovenia slaves]# multipath -f mpath5
```

- c. Flush all buffers from the system, and commit all pending I/O (Example 7-33).

Example 7-33 Flush disk buffers with blockdev

```
[root@slovenia ~]# blockdev --flushbufs /dev/sdi
[root@slovenia ~]# blockdev --flushbufs /dev/sdj
```

- d. Remove the paths from the system (Example 7-34).

Example 7-34 Remove paths (disk) from the system on Linux

```
[root@slovenia slaves]# echo 1 > /sys/block/sdi/device/delete
[root@slovenia slaves]# echo 1 > /sys/block/sdj/device/delete
```

Queue depth considerations

The queue depth defines the number of commands that the device keeps in its queue before sending the number of commands it to the device to process.

In VSCSI, the queue length holds 512 command elements. Two are reserved for the adapter itself and three are reserved for each VSCSI LUN for error recovery. The remainder are used for I/O request processing.

To obtain the best performance possible, adjust the queue depth of the virtualized disks to match the physical disks.

Sometimes, this method of tuning can reduce the number of disks that can be assigned to the VSCSI adapter in the VIOS, which leads to the creation of more VSCSI adapters per VIOS.

When changing the queue depth, use the following formula:

$$(512 - 2) / (Q + 3) = \text{Maximum number of disks mapped to a VIO}$$

The formula uses the following values:

- 512: Number of command elements implemented by VSCSI
- 2: Reserved for adapter
- Q: Desired queue depth
- 3: Reserved for each disk error manipulation operation

Important: Adding more VSCSI adapters into VIOS leads to the following situations:

- ▶ Increase in memory utilization by the hypervisor to build the VSCSI SRP
- ▶ Increase in processor utilization by the VIOS to processes VSCSI commands

On Linux, the queue depth must be defined during the execution time and the definition is not persistent; if the server is rebooted, it must be redefined by using the sysfs interface, as shown in Example 7-35.

Example 7-35 Define Queue Depth into a Linux Virtual disk

```
[root@slovenia bin]# echo 20 > /sys/block/sde/device/queue_depth
[root@slovenia bin]#
```

To determine the actual queue depth of a disk, use the **systool** command (Example 7-36).

Example 7-36 Determine the queue depth of a Virtual disk under Linux

```
[root@slovenia bin]# systool -c block sde -A queue_depth
Class = "block"
```

```
Class Device = "sde"
```

```
Device = "1:0:1:0"
queue_depth = "20"
```

```
[root@slovenia bin]#
```

On AIX, the queue depth can be defined by using the **chdev** command. This operation cannot be completed if the device is being used. An alternative way is to set the definition on system configuration (Object Data Manager, ODM) followed by rebooting the server, as shown in Example 7-37.

Example 7-37 Queue depth configuration of a virtual disk under AIX

```
usa:/#lsattr -El hdisk2 -a queue_depth
queue_depth 3 Queue DEPTH True
usa:/#chdev -l hdisk2 -a queue_depth=20 -P
hdisk2 changed
usa:/#lsattr -El hdisk2 -a queue_depth
queue_depth 20 Queue DEPTH True
usa:/# shutdown -Fr
```

7.1.5 N_Port ID Virtualization (NPIV)

NPIV allows the VIOS to dynamically create multiple N_PORT IDs and share it with the VIOC.

These N_PORT IDs are shared through the virtual Fibre Channel adapters and provides to the VIOC a worldwide port name (WWPN), which allows a direct connection to the storage area network (SAN).

The virtual Fibre Channel adapters support connection of the same set of devices that usually are connected to the physical Fibre Channel adapters, such as the following devices:

- ▶ SAN storage devices
- ▶ SAN tape devices

Unlike VSCSI and NPIV, VIOS does not work on a client-server model. Instead, the VIOS manages only the connection between the virtual Fibre Channel adapter and the physical adapter; the connection flow is managed by the hypervisor itself, resulting in a lighter implementation, requiring fewer processor resources to function when compared to VSCSI.

The NPIV functionality can be illustrated by Figure 7-5.

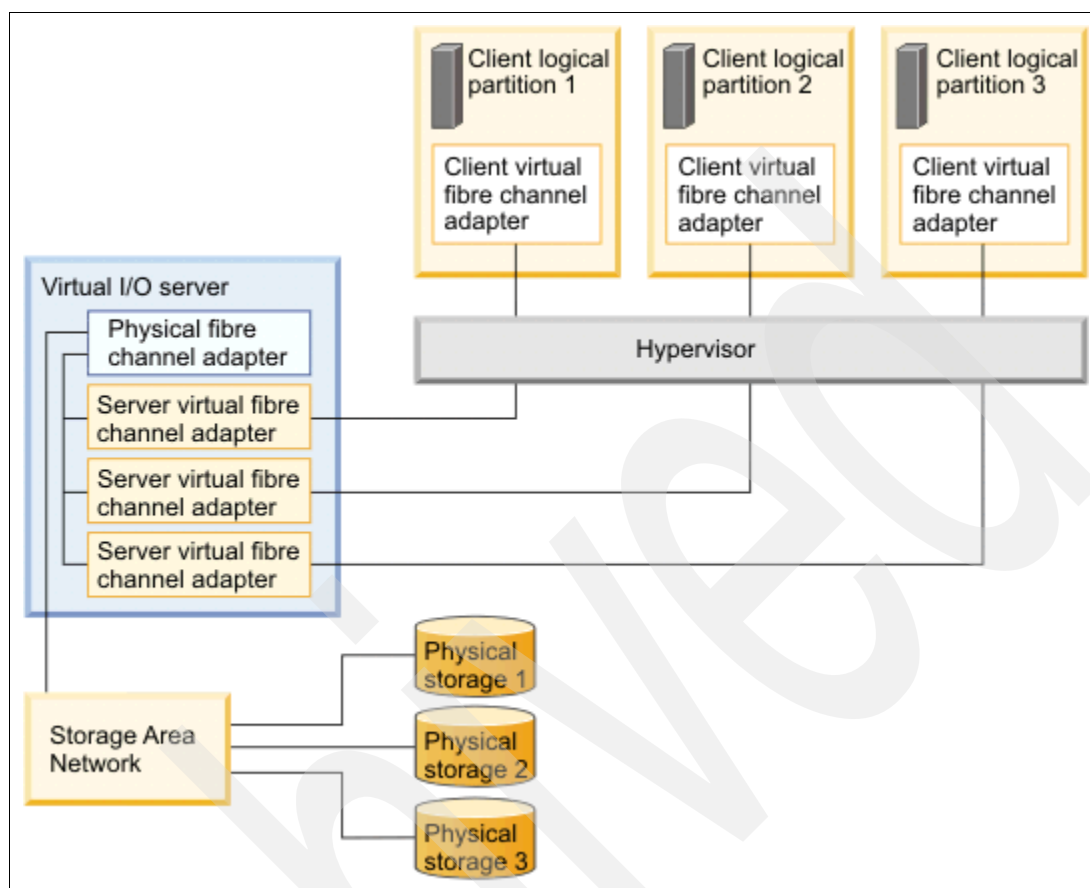


Figure 7-5 NPIV functionality illustration

Managing the VIOS

On the VIOS, the virtual Fibre Channel adapters are identified as vfchost devices as shown in Example 7-38.

Example 7-38 Virtual Fibre Channel adapters

```
$ lsdev -virtual | grep FC
vfchost0      Available    Virtual FC Server Adapter
vfchost1      Available    Virtual FC Server Adapter
vfchost2      Available    Virtual FC Server Adapter
vfchost3      Available    Virtual FC Server Adapter
$
```

After the virtual slots have been created and properly linked to the LPAR profiles, the management consists only of linking the physical adapters to the virtual Fibre Channel adapters on the VIOS.

Note: For information about creation of the virtual Fibre Channel slots, go to:

- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphad/iphatvfchmc.htm>
- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphad/iphatvfcivm.htm>

The management operations are accomplished with the assistance of three commands (**lsnports**, **lsmap**, and **vfcmmap**):

► **lsnports**

This command can list all NPIV capable ports in the VIOS, as shown in Example 7-39.

Example 7-39 Show NPIV capable physical Fibre Channel adapters with lsnports

```
$ lsnports
name physloc fabric tports aports swwpns awwpns
fcs0 U789D.001.DQDYKYW-P1-C1-T1 1 64 62 2048 2043
fcs1 U789D.001.DQDYKYW-P1-C1-T2 1 64 62 2048 2043
```

NPIV depends on a NPIV-capable physical Fibre Channel adapter and an enabled port SAN switch.

Of the fields listed in the **lsnports** command, the most important are aports and fabric:

- The aports field

Represents the number of available ports in the physical adapter; each virtual Fibre Channel connection consumes one port.

- The fabric field

Shows whether the NPIV adapter has fabric support in the switch.

If the SAN switch port where the NPIV adapter is connected does not support NPIV or the resource is not enabled, the fabric attribute is shown as 0 (zero), and the VIOS will not be able to create the connection with the virtual Fibre Channel adapter.

Note: The full definition of the fields in the **lsnports** command is in the system man page:

<http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=/ip/hcg/lsnports.htm>

► **lsmap**

This command is can list the allocation map of the physical resources to the virtual slots; when used in conjunction with the **-npiv** option, the command shows the allocation map for the NPIV connections, as demonstrated on Example 7-40.

Example 7-40 lsmap to shown all virtual Fibre Channel adapters

```
$ lsmap -all -npiv | head -15
Name Physloc CIntID CIntName CIntOS
-----
vfchost0 U9117.MMA.101F170-V1-C13 3 570_1_AIX AIX

Status:LOGGED_IN
FC name:fcs0 FC loc code:U789D.001.DQDYKYW-P1-C1-T1
Ports logged in:2
Flags:a<LOGGED_IN,STRIP_MERGE>
VFC client name:fcs0 VFC client DRC:U9117.MMA.101F170-V3-C6-T1

Name Physloc CIntID CIntName CIntOS
-----
vfchost1 U9117.MMA.101F170-V1-C14 4 570_1_LNX Linux

Status:LOGGED_IN
$
```


In the same way as a VSCSI server adapter, defining the **-vadpater** parameter to limit the output to a single vfchost is also possible.

► **vfcmmap**

This command is used to create and erase the connection between the virtual Fibre Channel adapter and the physical adapter.

The command accepts only two parameters:

- The **-vadapter** defines the vfchost to be modified.
- The **-fcp** parameter to defines which physical Fibre Channel will be bonded.

The connection creation is shown in Example 7-41.

Example 7-41 Connect a virtual Fibre Channel server adapter to a physical adapter

```
$ vfcmmap -vadapter vfchost0 -fcp fcs0
vfchost0 changed
$
```

To remove the connection between the virtual and physical Fibre Channel, the physical adapter must be omitted, as shown in Example 7-42.

Example 7-42 Remove the connection between the virtual fibre adapter and the physical adapter

```
$ vfcmmap -vadapter vfchost0 -fcp
vfchost0 changed
$
```

Important: Each time the connection is made, a new WWPN is created to the virtual Fibre Channel adapter.

A possibility is to force a specific WWPN in the virtual Fibre Channel adapter, defining it in the LPAR profile; however this approach requires extra care to avoid duplication of the WWPN into the SAN fabric.

Virtual I/O Clients

On the VIOC, the virtual Fibre Channel adapters are identified in the same way as a standard Fibre Channel adapter is. This identification is demonstrated in Example 7-43 and Example 7-44 on page 304.

Example 7-43 Virtual Fibre Channel adapter on Linux

```
[root@slovenia ~]# systool -c fc_host -a host6 -v
Class = "fc_host"

Class Device = "host6"
Class Device path = "/sys/class/fc_host/host6"
fabric_name       = "0xc05076000afe0054"
issue_lip         = <store method only>
maxframe_size    = "2048 bytes"
node_name        = "0xc05076000afe0054"
port_id          = "0x661302"
port_name        = "0xc05076000afe0054"
port_state       = "Online"
port_type        = "NPort (fabric via point-to-point)"
speed            = "2 Gbit"
supported_classes = "Class 2, Class 3"
```

```

tgtid_bind_type    = "wwpn (World Wide Port Name)"
uevent            = <store method only>

Device = "host6"
Device path = "/sys/devices/vio/30000006/host6"
uevent        = <store method only>

```

```
[root@slovenia ~]#
```

Example 7-44 Virtual Fibre Channel client adapter on AIX

```

usa:/#lscfg -l fcs0
      fcs0 U9117.MMA.101F170-V3-C6-T1 Virtual Fibre Channel Client Adapter
usa:/#

```

After configured in the VIOS, the virtual Fibre Channel client adapters operates on the same way as a physical adapter would. It requires zone and host connection in the storage or tape that will be mapped.

Tuning techniques such as the SCSI encapsulation parameters for multipath environments have to be set in all VIOC that use the virtual Fibre Channel adapters; on the servers for this book, the parameters **dyntrk** and **fc_err_recov** have been defined, as shown in Example 7-45.

Example 7-45 SCSI configuration on a virtual Fibre Channel adapter under AIX

```

usa:/#lsattr -El fscsi0
attach      switch    How this adapter is CONNECTED      False
dyntrk      yes       Dynamic Tracking of FC Devices      True
fc_err_recov fast_fail FC Fabric Event Error RECOVERY Policy True
scsi_id     0x661301    Adapter SCSI ID                     False
sw_fc_class 3         FC Class for Fabric                 True
usa:/#

```

Performance considerations

Usually, tuning guides recommend general tuning to device attributes such as `max_xfer_size`, `num_cmd_elems`, `lg_term_dma` (on AIX). These tuning devices reference only physical adapters and generally do not apply to the virtual Fibre Channel adapters. Any change in the defaults can lead to stability problems or possibly server hangs during rebooting. Therefore, tuning in the virtual Fibre Channel adapters have to be carefully considered, checking the physical adapter and in the storage device manual.

A best practice regarding optimizing I/O performance to the virtual Fibre Channel adapters (which also applies to physical adapters) consists of isolating traffic by the type of device attached to the Fibre Channel.

Tape devices is an example. Most tape drives can sustain a data throughput near 1 Gbps; an IBM TS1130 can have nearly 1.3 Gbps.

With eight tape drives attached to a single physical adapter linked to a 8 Gbps link; during the backups, the tape drivers saturate all links only with their traffic; consequently the disk I/O performance can be degraded.

Because of these factors, be sure to isolate the SAN disks from other devices on a physical adapter level, so that GPFS performance is not degraded when another type of operation that involved the Fibre Channel adapters is performed.

Considering the high availability aspects of a system, an optimal VIOS and VIOC configuration follows at least the same line pathways, as shown in Figure 7-6.

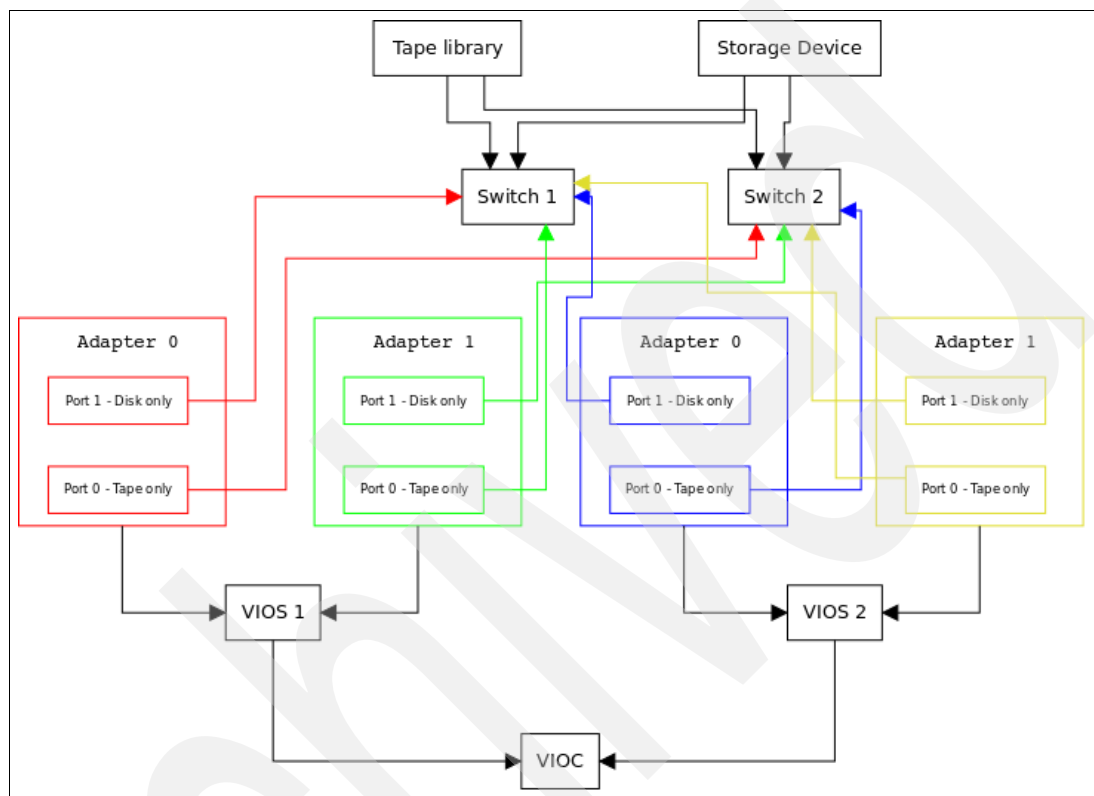


Figure 7-6 NPIV isolated traffic

In Figure 7-6, each VIOS has two dual-port NPIV adapters, each port on each NPIV adapter is used for specific traffic and is connected to a singular switch.

This way, the VIOC, has four paths to reach the storage device and the tape library: two paths per switch.

Note: For more information, see the following resources:

- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=iphcg/lsnports.htm>
- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=iphath/iphatvfc.htm>
- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphath/iphb1configvfc.htm>
- ▶ <http://www.ibm.com/systems/storage/tape/ts1130/specifications.html>
- ▶ <ftp://ftp.t11.org/t11/member/fc/da/02-340v1.pdf>

7.2 Shared Ethernet Adapter and Host Ethernet Adapter

This section describes the Shared Ethernet Adapter and Host Ethernet Adapter.

7.2.1 Shared Ethernet Adapter (SEA)

SEA allows the System p hypervisor to create a Virtual Ethernet network that enables communication across the LPARs, without requiring a physical network adapter assigned to each LPAR; all network operations are based on a virtual switch kept in the hypervisor memory. When one physical adapter is assigned to a VIOS, this adapter can be used to offload traffic into the external network, when needed.

Same as a standard Ethernet network, the SEA virtual switch can transport IP and all its subset protocols, such as Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Internet Control Message Protocol (ICMP), and Interior Gateway Routing Protocol (IGRP).

To isolate and prioritize each network and the traffic into the virtual switch, the Power Systems hypervisor uses VLAN tagging which can be spawned to the physical network. The Power Systems VLAN implementation is compatible with IEEE 802.1Q.

To provide high availability of the VIOS or even of the physical network adapters that are assigned to it, a transparent high availability (HA) feature is supplied. The disadvantage is that because the connection to the virtual switch is based on memory only, the SEA adapters do not implement the system calls to handle physical events, such as link failures. Therefore, standard link-state-based EtherChannels such as Link aggregation (802.3ad) or Active-failover, must not be used under the virtual adapters (SEA).

However, link aggregation (802.3ad) or active-failover is supported in the VIOS, and those types of devices can be used as offload devices to the SEA.

Tip: The POWER Hypervisor™ virtual Ethernet switch can support virtual Ethernet frames of up to 65408 bytes in size, which is much larger than what physical switches support (1522 bytes is standard and 9000 bytes are supported with Gigabit Ethernet jumbo frames). Therefore, with the POWER Hypervisor virtual Ethernet, you can increase the TCP/IP MTU size as follows:

- ▶ If you do not use VLAN: Increase to 65394 (equals 65408 minus 14 for the header, no CRC)
- ▶ If you use VLAN: Increase to 65390 (equals 65408 minus 14 minus 4 for the VLAN, no CRC)

Increasing the MTU size can benefit performance because it can improve the efficiency of the transport. This benefit is dependant on the communication data requirements of the running workload.

SEA high availability works on an active or backup model; only one server offloads the traffic and the others wait in standby mode until a failure event is detected in the active server. Its functionality is illustrated by Figure 7-7 on page 307.

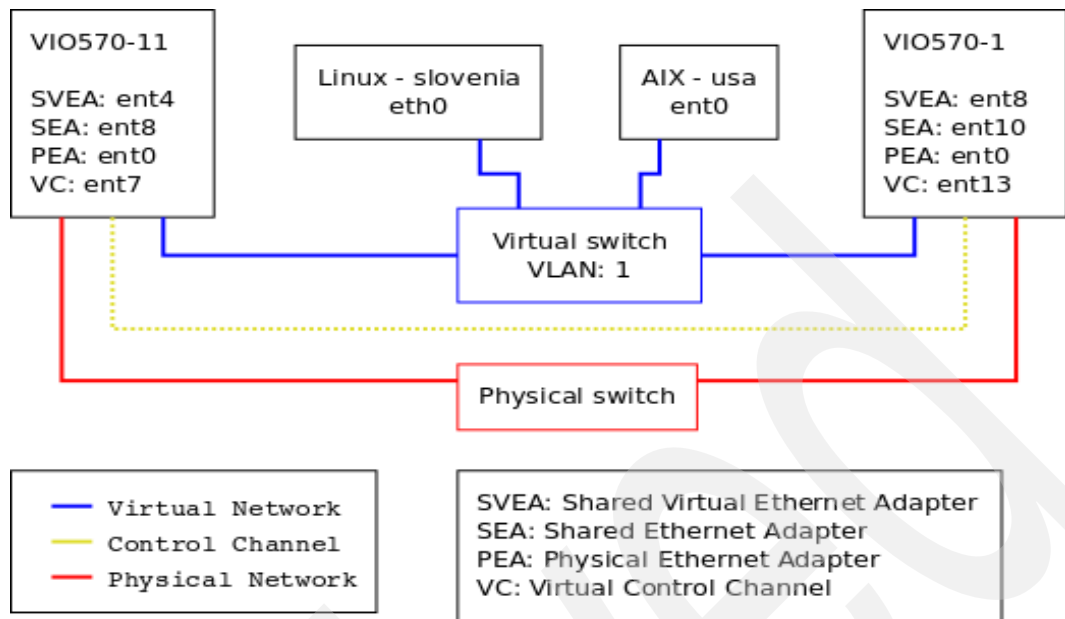


Figure 7-7 SEA functionality into the servers

SEA requires two types of virtual adapters to work:

- ▶ Standard adapter, used by the LPARs (servers) to connect into the virtual switch
- ▶ Trunk adapter, used by the VIOS to offload the traffic to the external network

When HA is enabled in the SEA, a secondary standard virtual Ethernet adapter must be attached to the VIOS. This adapter will be used to send heart beats and error events to the hypervisor, providing a way to determine the health of the SEA on the VIOS; this interface is called the *Control Channel adapter*.

Under the management console, determining which adapters are the trunk devices and which are not is possible. The determination happens with the `is_trunk` parameter. When it is set to 1 (one), the adapter is a trunk adapter and can be used to create the bond with the offload adapter into the VIOS. Otherwise, the adapter is not a trunk adapter.

In Example 7-46, the adapters that are placed in slot 20 and 21 on both VIOS are highlighted and are trunk adapters. The adapters in slot 20 manage the VLAN 1 and the adapters in slot 21 manage the VLAN 2. Other adapters are standard virtual Ethernet adapters.

Example 7-46 SEA slot and trunk configuration

```
hscroot@hmc4:~> lshwres -r virtualio --subtype eth --level lpar -m
570_1-9117-MMA-101F170 -F "lpar_name lpar_id slot_num is_trunk trunk_priority
port_vlan_id" --header --filter "lpar_ids=1"
lpar_name lpar_id slot_num is_trunk trunk_priority port_vlan_id
570_1_VIO_1 1 20 1 1 1
570_1_VIO_1 1 21 1 1 2
570_1_VIO_1 1 22 0 unavailable 3
570_1_VIO_1 1 30 0 unavailable 1
hscroot@hmc4:~> lshwres -r virtualio --subtype eth --level lpar -m
570_1-9117-MMA-101F170 -F "lpar_name lpar_id slot_num is_trunk trunk_priority
port_vlan_id" --header --filter "lpar_ids=2"
lpar_name lpar_id slot_num is_trunk trunk_priority port_vlan_id
570_1_VIO_2 2 20 1 2 1
```

```
570_1_VIO_2 2 21 1 2 2
570_1_VIO_2 2 22 0 unavailable 3
570_1_VIO_2 2 30 0 unavailable 1
hscroot@hmc4:~>
```

The adapters that are placed on slot 30, shown Example 7-46 on page 307, are standard adapters (not trunk adapters) that are used by the VIOS to access the virtual switch. The VIOS IP address is configured on these adapters.

SEA requires that only one VIOS controls the offload traffic for a specific network each time. The `trunk_priority` field controls the VIOS priority. The working VIOS with the lowest priority manages the traffic.

When the SEA configuration is defined in VIOS, another interface is created. This interface is called the Shared Virtual Ethernet Adapter (SVEA), and its existence can be verified with the command **lsmap**, as shown in Example 7-47.

Example 7-47 List virtual network adapters into VIOS, SEA

```
$ lsmap -net -all
SVEA  Physloc
-----
ent4  U9117.MMA.101F170-V2-C20-T1

SEA                                     ent8
Backing device                         ent0
Status                                Available
Physloc                               U789D.001.DQDYKYW-P1-C6-T1

SVEA  Physloc
-----
ent5  U9117.MMA.101F170-V2-C21-T1

SEA                                     NO SHARED ETHERNET ADAPTER FOUND

SVEA  Physloc
-----
ent6  U9117.MMA.101F170-V2-C30-T1

SEA                                     NO SHARED ETHERNET ADAPTER FOUND

SVEA  Physloc
-----
ent7  U9117.MMA.101F170-V2-C22-T1

SEA                                     NO SHARED ETHERNET ADAPTER FOUND
```

Before creating an SVEA adapter, be sure to identify which offload Ethernet adapters are SEA-capable and available in the VIOS. Identifying them can be achieved by the **lsdev** command, specifying the `ent4sea` device type, as shown in Example 7-48 on page 309.

Example 7-48 SEA capable Ethernet adapters in the VIOS

```
$ lsdev -type ent4sea
name status      description
ent0 Available    4-Port 10/100/1000 Base-TX PCI-Express Adapter (14106803)
name status      description
ent1 Available    4-Port 10/100/1000 Base-TX PCI-Express Adapter (14106803)
name status      description
ent2 Available    4-Port 10/100/1000 Base-TX PCI-Express Adapter (14106803)
name status      description
ent3 Available    4-Port 10/100/1000 Base-TX PCI-Express Adapter (14106803)
```

After the offload device has been identified, use the **mkvdev** command to create the SVEA adapter, as shown in Example 7-49.

Example 7-49 Create a SEA adapter in the VIOS

```
$ mkvdev -sea ent0 -vadapter ent4 -default ent4 -defaultid 1 -attr ha_mode=auto
ctl_chan=ent7
```

Note: The SVEA is a VIOS-specific device that is used only to keep the bond configuration between the SEA trunk adapter and the Ethernet offload adapter (generally a physical Ethernet port or a link aggregation adapter).

The adapter creation shown in Example 7-49 results in the configuration shown in Figure 7-7 on page 307.

Important: Example 7-49 assumes that multiple Virtual I/O Servers manage the SEA, implying that the **mkvdev** command must be replicated to all VIOS parts of this SEA.

If the SVEA adapter is created only on a fraction of the Virtual I/O Server's part of the SEA, the HA tends to not function properly.

To determine the current configuration of the SEA in the VIOS, the **-attr** parameter can be passed to the **lsdev** command, as shown in Example 7-50.

Example 7-50 Showing attributes of a SEA adapter in the VIOS

```
$ lsdev -dev ent8 -attr
attribute  value  description
user_settable

accounting disabled Enable per-client accounting of network statistics
True
ctl_chan   ent7   Control Channel adapter for SEA failover
True
gvrp       no     Enable GARP VLAN Registration Protocol (GVRP)
True
ha_mode     auto  High Availability Mode
True
jumbo_frames no     Enable Gigabit Ethernet Jumbo Frames
True
large_receive no     Enable receive TCP segment aggregation
True
```

```

largesend    0      Enable Hardware Transmit TCP Resegmentation
True
netaddr      0      Address to ping
True
pvid         1      PVID to use for the SEA device
True
pvid_adapter ent4    Default virtual adapter to use for non-VLAN-tagged packets
True
qos_mode     disabled N/A
True
real_adapter ent0    Physical adapter associated with the SEA
True
thread       1      Thread mode enabled (1) or disabled (0)
True
virt_adapters ent4    List of virtual adapters associated with the SEA (comma
separated) True

```

To change the attributes of the SEA adapter, use the **chdev** command. Example 7-51 shows one example of the command, which is also described in the **chdev** system manual.

Monitoring network utilization

Because SEA shares the physical adapters (or link aggregation devices) to offload the traffic to the external networks, tracking the network utilization is important for ensuring that no unexpected delay of important traffic occurs.

To track the traffic, use the **seastat** command, which requires that the promoter accounting is enabled in the SEA, as shown in Example 7-51.

Example 7-51 Enable accounting into a SEA device

```

$ chdev -dev ent8 -attr accounting=enabled
ent8 changed
$

```

Note: The accounting feature increases the processor utilization in the VIOS.

After the accounting feature is enabled into the SEA, the utilization statistics can be determined, as Example 7-52 shows.

Example 7-52 Partial output of the seastat command

```

$ seastat -d ent10

=====

Advanced Statistics for SEA
Device Name: ent10

=====
MAC: 6A:88:84:36:E2:03
-----

VLAN: None
VLAN Priority: None
Hostname: us
IP: 192.168.101.132

```


Transmit Statistics:

Packets: 58283

Bytes: 7810813

Receive Statistics:

Packets: 607642

Bytes: 51502830

To make the data simpler to read, and easier to track, the **sea_traf** script was written to monitor the SEA on VIOS servers.

The script calculates the average bandwidth utilization in bits, based on the amount of bytes transferred into the specified period of time, with the following formula:

$$\text{speed} = \text{bytes} * 8 / \text{time}$$

The formula has the following values:

speed: Average speed into the determined time
 bytes: Amount of bytes transferred into the period of time
 8: Factor to convert bytes into bits (1 byte = 8 bits)
 time: Amount of time taken to transfer the data

The script takes three parameters:

- ▶ Device: SEA device that will be monitored
- ▶ Seconds: Time interval between the checks, in seconds
- ▶ Number of checks: Number of times the script will collect and calculate SEA bandwidth utilization

The utilization is shown in Example 7-53.

Example 7-53 Using sea_traf.pl to calculate the network utilization into VIOS

```
# oem_setup_env
# sea_traf.pl
Usage:
  sea_traf <dev> <int> <count>
  <dev>   - SEA device to monitor
  <int>    - Seconds between the checks
  <count> - Number of checks
----
The sea interface must have accounting enabled before you use this script
# sea_traf.pl ent10 1 10
HOST                               Transmit      Receive  5
us                                 0 Bbit/s     960 Bbit/s
vio570-1                           0 Bbit/s     960 Bbit/s
slovenia                           0 Bbit/s     960 Bbit/s
FE80:0000:0000:0000:6888:87FF:FEA9:C703  0 Bbit/s     960 Bbit/s
vio570-11                          0 Bbit/s     960 Bbit/s
#
```

Note: The sea_traf.pl must be executed in the root environment to read the terminal capabilities.

Before executing it, be sure to execute the **oem_setup_env** command.

QoS and traffic control

The VIOS accepts a definition for quality of service (QoS) to prioritize traffic of specified VLANs. This definition is set with the SEA `quo_mode` attribute of the adapter.

The QoS mode is disabled as the default configuration, which means no prioritization to any network. The other modes are as follows:

- **Strict mode**

In this mode, more important traffic is bridged over less important traffic. Although this mode provides better performance and more bandwidth to more important traffic, it can result in substantial delays for less important traffic. The definition of the strict mode is shown in Example 7-54.

Example 7-54 Define strict QOS on SEA

```
chdev -dev <SEA device name> -attr qos_mode=strict
```

- **Loose mode**

A cap is placed on each priority level so that after a number of bytes is sent for each priority level, the following level is serviced. This method ensures that all packets are eventually sent. Although more important traffic is given less bandwidth with this mode than with strict mode, the caps in loose mode are such that more bytes are sent for the more important traffic, so it still gets more bandwidth than less important traffic. The definition of the loose mode is shown in Example 7-55.

Example 7-55 Define loose QOS into SEA

```
chdev -dev <SEA device name> -attr qos_mode=loose
```

In either strict or loose mode, because the Shared Ethernet Adapter uses several threads to bridge traffic, it is still possible for less important traffic from one thread to be sent before more important traffic of another thread.

Note: For more information, see the following resources:

- *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940 (“2.7 Virtual and Shared Ethernet introduction”)
- http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphb1/iphb1_vios_scenario_network_failover.htm
- http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphb1/iphb1_vios_concepts_network_sea.htm
- <http://www.ibm.com/support/docview.wss?uid=isg3T1011897>
- <http://www.ibm.com/support/docview.wss?uid=isg3T1011040>

Considerations

Consider using Shared Ethernet Adapters on the Virtual I/O Server in the following situations:

- When the capacity or the bandwidth requirement of the individual logical partition is inconsistent or is less than the total bandwidth of a physical Ethernet adapter or a link aggregation. Logical partitions that use the full bandwidth or capacity of a physical Ethernet adapter or link aggregation should use dedicated Ethernet adapters.
- If you plan to migrate a client logical partition from one system to another.
- Consider assigning a Shared Ethernet Adapter to a Logical Host Ethernet port when the number of Ethernet adapters that you need is more than the number of ports available on

the Logical Host Ethernet Adapters (LHEA), or you anticipate that your needs will grow beyond that number. If the number of Ethernet adapters that you need is fewer than or equal to the number of ports available on the LHEA, and you do not anticipate needing more ports in the future, you can use the ports of the LHEA for network connectivity rather than the Shared Ethernet Adapter.

7.2.2 Host Ethernet Adapter (HEA)

An HEA is a physical Ethernet adapter that is integrated directly into the GX+ bus on the system. The HEA is identified on the system as a Physical Host Ethernet adapter (PHEA or PHYS), and can be partitioned into several LHEAs that can be assigned directly from the hypervisor to the LPARs. The LHEA adapters offer high throughput, low latency. HEAs are also known as Integrated Virtual Ethernet (IVE) adapters.

The number and speed of the LHEAs that can be created in a PHEA vary according to the HEA model that is installed on the system. A brief list of the available models is in “Network design” on page 6.

The identification of the PHEA that is installed into the system can be done through the management console, as shown in Example 7-56.

Example 7-56 Identifying a PHEA into the system through the HMC management console

```
hscroot@hmc4:~> lshwres -r hea --rsubtype phys -m 570_1-9117-MMA-101F170 \
> --level port -F adapter_id,port_group,phys_port_type,curr_conn_speed
23000000,1,10000,10000
23000000,2,10000,10000
hscroot@hmc4:~>
```

System 570_1-9117-MMA-101F170 in Example 7-56 has two available 10 Gbps physical ports. Each PHEA port controls the connection of a set of LHEAs, organized into groups. Those groups are defined by the port_group parameter, shown in Example 7-57.

Example 7-57 PHEA port groups

```
hscroot@hmc4:~> lshwres -r hea --rsubtype phys -m 570_1-9117-MMA-101F170 \
> --level port_group -F
adapter_id,port_group,unassigned_logical_port_ids,curr_port_group_mcs_value
23000000,1,"3,4,5,6,7,8,9,10,11,12,13,14,15,16",4
23000000,2,"3,4,5,6,7,8,9,10,11,12,13,14,15,16",4
hscroot@hmc4:~>
```

As Example 7-57 shows, each port_group supports several logical ports; each free logical port is identified by the unassigned_logical_port_ids field. However, the number of available ports is limited by the curr_port_group_mcs_value field, which displays the number of logical ports supported by the physical port.

The system shown in Example 7-57 supports four logical ports per physical port, and has four logical ports in use, two on each physical adapter, as shown in Example 7-58 on page 314.

Example 7-58 List LHEA ports through HMC command line

```
hscroot@hmc4:~> lshwres -r hea --subtype logical -m 570_1-9117-MMA-101F170 \
> --level port -F \
> adapter_id,lpar_id,port_group,logical_port_id,lpar_id,lpar_name \
> | grep -v none
23000000,3,1,1,3,570_1_AIX
23000000,4,1,2,4,570_1_LNX
23000000,3,2,1,3,570_1_AIX
23000000,4,2,2,4,570_1_LNX
hscroot@hmc4:~>
```

The connection between the LHEA and the LPAR is made during the profile setup.

As shown in Figure 7-8, after selecting the LHEA tab in the profile setup, the PHEA (physical port) can be selected and configured.

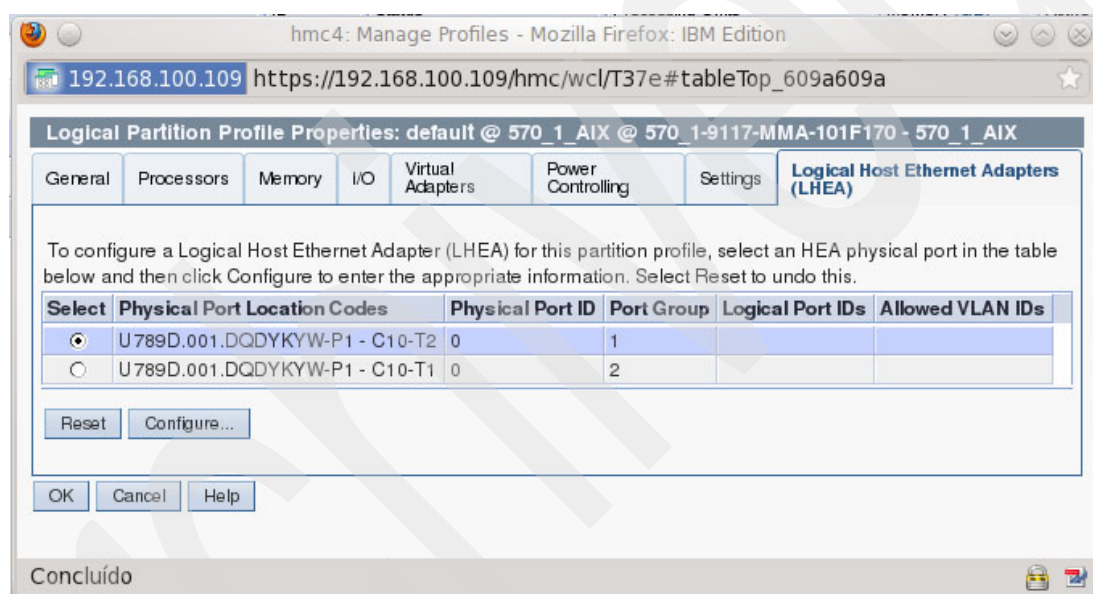


Figure 7-8 LHEA physical ports shown in HMC console

When the port that will be used is selected to be configured, a window similar to Figure 7-9 opens. The window allows selecting which one will be assigned to the LPAR.

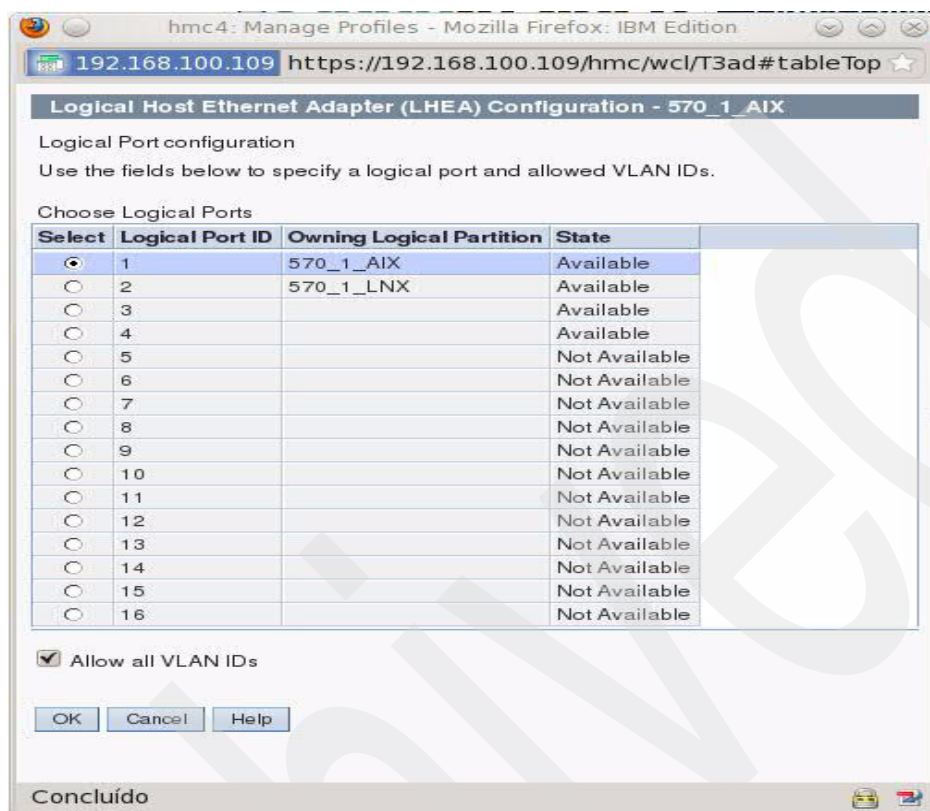


Figure 7-9 Select LHEA port to assign to a LPAR

The HEA can also be set up through the HMC command-line interface, with the assistance of the **chsyscfg** command. For more information about it, see the VIOS manual page.

Notice that a single LPAR cannot hold multiple ports in the same PHEA. HEA does not provide high availability capabilities by itself, such as SEA ha_mode.

To achieve adapter fault tolerance, other implementations have to be used. The most common implementation is through EtherChannel (bonding), which bonds multiple LHEA into a single logical Ethernet adapter, under the server level.

Note: For more information about HEA and EtherChannel and bonding, see the following resources:

- ▶ *Integrated Virtual Ethernet Adapter Technical Overview and Introduction*, REDP-4340
- ▶ <http://www.kernel.org/doc/Documentation/networking/bonding.txt>
- ▶ http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp?topic=/com.ibm.aix.commadmin/doc/commadmdita/etherchannel_consider.htm
- ▶ <http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=/iphat/iphathea.htm>

Archived

Information lifecycle management (ILM)

This chapter explains the technology and features that are used by GPFS to manage the information over its lifetime.

The chapter also shows the tools that are provided by GPFS for managing the information, based on several defined scenarios.

This chapter contains the following topics:

- ▶ 8.1, “Explaining the ILM concept” on page 318
- ▶ 8.2, “Tivoli Storage Manager” on page 329
- ▶ 8.3, “Information lifecycle management scenarios” on page 346
- ▶ 8.4, “Backup and restore” on page 376
- ▶ 8.5, “Advanced external pool scenario” on page 384

8.1 Explaining the ILM concept

GPFS has implemented several tools that allow the administrator to increase the ILM efficiencies through powerful policy-driven automated tiered storage management. Using these tools, GPFS can automatically determine where to physically store your data regardless of its placement in the logical directory structure. File, file sets, storage pools, and user-defined policies provide the ability to match the cost of your storage resources to the value of your data. Using GPFS you can manage the following information:

- ▶ Files and file sets (you may also automate the management of file data)
- ▶ Pools of storage

GPFS introduces several terms that are used with regard to the ILM. Several terms are also used in other software or hardware products, and can have similar or different meaning. To reduce the confusion, this section explains the terms and shows several examples.

8.1.1 Snapshot management tasks

A snapshot of an entire GPFS file system can be created to preserve the contents of the file system at a single point in time. The storage overhead for maintaining a snapshot keeps a copy of data blocks that would otherwise be changed or deleted after the time of the snapshot.

Snapshots of a file system are *read-only*; changes can only be made to the active, that is, normal, non-snapshot, files and directories.

The snapshot function allows a backup or mirror program to run concurrently with user updates and still obtain a consistent copy of the file system as of the time that the snapshot was created. Snapshots also provide an online backup capability that allows easy recovery from common problems such as accidental deletion of a file, and comparison with older versions of a file.

Note: Snapshots are *not* copies of the entire file system; do *not* use snapshots as protection against media failures.

Snapshots appear in the file system tree as hidden subdirectories of the root. Each snapshot of the file system, device, is identified by directory name on the **mmcrsnapshot** command, as in Example 8-1, where the **gpfs1** is the file system name and **firstsnap** is the snapshot name.

Example 8-1 Creating a snapshot

```
#mmcrsnapshot gpfs1 firstsnap
```

The following functions, listed with their commands, are the most commonly used regarding GPFS. For more information, see the GPFS manuals.

- ▶ Listing GPFS snapshots
 - Use the **mmllsnapshot** command to display existing snapshots of a file system and their attributes.
 - The **-d** flag is used to display the amount of storage used by a snapshot GPFS.

- The **-Q** flag displays whether or not quotas were automatically activated at mount time when the snapshot was created.
- When a snapshot is restored with the **mmrestorefs** command, restoring the quota activation with the **mmchfs -Q** command might be necessary.
- ▶ Restoring a GPFS file system from a snapshot:
 - Use the **mmrestorefs** command to restore user data and attribute files in an active file system from the specified snapshot.
 - The file system *must* be unmounted from all nodes in the cluster prior to issuing the **mmrestorefs** command.
 - The file system must *not* be remounted until the **mmrestorefs** command has successfully completed.

Note: The **mmrestorefs** command can take a long time to process, depending on the file system size.

- Existing snapshots, including the one being used in the restore operation, are not affected by the **mmrestorefs** command.
- ▶ Linking to the GPFS snapshots
 - Snapshot root directories appear in a special **.snapshots** directory under the file system root.
 - If you prefer to link directly to the snapshot rather than always traverse the root directory, you may use the **mmsnapdir** command to add a **.snapshots** subdirectory to all directories in the file system.
 - The **.snapshots** directories that are added by the **mmsnapdir** command are invisible to the **ls** command or to the **readdir()** function. This technique prevents recursive file system utilities, such as **find** or **tar**, from entering into the snapshot tree for each directory they process.
 - Specifying the **-r** option on the **mmsnapdir** command changes back to the default behavior, which is a single **.snapshots** directory in the file system root directory.
- ▶ Deleting a GPFS snapshot
 - Use the **mmde1snapshot** command to delete GPFS snapshots of a file system.
 - This command frees the space taken by the snapshot.

8.1.2 Storage pools

Physically, a *storage pool* is a collection of disks or RAID arrays. Storage pools also allow you to group multiple storage systems within a file system. Using storage pools, you can create tiers of storage by grouping storage devices based on performance, locality, or reliability characteristics. A storage pool is an attribute for the Network Shared Disk (NSD). For example, one pool can be an enterprise class storage system that hosts high-performance Fibre Channel disks, and another pool might consist of numerous disk controllers that host a large set of economical SATA disks.

Figure 8-1 on page 320 depicts an example of four nodes of a GPFS cluster that have defined several storage pools:

- ▶ System pool, for metadata information
- ▶ Gold pool, for files frequently accessed

- ▶ Silver pool, for files of large size
- ▶ External pool, for near online files that are seldom accessed

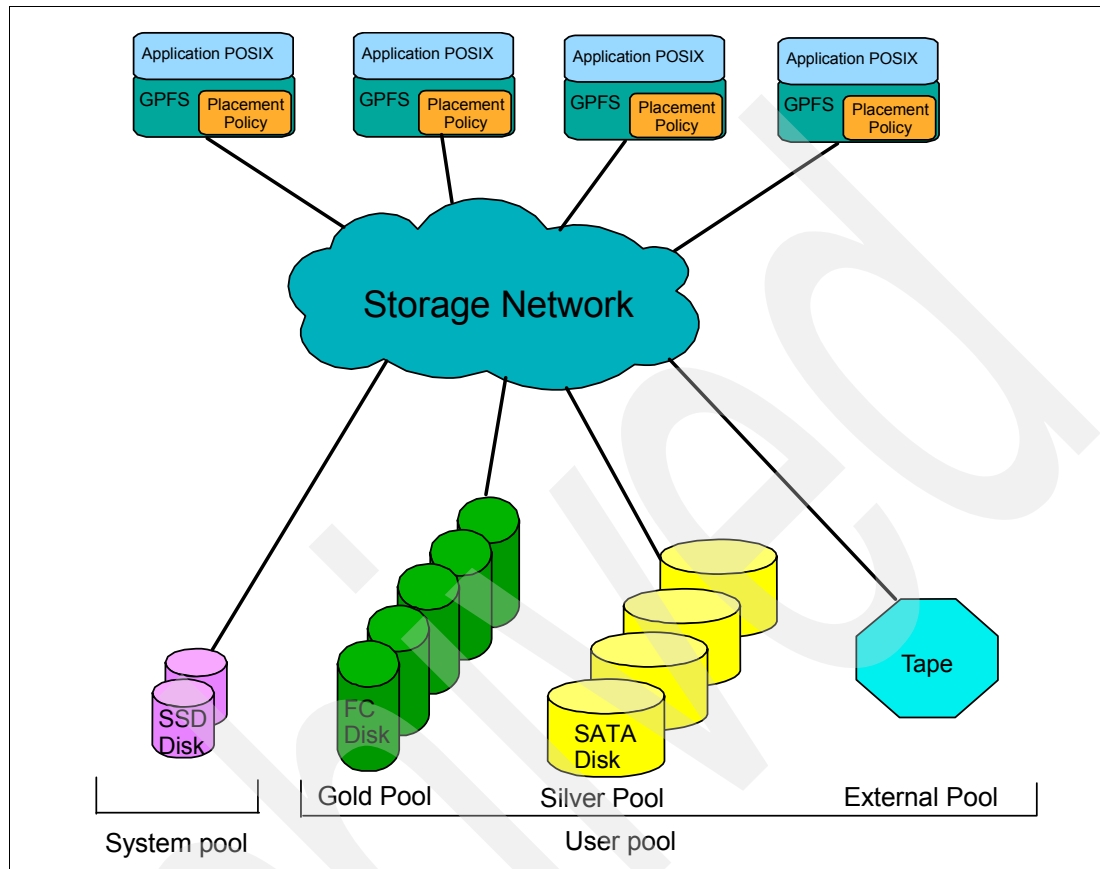


Figure 8-1 Storage pools

Two types of storage pools exist:

- ▶ Internal storage pools are managed within GPFS.
 - A system pool has the following characteristics:
 - Defined by default when the GPFS cluster is created
 - Can hold data and metadata
 - If no policy is defined all the data goes in system pool
 - When upgrading GPFS from a version without storage pools features, all the NSDs are defined in system pool unless otherwise specified.
 - Metadata cannot be moved out of system storage pool, through management policy, and it does not require placement policy to place data in.
 - There is only one system storage pool per file system; a file system cannot be created without the system storage pool.
 - System storage pool cannot be deleted unless deleting the entire file system.
 - Disks inside system pool can be deleted if there is at least one disk assigned to system pool or enough disks with space to store existing metadata.
 - Because system storage pool contains metadata, one should choose the premium storage to be assigned to system pool for reason such as better performance and failure protection.

- User pools are manually defined when NSDs are created and have the following characteristics:
 - Can hold only user data, not metadata. Because user storage pool stores only data, disks that are assigned to user storage pools can only have dataOnly usage type.
 - All user data for the file gets stored in the assigned storage pool, based on placement policy or using the **mmchattr** command.
 - A file's data can be migrated to a separate storage pool in the file system, based on management policy or using the **mmchattr** command.
 - Up to seven user storage pools can be created for each file system.
- External storage pools are managed by an external application such as Tivoli Storage Manager.
 - GPFS provides tools that allow you to define an interface that your external storage manager uses to access your data.
 - GPFS does not manage the data placed in external storage pools. Instead, GPFS manages the movement of data to and from external storage pools.

Storage pool example

To create an internal user storage pool, disk descriptor *must* contain the storage pool name; the sixth field of disk descriptor is reserved for the storage pool name, as in Example 8-2.

Example 8-2 Disk descriptor file

```
DiskName:ServerList::DiskUsage:FailureGroup:DesiredName:StoragePoolName
```

Alternately, you can specify the storage pool during file system creation, as in Example 8-3.

Example 8-3 Storage pool association

```
# mmcrfs /myfs myfs "gpfs64nsd:::dataOnly:::userPool1"
```

Important: If storage pool is not specified on the disk descriptor, it is by default assigned to the system storage pool.

When choosing a storage pool name, consider the following guidelines:

- It must be unique within a file system, but not across file systems.
- It must not be larger than 255 alphanumeric characters.
- It is case-sensitive; for example Mypool and myPool are distinct storage pools.

To create or define storage pools, use the following commands:

| | |
|------------------|---|
| mmcrfs | Use when file system is created. |
| mmadddisk | Use when one or more disk are added to the file system. |
| mmcrnsd | Use when one or more disk are added to the GPFS cluster without creating a file system. |

The following commands are also useful:

| | |
|------------------|---|
| mmldisk | Lists disks in a file system. |
| mmfsfs -P | Can be used to list storage pools in a file system. |
| mmdf -P | Lists disk statistics in the storage pools. |

Delete a storage pool

The following considerations apply when you delete a storage pool:

- ▶ User Storage pools can be deleted by deleting the last disk in the storage pool; system storage pool cannot be deleted.
- ▶ Only one storage pool can be deleted at a time with the **mmde1disk** command.
- ▶ Pools containing data cannot be deleted. You must either delete the data or migrate data to separate pool before deleting the disk.
- ▶ To delete system storage pool you must delete the entire file system. However, if you have multiple disks in the system storage pool, they can be deleted if the remaining disks have enough space to hold existing metadata.

Working with files in storage pools

To change a file storage pool assignment, consider the following information:

- ▶ The *root* user may change a file's assigned storage pool by issuing the **mmchattr -P** command.
- ▶ The default method is to migrate the data immediately (use the **-I yes** parameter).
- ▶ If the parameter **-I defer** is specified, the command does not move the existing data to the new storage pool immediately and is deferred until a later call to either the **mmrestripefs** or **mmrestripefile** command.

To rebalance or to restore files in a storage pool, consider the following information:

- ▶ The *root* user may rebalance or restore all files in a file system by issuing the **mmrestripefs** command.
- ▶ Specify the **-P** option (uppercase P), to repair only files that are assigned to the specified storage pool.
- ▶ Specify the **-p** option (lowercase p), to repair the file placement within the storage pool.
- ▶ Alternately, you may issue the **mmrestripefile -p** command to repair the placement of specific files within the storage pool.
- ▶ Files that are assigned to one storage pool, but with data in a separate pool, will have their data migrated to the correct pool.

Note: To use a user-defined storage pool, a *placement policy* has to be defined. By default, all files are written in the system storage pool, and when it is filled, an *out-of-space* error occurs even if the user-defined storage pools are empty.

8.1.3 File sets

A file set is a subtree of a file system namespace that in many respects behaves like an independent file system. File sets provide a means of partitioning the file system to allow administrative operations at a finer granularity than the entire file system.

Figure 8-2 on page 323 shows an example of several file sets, as follows:

- ▶ The Fileset_u1 starts from /dir4/dir_b directory and has /user1/ as the parent directory.
- ▶ The Fileset_u2, which starts at /user2 directory and has as its children the data1, data2, data3, and data4 directories.

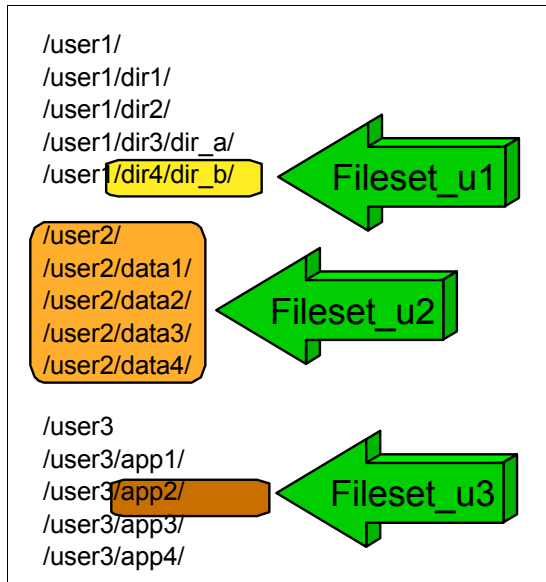


Figure 8-2 File sets

Consider the following information about file sets:

- ▶ File sets can be used to define quotas on both data blocks and inodes.
- ▶ The owning file set is an attribute of each file and can be specified in a policy to control the following items:
 - Initial data placement
 - Migration
 - Replication of the file's data
- ▶ Consider the following information about file sets implementation:
 - When the file system is first created, only one file set exists, which is called the root file set. The root file set contains the root directory and any system files, such as quota files.
 - As new files and directories are created, they automatically become part of the parent directory's file set.
 - The file set to which a file belongs is largely transparent for ordinary file access. However, to display the containing file set and the other attributes of each file, use the **mmfsattr -L** command
 - When upgrading an existing file system to the latest version of GPFS that supports file sets, all existing files and directories are assigned to the root file set.
- ▶ Consider the following information about file set namespaces:
 - A newly created file set consists of an empty directory for the root of the file set, and it is initially not linked into the file system's namespace.
 - A newly created file set is not visible to the user until it is attached to the namespace by issuing the **mmmlinkfileset** command.
 - File sets are attached to the namespace with a special link called a *junction*.
 - Only one junction is allowed per file set, so that a file set has a unique position in the namespace and a unique path to any of its directories.

- The target of the junction is referred to as the child file set, and a file set can have any number of children.
- After a file set is created and linked into the namespace, an administrator can unlink the file set from the namespace by issuing the `mmunlinkfileset` command.
- Unlinking makes all files and directories within the file set inaccessible. If other file sets are linked below it, the other file sets become inaccessible, but they do remain linked and will become accessible again when the file set is relinked.
- Unlinking a file set, like unmounting a file system, fails if there are open files. The `mmunlinkfileset` command has a force option to close the files and force the unlink.
- After a file set is unlinked, it can be re-linked into the namespace at its original location or any other location as shown in Figure 8-3.

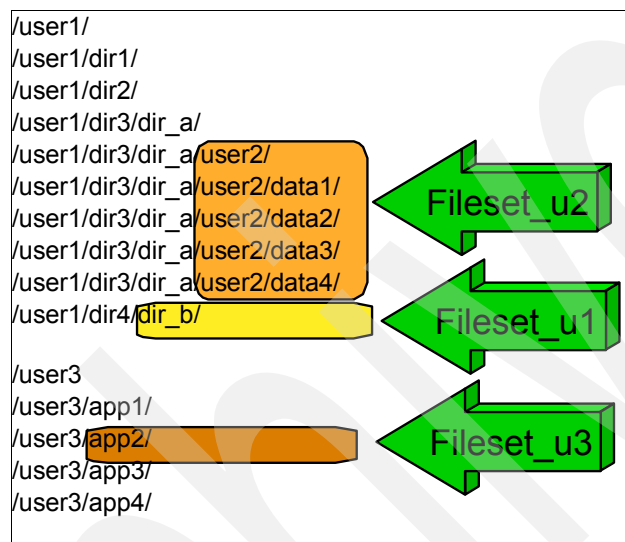


Figure 8-3 File sets linked in a separate location

- A file set has only one root directory and no other entry points such as hard links from directories in other file sets.
- The `mv` and `ln` commands cannot cross file set boundaries.
- Symbolic links can be used to provide shortcuts to any file system object in the namespace.
- The root directory of a GPFS file system is also the root of the root file set. The root file set is an exception. The root file set is attached to the local namespace by using the standard mount command. It cannot be created, linked, unlinked, or deleted by using the GPFS file set commands.
- Consider the following information about file sets and storage pools:
 - File sets are not specifically related to storage pools, although each file in a file set physically resides in blocks in a storage pool.
 - The relationship is many-to-many:
 - Each file in the file set can be stored in a separate user storage pool
 - A storage pool can contain files from many file sets, however, all of the data for a particular file is wholly contained within one storage pool.

- Using file-placement policies, you can specify that all files created in a particular file set are to be stored in a specific storage pool.
- Using file-management policies, you can define how files in a specific file set are to be moved or deleted during the file's life cycle.
- Consider the following information about file sets and snapshots:
 - A GPFS snapshot preserves the content of the entire file system, including all its file sets, *even* unlinked ones.
 - The saved file system can be accessed through the `.snapshots` directories; the namespace, including all linked file sets, appears as it did when the snapshot was created.
 - Unlinked file sets are inaccessible in the snapshot, as they were in the active file system. However, restoring a snapshot also restores the unlinked file sets, which can then be relinked and accessed.
 - A file set is included in a snapshot if the snapshot is created after the file set was created.
 - Deleted file sets appear in the output of the `mmfsfileset` command.
 - Using the `mmfsfileset -L` option you can display the latest snapshot that includes a file set. In particular, restoring a snapshot may undelete deleted file sets and change linked file sets to unlinked or vice versa.
- Consider the following information about file sets and backups:
 - The `mmbackup` command and Tivoli Storage Manager are unaware of the existence of file sets.
 - When restoring a file system that had been backed up to Tivoli Storage Manager, the files are restored to their original path names, regardless of the file sets of which they were originally a part.
 - During a full restore from backup, all file set information is lost and all files are restored into the root file set.
 - Saving the output of the `mmfsfileset` command aids in reconstruction of file set names and junction locations.
 - A partial restore can lead to confusion if file sets have been deleted, unlinked, or their junctions moved, since the backup was made.

Attention: If you are using the Tivoli Storage Manager Backup Archive client, you must use caution when you unlink file sets that contain data backed up by Tivoli Storage Manager.

Tivoli Storage Manager tracks files by path name and does not track file sets. As a result, when you unlink a file set, it appears to Tivoli Storage Manager that you deleted the contents of the file set. Therefore, the Tivoli Storage Manager Backup Archive client inactivates the data on the Tivoli Storage Manager server, which might result in the loss of backup data during the expiration process.

- Two commands help save and restore the structure of GPFS file systems: `mmbackupconfig` and `mmrestoreconfig` commands help with reconstructing the file system configuration before data is restored from backup.

- ▶ Quota can be defined per file set, so each file set can have its own quota with the following characteristics:
 - The quota limit on blocks and inodes in a file set are independent of the limits for specific users or groups of users.
 - File sets and quota information is saved when you use the `mmbackupconfig` command, and restored when using `mmrestoreconfig` commands.
 - Quota can be set for the following attributes:
 - Inode soft limit
 - Inode hard limit
 - Block soft limit
 - Block hard limit
 - File sets have been designed to better manage your space in a file system. They also add another dimension to quota management:
 - Managing space allocation based on file sets
 - Extending the quota management beyond the standard user and group

8.1.4 Policies and rules

GPFS provides a means to automate the management of files by using policies and rules. Properly managing your files allows you to efficiently use and balance your premium and less expensive storage resources.

Policies

GPFS supports the following policies:

- ▶ File placement policies are used to automatically place newly created files in a specific storage pool.
- ▶ File management policies are used to manage files during their life cycle by moving them to another storage pool, moving them to near-line storage, copying them to archival storage, changing their replication status, or deleting them.

A *policy* is a set of rules that describes the life cycle of user data based on the file's attributes. Each rule defines an operation or definition, such as migrate to a pool and replicate the file. The three uses for rules are as follows:

- ▶ Initial file placement
- ▶ File management
- ▶ Restoring file data

When a file is created or restored, the placement policy determines the location of the file's data and assigns the file to a storage pool. All data written to that file is placed in the assigned storage pool.

The placement policy defines the initial placement of newly created files; the rules for placement of restored data must be installed into GPFS with the `mmchpolicy` command. If a GPFS file system does not have a placement policy installed, all the data will be stored in the system storage pool. Only one placement policy can be installed at a time. If you switch from one placement policy to another, or make changes to a placement policy, that action has no effect on existing files. However, newly created files are always placed according to the currently installed placement policy.

Characteristics of a policy are as follows:

- ▶ A policy can contain any number of rules.
- ▶ A policy file is limited to a size of 1 MB.

Rules

A policy *rule* is an SQL-like statement that tells GPFS what to do with the data for a file in a specific storage pool, if the file meets specific criteria. A rule can apply to any file being created or only to files being created within a specific file set or group of file sets.

Rules specify conditions that, when true, cause the rule to be applied. Conditions that cause GPFS to apply a rule are as follows:

- ▶ Date and time when the rule is evaluated, that is, the current date and time
- ▶ Date and time when the file was last accessed
- ▶ Date and time when the file was last modified
- ▶ File set name
- ▶ File name or extension
- ▶ File size
- ▶ User ID and group ID

GPFS evaluates policy rules in order, from first to last, as they appear in the policy. The first rule that matches determines what is to be done with that file. For example, when a client creates a file, GPFS scans the list of rules in the active file-placement policy to determine which rule applies to the file. When a rule applies to the file, GPFS stops processing the rules and assigns the file to the appropriate storage pool. If no rule applies, an EINVAL error code is returned to the application.

Several rule types exist:

- ▶ Placement policies, evaluated at file creation, for example:
 - Rule *xxlfiles* set pool *gold* for file set *xxlfiles* rule *otherfiles* set pool *silver*
- ▶ Migration policies, evaluated periodically, for example:
 - Rule *cleangold* migrate from pool *gold* *threshold (90,70)* to pool *silver*
 - Rule *cleansilver* when *day_of_week()=monday* migrate from pool *silver* to pool *pewter* where *access_age > 30 days*
- ▶ Deletion policies, evaluated periodically, for example:
 - Rule *purgepewter* when *day_of_month() = 1* delete from pool *pewter* where *access_age > 365 days*

For the following tasks, use the guidelines and commands described:

- ▶ Creating a policy

Create a text file for your policy with the following guidelines:

- A policy must contain at least one rule.
- The last placement rule of a policy rule list must be as though no other placement rules apply to a file; the file will be assigned to a default pool.

- ▶ Installing a policy

Issue the **mmchpolicy** command.

- Changing a policy
Edit the text file containing the policy and issue the `mmchpolicy` command.
- Listing policies
The `mmfspolicy` command displays policy information for a given file system.
- Validating policies
The `mmchpolicy -I test` command validates but does not install a policy file.

8.1.5 ILM data flow

Figure 8-4 explains the ILM components.

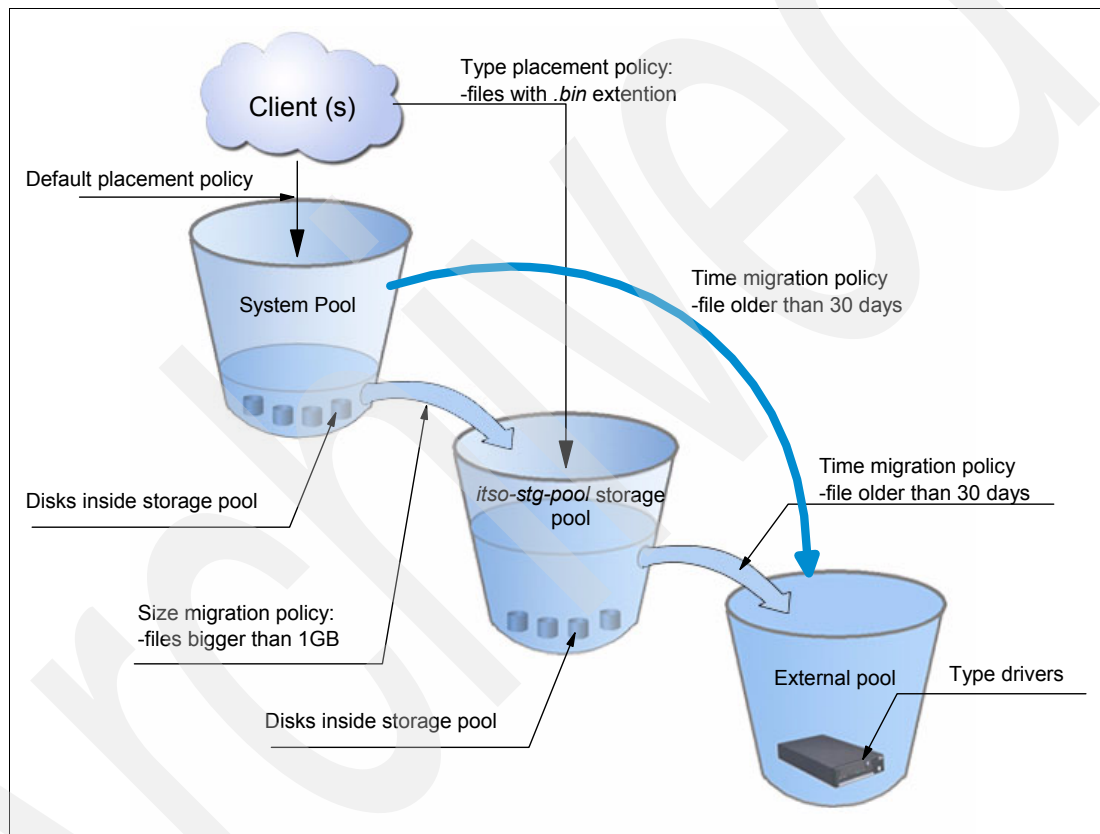


Figure 8-4 Storage pool hierarchy

The figure illustrates the following example:

- An application or user is writing to the GPFS file system.
- Three storage pools are defined:
 - The *system pool* with the fastest disks
 - The *itso-stg-pool* pool with slower disks
 - The *external* pool, Tivoli Storage Manager
- Policies are defined:
 - Initial placement
 - Files with the `.bin` extension are placed in the *itso-stg-pool* pool.
 - All other files are placed in the system pool.

- Migration policy
 - Files larger than 1 GB are moved to the *itso-stg-pool* pool.
 - Files older than 30 days are moved to the Tivoli Storage Manager.

The data flow is described as follows:

- ▶ A user or an application writes data on GPFS file system. Before data is actually written, the *initial file placement* policy is used. If the file has the *.bin* extension, it goes to the *itso-stg-pool* pool directly, otherwise it goes to system pool.
- ▶ When a file inside system pool, regardless the extension or type, reaches 1 GB of data, it is moved to the user pool transparently to the user or application. Even if the file shrinks below 1 GB, it remains on the user pool, unless another migration policy dictates otherwise.
- ▶ When a file, regardless where is located (system pool or user pool), has an access time older than 30 days, it is migrated to the Tivoli Storage Manager.
- ▶ When a file has the *.out* extension and has access time older the 90 days, it is deleted. If it was backed up by Tivoli Storage Manager, at the next Tivoli Storage Manager scan, it is marked as deleted and follows the expiration time for deleted files.

8.2 Tivoli Storage Manager

This section describes the Tivoli Storage Manager installation that is used as external pools for GPFS data. The section shows how GPFS can be integrated with an external storage pool such as Tivoli Storage Manager, and installed by using basic settings. Figure 8-5 describes the Tivoli Storage Manager (TSM) setup and environment.

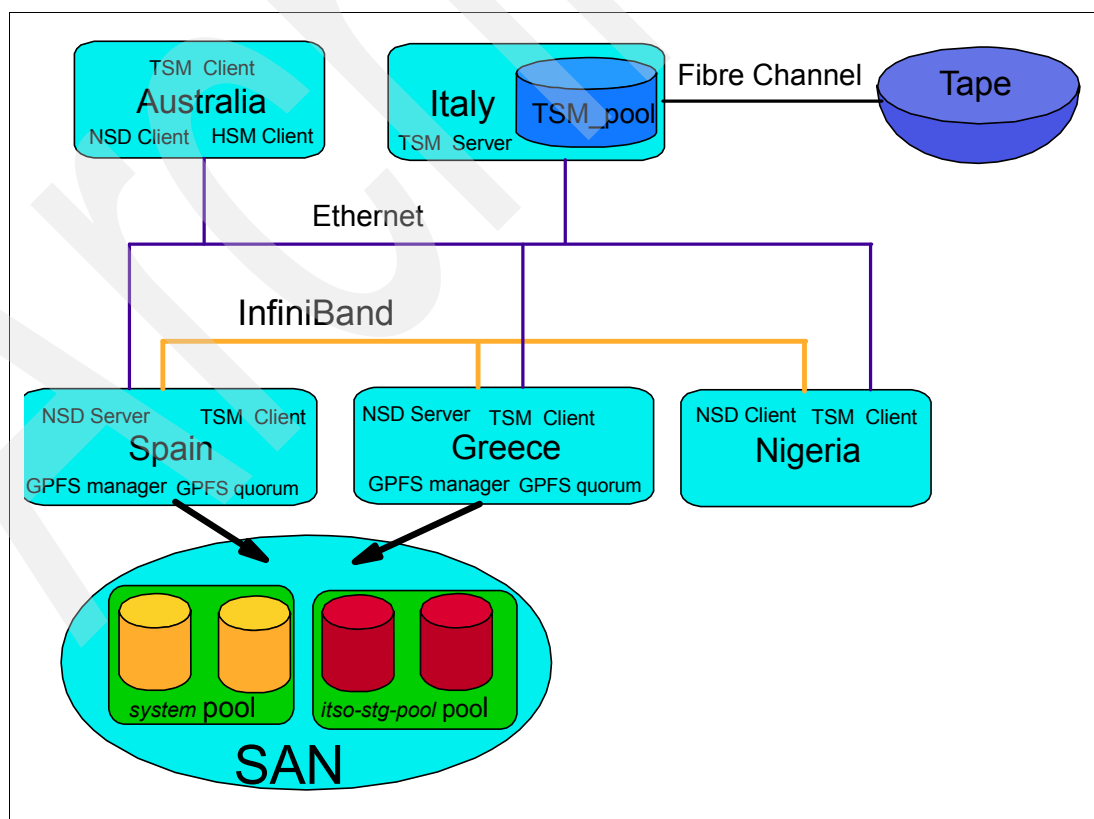


Figure 8-5 Tivoli Storage Manager environment

Figure 8-6 shows the selected servers from the general environment picture.

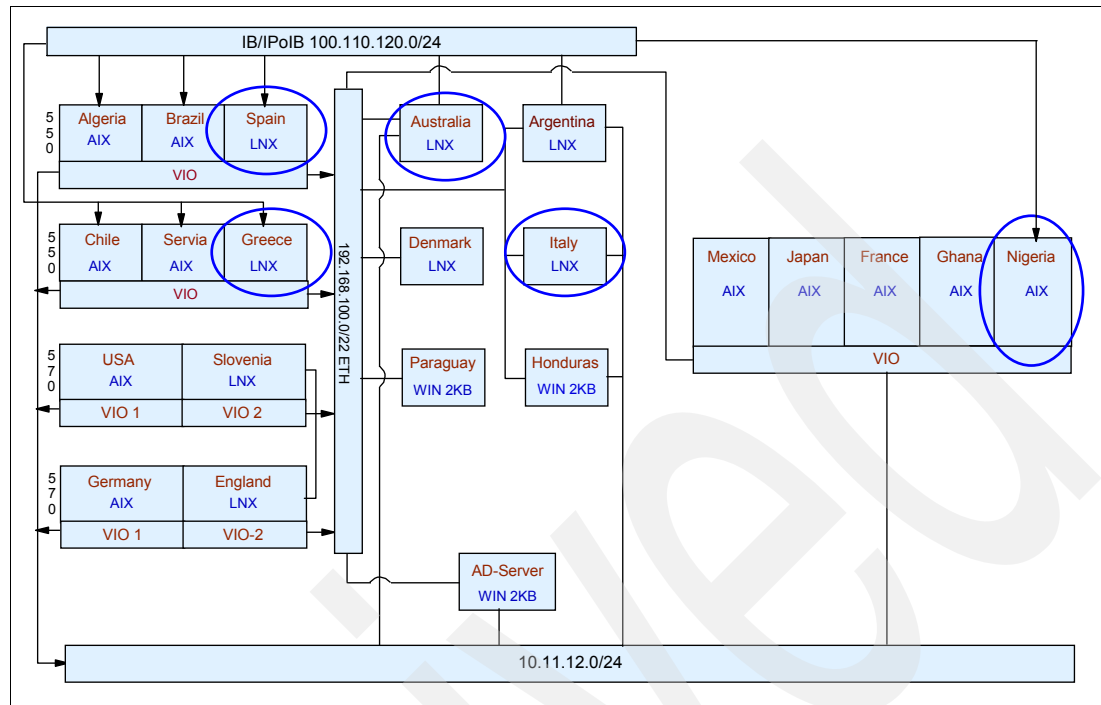


Figure 8-6 Selected servers from hardware environment

For details, best practices, and advanced configuration, see the Tivoli Storage Manager product manuals and other related books, such as *Tivoli Storage Manager V6.1 Technical Guide*, SG24-7718.

8.2.1 Preparing the Tivoli Storage Manager server

In our environment, we install the Tivoli Storage Manager server on an System x system with the following resources:

- ▶ Two CPUs Quad Core at 2.8 Ghz
- ▶ Eight GB RAM
- ▶ Four internal disks
- ▶ Four shared disks from a DS4xxx storage. These disks are shared with denmark, argentina and australia System x servers with similar hardware infrastructure.
- ▶ Three Fibre Channel adapters. Two adapters are used for shared disks access and each is zoned with one DS4xxx controller, so there are two path available for each shared disk. The third adapter is zoned with the tape library.
- ▶ In our laboratory, we use a TS3200 library with two drivers.
- ▶ Operating system is Red Hat Enterprise Linux 5 Update 5 for x86-64 bit processor with basic software and development packages.
- ▶ Host name is `italy` (for the primary network), and `italy-gpfs` (for the secondary network).

Note: All commands are run as *root* user, unless otherwise specified

To install the tape library drivers, perform the following steps:

1. Download the tape drivers for TS3200 for Red Hat 64-bit from the following location:

<http://www-933.ibm.com/support/fixcentral/>

Specify the tape you are using and the operating system. Save the file into a working directory.

2. Run the following command:

```
#rpmbuild --rebuild lin_tape-1.38.0-1.src.rpm
```

3. Install the rpm file by running the following command:

```
#rpm -i /usr/src/redhat/RPMS/x86_64/lin_tape-1.38.0-1.x86_64.rpm
```

4. Determine whether the drivers are operational, as shown in Example 8-4.

Example 8-4 The lin_tape kernel module

```
root@italy ts3200]# lsmod |grep lin
lin_tape          301336  0
scsi_mod          196953  11
lin_tape,scsi_dh_rdac,scsi_dh,st,sg,qla2xxx,scsi_transport_fc,libata,aic79xx,sc
si_transport_spi,sd_mod
```

5. Install the tape library commands package by running the following command:

```
#rpm -i lin_taped-1.38.0-rhel5.x86_64.rpm
```

6. Verify that the drivers are starting up at the boot time as shown in Example 8-5.

Example 8-5 Tape driver startup options

```
root@italy ts3200]# chkconfig lin_tape --list
lin_tape          0:off  1:off  2:on   3: on   4:off  5: on   6:off
```

7. Set up the multipath configuration, as shown in Example 8-6.

Example 8-6 Tape driver multipath configuration

```
root@italy ts3200]# lin_taped stop
root@italy ts3200]# rmmod lin_tape
root@italy ts3200]# echo "options lin_tape alternate_pathing=1" >> \
/etc/modprobe.conf
root@italy ts3200]# depmod
root@italy ts3200]# modprobe lin_tape
root@italy ts3200]# lin_taped start
[root@italy ts3200]# cat /proc/scsi/IBMchanger
lin_tape version: 1.38.0
lin_tape major number: 253
Attached Changer Devices:
Number  model      SN              HBA              F0 Path
0       3573-TL    00L4U78C9235_LL0 qla2xxx          Primary
[root@italy ts3200]# cat /proc/scsi/IBMtape
lin_tape version: 1.38.0
lin_tape major number: 253
Attached Tape Devices:
Number  model      SN              HBA              F0 Path
0       ULT3580-TD4 1310025521      qla2xxx          Primary
1       ULT3580-TD4 1310025518      qla2xxx          Primary
```

8.2.2 Tivoli Storage Manager installation

This section describes how to install Tivoli Storage Manager Version 6.2 for Linux 64-bit by using the wizard and basic settings. For setting up Tivoli Storage Manager environment that is ready for production, use the Tivoli Storage Manager manuals or web documentation:

<http://publib.boulder.ibm.com/infocenter/tsminfo/v6r2/index.jsp>

To install Tivoli Storage Manager, perform the following steps:

1. In a graphical environment, go to the Tivoli Storage Manager code root directory by running the `./install.bin` command.

The welcome window opens (Figure 8-7).



Figure 8-7 Installation wizard

2. Click **OK**.
3. Accept the license and click **Next**.
4. Select the components to install, as shown in Figure 8-8 on page 333, and click **Next**.



Figure 8-8 Select the components

5. Review the pre-installation Summary and click Next.
6. When the installation is finished click **Done**.

Disk configuration

To configure the trajectories and logical volume for Tivoli Storage Manager use the following steps. (We use Linux Logical Volume Manager to create a separate Volume Group: tsmvg)

1. Create a directory structure according to the steps in the following website, and see Example 8-7:

http://publib.boulder.ibm.com/infocenter/tsminfo/v6/index.jsp?topic=/com.ibm.it.sm.srv.install.doc/t_srv_inst_create_dirsid.html

Example 8-7 Directory structure

```
[root@italy ~]# mkdir /tsm
[root@italy ~]# mkdir /tsm/tsminst1
[root@italy ~]# mkdir /tsm/tsmdb001
[root@italy ~]# mkdir /tsm/tsmdb002
[root@italy ~]# mkdir /tsm/tsmdb003
[root@italy ~]# mkdir /tsm/tsmdb004
[root@italy ~]# mkdir /tsm/tsmlog
[root@italy ~]# mkdir /tsm/tsmlogmirror
[root@italy ~]# mkdir /tsm/tsmarchlog
[root@italy ~]# mkdir /tsm/home/tsminst1 -p
[root@italy ~]# lvs
LV          VG          Attr   LSize   Origin Snap%   Move Log Copy%  Convert
LogVol100   VolGroup00 -wi-ao 264.62G
LogVol101   VolGroup00 -wi-ao 14.66G
tsm_stg1_lv  tsmvg       -wi-ao 50.00G
```

| | | | |
|---------------|-------|--------|--------|
| tsmarchlog_lv | tsmvg | -wi-ao | 50.00G |
| tsmdb001_lv | tsmvg | -wi-ao | 50.00G |
| tsmdb002_lv | tsmvg | -wi-ao | 50.00G |
| tsmdb003_lv | tsmvg | -wi-ao | 50.00G |
| tsmdb004_lv | tsmvg | -wi-ao | 50.00G |
| tsmlog_lv | tsmvg | -wi-ao | 35.00G |

2. Create separate logical volumes for each directory, as shown in Example 8-8.

Note: Because our case uses three physical internal disks, we use the **-i 3 -I 64** parameters.

Example 8-8 Logical volumes

```
[root@italy ~]# lvcreate -A y -i 3 -I 32 -L 50G -n tsm_stg1_lv /dev/tsmvg
[root@italy ~]# lvcreate -A y -i 3 -I 64 -L 50G -n tsmdb001_lv /dev/tsmvg
[root@italy ~]# lvcreate -A y -i 3 -I 64 -L 50G -n tsmdb002_lv /dev/tsmvg
[root@italy ~]# lvcreate -A y -i 3 -I 64 -L 50G -n tsmdb003_lv /dev/tsmvg
[root@italy ~]# lvcreate -A y -i 3 -I 64 -L 50G -n tsmdb004_lv /dev/tsmvg
[root@italy ~]# lvcreate -A y -i 3 -I 64 -L 35G -n tsmlog_lv /dev/tsmvg
[root@italy ~]# lvcreate -A y -i 3 -I 64 -L 50G -n tsmarchlog_lv /dev/tsmvg
[root@italy ~]# lvs
```

| LV | VG | Attr | LSize | Origin | Snap% | Move | Log | Copy% | Convert |
|---------------|------------|--------|---------|--------|-------|------|-----|-------|---------|
| LogVol100 | VolGroup00 | -wi-ao | 264.62G | | | | | | |
| LogVol101 | VolGroup00 | -wi-ao | 14.66G | | | | | | |
| tsm_stg1_lv | tsmvg | -wi-ao | 50.00G | | | | | | |
| tsmarchlog_lv | tsmvg | -wi-ao | 50.00G | | | | | | |
| tsmdb001_lv | tsmvg | -wi-ao | 50.00G | | | | | | |
| tsmdb002_lv | tsmvg | -wi-ao | 50.00G | | | | | | |
| tsmdb003_lv | tsmvg | -wi-ao | 50.00G | | | | | | |
| tsmdb004_lv | tsmvg | -wi-ao | 50.00G | | | | | | |
| tsmlog_lv | tsmvg | -wi-ao | 35.00G | | | | | | |

3. Format the volumes by using the **mkfs.ext3** command and add the volumes in the **/etc/fstab** file, as shown in Example 8-9.

Example 8-9 The /etc/fstab file

```
[root@italy ~]# cat /etc/fstab
```

| | | | | |
|---------------------------|-----------------|--------|----------------|-----|
| /dev/VolGroup00/LogVol100 | / | ext3 | defaults | 1 1 |
| LABEL=/boot | /boot | ext3 | defaults | 1 2 |
| tmpfs | /dev/shm | tmpfs | defaults | 0 0 |
| devpts | /dev/pts | devpts | gid=5,mode=620 | 0 0 |
| sysfs | /sys | sysfs | defaults | 0 0 |
| proc | /proc | proc | defaults | 0 0 |
| /dev/VolGroup00/LogVol101 | swap | swap | defaults | 0 0 |
| /dev/tsmvg/tsm_stg1_lv | /tsm | ext3 | defaults | 0 0 |
| /dev/tsmvg/tsmdb001_lv | /tsm/tsmdb001 | ext3 | defaults | 0 0 |
| /dev/tsmvg/tsmdb002_lv | /tsm/tsmdb002 | ext3 | defaults | 0 0 |
| /dev/tsmvg/tsmdb003_lv | /tsm/tsmdb003 | ext3 | defaults | 0 0 |
| /dev/tsmvg/tsmdb004_lv | /tsm/tsmdb004 | ext3 | defaults | 0 0 |
| /dev/tsmvg/tsmlog_lv | /tsm/tsmlog | ext3 | defaults | 0 0 |
| /dev/tsmvg/tsmarchlog_lv | /tsm/tsmarchlog | ext3 | defaults | 0 0 |

4. Create a Tivoli Storage Manager user, as shown in Example 8-10 on page 335. The password is *itsoadmin*.

Example 8-10 Adding tsminst1 user

```
[root@italy ~]# groupadd tsmsrvrs
[root@italy ~]# adduser -d /tsm/home/tsminst1 -m -g tsmsrvrs -s /bin/bash
tsminst1
[root@italy ~]# passwd tsminst1
```

Tivoli Storage Manager Configuration tool

To finish the configuration of Tivoli Storage Manager, perform the following steps:

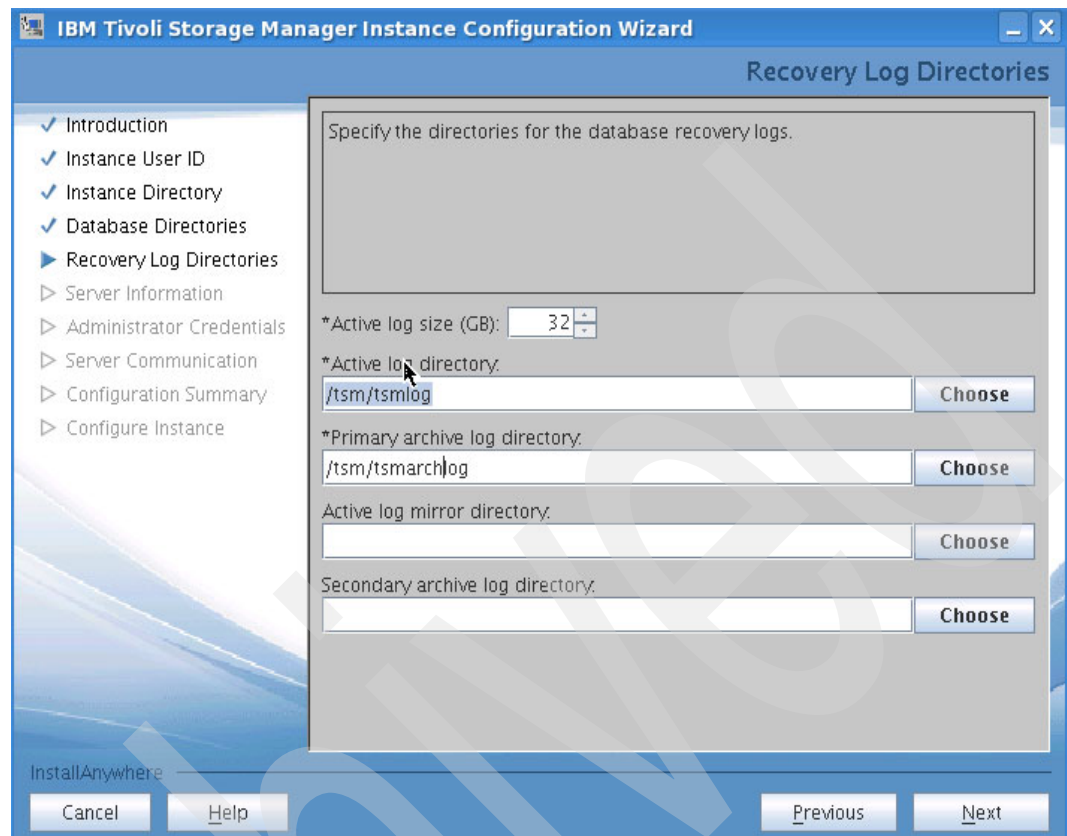
1. Log on as the tsminst1 user.
2. In a graphical environment, run the **dsmincfgx** program from `/opt/tivoli/tsm/server/bin` directory.
3. Select a language, in our case English, and click **OK**, as shown in Figure 8-9.



Figure 8-9 Language settings

4. Click **Next** on welcome window.
5. Specify the user tsminst1 and the password itsoadmin and click **Next**.
6. Specify the `/tsm/home/tsminst1/tsminst1/` directory and click **Next**.
7. Specify the following database directories and click **Next**.
 - `/tsm/tsmdb001`
 - `/tsm/tsmdb002`
 - `/tsm/tsmdb003`
 - `/tsm/tsmdb004`

8. Specify the log directories as shown in Figure 8-10 and click **Next**.



The screenshot shows the 'Recovery Log Directories' step of the IBM Tivoli Storage Manager Instance Configuration Wizard. The left sidebar lists the configuration steps: Introduction, Instance User ID, Instance Directory, Database Directories, Recovery Log Directories (selected), Server Information, Administrator Credentials, Server Communication, Configuration Summary, and Configure Instance. The main area contains a text box for specifying directories, followed by fields for: *Active log size (GB) set to 32, *Active log directory set to /tsm/tsmllog, *Primary archive log directory set to /tsm/tsmarchlog, Active log mirror directory, and Secondary archive log directory. Each directory field has a 'Choose' button. At the bottom are 'Cancel', 'Help', 'Previous', and 'Next' buttons.

IBM Tivoli Storage Manager Instance Configuration Wizard

Recovery Log Directories

Specify the directories for the database recovery logs.

✓ Introduction
✓ Instance User ID
✓ Instance Directory
✓ Database Directories
▶ Recovery Log Directories
▶ Server Information
▶ Administrator Credentials
▶ Server Communication
▶ Configuration Summary
▶ Configure Instance

*Active log size (GB): 32

*Active log directory: /tsm/tsmllog Choose

*Primary archive log directory: /tsm/tsmarchlog Choose

Active log mirror directory: Choose

Secondary archive log directory: Choose

InstallAnywhere

Cancel Help Previous Next

Figure 8-10 Recovery Log Directories

9. Specify the server name and starting option, as shown in Figure 8-11.

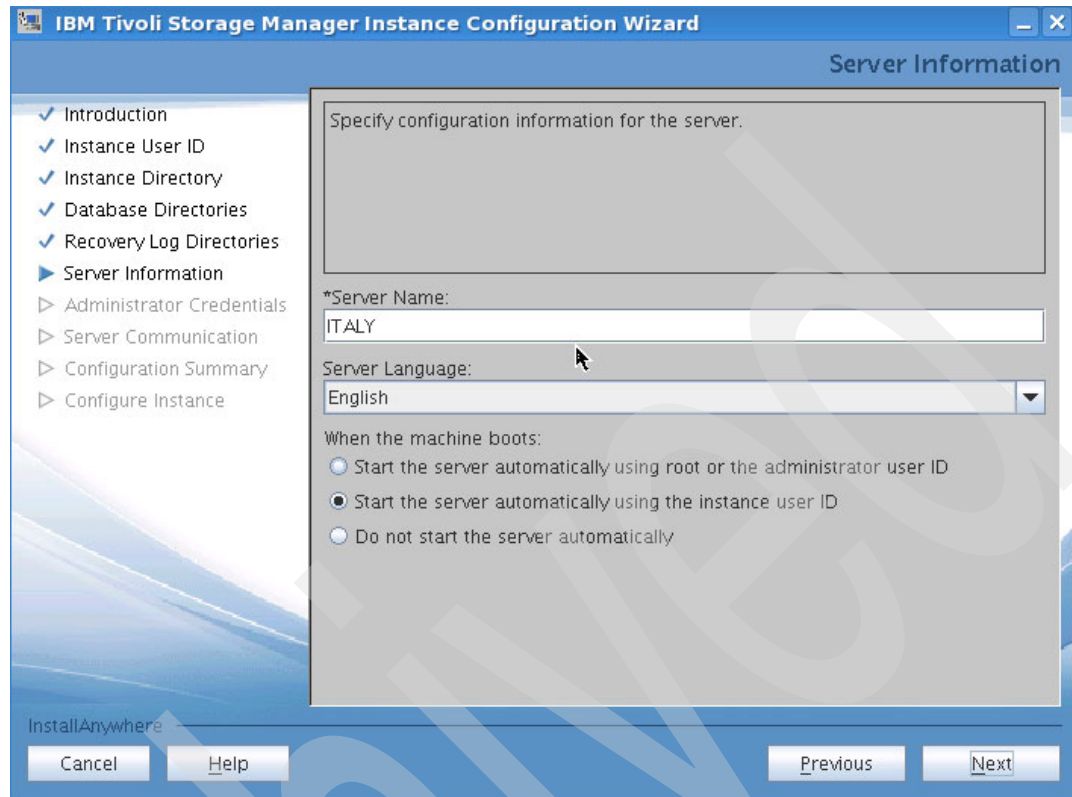


Figure 8-11 Server information

10. Specify user `tsmadmin` and password `its0admin` (`its<zero>admin`) as administrator user, and click **Next**.

11. Specify the client and administration ports as shown in Figure 8-12, and click **Next**.



Figure 8-12 Server communication

12. Review the settings, as shown in Figure 8-13, and click **Next** to start the configuration process.



Figure 8-13 Configuration summary

8.2.3 Administrator Center installation

The administration portal can be installed on any other node, but in our laboratory, we installed it in the same server.

To install the Tivoli Storage Manager administrator portal, perform the following steps:

1. In a graphical environment, go to the code location and run the `./install.bin` command.
2. In the welcome window, select a language and click Next.

The next window is a warning about root user installation (Figure 8-14 on page 340). Because we selected the simpler case, we ignored this message. Click **OK**, then Click **Next**.

If integration with other Tivoli components is a requirement, follow the instructions in the Tivoli Storage Manager manuals or the following web address:

<http://publib.boulder.ibm.com/infocenter/tsminfo/v6r2/index.jsp>

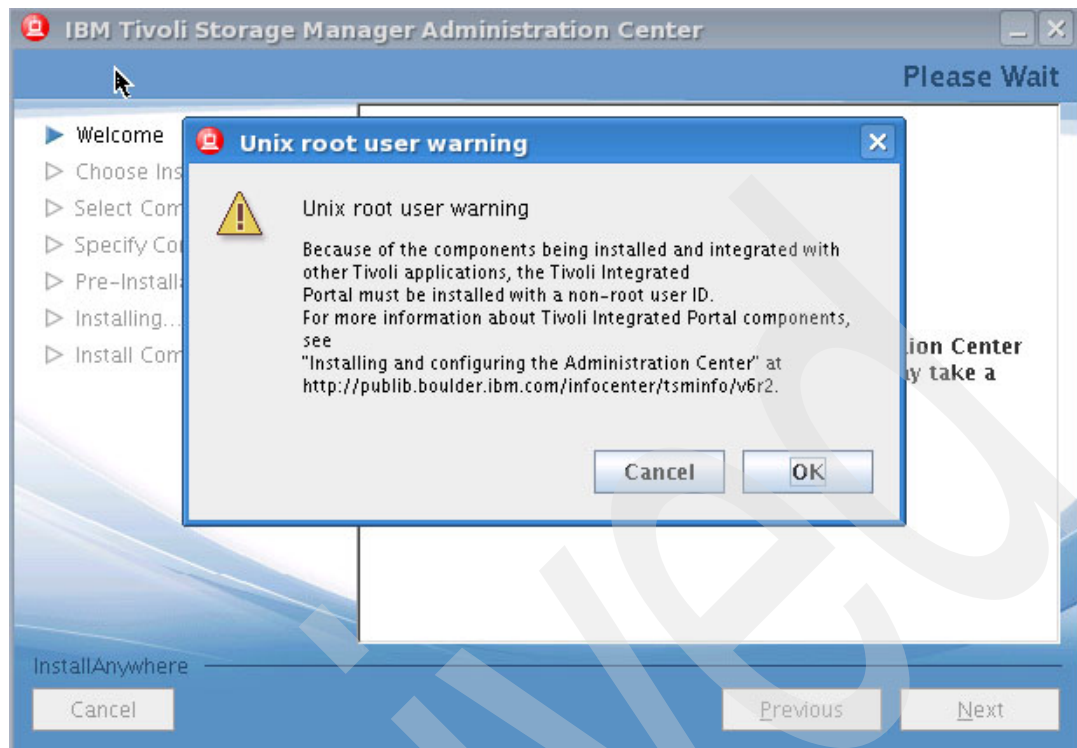


Figure 8-14 Non-root installation warning

3. Set up the `/tsm/ibm/ac` path as a new installation directory. Click **Next**.
4. Select the advanced installation and click **Next**.
5. Click **Yes, install authentication service**, and click **Next**.
6. Do *not* select any other external authentication system, as shown in Figure 8-15 on page 341. Click **Next**.

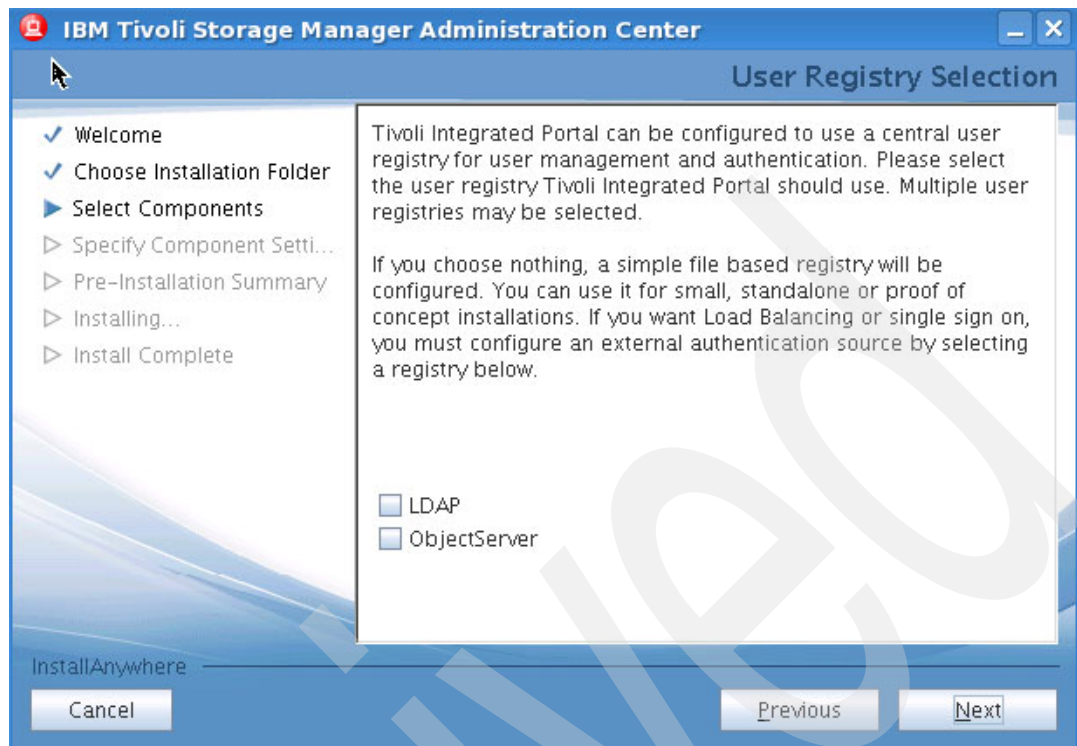


Figure 8-15 No external authentication system

7. Provide an administrative user ID (Figure 8-16) and click **Next**.



Figure 8-16 Password for tipadmin is its0admin (with a zero)

8. Review the summary installation and click **Next**.
9. At the installation-complete window, click **Done** to exit the graphical installation.
10. Log in on the administrative interface on port 16310.
11. Add the Tivoli Storage Manager server by specifying the following information:
 - Server name
 - Port
 - User name
 - Password

After the registration the portal looks similar to Figure 8-17.

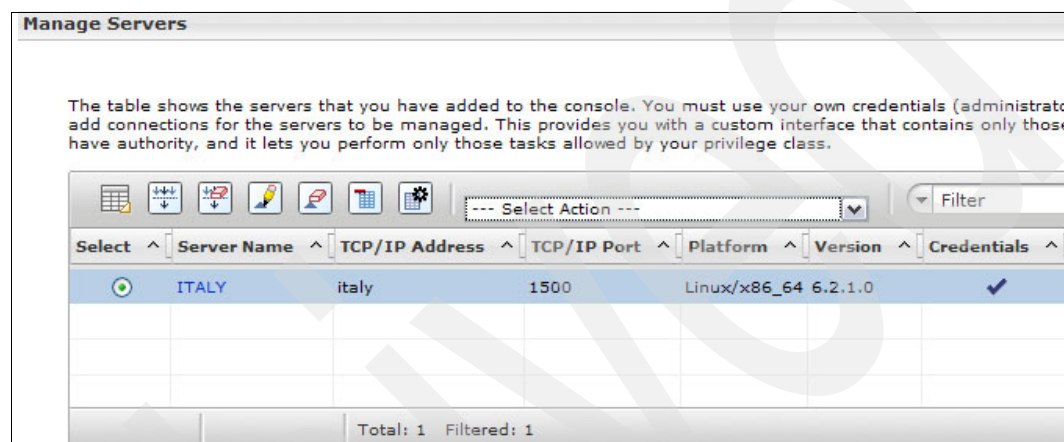


Figure 8-17 Tivoli Storage Manager registration into administrative portal

8.2.4 Tivoli Storage Manager server configuration

Now, define a disk storage pool and tape storage pool as the next storage pool, as shown in Example 8-11.

Example 8-11 Storage pool configuration

```
tsm: ITALY>q vol f=d
```

```

Volume Name: /tsm/home/tsminst1/tsminst1/ITS0-DISK
Storage Pool Name: ITS0-STG
Device Class Name: DISK
Estimated Capacity: 10.2 G
Scaled Capacity Applied:
Pct Util: 0.0
Volume Status: On-Line
Access: Read/Write

```

.....snipped.....

```
tsm: ITALY>q stgpool
```

| Storage Pool Name | Device Class Name | Estimated Capacity | Pct Util | Pct Migr | High Mig Pct | Low Mig Pct | Next Storage Pool |
|-------------------|-------------------|--------------------|----------|----------|--------------|-------------|-------------------|
| ARCHIVEPOOL | DISK | 0.0 M | 0.0 | 0.0 | 90 | 70 | |
| BACKUPPOOL | DISK | 0.0 M | 0.0 | 0.0 | 90 | 70 | |
| ITS0-STG | DISK | 10 G | 0.0 | 0.0 | 90 | 70 | ITS0-STG-LIB |
| ITS0-STG-LIB | LTO_CLASS_1 | 0.0 M | 0.0 | 90 | 70 | | |
| SPACEPOOL | DISK | 0.0 M | 0.0 | 0.0 | 90 | 70 | |

Next, we define the nodes argentina and australia, shown in Example 8-12.

Example 8-12 Node definitions

tsm: ITALY>q node

| Node Name | Platform | Policy Domain Name | Days Since Last Access | Days Since Password Set | Locked? |
|-----------|----------|--------------------|------------------------|-------------------------|---------|
| ARGENTINA | (?) | STANDARD | <1 | <1 | No |
| AUSTRALIA | (?) | STANDARD | <1 | <1 | No |
| ITALY | Linux86 | STANDARD | <1 | <1 | No |

The tape library definition and the tape media are shown in Example 8-13.

Example 8-13 Tape library definition

tsm: ITALY>q drive

| Library Name | Drive Name | Device Type | On-Line |
|--------------|----------------|-------------|---------|
| ITS0-LIB | ITS0-LIB-DR-V0 | LTO | Yes |
| ITS0-LIB | ITS0-LIB-DR-V1 | LTO | Yes |

tsm: ITALY>q libvol

| Library Name Device | Volume Name | Status | Owner | Last Use | Home Element |
|------------------------|-------------|---------|-------|----------|-----------------|
| Type | | | | | |
| ITS0-LIB LTO | 568AAAL4 | Scratch | | | 4,099 |
| ITS0-LIB LTO | 569AAAL4 | Scratch | | | 4,096 |
| ITS0-LIB LTO | 748AAFL4 | Scratch | | | 4,098 |
| ITS0-LIB LTO | 749AAFL4 | Scratch | | | 4,097 |

Note: To use self-client registration, clear the **Require Logon Password** option from the security page in the Tivoli Storage Manager administrative portal.

8.2.5 Tivoli Storage Manager client configuration

The following steps show the Tivoli Storage Manager client installation and configuration for the nodes australia, argentina, and italy as shown in Figure 8-5 on page 329. The installation is similar to all nodes.

1. On the argentina node, install the Tivoli Storage Manager client software, as shown in Example 8-14.

Example 8-14 Tivoli Storage Manager node installation

```
[root@argentina ~]# cd /kits/tsm/TSMCLI_LNX/tsmcli/linux86
[root@argentina linux86]# rpm -ivh gskcrypt32-8.0.13.3.linux.x86.rpm
gskcrypt64-8.0.13.3.linux.x86_64.rpm gskssl32-8.0.13.3.linux.x86.rpm
gskssl64-8.0.13.3.linux.x86_64.rpm TIVsm-API64.i386.rpm TIVsm-API.i386.rpm
TIVsm-BA.i386.rpm
Preparing... ##### [100%]
 1:gskcrypt64 ##### [ 14%]
 2:gskcrypt32 ##### [ 29%]
 3:gskssl32 ##### [ 43%]
 4:TIVsm-API ##### [ 57%]
Postinstall of the API

TSM Linux API installation complete.

Be sure to set up the configuration files!

 5:gskssl64 ##### [ 71%]
 6:TIVsm-API64 ##### [ 86%]
Postinstall of the API64

TSM Linux API64 installation complete.

Be sure to set up the configuration files!

 7:TIVsm-BA ##### [100%]
Postinstall of the Backup Archive client

TSM Linux client installation complete.

Be sure to set up the system configuration file
before starting the client!
[root@argentina linux86]#
```

2. Modify the /opt/tivoli/tsm/client/ba/bin/dsm.sys file, as shown in Example 7-15.

Example 8-15 The dsm.sys file

```
SErvername italy
COMMMethod TCPip
TCPPort 1500
TCPServeraddress italy
nodename argentina
passwordaccess generate
```

3. Test the Tivoli Storage Manager client backup function to verify that the backup process is working in this phase. We backup the /var/lib directory, as shown in Example 7-16.

Example 8-16 Testing the backup function

```
dsmc backup -quiet -su=yes /var/lib/
IBM Tivoli Storage Manager
Command Line Backup-Archive Client Interface
  Client Version 6, Release 2, Level 1.0
  Client date/time: 06/21/2010 08:46:04
(c) Copyright by IBM Corporation and other(s) 1990, 2010. All Rights Reserved.

Node Name: ARGENTINA
Session established with server ITALY: Linux/x86_64
  Server Version 6, Release 2, Level 1.0
  Server date/time: 06/22/2010 13:34:14 Last access: 06/22/2010 13:33:57
```

```
Total number of objects inspected:      342
Total number of objects backed up:      342
Total number of objects updated:         0
Total number of objects rebound:        0
Total number of objects deleted:         0
Total number of objects expired:         0
Total number of objects failed:          0
Total number of bytes inspected:        65.24 MB
Total number of bytes transferred:      65.04 MB
Data transfer time:                     2.16 sec
Network data transfer rate:             30,810.55 KB/sec
Aggregate data transfer rate:           19,314.04 KB/sec
Objects compressed by:                  0%
Total data reduction ratio:              0.32%
Elapsed processing time:                 00:00:03
```

4. Test the Tivoli Storage Manager client **restore** function to verify that the restore process is working in this phase. We restore the /var/lib/ directory in /tmp/var location as shown in Example 7-17.

Example 8-17 Tivoli Storage Manager client restore function

```
dsmc restore -quiet -su=yes /var/lib/ /tmp/var/
IBM Tivoli Storage Manager
Command Line Backup-Archive Client Interface
  Client Version 6, Release 2, Level 1.0
  Client date/time: 06/21/2010 08:46:48
(c) Copyright by IBM Corporation and other(s) 1990, 2010. All Rights Reserved.

Node Name: ARGENTINA
Session established with server ITALY: Linux/x86_64
  Server Version 6, Release 2, Level 1.0
  Server date/time: 06/22/2010 13:34:58 Last access: 06/22/2010 13:34:14
```

ANS1247I Waiting for files from the server...

```
Total number of objects restored:      342
Total number of objects failed:         0
Total number of bytes transferred:     65.02 MB
```

| | |
|-------------------------------|-------------------|
| Data transfer time: | 0.31 sec |
| Network data transfer rate: | 210,932.71 KB/sec |
| Aggregate data transfer rate: | 13,009.57 KB/sec |
| Elapsed processing time: | 00:00:05 |

Note: We add the Linux PowerPC nodes: Spain, Greece and Nigeria by using the procedure shown in 8.2.5, “Tivoli Storage Manager client configuration” on page 344

8.3 Information lifecycle management scenarios

Using GPFS ILM functions, we define several scenarios starting from basic and gradually to more complex ones. For this purpose we use the nodes: spain, greece and nigeria, which already form a cluster using InfiniBand network, although is not relevant for this chapter. For more information about creating a GPFS using InfiniBand cluster, see 3.4, “Linux InfiniBand cluster with RDMA and Linux for System x clients” on page 90

The current cluster configuration is shown in Example 8-18.

Example 8-18 GPFS cluster

```
[root@spain ~]# mmlscluster

GPFS cluster information
=====
GPFS cluster name:      GPFS-InfiniBand.spain-gpfs
GPFS cluster id:       723685802921743777
GPFS UID domain:      GPFS-InfiniBand.spain-gpfs
Remote shell command:  /usr/bin/ssh
Remote file copy command: /usr/bin/scp

GPFS cluster configuration servers:
-----
Primary server:  spain-gpfs
Secondary server: greece-gpfs
```

| Node | Daemon node name | IP address | Admin node name | Designation |
|------|------------------|--------------|-----------------|----------------|
| 1 | spain-gpfs | 10.11.12.103 | spain-gpfs | quorum-manager |
| 2 | greece-gpfs | 10.11.12.113 | greece-gpfs | quorum-manager |
| 3 | nigeria-gpfs | 10.11.12.225 | nigeria-gpfs | |

```
[root@spain ~]# mmlsconfig
Configuration data for cluster GPFS-InfiniBand.spain-gpfs:
-----
clusterName GPFS-InfiniBand.spain-gpfs
clusterId 723685802921743777
autoload no
minReleaseLevel 3.4.0.0
dmapifileHandleSize 32
verbsPorts ehca0/1
verbsRdma enable
adminMode central

File systems in cluster GPFS-InfiniBand.spain-gpfs:
```

```
-----
/dev/gpfs-ib
[root@spain ~]# mmlnsd -L
```

| File system | Disk name | NSD volume ID | NSD servers |
|-------------|-----------|------------------|------------------------|
| gpfs-ib | gpfs1nsd | 6567C0A84C24E60E | spain-gpfs,greece-gpfs |
| gpfs-ib | gpfs2nsd | 6571C0A84C24E608 | greece-gpfs,spain-gpfs |

```
[root@spain ~]# mmlnsd -m
```

| Disk name | NSD volume ID | Device | Node name | Remarks |
|-----------|------------------|----------|-------------|-------------|
| gpfs1nsd | 6567C0A84C24E60E | /dev/sdb | greece-gpfs | server node |
| gpfs1nsd | 6567C0A84C24E60E | /dev/sdb | spain-gpfs | server node |
| gpfs2nsd | 6571C0A84C24E608 | /dev/sdc | greece-gpfs | server node |
| gpfs2nsd | 6571C0A84C24E608 | /dev/sdc | spain-gpfs | server node |

8.3.1 Working with snapshots

This section presents the following tasks for working with snapshots:

- ▶ Generate data on GPFS file system.
- ▶ Create several snapshots.
- ▶ Delete data from the file system and notice the space allocation when using snapshots.
- ▶ Back up the data from a snapshot.
- ▶ Delete a snapshot.

The following steps show how to work with snapshots:

1. Create a file by using the following command, which generates data on the file system:

```
dd if=/dev/urandom of=/gpfs-ib/gogo bs=1M
```

Example 8-19 shows the GPFS file system utilization after the file is created.

Example 8-19 The mmdf command

```
root@spain ~]# mmdf gpfs-ib
```

| disk name | disk size in KB | failure group | holds metadata | holds data | free KB in full blocks | free KB in fragments |
|---|-----------------|---------------|----------------|------------|------------------------|----------------------|
| Disks in storage pool: system (Maximum disk size allowed is 219 GB) | | | | | | |
| gpfs1nsd | 26214400 | 1 | yes | yes | 24977664 (95%) | 536 (0%) |
| gpfs2nsd | 26214400 | 2 | yes | yes | 24974080 (95%) | 592 (0%) |
| (pool total) | 52428800 | | | | 49951744 (95%) | 1128 (0%) |
| ===== | | | | | | |
| (total) | 52428800 | | | | 49951744 (95%) | 1128 (0%) |

Inode Information

```
-----
```

| | |
|-----------------------------|-------|
| Number of used inodes: | 4040 |
| Number of free inodes: | 48184 |
| Number of allocated inodes: | 52224 |
| Maximum number of inodes: | 52224 |

2. Create a snapshot by using the **mmcrsnapshot gpfs-ib snap1** command.
3. Delete the file by running the **rm /gpfs-ib/gogo -f** command.
4. The file system space usage is not changed, although the file is deleted because the information is still in snapshot, as shown in Example 8-20.

Notice that the used inodes number has decreased automatically because it is required by the snapshot procedure.

Example 8-20 The mmdf command after the file /gpfs-ib/gogo is removed

```
[root@spain ~]# mmdf gpfs-ib
```

| disk name | disk size in KB | failure holds group metadata | holds data | free KB in full blocks | free KB in fragments |
|---|--------------------|---------------------------------|---------------|---------------------------|-------------------------|
| ----- | | | | | |
| Disks in storage pool: system (Maximum disk size allowed is 219 GB) | | | | | |
| gpfs1nsd | 26214400 | 1 yes | yes | 24977408 (95%) | 392 (0%) |
| gpfs2nsd | 26214400 | 2 yes | yes | 24973824 (95%) | 464 (0%) |
| ----- | | | | | |
| (pool total) | 52428800 | | | 49951232 (95%) | 856 (0%) |
| ===== | | | | | |
| (total) | 52428800 | | | 49951232 (95%) | 856 (0%) |


```
Inode Information
```

| | |
|-----------------------------|-------------|
| Number of used inodes: | 4039 |
| Number of free inodes: | 48185 |
| Number of allocated inodes: | 52224 |
| Maximum number of inodes: | 52224 |

The deleted file is still available in the snapshot *snap1*, because it was available in the file system at the creation of the snapshot. The snapshot data is still saved on the file system and you can determine the space usage, as shown in Example 8-21.

Example 8-21 Snapshot space utilization

```
[root@spain ~]# mmlssnapshot gpfs-ib -d
```

Snapshots in file system gpfs-ib: [data and metadata in KB]

| Directory | SnapId | Status | Created | Data | Metadata |
|-----------|--------|--------|-------------------------|----------------|------------|
| snap1 | 2 | Valid | Tue Jul 6 13:28:55 2010 | 2424864 | 736 |

5. Create a second snapshot by using the **mmcrsnapshot gpfs-ib snap2** command.

The file *gogo* is present only in the *snap1* snapshot because it was deleted from the file system, as shown in Example 8-22.

Example 8-22 The snap1 file listing

```
[root@spain ~]# ll /gpfs-ib/
total 40
drwxr-xr-x 7 root root 8192 Jun 29 17:31 user1
drwxr-xr-x 7 root root 8192 Jun 29 17:31 user2
drwxr-xr-x 7 root root 8192 Jun 29 17:31 user3
drwxr-xr-x 7 root root 8192 Jun 29 17:31 user4
drwxr-xr-x 7 root root 8192 Jun 29 17:31 user5
[root@spain ~]# ll /gpfs-ib/.snapshots/snap1/
total 2424872
-rw-r--r-- 1 root root 2483027968 Jul 6 13:27 gogo
```

```

drwxr-xr-x 7 root root      8192 Jun 29 17:31 user1
drwxr-xr-x 7 root root      8192 Jun 29 17:31 user2
drwxr-xr-x 7 root root      8192 Jun 29 17:31 user3
drwxr-xr-x 7 root root      8192 Jun 29 17:31 user4
drwxr-xr-x 7 root root      8192 Jun 29 17:31 user5
[root@spain ~]# ll /gpfs-ib/.snapshots/snap2/
total 40
drwxr-xr-x 7 root root 8192 Jun 29 17:31 user1
drwxr-xr-x 7 root root 8192 Jun 29 17:31 user2
drwxr-xr-x 7 root root 8192 Jun 29 17:31 user3
drwxr-xr-x 7 root root 8192 Jun 29 17:31 user4
drwxr-xr-x 7 root root 8192 Jun 29 17:31 user5

```

At this point, the file system space usage is still at 95%, as shown in Example 8-20 on page 348.

6. Backup the *snap1* snapshot by using the Tivoli Storage Manager client as shown in Example 8-23.

Note: Use one Tivoli Storage Manager *nodename* for all servers that are part of the same GPFS cluster. This requirement is so that Tivoli Storage Manager can know that the file system is the same, regardless which GPFS node is used for backup.

From now on, use the *gpfs-ib* nodename in the following file for all GOFS cluster nodes:

```
/opt/tivoli/tsm/client/ba/bin/dsm.sys
```

For local files backup, a separate Tivoli Storage Manager *dsm.sys* file must be used. Use the variable *DSM_DIR* or *DSM_CONFIG* to specify which *dsm.sys* file to use every time the backup is started.

For more information, read the Tivoli Storage Manager manuals or the following resource:

<http://publib.boulder.ibm.com/infocenter/tsminfo/v6r2/index.jsp>

Example 8-23 The *snap1* snapshot backup

```

[root@spain bin]# dsmc backup -quiet -su=yes /gpfs-ib/.snapshots/snap1/gogo
ANS0990W Options file '/opt/tivoli/tsm/client/ba/bin/dsm.opt' could not be
found. Default option values will be used.
IBM Tivoli Storage Manager
Command Line Backup-Archive Client Interface
  Client Version 6, Release 2, Level 1.0
  Client date/time: 07/06/2010 15:25:14
(c) Copyright by IBM Corporation and other(s) 1990, 2010. All Rights Reserved.

Node Name: GPFS-IB
Session established with server ITALY: Linux/x86_64
  Server Version 6, Release 2, Level 1.0
  Server date/time: 07/06/2010 15:33:11  Last access: 07/06/2010 13:43:02

```

```

Total number of objects inspected:      63
Total number of objects backed up:      3
Total number of objects updated:        0
Total number of objects rebound:       0

```

```

Total number of objects deleted:      0
Total number of objects expired:      0
Total number of objects failed:       0
Total number of bytes inspected:      2.31 GB
Total number of bytes transferred:    2.31 GB
Data transfer time:                   76.33 sec
Network data transfer rate:           31,778.08 KB/sec
Aggregate data transfer rate:         29,602.20 KB/sec
Objects compressed by:                0%
Total data reduction ratio:           0.00%
Elapsed processing time:               00:01:21

```

7. Delete the snapshot *snap1* by using the `mmdel snapshot gpfs-ib snap1` command, and check the usage of both the file system and snapshot, as shown in Example 8-24.

Example 8-24 Delete snapshot

```

[root@spain ~]# mmdf gpfs-ib
disk          disk size  failure holds   holds          free KB          free KB
name          in KB     group metadata data          in full blocks  in fragments
-----
Disks in storage pool: system (Maximum disk size allowed is 219 GB)
gpfs1nsd      26214400      1 yes    yes    26190080 (100%)    616 ( 0%)
gpfs2nsd      26214400      2 yes    yes    26186496 (100%)    448 ( 0%)
-----
(pool total)   52428800                      52376576 (100%)    1064 ( 0%)
=====
(total)        52428800                      52376576 (100%)    1064 ( 0%)

..... snipped .....

[root@spain ~]# mmlssnapshot gpfs-ib -d
Snapshots in file system gpfs-ib: [data and metadata in KB]
Directory      SnapId  Status  Created          Data  Metadata
snap2          3      Valid  Tue Jul  6 13:31:00 2010    0    256

```

8.3.2 Working with storage pools and policies

This chapter describes several tasks to perform while using storage pools and policies:

- ▶ Create a second storage pool.
- ▶ Work with default policy.
- ▶ Create new placement and migration policies.
- ▶ Use callback functions to apply policies on demand.

Create a user storage pool and placement policy

To create a user storage pool and placement policy, perform the following steps:

1. Add more NSD disks to the GPFS, as shown in Example 8-25, and define a user storage pool: `itso-stg-pool`

Example 8-25 Create NSD disks

```
[root@spain ~]# cat /work/nsd_stg_pool
/dev/sdd:spain-gpfs,greece-gpfs::dataOnly:1::itso-stg-pool:
/dev/sde:greece-gpfs,spain-gpfs::dataOnly:2::itso-stg-pool:
[root@spain ~]# mmcrnsd -F /work/nsd_stg_pool
mmcrnsd: Processing disk sdd
mmcrnsd: Processing disk sde
mmcrnsd: Propagating the cluster configuration data to all
        affected nodes. This is an asynchronous process.
[root@spain ~]# mmlsnsd
```

| File system | Disk name | NSD servers |
|-------------|-----------|------------------------|
| gpfs-ib | gpfs1nsd | spain-gpfs,greece-gpfs |
| gpfs-ib | gpfs2nsd | greece-gpfs,spain-gpfs |
| (free disk) | gpfs3nsd | spain-gpfs,greece-gpfs |
| (free disk) | gpfs4nsd | greece-gpfs,spain-gpfs |

2. Add the defined disk to the current file system online, shown in Example 8-26.

Example 8-26 Adding disk to the file system

```
[root@spain ~]# mmadddisk gpfs-ib -F /work/nsd_stg_pool

The following disks of gpfs-ib will be formatted on node spain:
gpfs3nsd: size 26214400 KB
gpfs4nsd: size 26214400 KB
Extending Allocation Map
Creating Allocation Map for storage pool 'itso-stg-pool'
Flushing Allocation Map for storage pool 'itso-stg-pool'
Disks up to size 219 GB can be added to storage pool 'itso-stg-pool'.
Checking Allocation Map for storage pool 'itso-stg-pool'
Completed adding disks to file system gpfs-ib.
mmadddisk: Propagating the cluster configuration data to all
        affected nodes. This is an asynchronous process.
- run mmfsrestripe command
NOTE: When added disk in the file we could also specify the -r to rebalance the
file system
```

Note: Running the `mmrestripefs` command in this phase does not rebalance the data between the disk belonging to the *system storage pool* and the *itso-stg-pool* storage pool because there is no policy defined that tells GPFS to write data in *itso-stg-pool* storage pool.

3. Create a file by using the following command to generate data on file system:
`dd if=/dev/zero of=/gpfs-ib/gogo1 bs=1M count=5000`

4. Check the space allocation on the GPFS file system, as shown in Example 8-27. As expected, all the data is on the system pool. Now, if the space on system pool becomes full, the GPFS will report a disk-full error, although the *itso-stg-pool* is empty.

Example 8-27 Generate data on file system

```

root@spain ~]# mmdf gpfs-ib
disk          disk size  failure holds   holds          free KB          free KB
name          in KB     group metadata data          in full blocks  in fragments
-----
Disks in storage pool: system (Maximum disk size allowed is 219 GB)
gpfs1nsd      26214400      1 yes    yes    23625984 ( 90%)    1344 ( 0%)
gpfs2nsd      26214400      2 yes    yes    23622144 ( 90%)    720 ( 0%)
-----
(pool total)   52428800                                47248128 ( 90%)    2064 ( 0%)

Disks in storage pool: itso-stg-pool (Maximum disk size allowed is 219 GB)
gpfs3nsd      26214400      1 no     yes    26212096 (100%)    248 ( 0%)
gpfs4nsd      26214400      2 no     yes    26212096 (100%)    248 ( 0%)
-----
(pool total)   52428800                                52424192 (100%)    496 ( 0%)

=====
(data)         104857600                                99672320 ( 95%)    2560 ( 0%)
(metadata)     52428800                                47248128 ( 90%)    2064 ( 0%)
=====
(total)        104857600                                99672320 ( 95%)    2560 ( 0%)

Inode Information
-----
..... snipped .....

```

5. Create a policy and save it in the `/work/policies.txt` file as shown in Example 8-28. This policy tells GPFS that all files with the `.bin` extension will go *itso-stg-pool*. Run the `mmchpolicy gpfs-ib /work/policies.txt` command to apply the policy.

Example 8-28 Create a policy file

```

[root@spain ~]# cat /work.policies.txt
/* Rule named "execute-files" places files in pool "itso-stg-pool", using
criteria: file extension ".bin" */
RULE 'execute-files' SET POOL 'itso-stg-pool' WHERE (name) like '%.bin'

```

6. Create a file using the following command:
`dd if=/dev/zero of=/gpfs-ib/gogo.bin bs=1M count=2000`
7. Check the space utilization, as shown in Example 8-29.

Example 8-29 New policy is applied

```

[root@spain ~]# mmdf gpfs-ib
disk          disk size  failure holds   holds          free KB          free KB
name          in KB     group metadata data          in full blocks  in fragments
-----
Disks in storage pool: system (Maximum disk size allowed is 219 GB)
gpfs1nsd      26214400      1 yes    yes    23625728 ( 90%)    1248 ( 0%)
gpfs2nsd      26214400      2 yes    yes    23622144 ( 90%)    624 ( 0%)
-----

```

| | | | | |
|--------------|----------|--|-----------------|------------|
| (pool total) | 52428800 | | 47247872 (90%) | 1872 (0%) |
|--------------|----------|--|-----------------|------------|

Disks in storage pool: itso-stg-pool (Maximum disk size allowed is 219 GB)

| | | | | | |
|----------|----------|------|-----|-----------------|-----------|
| gpfs3nsd | 26214400 | 1 no | yes | 25188096 (96%) | 248 (0%) |
| gpfs4nsd | 26214400 | 2 no | yes | 25188096 (96%) | 248 (0%) |

| | | | | |
|--------------|----------|--|-----------------|-----------|
| (pool total) | 52428800 | | 50376192 (96%) | 496 (0%) |
|--------------|----------|--|-----------------|-----------|

| | | | | |
|------------|-----------|--|-----------------|------------|
| (data) | 104857600 | | 97624064 (93%) | 2368 (0%) |
| (metadata) | 52428800 | | 47247872 (90%) | 1872 (0%) |

| | | | | |
|---------|-----------|--|-----------------|------------|
| (total) | 104857600 | | 97624064 (93%) | 2368 (0%) |
|---------|-----------|--|-----------------|------------|

..... snipped

8. Try to create other files as shown in Example 8-30. At this time, only one policy is enabled, which permits files only with the .bin extension on the file system.

Example 8-30 Create a file

```
root@spain ~]# touch /gpfs-ib/gogo.text
touch: setting times of `/gpfs-ib/gogo.text': No such file or directory
```

For all other files, there is no placement policy so GPFS does not allow the files on the file system. Add a general rule that tells GPFS where to write all other files, as shown in Example 8-31. Run the **mmchpolicy gpfs-ib /work/policies.txt** command to apply the new policies; creating files is now possible.

Example 8-31 Add a default policy

```
cat /work/policies.txt
/* Rule named "execute-files" places files in pool "itso-stg-pool", using
criteria: file extension ".bin" */
RULE 'execute-files' SET POOL 'itso-stg-pool' WHERE (name) like '%.bin'
/* NOTE: THIS IS MANDATORY, otherwise you may end up not being able to create
other files */
RULE 'default' set POOL 'system'
```

Note: The default rule may specify other pools in addition to system pools. This approach is useful when the system pool is used only for metadata and other pools are for data only.

Create a migration policy

To create a migration policy, perform the following steps:

1. The policy in Example 8-32 tells GPFS that all files starting with the characters big to be moved to the itso-stg-pool pool. Apply the policy by using the following command:

```
mmchpolicy gpfs-ib /work/policies.txt
```

Example 8-32 Create a migration policy

```
[root@spain ~]# cat /work/policies.txt
/* Rule named "execute-files" places files in pool "itso-stg-pool", using
criteria: file extension ".bin" */
RULE 'execute-files' SET POOL 'itso-stg-pool' WHERE (name) like '%.bin'
```

```
/* NOTE: THIS IS MANDATORY, otherwise you may end up not being able to create
other files */
RULE 'default' set POOL 'system'
```

```
/* Move all files having the name starting with "big" */
RULE 'bigfiles' MIGRATE TO POOL 'itso-stg-pool' WHERE (name) like 'big%'
```

2. Create a file by using the following command:
dd if=/dev/zero of=/gpfs-ib/big-gogo bs=1M count=1000
3. Run the **mmfsattr** command, shown in Example 8-33, to verify where the file is placed.

Example 8-33 The mmfsattr command

```
[root@spain ~]# mmfsattr -L /gpfs-ib/big-gogo
file name:          /gpfs-ib/big-gogo
metadata replication: 1 max 2
data replication:    1 max 2
immutable:          no
appendOnly:         no
flags:
storage pool name:   system
fileset name:        root
snapshot name:
```

4. To enable the migration policy to take effect, run the **mmapplypolicy gpfs-ib** command (Example 8-34).

Note: To apply a migration policy, the **mmapplypolicy** command is used. This process is manual. Although ways exist to automate the process, several considerations apply:

- ▶ A migration policy, when run, generates I/O, so it can have an impact on production systems.
- ▶ The **mmapplypolicy** command might take longer to finish depending on the data that it has to move.

Example 8-34 The mmapplypolicy command

```
[root@spain ~]# mmapplypolicy gpfs-ib
... snipped...
Evaluating MIGRATE/DELETE/EXCLUDE rules with CURRENT_TIMESTAMP =
2010-07-07@14:15:36 UTC
parsed 0 Placement Rules, 0 Restore Rules, 1 Migrate/Delete/Exclude Rules,
0 List Rules, 0 External Pool/List Rules
/* Move all files having the name starting with "big" */
RULE 'bigfiles' MIGRATE TO POOL 'itso-stg-pool' WHERE (name) like 'big%'
[I]2010-07-07@14:15:36.814 Directory entries scanned: 35.
[I] Directories scan: 4 files, 31 directories, 0 other objects, 0 'skipped'
files and/or errors.
[I]2010-07-07@14:15:36.817 Sorting 35 file list records.
[I] Inodes scan: 4 files, 31 directories, 0 other objects, 0 'skipped' files
and/or errors.
[I]2010-07-07@14:15:36.837 Policy evaluation. 35 files scanned.
[I]2010-07-07@14:15:36.840 Sorting 1 candidate file list records.
[I]2010-07-07@14:15:36.841 Choosing candidate files. 1 records scanned.
[I] Summary of Rule Applicability and File Choices:
```

| Rule# | Hit_Cnt | KB_Hit | Chosen | KB_Chosen | KB_Ill | Rule |
|-------|---------|---------|--------|-----------|--------|--|
| 0 | 1 | 1024000 | 1 | 1024000 | 0 | RULE 'bigfiles' MIGRATE TO POOL 'itso-stg-pool' WHERE(.) |

[I] Filesystem objects with no applicable rules: 34.

[I] GPFS Policy Decisions and File Choice Totals:

Chose to migrate 1024000KB: 1 of 1 candidates;
 Chose to premigrate OKB: 0 candidates;
 Already co-managed OKB: 0 candidates;
 Chose to delete OKB: 0 of 0 candidates;
 Chose to list OKB: 0 of 0 candidates;
 OKB of chosen data is illplaced or illreplicated;
 Predicted Data Pool Utilization in KB and %:
itso-stg-pool 3076608 52428800 5.868164%
system 5181952 52428800 9.883789%

[I]2010-07-07@14:15:55.443 Policy execution. 1 files dispatched.

[I] A total of 1 files have been migrated, deleted or processed by an EXTERNAL EXEC/script;
 0 'skipped' files and/or errors.

5. Use the `mmfsattr` command to verify that the big-gogo file is in itso-stg-pool.

As an example, we add a new migration policy that moves all files larger than 1 GB to itso-stg-pool, as shown in Example 8-35.

Example 8-35 Large files policy

```
/* Move all files bigger then 1GB to pool "itso-stg-pool" */
RULE 'largefiles' MIGRATE FROM POOL 'system' TO POOL 'itso-stg-pool' WHERE
FILE_SIZE > 1073741824
```

6. Create a file larger that 1 GB and run `mmapplypolicy` to test the results.

Create an automated migration policy

To apply the migration policy automatically, the `mmapplypolicy` command must be run by a scheduler. Although a common way is to use crontab to start the `mmapplypolicy` command on off-peak hours, the disadvantage to this approach is that the `mmapplypolicy` runs at a predefined time, such as every night.

If, during the day, a migration policy has to run to be able to free space on various storage pools, the process should be a manual process. If the process fails by human error, the file system might fill up.

To run `mmapplypolicy` on-demand (when it is needed), use GPFS event triggers functions. For example, create a *CallbackIdentifier* action that is triggered by *lowDiskSpace event* and runs the `mmapplypolicy` command through a script. The *lowDiskSpace* is triggered when the file system manager detects that disk space is below the low threshold that is specified in the current policy rule.

The entire process works as follows:

1. Create files of varying sizes on the system pool by using the `dd` command, but not larger than 1 GB, so it does not match the previous migration policies you created.
2. Add a new migration policy (shown in Example 8-36) that moves all the data from the system pool when the system pool gets to 30% utilization and stops at 10%, starting with larger files first.

Example 8-36 The threshold migration policy

```
/* Move all data from the system pool when it gets to 30% utilization and
stops at 10% */
RULE 'overflow' MIGRATE FROM POOL 'system' THRESHOLD(30,10) TO POOL
'itso-stg-pool' WEIGHT(KB_ALLOCATED)
```

After creating the files, the system pool utilization drops below 30%, as shown in Example 8-37

Example 8-37 Dropping below 30%, system pool utilization

```
[root@greece gpfs-ib]# mmdf gpfs-ib
```

| disk name | disk size in KB | failure group | holds metadata | holds data | free KB in full blocks | free KB in fragments |
|---|--------------------|------------------|-------------------|---------------|---------------------------|-------------------------|
| Disks in storage pool: system (Maximum disk size allowed is 219 GB) | | | | | | |
| gpfs1nsd | 26214400 | 1 | yes | yes | 16449536 (63%) | 6632 (0%) |
| gpfs2nsd | 26214400 | 2 | yes | yes | 16446208 (63%) | 9056 (0%) |
| ----- | | | | | | |
| (pool total) | 52428800 | | | | 32895744 (63%) | 15688 (0%) |
| Disks in storage pool: itso-stg-pool (Maximum disk size allowed is 219 GB) | | | | | | |
| gpfs3nsd | 26214400 | 1 | no | yes | 22121728 (84%) | 248 (0%) |
| gpfs4nsd | 26214400 | 2 | no | yes | 22110464 (84%) | 248 (0%) |
| ----- | | | | | | |
| (pool total) | 52428800 | | | | 44232192 (84%) | 496 (0%) |
| ===== | | | | | | |
| (data) | 104857600 | | | | 77127936 (74%) | 16184 (0%) |
| (metadata) | 52428800 | | | | 32895744 (63%) | 15688 (0%) |
| ===== | | | | | | |
| (total) | 104857600 | | | | 77127936 (74%) | 16184 (0%) |
| ...snipped... | | | | | | |

3. Run the **mmapplypolicy** command in test and verbose mode to determine which file is a candidate for moving, as shown in Example 8-38, and note the ordering process (it starts with larger files) until the system pool reaches under 10% utilization.

Example 8-38 Testing the migration policy

```
[root@spain ~]# mmapplypolicy gpfs-ib -I test -L 3
...snipped...
/* Move all data from the system pool when it gets to 30% utilization and stops at 10% */
RULE 'overflow' MIGRATE FROM POOL 'system' THRESHOLD(30,10) WEIGHT(KB_ALLOCATED) TO POOL
'itso-stg-pool'
...snipped...
[I] Inodes scan: 34 files, 31 directories, 0 other objects, 0 'skipped' files and/or errors.
[I]2010-07-07@15:50:54.767 Policy evaluation. 65 files scanned.
[I]2010-07-07@15:50:54.770 Sorting 32 candidate file list records.
...snipped...
WEIGHT(921600.000000) MIGRATE /gpfs-ib/user1/file5user1 TO POOL itso-stg-pool SHOW()
WEIGHT(921600.000000) MIGRATE /gpfs-ib/user2/file5user2 TO POOL itso-stg-pool SHOW()
WEIGHT(921600.000000) MIGRATE /gpfs-ib/user3/file5user3 TO POOL itso-stg-pool SHOW()
WEIGHT(921600.000000) MIGRATE /gpfs-ib/user4/file5user4 TO POOL itso-stg-pool SHOW()
WEIGHT(921600.000000) MIGRATE /gpfs-ib/user5/file5user5 TO POOL itso-stg-pool SHOW()
WEIGHT(512000.000000) MIGRATE /gpfs-ib/user1/fileuser1 TO POOL itso-stg-pool SHOW()
WEIGHT(512000.000000) MIGRATE /gpfs-ib/user2/fileuser2 TO POOL itso-stg-pool SHOW()
```

...snipped..

[I]2010-07-07@15:50:54.771 Choosing candidate files. 32 records scanned.

[I] Summary of Rule Applicability and File Choices:

| Rule# | Hit_Cnt | KB_Hit | Chosen | KB_Chosen | KB_ILL | Rule |
|----------|---------|----------|--------|-----------|--------|---|
| 0 | 1 | 1024000 | 0 | 0 | 0 | RULE 'bigfiles' MIGRATE TO POOL 'itso-stg-pool' |
| WHERE(.) | | | | | | |
| 1 | 0 | 0 | 0 | 0 | 0 | RULE 'largefiles' MIGRATE FROM POOL 'system' TO POOL 'itso-stg-pool' WHERE(.) |
| 2 | 31 | 19456000 | 20 | 14336000 | 0 | RULE 'overflow' MIGRATE FROM POOL 'system' THRESHOLD(30,10) WEIGHT(.) TO POOL 'itso-stg-pool' |

[I] Filesystem objects with no applicable rules: 33.

[I] GPFS Policy Decisions and File Choice Totals:

Chose to migrate 14336000KB: 20 of 32 candidates;

Chose to premigrate 0KB: 0 candidates;

Already co-managed 0KB: 0 candidates;

Chose to delete 0KB: 0 of 0 candidates;

Chose to list 0KB: 0 of 0 candidates;

0KB of chosen data is illplaced or illreplicated;

Predicted Data Pool Utilization in KB and %:

| | | | |
|---------------|----------------|-----------------|------------------|
| itso-stg-pool | 22532608 | 52428800 | 42.977539% |
| system | 5197056 | 52428800 | 9.912598% |

4. Apply the policy by running the **mmappolicy gpfs-ib** command.

5. Review the **mmdf** command output, as shown in Example 8-39. Compare it with the output from the Example 8-37 on page 356.

Example 8-39 Rearrange data according to policies

```
[root@greece gpfs-ib]# mmdf gpfs-ib
```

| disk name | disk size in KB | failure group | holds metadata | holds data | free KB in full blocks | free KB in fragments |
|---|-----------------|---------------|----------------|------------|------------------------|----------------------|
| Disks in storage pool: system (Maximum disk size allowed is 219 GB) | | | | | | |
| gpfs1nsd | 26214400 | 1 | yes | yes | 23617792 (90%) | 6632 (0%) |
| gpfs2nsd | 26214400 | 2 | yes | yes | 23613952 (90%) | 9056 (0%) |
| (pool total) | 52428800 | | | | 47231744 (90%) | 15688 (0%) |
| Disks in storage pool: itso-stg-pool (Maximum disk size allowed is 219 GB) | | | | | | |
| gpfs3nsd | 26214400 | 1 | no | yes | 14981888 (57%) | 248 (0%) |
| gpfs4nsd | 26214400 | 2 | no | yes | 14914304 (57%) | 248 (0%) |
| (pool total) | 52428800 | | | | 29896192 (57%) | 496 (0%) |
| ===== | | | | | | |
| (data) | 104857600 | | | | 77127936 (74%) | 16184 (0%) |
| (metadata) | 52428800 | | | | 47231744 (90%) | 15688 (0%) |
| ===== | | | | | | |
| (total) | 104857600 | | | | 77127936 (74%) | 16184 (0%) |

Inode Information

Number of used inodes: 4073

| | |
|-----------------------------|-------|
| Number of free inodes: | 48151 |
| Number of allocated inodes: | 52224 |
| Maximum number of inodes: | 52224 |

6. Create a callback script that runs when the *lowDiskSpace* event triggers. A sample script is shown in Example 8-40.

Note: The *lowDiskSpace* event triggers when a migration policy using threshold is defined and then applied for a specific storage pool. If multiple migration policies using this threshold exist, then the first one that matches it will trigger the event.

Example 8-40 The mmappolicy script

```
[root@spain ~]# cat /work/test.sh
#!/bin/bash
#Setup Variables
SCRIPT_NAME=$0
EVENT_NAME=$1
STG_POOL=$2
FS=$3
#Find out the mount point for the monitored file system
MOUNT_POINT=`/usr/lpp/mmfs/bin/mmfsfs $FS |grep "\-T" |awk '{ print $2}'`

#Create a timestamp on the monitored files systems
#Warning: If the file system gets full the timestamp file cannot be created
if [ -f $MOUNT_POINT/.${EVENT_NAME} ]
then
/bin/echo EVENT_$SCRIPT_NAME: The $EVENT_NAME event is still running since \
`/usr/bin/stat -c %y $MOUNT_POINT/.${EVENT_NAME} |awk '{ print $1 " " $2 }'` \
Exiting ...
exit 0
else
/bin/touch $MOUNT_POINT/.${EVENT_NAME}
fi
/bin/echo
#Apply the policy
/bin/echo running mmappolicy for the file system: $FS
/usr/lpp/mmfs/bin/mmappolicy $FS
#Remove the lock
/bin/rm $MOUNT_POINT/.${EVENT_NAME} -f
/bin/echo Event name $EVENT_NAME is DONE
/bin/echo
```

7. Add the callback function by running the following command:

```
mmaddcallback LOWDISK --command /work/test.sh --event lowDiskSpace -N \
spain-gpfs --parms "%eventname %storagePool %fsName"
```
8. Verify whether the script is installed by running the **mm1scallback** command, as shown in Example 8-41.

Example 8-41 The mm1scallback command

```
[root@spain ~]# mm1scallback
LOWDISK
      command      = /work/test.sh
      event         = lowDiskSpace
```



```
node          = spain-gpfs
parms         = %eventName %storagePool %fsName
```

Note: After the *lowDiskSpace* event triggers, it continues to trigger every two minutes until the condition that triggered it stops, in our case the storage pool utilization drops below 30%. This means that the script that is started by the event will start again every two minutes. We build a rudimentary (as example) locking system into the script to avoid multiple starts of **mmapplypolicy** command.

The script performs the following tasks:

- ▶ Receives the event name, storage name, and file system name as parameters from the callback function.
- ▶ Verifies whether the lock file is on the file system:
 - If it is, the script exits because another script is running.
 - If it is not, the script creates a lock on the GPFS file system where the event triggered, and starts the **mmapplypolicy** command. When the command finishes the lock file, the script erases the lock file and exits.

For a production system, such a script must perform more checking and be more save-proof as follows:

- ▶ The lock file must be present all the time; for example, if the file system is full, the lock file is already present.
- ▶ To have high availability, several nodes must trigger the same event. Our case uses only one node.
- ▶ If the script fails, the lock is never erased so the migration never starts. In this case, the script must verify that the lock is still valid and there is a process currently running regarding the lock file. The verification function must be able to check all the nodes where the event is triggered.

Example 8-42 shows the GPFS logs when the event is triggered.

Example 8-42 Event-triggered logs

```
Wed Jul  7 16:20:53.479 2010: Calling User Exit Script LOWDISK: event
lowDiskSpace, command /work/test.sh.

running mmapplypolicy for the file system: gpfs-ib
[I] GPFS Current Data Pool Utilization in KB and %
itso-stg-pool 12599808      52428800      24.032227%
system 17180672      52428800      32.769531%
[I] 4074 of 52224 inodes used: 7.801011%.
[I] Loaded policy rules from /var/mmfs/tmp/tspolicyFile.mmapplypolicy.1283.
Evaluating MIGRATE/DELETE/EXCLUDE rules with CURRENT_TIMESTAMP =
2010-07-07@20:20:54 UTC
parsed 2 Placement Rules, 0 Restore Rules, 3 Migrate/Delete/Exclude Rules,
      0 List Rules, 0 External Pool/List Rules
...snipped...
/* Move all data from the system pool when it gets to 30% utilization and stops
at 10% */
RULE 'overflow' MIGRATE FROM POOL 'system' THRESHOLD(30,10) WEIGHT(KB_ALLOCATED)
TO POOL 'itso-stg-pool'
...snipped..
```

```

/gpfs-ib/user4/file1user4      RULE 'overflow' MIGRATE FROM POOL 'system' TO
POOL 'itso-stg-pool' WEIGHT(512000.000000)
/gpfs-ib/user5/file1user5      RULE 'overflow' MIGRATE FROM POOL 'system' TO
POOL 'itso-stg-pool' WEIGHT(512000.000000)
/gpfs-ib/.lowDiskSpace        RULE 'overflow' MIGRATE FROM POOL 'system' TO POOL
'itso-stg-pool' WEIGHT(0.000000)
/gpfs-ib/big-gogo             RULE 'bigfiles' MIGRATE TO POOL 'itso-stg-pool'
WEIGHT(inf)
[I] Inodes scan: 35 files, 31 directories, 0 other objects, 0 'skipped' files
and/or errors.
[I] 2010-07-07@20:20:54.340 Policy evaluation. 66 files scanned.
[I] 2010-07-07@20:20:54.343 Sorting 12 candidate file list records.
[I] 2010-07-07@20:20:54.344 Choosing candidate files. 12 records scanned.
[I] Summary of Rule Applicability and File Choices:
  Rule#  Hit_Cnt KB_Hit Chosen KB_Chosen      KB_I11 Rule
    0      1    1024000 0      0      0      RULE 'bigfiles' MIGRATE TO POOL
'itso-stg-pool' WHERE(.)
    1      0      0      0      0      0      RULE 'largefiles' MIGRATE FROM
POOL 'system' TO POOL 'itso-stg-pool' WHERE(.)
    2     11    5120000 11    5120000 0      RULE 'overflow' MIGRATE FROM POOL
'system' THRESHOLD(30,10) WEIGHT(.) TO POOL 'itso-stg-pool'

[I] Filesystem objects with no applicable rules: 54.

...snipped..
[I] MIGRATED /gpfs-ib/user1/file1user1 TO POOL itso-stg-pool
[I] MIGRATED /gpfs-ib/user5/file2user5 TO POOL itso-stg-pool
[I] MIGRATED /gpfs-ib/user5/file1user5 TO POOL itso-stg-pool
[I] MIGRATED /gpfs-ib/user2/file1user2 TO POOL itso-stg-pool
[I] 2010-07-07@20:21:44.561 Policy execution. 11 files dispatched.
[I] A total of 11 files have been migrated, deleted or processed by an EXTERNAL
EXEC/script;
    0 'skipped' files and/or errors.
Event name lowDiskSpace is DONE

```

8.3.3 Working with external storage pool and GPFS policies

An external storage pool can be used in a variety of situations. As an example, we define as an external storage pool, a program that archives files on a DVD ISO image that can be created. It works as follows:

1. We define a policy in GPFS that, based on certain criteria, creates a list of candidate files to be archived.
2. The list is passed to the **mkisofs** program that creates a DVD ISO image.
3. After the DVD ISO image is created, the files are deleted.

Several disadvantages to this approach are as follows:

- ▶ The archived data cannot be recalled. The process works in only one direction.
- ▶ GPFS does not erase the files by itself. An external program has to delete them. After the files are deleted, GPFS has no knowledge of them. Retrieving them requires a manual process.

Note: The archive policy is only an example and is not meant to be used for production. Building a production-ready policy is beyond the scope of this book.

To install the archive policy, perform the following steps:

1. Define the external pool policy and rule (see Example 8-43):
 - The first line defines an external list that, when called, is running the `/work/mkiso.sh` script.
 - The second line defines the criteria that are used to build the list. In our case, all the files from storage pool `itso-stg-pool` that are starting with the letters `big` are to be archived.

Example 8-43 Archive policy

```
[root@spain work]# cat iso.policy
RULE EXTERNAL LIST 'mkiso' EXEC '/work/mkiso.sh'
RULE 'itso-stg-pool_to_iso' LIST 'mkiso' FROM POOL 'itso-stg-pool'
DIRECTORIES_PLUS WHERE (name) like 'big%'
```

2. Create the `/work/mkiso.sh` file as shown in Example 8-44. The script functions as follows:
 - When we apply the GFSP policy, using the `mmapplypolicy` command, the script is called with two parameters: the command to execute, in our case `LIST`, and the path of the temporary file that contains the file's candidates for archiving according to the policy we defined.
 - For the `LIST` command, the script removes the previous backup list files, takes the files from the GPFS temporary file list (that contains files to archive) and passes it to the `mkisofs` DVD ISO image creator. After the DVD ISO image is done, the script erases the files from the list (we comment those lines for safety).

Example 8-44 The mkiso.sh file

```
[root@spain work]# cat mkiso.sh
#!/bin/sh
case $1 in
    LIST)
        rm -f /tmp/backup*
        rm -f /gpfs-ib/backup*
        BACKUP_DATE=`date +%F-%H:%M`
        cat $2 | awk '{ print $5 }' >>/tmp/backup_${BACKUP_DATE}.filelist
        mkisofs -o /gpfs-ib/backup_${BACKUP_DATE}.iso
        -path-list=/tmp/backup_${BACKUP_DATE}.filelist
        # for i in `cat /tmp/backup_${BACKUP_DATE}.filelist`
        # do
        #     rm -fr $i
        # done
        rc=0
        ;;
    TEST)      # Respond with success
        rc=0
        ;;
    *)         # Command not supported by this script
```

```

rc=1
;;
esac
exit $rc

```

3. Apply the policy by running the **mmapplypolicy gpfs-ib -P iso.policy -L 3** command, as shown in Example 8-45. Note the order in which the policies are executed and that the **mkisofs** program is running. The policy can be automated by creating a user exit script, as described in 8.3.2, “Working with storage pools and policies” on page 350.

Example 8-45 The iso.policy is applied

```

[root@spain work]# mmapplypolicy gpfs-ib -P iso.policy -L 3
[I] GPFS Current Data Pool Utilization in KB and %
itso-stg-pool 199680 52428800 0.380859%
system 2112512 52428800 4.029297%
[I] 4077 of 52224 inodes used: 7.806756%.
[I] Loaded policy rules from iso.policy.
Evaluating MIGRATE/DELETE/EXCLUDE rules with CURRENT_TIMESTAMP = 2010-07-09@16:16:44 UTC
parsed 0 Placement Rules, 0 Restore Rules, 0 Migrate/Delete/Exclude Rules,
  1 List Rules, 1 External Pool/List Rules
RULE EXTERNAL LIST 'mkiso' EXEC '/work/mkiso.sh'
RULE 'itso-stg-pool_to_iso' LIST 'mkiso' FROM POOL 'itso-stg-pool' DIRECTORIES_PLUS WHERE (name)
like 'big%'
[I]2010-07-09@16:16:44.855 Directory entries scanned: 69.
[I] Directories scan: 38 files, 31 directories, 0 other objects, 0 'skipped' files and/or
errors.
[I]2010-07-09@16:16:44.858 Sorting 69 file list records.
/gpfs-ib/big-gogo.gz44.8 RULE 'itso-stg-pool_to_iso' LIST 'mkiso' DIRECTORIES_PLUS FROM POOL
'itso-stg-pool' WEIGHT(inf)
/gpfs-ib/big-gogo2.bin RULE 'itso-stg-pool_to_iso' LIST 'mkiso' DIRECTORIES_PLUS FROM POOL
'itso-stg-pool' WEIGHT(inf)
[I] Inodes scan: 38 files, 31 directories, 0 other objects, 0 'skipped' files and/or errors.
[I]2010-07-09@16:16:44.937 Policy evaluation. 69 files scanned.
[I]2010-07-09@16:16:44.940 Sorting 2 candidate file list records.
[I]2010-07-09@16:16:44.941 Choosing candidate files. 2 records scanned.
[I] Summary of Rule Applicability and File Choices:
Rule# Hit_Cnt KB_Hit Chosen KB_Chosen KB_ILL Rule
0 2 102632 2 102632 0 RULE 'itso-stg-pool_to_iso' LIST 'mkiso'
DIRECTORIES_PLUS FROM POOL 'itso-stg-pool' WHERE(.)
[I] Filesystem objects with no applicable rules: 67.

[I] GPFS Policy Decisions and File Choice Totals:
Chose to migrate OKB: 0 of 0 candidates;
Chose to premigrate OKB: 0 candidates;
Already co-managed OKB: 0 candidates;
Chose to delete OKB: 0 of 0 candidates;
Chose to list 102632KB: 2 of 2 candidates;
OKB of chosen data is illplaced or illreplicated;
Predicted Data Pool Utilization in KB and %:
itso-stg-pool 199680 52428800 0.380859%
system 2112512 52428800 4.029297%
INFO:10-UTF-8 character encoding detected by locale settings.ed. .\.....
Assuming UTF-8 encoded filenames on source filesystem,
use -input-charset to override.

```

```

 9.64% done, estimate finish Fri Jul 9 16:16:54 2010
19.29% done, estimate finish Fri Jul 9 16:16:49 2010
28.92% done, estimate finish Fri Jul 9 16:16:47 2010
38.57% done, estimate finish Fri Jul 9 16:16:46 2010
48.20% done, estimate finish Fri Jul 9 16:16:46 2010
57.85% done, estimate finish Fri Jul 9 16:16:45 2010
67.48% done, estimate finish Fri Jul 9 16:16:45 2010
77.13% done, estimate finish Fri Jul 9 16:16:45 2010
86.75% done, estimate finish Fri Jul 9 16:16:45 2010
96.41% done, estimate finish Fri Jul 9 16:16:46 2010
Total translation table size: 0
Total rockridge attributes bytes: 0
Total directory bytes: 0
Path table size(bytes): 10
Max brk space used 0
51871 extents written (101 MB)
[I]2010-07-09@16:16:46.031 Policy execution. 2 files dispatched.
[I] A total of 2 files have been migrated, deleted or processed by an EXTERNAL EXEC/script;
    0 'skipped' files and/or errors.

```

8.3.4 Working with hierarchical storage management

In this scenario, the australia node joins the GPFS cluster. In the scenario, spain, greece, and nigeria are Linux PowerPC nodes, and australia is a Linux86 node. At the time of writing, Tivoli Hierarchical Storage Manager (HSM) code does not run on the Linux PowerPC platform.

This scenario uses HSM client to migrate data from the GPFS file system using its own policies, like with any other file system

The following considerations apply when you use a hierarchical storage manager in a GPFS cluster:

- ▶ The HSM client manages file systems that only belong to the local (home) GPFS cluster. It does not manage remotely mounted file systems.
- ▶ In a GPFS environment, a small file that is less than 8 KB (smaller than a GPFS block size) can become larger after an HSM migration because GPFS adds meta information to the file during the migration. Because another block on the file system is allocated for the meta information, the space allocated for the file is increased. If a file system is filled to its maximum capacity with many small files, the file system might possibly run out of space during the file migration
- ▶ The HSM policy applies to the entire GPFS file system regardless of defined storage pool, file sets, and so on.

Install HSM on the australia node and configure it as follows:

1. Install the code on australia as shown Example 8-46.

Example 8-46 HSM installation

```

root@australia linux86]# rpm -ivh TIVsm-HSM.i386.rpm
Preparing...                               ##### [100%]
 1:TIVsm-HSM                               ##### [100%]

```

```

Performing configuration tasks for the TSM Space Management Client.
Removing the GPFS HSM entry from /etc/inittab.

```

```
Added the following entry to /etc/inittab right after mmfs:
    "ghsm:2345:respawn:/opt/tivoli/tsm/client/hsm/bin/dsmwatchd nodetach >
/dev/console 2>&1 # TSM SpaceMan"
The file /var/mmfs/etc/gpfsready does not exist
Copying /usr/lpp/mmfs/samples/gpfsready.sample to /var/mmfs/etc/gpfsready
Adding HSM to GPFS startup script
```

```
-----
The file '/opt/tivoli/tsm/client/ba/bin/dsm.sys' must be updated
to point to an TSM Server.
-----
```

```
-----
The file '/opt/tivoli/tsm/client/ba/bin/dsm.opt' must be updated
to point to an TSM Server.
-----
```

```
Starting dsmwatchd . . . ok!
--- loading dsmrootd
--- loading dsmreca1d
--- loading dsmmonitord
--- loading dsmscouth
```

Configuration tasks for the TSM Space Management Client completed.

2. Verify that the dsm.sys and dsm.opt files have the correct server definition, as shown in Example 8-47. The passwordaccess parameter has to be set to generate so that HSM can work.

Example 8-47 The dsm.sys file

```
[root@australia ~]# cat /opt/tivoli/tsm/client/ba/bin/dsm.sys
...snipped...
SErvername  italy
COMMMethod      TCPip
TCPPort         1500
TCPServeraddress italy
nodename gpfs-ib
passwordaccess generate
```

3. Enable DMAPI on the GPFS file system, as shown in Example 8-48.

Note: To enable DMAPI, unmount the file system from all GPFS nodes.

Example 8-48 Enable DMAPI

```
[root@australia work]# mkdir /gpfs-ib/.SpaceMan
[root@australia work]# mmlsfs gpfs-ib -z
flag          value          description
-----
-z            no              Is DMAPI enabled?
[root@australia work]# mmchfs gpfs-ib -z yes
The option '-z' could not be changed. gpfs-ib is still in use.
mmchfs: tschfs failed.
mmchfs: Command failed. Examine previous error messages to determine cause.
```

```
[root@australia work]# mmumount gpfs-ib -a
Sun Jul 11 07:10:30 EDT 2010: mmumount: Unmounting file systems ...
```

```
[root@australia work]# mmchfs gpfs-ib -z yes
[root@australia work]# mmmount gpfs-ib -a
Sun Jul 11 07:10:45 EDT 2010: mmmount: Mounting file systems ...
[root@australia work]# mmlsfs gpfs-ib -z
```

| flag | value | description |
|------|-------|-------------------|
| -z | yes | Is DMAPI enabled? |

4. Enable space management by running the **dsmmigfs** command (Example 8-49).

Example 8-49 Enable space management

```
[root@australia work]# dsmmigfs Add -HT=40 -L=10 -ST=1024k /gpfs-ib
IBM Tivoli Storage Manager
Command Line Space Management Client Interface
  Client Version 6, Release 2, Level 1.0
  Client date/time: 07/11/2010 07:20:20
(c) Copyright by IBM Corporation and other(s) 1990, 2010. All Rights Reserved.
```

```
Adding HSM support for /gpfs-ib ...
ANS9087I Space management is successfully added to file system /gpfs-ib.
ANS9360W dsmmigfs: /gpfs-ib: configured 10 low threshold is below the
recommended minimum 11 low threshold.
```

Note: When DMAPI is enabled on a GPFS file system, the file system can be mounted only if a **dsmrecalld** daemon is already set up on one of the cluster nodes within the GPFS cluster.

5. Verify that the space management policy is allowed in Tivoli Storage Manager server. Modify the default management class to allow the HSM policy, as shown in Figure 8-18 on page 365, and then activate it.

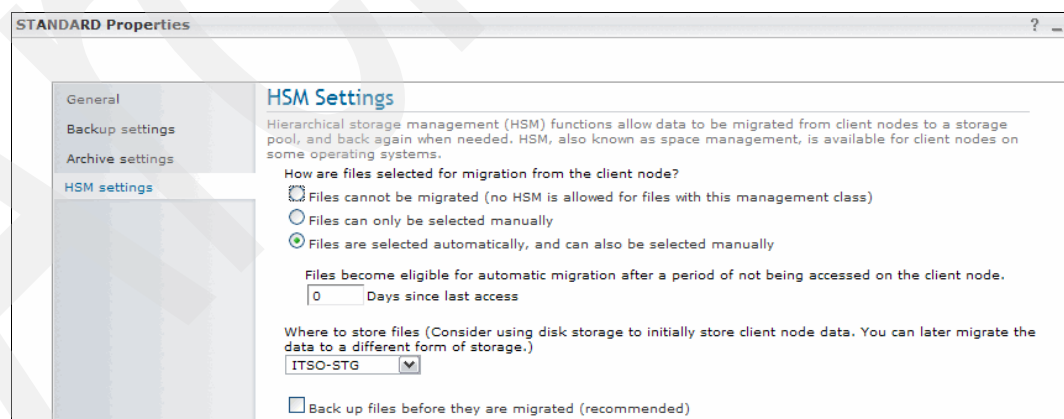


Figure 8-18 HSM settings

6. Fill the file system either by writing data to it with the **dd** command, or by copying some files. When the file system becomes more than 40% full, according to the defined policy, the **dsmautomig** process is started. The process does not start immediately and might take a while to process, depending on the data to be migrated

When the **dsmautomig** process completes, the data utilization on the file system should be below the threshold. The migrated files are still present on the file system although part of them are on the Tivoli Storage Manager storage pool, as shown in Example 8-50. (The resident part is the *stub* file that we defined in Example 8-49 on page 365.)

Example 8-50 dsmls /gpfs-ib

```
[root@australia ~]# dsmls /gpfs-ib/
IBM Tivoli Storage Manager
Command Line Space Management Client Interface
  Client Version 6, Release 2, Level 1.0
  Client date/time: 07/12/2010 13:05:44
(c) Copyright by IBM Corporation and other(s) 1990, 2010. All Rights Reserved.
```

| Actual Size <mnt> | Resident Size 32768 | Resident Blk (KB) 32 | File State - | File Name gpfs-ib/ |
|-------------------------|---------------------------|----------------------------|--------------------|--------------------------|
| /gpfs-ib: | | | | |
| <dir> | 8192 | 8 | - | .SpaceMan/ |
| <dir> | 8192 | 1 | - | .snapshots/ |
| 10485760000 | 10485760000 | 10240000 | r | big-gogo |
| 5242880000 | 5242880000 | 5120000 | r | big-gogo1 |
| 1047527424 | 1047527424 | 1022976 | r | gogo1 |
| 8388608000 | 1048576 | 1024 | m | gogo1.bin |
| 1047527424 | 1047527424 | 1022976 | r | gogo2 |
| 8388608000 | 1048576 | 1024 | m | gogo2.bin |
| 1047527424 | 1047527424 | 1022976 | r | gogo3 |
| 8388608000 | 1048576 | 1024 | m | gogo3.bin |
| 1047527424 | 1047527424 | 1022976 | r | gogo4 |
| 10485760000 | 1048576 | 1024 | m | gogo4.bin |
| 10485760000 | 1048576 | 1024 | m | gogo5.bin |
| 3145728000 | 1048576 | 1024 | m | gogo6.bin |

```
[root@australia ~]# dsmdf -Detail /gpfs-ib
IBM Tivoli Storage Manager
Command Line Space Management Client Interface
  Client Version 6, Release 2, Level 1.0
  Client date/time: 07/12/2010 13:08:14
(c) Copyright by IBM Corporation and other(s) 1990, 2010. All Rights Reserved.
```

| | |
|--------------------|----------|
| HSM Filesystem: | /gpfs-ib |
| FS State: | active |
| Migrated Size: | 48128000 |
| Premigrated Size: | 11837440 |
| Migrated Files: | 6 |
| Premigrated Files: | 11 |
| Unused Inodes: | 48171 |
| Free Size: | 82408192 |

Note: Newly-created files that you migrate either automatically or selectively must be older than two minutes before you can migrate them. Migrating newly-created files less than five minutes old might display incorrect results (resident size) when you use the **dsmdf** and **dsmdu** commands because the GPFS is not synchronized on all nodes when you migrate files. The **dsmdf** command displays correct results after GPFS synchronization and after the next reconciliation of the file system.

8.3.5 Working with hierarchical storage management and GPFS policies

This scenario uses previous HSM client and GPFS policies to migrate data from the GPFS file system to Tivoli Storage Manager. In this case, the HSM client is only a framework for GPFS, because all the logic is inside GPFS

Note: HSM has its own policy management separate from GPFS. However, use only one tool for policy management, either HSM or GPFS, to prevent confusion and misbehavior for the migration system.

Using GPFS policy management tools has several advantages:

- ▶ One tool manages all the policies regarding GPFS.
- ▶ GPFS policies are more flexible and permit more thorough selection of files to migrate.

The configuration steps are as follows:

1. Disable the HSM auto-migration capability by running the following command:

```
dsmmigfs update -HT=100 /gpfs-ib
```
2. Add an external HSM storage pool to as shown in Example 8-51. This rule indicates that the external pool named *hsm* is managed by the */work/hsm.sh* script.

Example 8-51 The external pool

```
/* Define hsm storage manager as an external pool */
RULE EXTERNAL POOL 'hsm' EXEC '/work/hsm.sh' OPTS '-v'
```

3. Add a policy to migrate the files from the GPFS file system to the Tivoli Storage Manager using HSM functions, as shown in Example 8-52. This rule migrates the files from the *itso-stg-pool*, when it is 30% full until it gets to 20%, to the *hsm* pool, larger files first.

Example 8-52 HSM policy

```
/* Move unused data off-line during the week */
RULE 'itso-stg-pool_to_hsm' MIGRATE FROM POOL 'itso-stg-pool'
THRESHOLD(30,20,15) TO POOL 'hsm' WEIGHT(KB_ALLOCATED)
```

4. Fill up the file system by using the **dd** command or copy some files as shown in Example 8-49 on page 365. Note that *itso-stg-pool* is more then 30% utilized (50% free space), but the entire file system is only 29% utilized.

Example 8-53 Fill up the GPFS file system

```
[root@spain ~]# mmdf gpfs-ib
```

| disk name | disk size in KB | failure holds group metadata | holds data | free KB in full blocks | free KB in fragments |
|---|--------------------|---------------------------------|---------------|---------------------------|-------------------------|
| ----- | | | | | |
| Disks in storage pool: system (Maximum disk size allowed is 219 GB) | | | | | |

| | | | | | |
|--|-----------|-------|-----|-----------------|------------|
| gpfs1nsd | 26214400 | 1 yes | yes | 24009728 (92%) | 2248 (0%) |
| gpfs2nsd | 26214400 | 2 yes | yes | 24001792 (92%) | 2240 (0%) |
| ----- | | | | | |
| (pool total) | 52428800 | | | 48011520 (92%) | 4488 (0%) |
| Disks in storage pool: itso-stg-pool (Maximum disk size allowed is 219 GB) | | | | | |
| gpfs3nsd | 26214400 | 1 no | yes | 13123840 (50%) | 248 (0%) |
| gpfs4nsd | 26214400 | 2 no | yes | 13051136 (50%) | 248 (0%) |
| ----- | | | | | |
| (pool total) | 52428800 | | | 26174976 (50%) | 496 (0%) |
| ===== | | | | | |
| (data) | 104857600 | | | 74186496 (71%) | 4984 (0%) |
| (metadata) | 52428800 | | | 48011520 (92%) | 4488 (0%) |
| ===== | | | | | |
| (total) | 104857600 | | | 74186496 (71%) | 4984 (0%) |

...snipped...

5. Test the policy by running the following command:

```
mmapplypolicy gpfs-ib -P /work/to_hsm.policy -I test -L3
```

Notice that during the test, the .SpaceMan directory is also investigated as a possible candidate. As shown in Example 8-54, this directory should not be migrated from the GPFS file system by using an exception rule. Also notice the effect that policy has on the space utilization and the effective file that will be migrated: the largest files.

Example 8-54 The .SpaceMan directory

```
...snipped...
[root@australia work]# mmapplypolicy gpfs-ib -P /work/to_hsm.policy -I test -L3
[I] GPFS Current Data Pool Utilization in KB and %
itso-stg-pool 26225152      52428800      50.020508%
system 4417280 52428800      8.425293%
[I] 4054 of 52224 inodes used: 7.762714%.
/gpfs-ib/gogo3.bin      RULE 'itso-stg-pool_to_hsm' MIGRATE FROM POOL 'itso-stg-pool' TO POOL
'hsm' WEIGHT(1024.000000)
/gpfs-ib/.SpaceMan/metadata/.gpfs-ib.meta1      RULE 'itso-stg-pool_to_hsm' MIGRATE FROM POOL
'itso-stg-pool' TO POOL 'hsm' WEIGHT(1639424.000000)
/gpfs-ib/gogo1      RULE 'itso-stg-pool_to_hsm' MIGRATE FROM POOL 'itso-stg-pool' TO POOL 'hsm'
WEIGHT(1022976.000000)
[I]2010-07-12@17:42:22.500 Sorting 11 candidate file list records.
WEIGHT(10240000.000000) MIGRATE /gpfs-ib/big-gogo TO POOL hsm SHOW()ed. .\.....
WEIGHT(8192000.000000) MIGRATE /gpfs-ib/gogo7.bin TO POOL hsm SHOW()
[I]2010-07-12@17:42:22.500 Choosing candidate files. 11 records scanned.
[I] Summary of Rule Applicability and File Choices:
Rule# Hit_Cnt KB_Hit Chosen KB_Chosen KB_Ill Rule
0 11 26220544 2 18432000 0 RULE 'itso-stg-pool_to_hsm'
Predicted Data Pool Utilization in KB and %:
itso-stg-pool 7793152 52428800 14.864258%
system 4417280 52428800 8.425293%
```

6. Apply the policy by running the following command:

```
mmapplypolicy gpfs-ib -P /work/to_hsm.policy
```

The migrated files are shown in Example 8-55.

Example 8-55 Migrated files

```
[root@australia work]# dsmls /gpfs-ib/
IBM Tivoli Storage Manager
Command Line Space Management Client Interface
  Client Version 6, Release 2, Level 1.0
  Client date/time: 07/12/2010 13:53:18
(c) Copyright by IBM Corporation and other(s) 1990, 2010. All Rights Reserved.
```

| Actual Size <mnt> | Resident Size 32768 | Resident Blk (KB) 32 | File State - | File Name gpfs-ib/ |
|---------------------------|---------------------------|----------------------------|--------------------|--------------------------|
| /gpfs-ib: ...snipped.. | | | | |
| 10485760000 | 1048576 | 1024 | m | gogo4.bin |
| 10485760000 | 1048576 | 1024 | m | gogo5.bin |
| 3145728000 | 1048576 | 1024 | m | gogo6.bin |
| 8388608000 | 1048576 | 1024 | m | gogo7.bin |

7. The files are still accessible by any application. To test it, run the following command:

```
cat /gpfs-ib/gogo7.bin > /dev/null
```

Example 8-56 shows the file system space utilization before running the **cat** command.

Example 8-56 Before reading the gogo7.bin file

```
[root@spain ~]# mmdf gpfs-ib
disk          disk size  failure holds    holds          free KB          free KB
name          in KB     group metadata data          in full blocks  in fragments
-----
Disks in storage pool: system (Maximum disk size allowed is 219 GB)
gpfs1nsd      26214400      1 yes      yes      24009728 ( 92%)      2344 ( 0%)
gpfs2nsd      26214400      2 yes      yes      24001792 ( 92%)      2336 ( 0%)
-----
(pool total)   52428800                                48011520 ( 92%)      4680 ( 0%)

Disks in storage pool: itso-stg-pool (Maximum disk size allowed is 219 GB)
gpfs3nsd      26214400      1 no       yes      17233664 ( 66%)      248 ( 0%)
gpfs4nsd      26214400      2 no       yes      17160960 ( 65%)      248 ( 0%)
-----
(pool total)   52428800                                34394624 ( 66%)      496 ( 0%)

=====
(data)         104857600                                82406144 ( 79%)      5176 ( 0%)
(metadata)     52428800                                48011520 ( 92%)      4680 ( 0%)
=====
(total)        104857600                                82406144 ( 79%)      5176 ( 0%)
...snipped...
```

After the **cat** command is run, the file system utilization has increased because the **gogo7.bin** file is transferred from the Tivoli Storage Manager to GPFS, as shown in Example 8-57.

Example 8-57 After reading the gogo7.bin file

```
[root@spain ~]# mmdf gpfs-ib
```

| disk name | disk size in KB | failure holds group metadata | holds data | free KB in full blocks | free KB in fragments |
|--|-----------------|------------------------------|------------|------------------------|----------------------|
| Disks in storage pool: system (Maximum disk size allowed is 219 GB) | | | | | |
| gpfs1nsd | 26214400 | 1 yes | yes | 24009728 (92%) | 2344 (0%) |
| gpfs2nsd | 26214400 | 2 yes | yes | 24001792 (92%) | 2336 (0%) |
| (pool total) | 52428800 | | | 48011520 (92%) | 4680 (0%) |
| Disks in storage pool: itso-stg-pool (Maximum disk size allowed is 219 GB) | | | | | |
| gpfs3nsd | 26214400 | 1 no | yes | 13138176 (50%) | 248 (0%) |
| gpfs4nsd | 26214400 | 2 no | yes | 13065472 (50%) | 248 (0%) |
| (pool total) | 52428800 | | | 26203648 (50%) | 496 (0%) |
| (data) | 104857600 | | | 74215168 (71%) | 5176 (0%) |
| (metadata) | 52428800 | | | 48011520 (92%) | 4680 (0%) |
| (total) | 104857600 | | | 74215168 (71%) | 5176 (0%) |
| ...snipped... | | | | | |

8. The data can be restored by using a restore policy as show in Example 8-58. The policy restores the files until 80% utilization of the itso-stg-pool is reached.

Example 8-58 Restore policy

```
[root@australia ~]# cat /work/from_hsm.policy
/* Define hsm storage manager as an external pool */
RULE EXTERNAL POOL 'hsm' EXEC '/work/hsm.sh' OPTS '-v'
/* Recall data on the weekend for batch processing */
RULE 'hsm_to_itso-stg-pool' MIGRATE FROM POOL 'hsm' TO POOL 'itso-stg-pool' \
LIMIT(80)
```

9. Test and apply the policy by running the following command, shown in Example 8-59:

```
mmapplypolicy gpfs-ib -P /work/from_hsm.policy
```

Note: The restoration of the gogo7.bin file fails because the file was already restored at the previous step.

Example 8-59 Restore policy

...snipped...

```
Recalling 10,485,760,000 /gpfs-ib/gogo4.bin [Done]
Recalling 10,485,760,000 /gpfs-ib/gogo5.bin [Done]
Recalling 3,145,728,000 /gpfs-ib/gogo6.bin ANS9101I No migrated files matching
'/gpfs-ib/gogo7.bin' were found.
[Done]
Recalling 8,388,608,000 /gpfs-ib/gogo1.bin [Done]
Recalling 8,388,608,000 /gpfs-ib/gogo2.bin [Done]
Recalling 8,388,608,000 /gpfs-ib/gogo3.bin [Done]

Recall processing finished.
rm /tmp/mmPolicy.ix.18704.D660D269.1.hsm
/work/hsm.sh: RECALL /tmp/mmPolicy.ix.18704.D660D269.1 -v: 0k
```

```
[I]2010-07-12@18:25:38.940 Policy execution. 7 files dispatched.  
[I] A total of 7 files have been migrated, deleted or processed by an EXTERNAL  
EXEC/script;  
0 'skipped' files and/or errors.
```

8.3.6 End-to-end implementation of ILM and HSM

This scenario implements an end-to-end solution for GPFS policies and HSM migration. The cluster consists of spain, greece, nigeria and australia; spain and greece are quorum and managers nodes; nigeria is a NSD client; and australia is a NSD and HSM client. See Figure 8-5 on page 329.

Note: Certain events, including *lowDiskSpace*, are executed only on the file system manager, whichever is designated at that time.

Note the following information:

- ▶ By default, data goes into the *system pool*.
- ▶ At 50% *system* pool utilization, data is migrated automatically to the *itso-stg-pool* storage pool.
- ▶ At 70% *itso-stg-pool* storage pool utilization, data is migrated on Tivoli Storage Manager automatically.

Implementation

The end-to-end solution is implemented as follows:

1. Erase all files in the *gpfs-ib* file system. This process takes longer because there are files on the Tivoli Storage Manager.
2. Create the policy file and apply it as shown in Example 8-60 on page 371. The external rule defines the */work/local.hsm.sh* file as the one that manages the HSM migration.

Example 8-60 The end-to-end policy

```
[root@spain work]# cat end-to-end.policy  
RULE 'default' set POOL 'system'  
RULE EXTERNAL POOL 'hsm' EXEC '/work/local.hsm.sh' OPTS '-v'  
RULE 'system_to_itso-stg-pool' MIGRATE FROM POOL 'system' THRESHOLD(50,30) TO  
POOL 'itso-stg-pool' WEIGHT(KB_ALLOCATED)  
RULE 'itso-stg-pool_to_hsm' MIGRATE FROM POOL 'itso-stg-pool' THRESHOLD(70,50)  
TO POOL 'hsm' WEIGHT(KB_ALLOCATED)  
  
[root@spain work]# mmchpolicy gpfs-ib end-to-end.policy  
Policy `end-to-end.policy' installed and broadcast to all nodes.  
Validated policy `end-to-end.policy': parsed 1 Placement Rules, 0 Restore  
Rules, 2 Migrate/Delete/Exclude Rules,  
0 List Rules, 1 External Pool/List Rules  
[root@spain work]# mmlspolicy gpfs-ib -L  
RULE 'default' set POOL 'system'  
RULE EXTERNAL POOL 'hsm' EXEC '/work/local.hsm.sh' OPTS '-v'  
RULE 'system_to_itso-stg-pool' MIGRATE FROM POOL 'system' THRESHOLD(50,30) TO  
POOL 'itso-stg-pool' WEIGHT(KB_ALLOCATED)  
RULE 'itso-stg-pool_to_hsm' MIGRATE FROM POOL 'itso-stg-pool' THRESHOLD(70,50)  
TO POOL 'hsm' WEIGHT(KB_ALLOCATED)
```

3. Add the callback function that triggers the *lowDiskSpace* event. The event triggers when one of the migration policies matches. In this case, either the *system* pool reaches 50% utilization or *itso-stg-pool* reaches 70%, whichever matches first. Also, when a migration policy matches, the remaining migration policies are not started at the same time. They will start the next time the event triggers.

Note: For example, if the system pool reaches 50% utilization, the event triggers and starts the migration policy program, which creates a lock file and is not removed until the migration completes. Meanwhile, the *itso-stg-pool* can fill up with data from the *system* storage pool and triggers the migration policy to move data to Tivoli Storage Manager, but the migration policy program is not started because the lock file exists from the previous migration. In this situation, the *itso-stg-pool* might run out of space in generating errors.

To avoid this situation, choose the migration thresholds in such a way that migration of one storage pool does not fill up the other. In this scenario, the *system storage pool* migrates about 20% of data. Even if the *itso-stg-pool* is at 69% utilization, there is still room for another 20% of data from the *system storage pool*, assuming the storage pool has the same amount of storage capacity.

The callback trigger event is shown in Example 8-61 on page 372.

Important: The *lowDiskSpace* event triggers only on the file system manager, which in this case is the *spain* node. This means that the */work/end-to-end.sh* script is run only on the *spain* node.

Example 8-61 Event handling

```
[root@spain work]# mmlscallback
MIGDISK
      command      = /work/end-to-end.sh
      event         = lowDiskSpace
      parms         = %eventName %storagePool %fsName
[root@spain work]# cat end-to-end.sh
#!/bin/bash
#Setup Variables
SCRIPT_NAME=$0
EVENT_NAME=$1
STG_POOL=$2
FS=$3
#Find out the mount point for the monitored file system
MOUNT_POINT=`/usr/lpp/mmfs/bin/mmlsfs $FS |grep "\-T" |awk '{ print $2}'`

#Create a timestamp on the monitored files systems
#If the files system gets full the timestamp file cannot be created
if [ -f $MOUNT_POINT/.${EVENT_NAME} ]
then
  /bin/echo  EVENT_$SCRIPT_NAME: The $EVENT_NAME event is still running since
  `/usr/bin/stat -c %y $MOUNT_POINT/.${EVENT_NAME} |awk '{ print $1 " " $2 }'`
  Exiting ...
  exit 0
else
  /bin/touch $MOUNT_POINT/.${EVENT_NAME}
fi
```

```

/bin/echo
/bin/echo running mmapplypolicy for the file system: $FS
/usr/lpp/mmfs/bin/mmapplypolicy $FS -L3
/bin/rm $MOUNT_POINT/.${EVENT_NAME} -f
/bin/echo Event name $EVENT_NAME is DONE
/bin/echo

```

4. Create the `/work/local.hsm.sh` file. This file runs on the spain node, but the HSM client is installed on the australia node. The script has to capture the parameters that the migration policy generates and pass them to the `/work/hsm.sh` script that actually migrates the data by using HSM commands. The parameters are as follows:

- The action for the script to take (for example, MIGRATE or TEST)
- The file containing the files list to be migrated

5. Return any running errors that might occur.

The script is shown in Example 8-62. The `/work/hsm.sh` file is the same file as the `/usr/lpp/mmfs/samples/ilm/mmpolicyExec-hsm.sample` file that is provided by the GPFS code, but it has fewer comments.

Example 8-62 The `/work/hsm.sh` script

```

[root@spain work]# cat local.hsm.sh
#!/bin/sh
scp $2 australia:$2
ssh australia "/work/hsm.sh $1 $2 $3"
ERR=$?
ssh asutralia rm -f $2
exit $ERR

```

6. Create files to fill up the space, by running the `dd` command.

When the system storage pool gets to more than 50% utilization (39% free space), as shown in Example 8-63, the `lowDiskSpace` event triggers and starts the migration, as shown in Example 8-64.

Note: The `lowDiskSpace` event is checked every two minutes. If writing is intensive, as in our case, by the time the `lowDiskSpace` event is triggered, more than 50% of the available space has been used.

Also notice the time difference between when the file system reaches 50% utilization (Example 8-63) and the time when the migration process starts (Example 8-64); the system pool is already at 61% utilization.

Example 8-63 More than 50% utilization

Tue Jul 13 19:29:20 EDT 2010

| disk name | disk size in KB | failure holds group metadata | holds data | free KB in full blocks | free KB in fragments |
|---|--------------------|---------------------------------|---------------|---------------------------|-------------------------|
| ----- | | | | | |
| Disks in storage pool: system (Maximum disk size allowed is 219 GB) | | | | | |
| gpfs1nsd | 26214400 | 1 yes | yes | 13061632 (50%) | 10184 (0%) |
| gpfs2nsd | 26214400 | 2 yes | yes | 13053440 (50%) | 11424 (0%) |
| ----- | | | | | |
| (pool total) | 52428800 | | | 26115072 (50%) | 21608 (0%) |

Disks in storage pool: itso-stg-pool (Maximum disk size allowed is 219 GB)

| | | | | | |
|---------------|-----------|------|-----|-----------------|-------------|
| gpfs3nsd | 26214400 | 1 no | yes | 25393920 (97%) | 248 (0%) |
| gpfs4nsd | 26214400 | 2 no | yes | 25390848 (97%) | 248 (0%) |
| ----- | | | | | |
| (pool total) | 52428800 | | | 50784768 (97%) | 496 (0%) |
| ===== | | | | | |
| (data) | 104857600 | | | 76899840 (73%) | 22104 (0%) |
| (metadata) | 52428800 | | | 26115072 (50%) | 21608 (0%) |
| ===== | | | | | |
| (total) | 104857600 | | | 76899840 (73%) | 22104 (0%) |
| ...snipped... | | | | | |

Example 8-64 Data migration from system to itso-stg-pool

```

Tue Jul 13 19:30:21.107 2010: Calling User Exit Script MIGDISK: event lowDiskSpace, command
/work/end-to-end.sh.
[I] GPFS Current Data Pool Utilization in KB and %
itso-stg-pool 1644032 52428800 3.135742%
system 32086272 52428800 61.199707%
[I] 4054 of 52224 inodes used: 7.762714%.
[I] Loaded policy rules from /var/mmfs/tmp/tspolicyFile.mmapplypolicy.28706.
...snipped...
[I]2010-07-13@23:30:23.042 Policy evaluation. 45 files scanned.
[I]2010-07-13@23:30:23.046 Sorting 37 candidate file list records.
[I]2010-07-13@23:30:23.046 Choosing candidate files. 37 records scanned.
Chose to migrate 17920000KB: 7 of 37 candidates;
Chose to premigrate 0KB: 0 candidates;
Already co-managed 0KB: 0 candidates;
Chose to delete 0KB: 0 of 0 candidates;
Chose to list 0KB: 0 of 0 candidates;
0KB of chosen data is illplaced or illreplicated;
Predicted Data Pool Utilization in KB and %:
itso-stg-pool 19564032 52428800 37.315430%
system 14166272 52428800 27.020020%
Tue Jul 13 19:32:57.277 2010: Command: mmdf /dev/gpfs-ib atched. ....*....
Tue Jul 13 19:32:57.810 2010: Command: err 0: mmdf /dev/gpfs-ib
[I] MIGRATED /gpfs-ib/end-to-end.disk11 TO POOL itso-stg-pooled. ....@..
[I] MIGRATED /gpfs-ib/end-to-end.disk03 TO POOL itso-stg-pool
[I] MIGRATED /gpfs-ib/end-to-end.disk09 TO POOL itso-stg-pool
[I] MIGRATED /gpfs-ib/end-to-end.disk02 TO POOL itso-stg-pool
[I] MIGRATED /gpfs-ib/end-to-end.disk01 TO POOL itso-stg-pool
[I] MIGRATED /gpfs-ib/end-to-end.disk00 TO POOL itso-stg-pool
[I] MIGRATED /gpfs-ib/end-to-end.disk10 TO POOL itso-stg-pool
[I]2010-07-13@23:33:26.006 Policy execution. 7 files dispatched.
[I] A total of 7 files have been migrated, deleted or processed by an EXTERNAL EXEC/script;
0 'skipped' files and/or errors.

```

7. Write data until itso-stg-pool reaches 70% by using the **dd** command or a similar command. Because data goes only on the system pool, several migration policies start to migrate data from the system pool to the itso-stg-pool until the system pool processes 70% utilization. Because the migration process takes a while, the lowDiskSpace event is triggered several times, but the locking mechanism is prevented from starting the migration several times in parallel as shown in Example 8-65.

Example 8-65 Multiple runs of ene-to-end.sh

```
Wed Jul 14 08:55:57.804 2010: Calling User Exit Script MIGDISK: event lowDiskSpace,
command /work/end-to-end.sh.
EVENT_/work/end-to-end.sh: The lowDiskSpace event is still running since 2010-07-14
08:53:58.238792000 Exiting ...
```

8. As the *itso-stg-pool* fills up to 70%, as shown in Example 8-66, and the migration from system to itso-stg-pool finishes, by releasing the lock file, the migration from itso-stg-pool to Tivoli Storage Manager through external pool begins, as shown in Example 8-67 on page 375.

Example 8-66 Showing itso-stg-pool utilization

| disk name | disk size in KB | failure holds group metadata | holds data | free KB in full blocks | free KB in fragments |
|--|-----------------|------------------------------|------------|------------------------|----------------------|
| Disks in storage pool: system (Maximum disk size allowed is 219 GB) | | | | | |
| gpfs1nsd | 26214400 | 1 yes | yes | 18710528 (71%) | 20328 (0%) |
| gpfs2nsd | 26214400 | 2 yes | yes | 18703104 (71%) | 20272 (0%) |
| (pool total) | 52428800 | | | 37413632 (71%) | 40600 (0%) |
| Disks in storage pool: itso-stg-pool (Maximum disk size allowed is 219 GB) | | | | | |
| gpfs3nsd | 26214400 | 1 no | yes | 2302720 (9%) | 248 (0%) |
| gpfs4nsd | 26214400 | 2 no | yes | 2402048 (9%) | 248 (0%) |
| (pool total) | 52428800 | | | 4704768 (9%) | 496 (0%) |
| (data) | 104857600 | | | 42118400 (40%) | 41096 (0%) |
| (metadata) | 52428800 | | | 37413632 (71%) | 40600 (0%) |
| (total) | 104857600 | | | 42118400 (40%) | 41096 (0%) |

Example 8-67 Data migration

```
Wed Jul 14 08:57:57.857 2010: Calling User Exit Script MIGDISK: event lowDiskSpace, command
/work/end-to-end.sh.

running mmapplypolicy for the file system: gpfs-ib
[I] GPFS Current Data Pool Utilization in KB and %
itso-stg-pool 47724032 52428800 91.026367%
system 15014912 52428800 28.638672%
[I] 4066 of 52224 inodes used: 7.785692%.
[I] Loaded policy rules from /var/mmfs/tmp/tspolicyFile.mmapplypolicy.718.
Evaluating MIGRATE/DELETE/EXCLUDE rules with CURRENT_TIMESTAMP = 2010-07-14@12:57:58 UTC
parsed 1 Placement Rules, 0 Restore Rules, 2 Migrate/Delete/Exclude Rules,
0 List Rules, 1 External Pool/List Rules
RULE 'default' set POOL 'system'
RULE EXTERNAL POOL 'hsm' EXEC '/work/local.hsm.sh' OPTS '-v'
RULE 'system_to_itso-stg-pool' MIGRATE FROM POOL 'system' THRESHOLD(50,30) TO POOL 'itso-stg-pool'
WEIGHT(KB_ALLOCATED)
RULE 'itso-stg-pool_to_hsm' MIGRATE FROM POOL 'itso-stg-pool' THRESHOLD(70,50) TO POOL 'hsm'
WEIGHT(KB_ALLOCATED)
Wed Jul 14 09:03:58.005 2010: Calling User Exit Script MIGDISK: event lowDiskSpace, command
/work/end-to-end.sh.
EVENT_/work/end-to-end.sh: The lowDiskSpace event is still running since 2010-07-14 08:57:58.359696000
Exiting ...
...snipped...
/work/hsm.sh: MIGRATE /tmp/mmPolicy.ix.770.20EDB72A.1 -v: Ok
```

```
[I]2010-07-14@13:11:12.826 Policy execution. 9 files dispatched.  
[I] A total of 9 files have been migrated, deleted or processed by an EXTERNAL EXEC/script;  
    0 'skipped' files and/or errors.  
Event name lowDiskSpace is DONE
```

Conclusions

The GPSF policy engine is flexible but complex. The following guidelines can help you work with policies:

- ▶ Each rule applies to a specific file. Use as much selection criteria as possible to avoid matching files that should not match.
- ▶ When a policy runs, it creates a list of files to be migrated. After the list is created, only the files in that list are migrated, even if, meanwhile, other files might be written and match the selection criteria. Those files will be migrated at the next migration. This means that if the migration process takes a long time, while the file system is written to, it can fill up.
- ▶ Applying a policy generates considerable I/O, which can affect production. Also it can be time consuming so that when the policy finishes its I/O, the production application might start.
- ▶ Choose the thresholds wisely and leave enough free space on the file system. When a file is read, its contents are restored on the file system. After the file is located in the file system, it will be migrated back when the next migration policy runs.
- ▶ The thresholds use a percentage to specify the size of file system. The actual amount of data varies with the size of file system or storage pools. Do not copy policies from one file system to another without verifying the threshold values.
- ▶ If several migration processes overlap, the data that one migration process is moving is not moved by the other so that no data corruption occurs. However, avoid this situation if possible.
- ▶ When using Tivoli Storage Manager and HSM, choose the stub size (the part of the file that remains on the file system) based on application behavior. For example, if the application usually reads the first 20% of the file, the stub file should be 21% of the file size.
- ▶ In the file system, the HSM client creates a .SpaceMan directory, which can take some space from the file system. In our case it is about 1% of the space.

8.4 Backup and restore

Several ways are available to back up and restore a GPFS file system. One way is to use policies because they provide more flexibility. This way is described in 8.3, “Information lifecycle management scenarios” on page 346. Another way is to use GPFS backup tools as described in the following sections.

8.4.1 GPFS backup tools

The **mmbackup** command backs up file system structure such as the following structures:

- ▶ Entire file system
- ▶ Subdirectory
- ▶ Snapshot

Several differences exist between using the **mmbackup** and **mmrestore** commands, and using policies:

- ▶ The **mmbackup** command backs up the entire file system.
- ▶ By using policies for backup, the files that match the selection criteria are backed up. For example, symlinks are small so might not be selected when using the **WEIGHT(KB_ALLOCATED)** function.
- ▶ The **mmrestorefs** command requires the file system to be unmounted.

To back up and restore the gpfs-ib file system, perform the following steps:

1. Generate some data on the file system by using the **dd** command or similar command as shown in Example 8-68.

Example 8-68 Generating data by using the dd command

```
[root@australia gpfs-ib]# ll
total 15360000
-rw-r--r-- 1 root root 2621440000 Jun 14 16:59 data1
-rw-r--r-- 1 root root 2621440000 Jun 14 16:59 data2
-rw-r--r-- 1 root root 2621440000 Jun 14 17:00 data3
-rw-r--r-- 1 root root 2621440000 Jun 14 17:00 data4
-rw-r--r-- 1 root root 2621440000 Jun 14 17:01 data5
-rw-r--r-- 1 root root 2621440000 Jun 14 17:01 data6
```

2. Create a file set and link it in the gpfs-ib file system, as shown in Example 8-69, and then create several files.

Example 8-69 Create a file set and several files

```
[root@australia gpfs-ib]# mmcrfileset gpfs-ib itso-fileset
Fileset 'itso-fileset' created.
[root@australia gpfs-ib]# mmlinkfileset gpfs-ib itso-fileset -J itso
Fileset 'itso-fileset' linked at '/gpfs-ib/itso'.
[root@australia ~]# for i in `seq -w 1 6`; do dd if=/dev/zero
of=/gpfs-ib/data$i bs=1M count=2500; done
[root@australia gpfs-ib]# for i in `seq -w 1 6`; do dd if=/dev/zero
of=/gpfs-ib/itso/user$i bs=1M count=25; done

[root@australia gpfs-ib]# ll -R
.:
total 15360016
-rw-r--r-- 1 root root 2621440000 Jun 14 18:31 data1
-rw-r--r-- 1 root root 2621440000 Jun 14 18:32 data2
-rw-r--r-- 1 root root 2621440000 Jun 14 18:32 data3
-rw-r--r-- 1 root root 2621440000 Jun 14 18:32 data4
-rw-r--r-- 1 root root 2621440000 Jun 14 18:33 data5
-rw-r--r-- 1 root root 2621440000 Jun 14 18:33 data6
-r----- 1 root root      6076 Jun 14 18:45 dsmerror.log
drwx----- 2 root root      8192 Jul 14 2010 itso

./itso:
total 153600
-rw-r--r-- 1 root root 26214400 Jun 14 18:39 user1
-rw-r--r-- 1 root root 26214400 Jun 14 18:39 user2
-rw-r--r-- 1 root root 26214400 Jun 14 18:39 user3
-rw-r--r-- 1 root root 26214400 Jun 14 18:39 user4
```

```
-rw-r--r-- 1 root root 26214400 Jun 14 18:39 user5
-rw-r--r-- 1 root root 26214400 Jun 14 18:39 user6
```

The files in /gpfs/itso belong to itso-fileset, as shown in Example 8-70.

Example 8-70 List file in itso-fileset

```
[root@australia gpfs-ib]# mmlsattr -L /gpfs-ib/itso/user1
file name:          /gpfs-ib/itso/user1
metadata replication: 1 max 2
data replication:    1 max 2
immutable:          no
appendOnly:         no
flags:
storage pool name:  system
fileset name:      itso-fileset
snapshot name:
```

3. Run the **mmbackup** command and check the backed up files, as shown in Example 8-71 on page 378.

Example 8-71 The mmbackup command

```
[root@australia gpfs-ib]# mmbackup gpfs-ib
-----
Backup of /gpfs-ib begins at Mon Jun 14 17:14:09 EDT 2010.
-----
File system scan of gpfs-ib is complete.
Determining file system changes for gpfs-ib.
Sending files to the TSM server [14 changed, 7 expired].

mmbackup: TSM Summary Information:
    Total number of objects inspected:    23
    Total number of objects backed up:    15
    Total number of objects updated:      0
    Total number of objects rebound:     0
    Total number of objects deleted:      0
    Total number of objects expired:      8
    Total number of objects failed:       0
/opt/tivoli/tsm/client/ba/bin/dsmc selective /gpfs-ib/.mmbackupShadow.1.italy -servername=italy returned
rc=3072. Continuing with output analysisDone backing files to the TSM Server.
-----
mmbackup: Backup of /gpfs-ib completed on Mon Jun 14 17:24:38 EDT 2010.
-----
[root@australia gpfs-ib]# dsmc query backup -su=yes /gpfs-ib/
IBM Tivoli Storage Manager
Command Line Backup-Archive Client Interface
  Client Version 6, Release 2, Level 1.0
  Client date/time: 06/14/2010 19:09:35
(c) Copyright by IBM Corporation and other(s) 1990, 2010. All Rights Reserved.

Node Name: GPFS-IB
Session established with server ITALY: Linux/x86_64
  Server Version 6, Release 2, Level 1.0
  Data compression forced on by the server
  Server date/time: 07/14/2010 19:17:53  Last access: 07/14/2010 19:16:45
```

| Size | Backup Date | Mgmt Class | A/I File |
|------|-------------|------------|----------|
| ---- | ----- | ----- | --- ---- |

| | | | | | |
|---------------|---|---------------------|----------|---|----------------------------------|
| 32,768 | B | 07/14/2010 19:14:47 | STANDARD | A | /gpfs-ib/ |
| 8,192 | B | 07/14/2010 19:14:41 | STANDARD | A | /gpfs-ib/itso |
| 2,805 | B | 07/14/2010 19:14:47 | DEFAULT | A | /gpfs-ib/.mmbackupShadow.1.italy |
| 2,621,440,000 | B | 07/14/2010 19:04:28 | DEFAULT | A | /gpfs-ib/data1 |
| 2,621,440,000 | B | 07/14/2010 19:06:11 | DEFAULT | A | /gpfs-ib/data2 |
| 2,621,440,000 | B | 07/14/2010 19:07:51 | DEFAULT | A | /gpfs-ib/data3 |
| 2,621,440,000 | B | 07/14/2010 19:09:33 | DEFAULT | A | /gpfs-ib/data4 |
| 2,621,440,000 | B | 07/14/2010 19:11:14 | DEFAULT | A | /gpfs-ib/data5 |
| 2,621,440,000 | B | 07/14/2010 19:13:03 | DEFAULT | A | /gpfs-ib/data6 |
| 6,076 | B | 07/14/2010 19:14:41 | DEFAULT | A | /gpfs-ib/dsmerror.log |
| 26,214,400 | B | 07/14/2010 19:14:41 | DEFAULT | A | /gpfs-ib/itso/user1 |
| 26,214,400 | B | 07/14/2010 19:14:42 | DEFAULT | A | /gpfs-ib/itso/user2 |
| 26,214,400 | B | 07/14/2010 19:14:43 | DEFAULT | A | /gpfs-ib/itso/user3 |
| 26,214,400 | B | 07/14/2010 19:14:44 | DEFAULT | A | /gpfs-ib/itso/user4 |
| 26,214,400 | B | 07/14/2010 19:14:44 | DEFAULT | A | /gpfs-ib/itso/user5 |
| 26,214,400 | B | 07/14/2010 19:14:45 | DEFAULT | A | /gpfs-ib/itso/user6 |

- To test the **mmbackup** command, unlink the file set and retry the command, as shown in Example 8-72 on page 379. If unlinked file sets exist, **mmbackup** fails, by default. To overwrite, run the following command:

```
mmbackup gpfs-ib -f
```

Note: When a file set is unlinked, Tivoli Storage Manager marks the files belonging to the file set as *deleted*, because they do not appear on the file system. In Tivoli Storage Manager, deleted files have an expiration policy and are deleted when the policy time expires.

Example 8-72 The mmbackup fails if file sets are unlinked

```
[root@australia gpfs-ib]# mmunlinkfileset -h
Usage:
    mmunlinkfileset Device {FilesetName | -J JunctionPath} [-f]

[root@australia gpfs-ib]# mmunlinkfileset gpfs-ib -J itso
Fileset 'itso-fileset' unlinked.
[root@australia gpfs-ib]# mmbackup gpfs-ib
-----
Backup of /gpfs-ib begins at Mon Jun 14 19:21:34 EDT 2010.
-----
mmbackup: An unlinked fileset exists. Please check your filesets and try
again.
mmbackup: Command failed. Examine previous error messages to determine cause.
```

- Unmount the file system and delete it, as shown in Example 8-73.

Example 8-73 unmount and delete gpfs-ib file system

```
[root@australia ~]# mmumount gpfs-ib -a
Mon Jun 14 19:39:32 EDT 2010: mmumount: Unmounting file systems ...
[root@australia ~]# mmdelfs gpfs-ib
All data on following disks of gpfs-ib will be destroyed:
    gpfs1nsd
    gpfs2nsd
    gpfs3nsd
    gpfs4nsd
Completed deletion of file system /dev/gpfs-ib.
mmdelfs: Propagating the cluster configuration data to all
```

affected nodes. This is an asynchronous process.

6. Re-create the file system, as shown in Example 8-74.

Example 8-74 Re-creating the file system

```
[root@spain work]# cat nsd
# /dev/sdb:spain-gpfs,greece-gpfs::dataAndMetadata:1:::
gpfs1nsd:::dataAndMetadata:1:::
# /dev/sdc:greece-gpfs,spain-gpfs::dataAndMetadata:2:::
gpfs2nsd:::dataAndMetadata:2:::
[root@spain work]# cat nsd_stg_pool
# /dev/sdd:spain-gpfs,greece-gpfs::dataOnly:1::itso-stg-pool:
gpfs3nsd:::dataOnly:1::itso-stg-pool
# /dev/sde:greece-gpfs,spain-gpfs::dataOnly:2::itso-stg-pool:
gpfs4nsd:::dataOnly:2::itso-stg-pool
[root@spain work]# mmcrfs gpfs-ib -F nsd -z yes
```

The following disks of gpfs-ib will be formatted on node spain:

- gpfs1nsd: size 26214400 KB
- gpfs2nsd: size 26214400 KB

Formatting file system ...
Disks up to size 234 GB can be added to storage pool 'system'.
Creating Inode File
Creating Allocation Maps
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool 'system'
Completed creation of file system /dev/gpfs-ib.
mmcrfs: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.

```
[root@spain work]# mmcrfs gpfs-ib -F nsd -z yes -T /gpfs-ib
```

The following disks of gpfs-ib will be formatted on node spain:

- gpfs1nsd: size 26214400 KB
- gpfs2nsd: size 26214400 KB

Formatting file system ...
Disks up to size 234 GB can be added to storage pool 'system'.
Creating Inode File
Creating Allocation Maps
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool 'system'
Completed creation of file system /dev/gpfs-ib.
mmcrfs: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.

```
[root@spain work]# mmadddisk gpfs-ib -F nsd_stg_pool
```

The following disks of gpfs-ib will be formatted on node spain:

- gpfs3nsd: size 26214400 KB
- gpfs4nsd: size 26214400 KB

Extending Allocation Map
Creating Allocation Map for storage pool 'itso-stg-pool'
Flushing Allocation Map for storage pool 'itso-stg-pool'
Disks up to size 234 GB can be added to storage pool 'itso-stg-pool'.
Checking Allocation Map for storage pool 'itso-stg-pool'
Completed adding disks to file system gpfs-ib.

```
mmadddisk: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root@spain work]# mmmount gpfs-ib -a
Wed Jul 14 20:01:41 EDT 2010: mmmount: Mounting file systems ...
```

7. Restore the file system by running Tivoli Storage Manager command shown in Example 8-75.

Example 8-75 Restore the file system

```
[root@australia gpfs-ib]# dsmc restore /gpfs-ib/ -su=yes -replace=no
IBM Tivoli Storage Manager
Command Line Backup-Archive Client Interface
Client Version 6, Release 2, Level 1.0
Client date/time: 06/14/2010 20:22:46
(c) Copyright by IBM Corporation and other(s) 1990, 2010. All Rights Reserved.
```

```
Node Name: GPFS-IB
Session established with server ITALY: Linux/x86_64
Server Version 6, Release 2, Level 1.0
Data compression forced on by the server
Server date/time: 07/14/2010 20:31:05 Last access: 07/14/2010 20:30:48
```

Restore function invoked.

```
ANS1247I Waiting for files from the server...
Restoring 2,621,440,000 /gpfs-ib/data1 [Done]
Restoring 2,621,440,000 /gpfs-ib/data2 [Done]
Restoring 2,621,440,000 /gpfs-ib/data3 [Done]
Restoring 2,621,440,000 /gpfs-ib/data4 [Done]
Restoring 2,621,440,000 /gpfs-ib/data5 [Done]
Restoring 2,621,440,000 /gpfs-ib/data6 [Done]
ANS1946W File /gpfs-ib/dsmerror.log exists, skipping
Restoring 8,192 /gpfs-ib/itso [Done]
Restoring 26,214,400 /gpfs-ib/itso/user1 [Done]
Restoring 26,214,400 /gpfs-ib/itso/user2 [Done]
Restoring 26,214,400 /gpfs-ib/itso/user3 [Done]
Restoring 26,214,400 /gpfs-ib/itso/user4 [Done]
Restoring 26,214,400 /gpfs-ib/itso/user5 [Done]
Restoring 26,214,400 /gpfs-ib/itso/user6 [Done]
Restoring 32,768 /gpfs-ib/ [Done]
Restoring 2,805 /gpfs-ib/.mmbackupShadow.1.italy [Done]
```

Restore processing finished.

```
Total number of objects restored:      15
Total number of objects failed:         0
Total number of bytes transferred:    11.15 MB
Data transfer time:                    0.03 sec
Network data transfer rate:            293,777.99 KB/sec
Aggregate data transfer rate:          72.99 KB/sec
Elapsed processing time:                00:02:36
```

Note: The high transfer rate is shown because we are using files created from /dev/zero, which is highly compressible.

8. List the file attributes, as show in Example 8-76. The file set information is lost because it was not saved.

Example 8-76 Listing the file attributes

```
[root@australia gpfs-ib]# mmlsattr -L /gpfs-ib/itso/user1
file name:           /gpfs-ib/itso/user1
metadata replication: 1 max 2
data replication:    1 max 2
immutable:           no
appendOnly:          no
flags:
storage pool name:   system
fileset name:        root
snapshot name:
[root@australia gpfs-ib]# mmlsfileset gpfs-ib
Filesets in file system 'gpfs-ib':
Name                Status      Path
root                 Linked     /gpfs-ib
```

8.4.2 GPFS advanced backup tools

This section provide information about the GPFS advance backup tools. The following steps guide you to try these backup utilities within GPFS:

1. Re-create the environment from 8.4.1, “GPFS backup tools” on page 376, step 1 on page 377, as follows:
 - a. Unmount and delete the gpfs-ib file system.
 - b. Delete the backup.
 - c. Create the gpfs-ib file system.
 - d. Create a file set.
 - e. Create the same files within the itso-fileset file set, mounted in /gpfs-ib/itso.
2. Back up the file system by using the **dsmc backup /gpfs-ib/** command.
3. Back up the file system configuration by running the **mmbackupconfig** command, as shown in Example 8-77.

Example 8-77 Back up the file system configuration

```
[root@australia gpfs-ib]# mmbackupconfig gpfs-ib -o /work_gpfs-ib_mmbackupconfig

mmbackupconfig: Processing file system gpfs-ib ...
mmbackupconfig: Command successfully completed
```

4. Reset the environment as follows:
 - a. Unmount and delete the gpfs-ib file system.
 - b. Create the gpfs-ib file system.
5. Restore the file system configuration by running the **mmrestoreconfig** command, as shown in Example 8-78.

Example 8-78 Restore the file system configuration

```
[root@australia /]# mmrestoreconfig gpfs-ib -i /work_gpfs-ib_mmbackupconfig
-----
Configuration restore of gpfs-ib begins at Mon Jun 14 21:03:20 EDT 2010.
-----
```

```

mmrestoreconfig: Checking disk settings for gpfs-ib:
mmrestoreconfig: Checking the number of storage pools defined for gpfs-ib.
mmrestoreconfig: Checking storage pool names defined for gpfs-ib.
mmrestoreconfig: Checking storage pool size for 'system'.
mmrestoreconfig: Checking storage pool size for 'itso-stg-pool'.

mmrestoreconfig: Checking filesystem attribute configuration for gpfs-ib:
Filesystem attribute value for stripeMethod restored.
Filesystem attribute value for logicalSectorSize restored.
Filesystem attribute value for minFragmentSize restored.
Filesystem attribute value for inodeSize restored.
Filesystem attribute value for indirectBlockSize restored.
Filesystem attribute value for defaultMetadataReplicas restored.
Filesystem attribute value for maxMetadataReplicas restored.
Filesystem attribute value for prefetchBuffers restored.
Filesystem attribute value for defaultDataReplicas restored.
Filesystem attribute value for maxDataReplicas restored.
Filesystem attribute value for blockAllocationType restored.
Filesystem attribute value for maxExpectedDiskI/Olatency restored.
Filesystem attribute value for fileLockingSemantics restored.
Filesystem attribute value for ACLSemantics restored.
Filesystem attribute value for estimatedAverageFileSize restored.
Filesystem attribute value for numNodes restored.
Filesystem attribute value for maxConcurrentI/OoperationsPerDisk restored.
Filesystem attribute value for blockSize restored.
Filesystem attribute value for quotasEnforced restored.
Filesystem attribute value for defaultQuotasEnabled restored.
Filesystem attribute value for filesystemVersion restored.
Filesystem attribute value for filesystemVersionLocal restored.
Filesystem attribute value for filesystemVersionManager restored.
Filesystem attribute value for filesystemVersionOriginal restored.
Filesystem attribute value for filesystemHighestSupported restored.
Filesystem attribute value for aggressivenessLevelOfTokensPrefetch restored.
Filesystem attribute value for supportForLargeLUNs restored.
Filesystem attribute value for DMAPIEnabled restored.
Filesystem attribute value for logfileSize restored.
Filesystem attribute value for exactMtime restored.
Filesystem attribute value for suppressAtime restored.
Filesystem attribute value for strictReplication restored.
Filesystem attribute value for create-time restored.
Filesystem attribute value for fastEAenabled restored.
Filesystem attribute value for filesystemFeatures restored.
Filesystem attribute value for filesetdfEnabled restored.
Filesystem attribute value for maxNumberOfInodes restored.
Filesystem attribute value for maxSnapshotId restored.
Filesystem attribute value for storagePools restored.
Filesystem attribute value for defaultMountPoint restored.
Filesystem attribute value for automaticMountOption restored.
Filesystem attribute value for additionalMountOptions restored.

mmrestoreconfig: Checking fileset configurations for gpfs-ib:
Fileset 'itso-fileset' created.

mmrestoreconfig: Mounting file system gpfs-ib to link filesets.
Fileset 'itso-fileset' linked at '/gpfs-ib/itso'.
mmrestoreconfig: Fileset link satus has been restored. Unmounting file system gpfs-ib.
Mon Jun 14 21:03:33 EDT 2010: mmumount: Unmounting file systems ...

mmrestoreconfig: Checking policy rule configuration for gpfs-ib:
mmrestoreconfig: No policy rules installed in backed up filesystem gpfs-ib.

mmrestoreconfig: Checking quota settings for gpfs-ib:
mmrestoreconfig: The tscheckquota command failed.

```

```
/usr/lpp/mmfs/bin/mmrestoreconfig[3244]: /var/mmfs/mmbbackup/quotafile.restore: cannot open  
[No such file or directory]
```

6. Restore the data from Tivoli Storage Manager by running the following command:

```
dsmc restore /gpfs-ib/ -su=yes
```
7. Check the file attributes, as shown in Example 8-79.

Example 8-79 file attributes

```
[root@australia gpfs-ib]# mmlsattr -L /gpfs-ib/itso/user1  
file name:          /gpfs-ib/itso/user1  
metadata replication: 1 max 2  
data replication:    1 max 2  
immutable:          no  
appendOnly:         no  
flags:  
storage pool name:   system  
fileset name:        itso-fileset  
snapshot name:
```

8.4.3 Conclusions

The **mmbbackup** command relies on the Tivoli Storage Manager commands to back up the file system. Although the user data is saved, not all the metadata is saved because the Tivoli Storage Manager tools do not know about it.

The **mmbbackupconfig** command saves the metadata information:

- ▶ Block size
- ▶ Replication factors
- ▶ Number and size of disks
- ▶ Storage pool layout
- ▶ File sets and junction points
- ▶ Policy rules
- ▶ Quota information
- ▶ A number of other file system attributes.

The output of the **mmbbackupconfig** command must also be backed up. The backup step can be integrated in the backup script. The **mmrestoreconfig** command restores the configuration of the file system from a text file, which has to be retrieved from Tivoli Storage Manager, which is usually a manual process.

8.5 Advanced external pool scenario

In this section, we show several possible scenarios using a custom built external pool. The purpose is to provide a guideline for how to integrate other external storage pools.

Note: The information in this section is provided *as is*, and is not appropriate for a production environment without improvements.

8.5.1 Off-line compressed storage

Off-line compressed storage means that we compress the files that are on the GPFS file system, based on policy-defined criteria and by using Linux compression tools. When needed a file can be decompressed and used by the application. The off-line compressed storage has the following characteristics:

- ▶ It is referred to as *off-line* because after the files are compressed an application cannot use them, by default. An external action must be performed to decompress them before the application can use the files again.
- ▶ Compression is done on each file, so the directory structure remains the same. For example, the `/gpfs-ib/file1.txt` file becomes `/gpfs-ib/file1.txt.gz` after compression.
- ▶ Selection criteria is based on the GPFS policy engine, using the same flexible rules as with any other rule.
- ▶ To decompress the files, another policy has to be active or based on external criteria that the `mmapplypolicy` command uses.
- ▶ The compressed files are still in the GPFS file system, they can match different rules and can use the selection criteria as any other rule.

Using off-line compressed storage offers several advantages:

- ▶ Some files can be archived with high-ratio compression rate, depending on the file structure. Files such as MP3 or video format are already compressed.
- ▶ Accessing the files can be faster and simpler because they are already on the file system (simply decompress them before using the files).
- ▶ For files that are seldom accessed, compressing them saves space.

Note: Compressing files does not exclude using backup tools, such as Tivoli Storage Manager, because the files are still on the same disks that are subject to failure.

8.5.2 Implementation scenario

We use `gzip` and `gunzip` commands for compressing and decompressing files.

Compressing the files works as follows:

1. Based on several selection criteria, the GPFS policy engine creates a list of matching files.
2. The `gzip` tool uses the list file to compress each file in the list.

Decompressing the files works as follows:

1. Based on several selection criteria, the GPFS policy engine creates a list of matching compressed files. In our case, the compressed files have the `.gz` file name extension.
2. The `gunzip` tool uses the list file to decompress each file in the list.

The following steps describe how to create a policy to compress the files:

1. Delete all file in the `gpfs-ib` file system.
2. Create the policy that compresses the files, as shown in Example 8-80. Each rule has the following meaning:
 - First rule defines the external list and the script that will handle it.

- Second rule defines a rule that acts when the system storage pool reaches 20% utilization, selects all the files that have the .out extension, calls the gzip external list and sorts the files by size, in descending order.
- Third rule defines a rule that places the files with the .gz extension in itso-stg-pool. In our example, all compressed files are placed in a dedicated storage that can use slower disks.
- Fourth rule specifies that all other files are placed in system pool.

Example 8-80 the compress policy

```
[root@spain work]# cat gzip.policy
RULE EXTERNAL LIST 'gzip' EXEC '/work/gzip.sh'
RULE 'compress' LIST 'gzip' FROM POOL 'system' THRESHOLD(20,10)
WEIGHT(KB_ALLOCATED) WHERE (name) like '%.out'
RULE 'compress-files' SET POOL 'itso-stg-pool' WHERE (name) like '%.gz'
RULE 'default' set POOL 'system'
```

The **gzip.sh** script is compressing the files based on the list that the policy engine generated, as shown in Example 8-81 on page 386. For troubleshooting, export the received parameters, for example, by using **echo param0=\$0** for the first parameter.

Example 8-81 The gzip.sh file

```
[root@spain work]# cat gzip.sh
#!/bin/sh
echo param0=$0
echo param1=$1
echo param2=$2
echo param3=$3
echo param4=$4
case $1 in
    LIST)          # Simply cat the filelist to stdout
        for i in `cat $2 | awk '{ print $5 }'`
        do
            echo gzip file $i
            gzip $i
        done
        echo END
        rc=0
        ;;
    TEST)          # Respond with success
        rc=0
        ;;
    *)              # Command not supported by this script
        rc=1
        ;;
esac
exit $rc
```

3. Apply the policy by running the following command:

```
mmchpolicy gpfs-ib /work/gzip.policy
```

4. Generate some files with the .out extension and make sure the file system reaches more than 20% utilization so the policy will match.
5. Apply the policy by running the following command:

```
mmapplypolicy gpfs-ib -P /work/gzip.policy
```

Note: A callback script can be used to trigger the policy automatically based on *lowDiskSpace* event, but for ease of understanding, we use a manual process.

The result of applying the policy is shown in Example 8-82.

Example 8-82 Applying the compress policy

```
[root@spain work]# ll /gpfs-ib/
total 11264008
-rw-r--r-- 1 root root 1048576000 Jul 15 18:17 data1
-rw-r--r-- 1 root root 1048576000 Jul 15 18:17 data2
-rw-r--r-- 1 root root 3145728000 Jul 15 18:18 data3
drwx----- 2 root root      8192 Jun 15 17:42 itso
-rw-r--r-- 1 root root 2097152000 Jul 15 18:18 user1.out
-rw-r--r-- 1 root root 2097152000 Jul 15 18:19 user2.out
-rw-r--r-- 1 root root 2097152000 Jul 15 18:19 user3.out
[root@spain work]# mmapplypolicy gpfs-ib -P gzip.policy -L3
...snipped...
parsed 2 Placement Rules, 0 Restore Rules, 0 Migrate/Delete/Exclude Rules,
      1 List Rules, 1 External Pool/List Rules
RULE EXTERNAL LIST 'gzip' EXEC '/work/gzip.sh'
RULE 'compress' LIST 'gzip' FROM POOL 'system' THRESHOLD(20,10)
WEIGHT(KB_ALLOCATED) WHERE (name) like '%.out'
RULE 'compress-files' SET POOL 'itso-stg-pool' WHERE (name) like '%.gz'
RULE 'default' set POOL 'system'
param0=/work/gzip.sh
param1=TEST
param2=/gpfs-ib
param3=
param4=
[I]2010-07-16@01:24:49.012 Directory entries scanned: 8.
[I] Directories scan: 6 files, 2 directories, 0 other objects, 0 'skipped'
files and/or errors.
[I]2010-07-16@01:24:49.015 Sorting 8 file list records.
/gpfs-ib/user2.out4:49.0 RULE 'compress' LIST 'gzip' FROM POOL 'system'
WEIGHT(2048000.000000)
/gpfs-ib/user3.out      RULE 'compress' LIST 'gzip' FROM POOL 'system'
WEIGHT(2048000.000000)
/gpfs-ib/user1.out      RULE 'compress' LIST 'gzip' FROM POOL 'system'
WEIGHT(2048000.000000)
[I] Inodes scan: 6 files, 2 directories, 0 other objects, 0 'skipped' files
and/or errors.
[I]2010-07-16@01:24:49.056 Policy evaluation. 8 files scanned.
[I]2010-07-16@01:24:49.059 Sorting 3 candidate file list records.
[I]2010-07-16@01:24:49.060 Choosing candidate files. 3 records scanned.
[I] Summary of Rule Applicability and File Choices:
Rule#  Hit_Cnt KB_Hit  Chosen  KB_Chosen      KB_Ill Rule
   0      3      6144000  3      6144000  0      RULE 'compress' LIST 'gzip'
FROM POOL 'system' THRESHOLD(20,10) WEIGHT(.) WHERE(.)
```

[I] Filesystem objects with no applicable rules: 5.

[I] GPFS Policy Decisions and File Choice Totals:

Chose to migrate OKB: 0 of 0 candidates;

Chose to premigrate OKB: 0 candidates;

Already co-managed OKB: 0 candidates;

Chose to delete OKB: 0 of 0 candidates;

Chose to list 614400KB: 3 of 3 candidates;

OKB of chosen data is illplaced or illreplicated;

Predicted Data Pool Utilization in KB and %:

itso-stg-pool 10240 52428800 0.019531%

system 11452928 52428800 21.844727%

param0=/work/gzip.sh49.060 Policy execution. 0 files dispatched. .\.....

param1=LIST

param2=/tmp/mmPolicy.ix.9667.B4633B10.1

param3=

param4=

gzip file /gpfs-ib/user2.out

gzip file /gpfs-ib/user3.outolicy execution. 3 files dispatched.<...

gzip file /gpfs-ib/user1.outolicy execution. 3 files dispatched. ..%.....

END2010-07-16@01:27:49.010 Policy execution. 3 files dispatched.@..

[I]2010-07-16@01:27:52.366 Policy execution. 3 files dispatched.

[I] A total of 3 files have been migrated, deleted or processed by an EXTERNAL EXEC/script;

0 'skipped' files and/or errors.

[root@spain work]# ll /gpfs-ib/

total 5125984

-rw-r--r-- 1 root root 1048576000 Jul 15 18:17 data1

-rw-r--r-- 1 root root 1048576000 Jul 15 18:17 data2

-rw-r--r-- 1 root root 3145728000 Jul 15 18:18 data3

drwx----- 2 root root 8192 Jun 15 17:42 itso

-rw-r--r-- 1 root root 2035269 Jul 15 18:18 **user1.out.gz**

-rw-r--r-- 1 root root 2035269 Jul 15 18:19 **user2.out.gz**

-rw-r--r-- 1 root root 2035269 Jul 15 18:19 **user3.out.gz**

The placement policy also matches for the .gz files and places them in itso-stg-pool, as shown in Example 8-83.

Example 8-83 Placement policy

[root@spain work]# mmlsattr -L /gpfs-ib/user1.out.gz

file name: /gpfs-ib/user1.out.gz

metadata replication: 1 max 2

data replication: 1 max 2

immutable: no

appendOnly: no

flags:

storage pool name: itso-stg-pool

fileset name: root

snapshot name:

6. To decompress, create the a policy shown in Example 8-84. This policy decompress all files that are placed in itso-stg-pool and have the .gz extension. According to the default placement policy, the decompressed file are placed in system pool.

Example 8-84 The gunzip.policy file

```
[root@spain work]# cat gunzip.policy
RULE EXTERNAL LIST 'gunzip' EXEC '/work/gunzip.sh'
RULE 'decompress' LIST 'gunzip' FROM POOL 'itso-stg-pool' WHERE (name) like
'%.gz'
```

The /work/gunzip.sh file is shown in Example 8-85. It is similar to the /work/gzip.sh file except it decompresses files.

Example 8-85 The gunzip.sh file

```
[root@spain work]# cat gunzip.sh
...snipped...
LIST)      # Simply cat the filelist to stdout
            for i in `cat $2 | awk '{ print $5 }'`
            do
            echo gzip file $i
            gunzip $i
            done
            echo END
...snipped...
```

7. Apply the decompress policy by running the following command, also shown in Example 8-86:

```
mmapplypolicy gpfs-ib -P gunzip.policy
```

Example 8-86 Applying the decompress policy

```
[root@spain work]# mmapplypolicy gpfs-ib -P gunzip.policy -L3
[...snipped...]
[I]2010-07-16@01:43:44.172 Sorting 8 file list records.
/gpfs-ib/user2.out.gz4.1 RULE 'decompress' LIST 'gunzip' FROM POOL
'itso-stg-pool' WEIGHT(inf)
/gpfs-ib/user3.out.gz    RULE 'decompress' LIST 'gunzip' FROM POOL
'itso-stg-pool' WEIGHT(inf)
/gpfs-ib/user1.out.gz    RULE 'decompress' LIST 'gunzip' FROM POOL
'itso-stg-pool' WEIGHT(inf)
[I] Inodes scan: 6 files, 2 directories, 0 other objects, 0 'skipped' files
and/or errors.
[I]2010-07-16@01:43:44.214 Policy evaluation. 8 files scanned.
[I]2010-07-16@01:43:44.216 Sorting 3 candidate file list records.
[I]2010-07-16@01:43:44.217 Choosing candidate files. 3 records scanned.
[I] Summary of Rule Applicability and File Choices:
Rule#  Hit_Cnt KB_Hit  Chosen  KB_Chosen    KB_Ill  Rule
    0      3    5976    3    5976    0    RULE 'decompress' LIST 'gunzip'
FROM POOL 'itso-stg-pool' WHERE(.)

[I] Filesystem objects with no applicable rules: 5.

[I] GPFS Policy Decisions and File Choice Totals:
Chose to migrate OKB: 0 of 0 candidates;
Chose to premigrate OKB: 0 candidates;
Already co-managed OKB: 0 candidates;
Chose to delete OKB: 0 of 0 candidates;
Chose to list 5976KB: 3 of 3 candidates;
OKB of chosen data is illplaced or illreplicated;
```

```
...snipped...
gzip file /gpfs-ib/user2.out.gz
gzip file /gpfs-ib/user3.out.gzcy execution. 3 files dispatched. ..|.....
gzip file /gpfs-ib/user1.out.gzcy execution. 3 files dispatched. .../.....
END2010-07-16@01:44:44.160 Policy execution. 3 files dispatched. .....<...
[I]2010-07-16@01:44:47.021 Policy execution. 3 files dispatched.
[I] A total of 3 files have been migrated, deleted or processed by an EXTERNAL
EXEC/script;
    0 'skipped' files and/or errors.
```

After decompressing, review the files and their location, shown Example 8-87.

Example 8-87 Placement policy

```
[root@spain work]# ll /gpfs-ib/
total 11264008
-rw-r--r-- 1 root root 1048576000 Jul 15 18:17 data1
-rw-r--r-- 1 root root 1048576000 Jul 15 18:17 data2
-rw-r--r-- 1 root root 3145728000 Jul 15 18:18 data3
drwx----- 2 root root      8192 Jun 15 17:42 itso
-rw-r--r-- 1 root root 2097152000 Jul 15 18:18 user1.out
-rw-r--r-- 1 root root 2097152000 Jul 15 18:19 user2.out
-rw-r--r-- 1 root root 2097152000 Jul 15 18:19 user3.out
[root@spain work]# mmlsattr -L /gpfs-ib/user1.out
file name:           /gpfs-ib/user1.out
metadata replication: 1 max 2
data replication:     1 max 2
immutable:            no
appendOnly:           no
flags:
storage pool name:  system
fileset name:         root
snapshot name:
```

Archived

Disaster recovery using GPFS

This chapter describes disaster recovery (DR) configurations and solutions using GPFS. Disaster recovery solutions seek to address the high-availability requirement for large GPFS customers with database systems. To support and manage a DR environment, GPFS introduces the `mmfsctl` command, described in this chapter.

This chapter contains the following topics:

- ▶ 9.1, “Disaster recovery solution using GPFS replication” on page 394
- ▶ 9.2, “The GPFS `mmfsctl` command” on page 396

9.1 Disaster recovery solution using GPFS replication

This section describes how GPFS replication can be implemented as a DR solution.

The DR model using GPFS replication consists of a single GPFS cluster that is defined across two geographic sites and a third tiebreaker site. The goal is to keep one of the sites operational in case the other site fails.

9.1.1 Configuration

This DR configuration consist of production sites (A and B) plus a tiebreaker site (C), as shown in Figure 9-1.

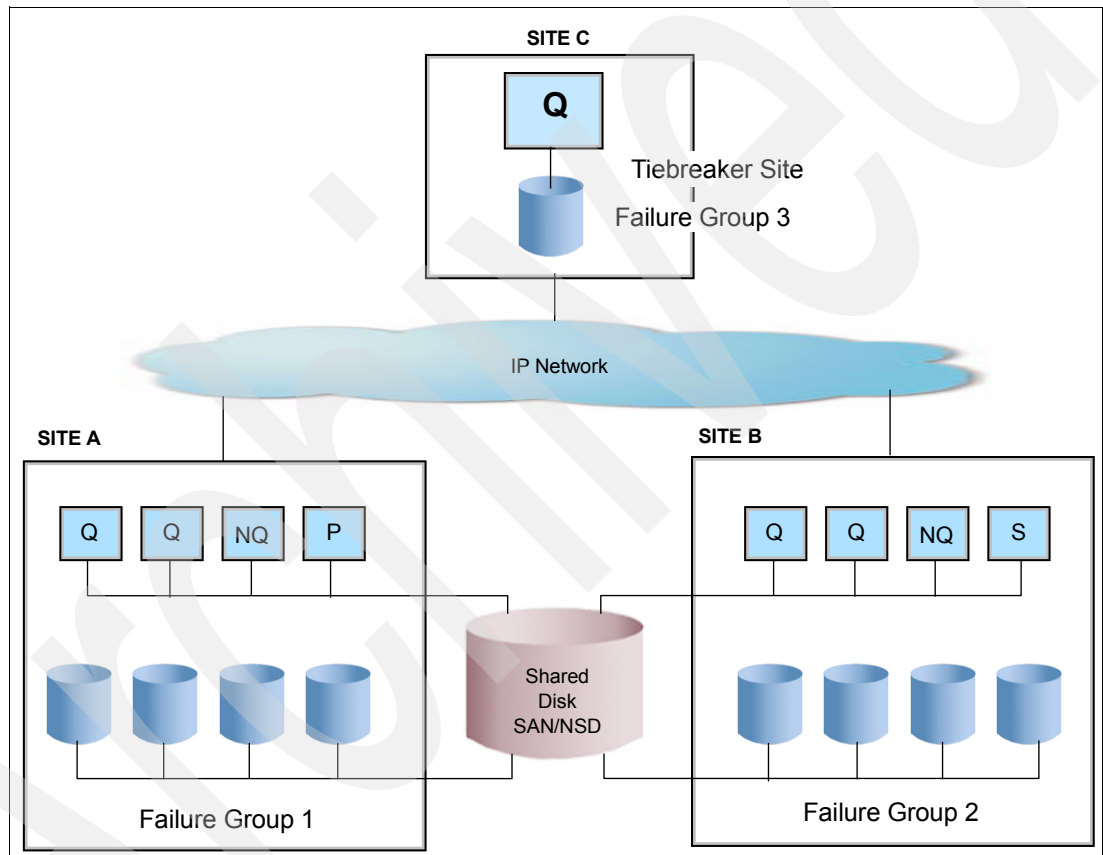


Figure 9-1 Three-site DR configuration with GPFS replication

9.1.2 Characteristics of this DR configuration

The production sites (A and B) are used for daily production work. The tiebreaker site (C) has the following roles:

- ▶ Serve to maintain node and file-system descriptor quorums after a site failure.
- ▶ Hold an additional quorum node.
- ▶ Hold a *file system descriptor-only* (descOnly) disk in a third failure group.

Additional characteristics of the DR configuration are as follows:

- ▶ Storage on the production sites is shared.
- ▶ Number of quorum nodes at each production site must be the same.
- ▶ Disks are split into two failure groups (one at each site).
- ▶ Configuration consists of one or more site-wide replicated GPFS file systems.
- ▶ Primary cluster configuration data server is configured on site A.
- ▶ Secondary cluster configuration data server is configured on site B.

After a production site failure

After a production site failure, no administrative intervention is required.

GPFS detects the failure and reacts to it as follows:

- ▶ The failed nodes are marked as *down*.
- ▶ The failed disks are marked as *unavailable*.
- ▶ The application continues running at the surviving site.

After site recovery

Perform the following steps:

1. Restart GPFS on all nodes at the recovered site:

```
mmstartup -a
```

2. Bring the recovered disks online:

```
mmchdisk ... start
```

Failure on the production site and the tiebreaker site

GPFS loses quorum and the file system is unmounted after the failure occurs.

The administrator initiates the manual takeover procedure:

1. Relaxes the node quorum:

```
mmchconfig designation=nonquorum ...
```

2. Relaxes the file system descriptor quorum:

```
mmfsctl ... exclude
```

9.2 The GPFS mmfsctl command

The **mmfsctl** GPFS command implements all disaster recovery functionality:

- ▶ Use the following command before creating a FlashCopy backup. It suspends file-system I/O and *flushes the GPFS cache* to ensure the integrity of the FlashCopy image:

```
mmfsctl Device {suspend | resume}
```

- ▶ Use this command with active-passive PPRC-based configurations. It synchronizes the file system's configuration state between peer recovery clusters:

```
mmfsctl Device syncFSconfig {-n RemoteNodesFile | -C RemoteCluster} [-S  
SpecFile]
```

- ▶ Use this command for minority takeover in Active-Active replicated configurations. It tells GPFS to exclude the specified disks or failure groups from the file system descriptor quorum:

```
mmfsctl Device {exclude | include} {-d DiskList | -F DiskFile | -G  
FailureGroup}
```

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 399. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *HPC Clusters Using InfiniBand on IBM Power Systems Servers*, SG24-7767
- ▶ *IBM Midrange System Storage Implementation and Best Practices Guide*, SG24-6363
- ▶ *IBM PowerVM Virtualization Managing and Monitoring*, SG24-7590
- ▶ *IBM TotalStorage Enterprise Storage Server Implementing ESS Copy Services in Open Environments*, SG24-5757
- ▶ *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940
- ▶ *Tivoli Storage Manager V6.1 Technical Guide*, SG24-7718

Other publications

These publications are also relevant as further information sources:

- ▶ *General Parallel File System Administration and Programming Reference Version 3 Release 3*, SA23-2221
- ▶ *General Parallel File System Administration and Programming Reference Version 3 Release 4*, SA23-2221
- ▶ *General Parallel File System Advanced Administration Guide Version 3 Release 3*, SC23-5182
- ▶ *General Parallel File System Advanced Administration Guide Version 3 Release 4*, SC23-5182
- ▶ *General Parallel File System Concepts, Planning, and Installation Guide Version 3 Release 3*, GA76-0413
- ▶ *General Parallel File System Concepts, Planning, and Installation Guide Version 3 Release 4*, GA76-0413
- ▶ *General Parallel File System Data Management API Guide Version 3 Release 3*, GA76-0414
- ▶ *General Parallel File System Data Management API Guide Version 3 Release 4*, GA76-0414

Online resources

These websites are also relevant as further information sources:

- ▶ IBM General Parallel File System main page
<http://www.ibm.com/systems/software/gpfs/index.html>
- ▶ IBM General Parallel File System resources
<http://www.ibm.com/systems/software/gpfs/resources.html>
- ▶ IBM Redbooks GPFS search
<http://publib-b.boulder.ibm.com/Redbooks.nsf/Redbooks?SearchView&Query=GPFS&SearchMax=4999>
- ▶ GPFS FAQs
http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.cluster.gpfs.doc/gpfs_faqs/gpfs_faqs.html
- ▶ GPFS technical discussion forum
http://www.ibm.com/developerworks/forums/dw_forum.jsp?forum=479&cat=13
- ▶ Operating system support
http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.cluster.gpfs.doc/gpfs_faqs/gpfsclustersfaq.html
- ▶ InfiniBand
<http://www.infinibandta.org>
- ▶ IBM Intelligent Cluster
<http://www.ibm.com/systems/x/hardware/cluster/index.html>
- ▶ Power Systems high performance computing solutions
<http://www.ibm.com/systems/power/hpc.html>
- ▶ SAN Volume Controller
 - <http://www.ibm.com/systems/storage/software/virtualization/svc/interop.html>
 - http://www.ibm.com/support/docview.wss?rs=540&context=ST52G7&dc=D430&uid=ssglS4000350&loc=en_US&cs=utf-8&lang=en#SVC
- ▶ IBM Smart Analytics System:
<http://www.ibm.com/software/data/infosphere/support/smart-analytics-system/>
- ▶ IBM and SAP
<http://www.ibm.com/solutions/sap/us/en/>
- ▶ NFSv4 protocol
<http://www.nfsv4.org/>
- ▶ GPFS fixes from the IBM GPFS support
<http://www14.software.ibm.com/webapp/set2/sas/f/gpfs/home.html>
- ▶ VIOS documentation
<http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphat/iphblconfigurelparp6.htm>

- ▶ Virtual fibre channel for HMC-managed systems
<http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphathatvfc.htm>
- ▶ Virtual fibre channel on IVM-managed systems
<http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/topic/iphathatvfcivm.htm>
- ▶ Tivoli Storage Manager manuals or web documentation:
<http://publib.boulder.ibm.com/infocenter/tsminfo/v6r2/index.jsp>

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks publications, at this website:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Archived

Index

Symbols

.awk all 265

Numerics

0xFFFFFFFFFFFFFFFFF usa 261–262, 265, 272–274, 276–278

1M count 351–352, 354, 377

A

ACL semantics 138, 218

Active Directory 98, 101, 104

Admin IP 149

affected node 79–80, 82, 84, 92, 95, 110–112, 114, 117–118, 154, 156, 158–161, 165–166, 168–170, 184–185, 196, 198–199, 202, 204, 206, 209, 211–212, 215, 218–219, 351, 380–381

AIX client

node 97

nodes access 97

al command 199

algeria 121, 134, 137–142, 144–149, 151, 155–156, 158–161

Allocation Map 82, 93, 112, 169, 197–198, 204, 211, 215, 285, 351, 380

argentina-gpfs 78

asynchronous process 79–80, 82, 84, 92, 95, 154, 156, 159–161, 165–166, 168–170, 184–185, 196, 198–199, 202, 204, 206, 209, 211–212, 215, 218–219, 351, 380–381

australia-gpfs 74, 78–81, 83, 86, 88–89, 149, 152–153, 157

B

bin/echo Event 358, 373

block size 138, 212, 217, 232–233, 363, 384

Bronx cluster 150–151, 153, 155–156, 159–160

file systems 161

GPFS communication network 155

gpfs daemons 153

remote cluster definitions 159

Brooklyn cluster 150–151

file systems 159

proper location 154

remote cluster definitions 161

SSL keys 154, 156

BUF site 163, 182

single disk 167

C

cat gunzip.sh 389

chdev command 293, 300, 310

Client date/time 345, 349, 365–366, 369, 378, 381

client_reserve parameter 295

cluster configuration data 79–80, 82, 84, 92–93, 95, 110–112, 114, 117–118, 154–156, 158–161, 165–170, 184–185, 196–199, 202, 204, 206–207, 209–212, 215, 218–219, 351, 379–381

cluster node 74, 77, 88, 199, 203, 207, 349, 365

adapter failure 87

current licenses 119

file system 83

GPFS licenses 79

cluster setup 109, 122, 150

collisions

0 txqueuelen 1000 86–88

results 246

command 263

cat 369

cfgmgr 290

chattr 240

chdev 285–286, 289, 291

chpath 293

chsyscfg 315

cp 267

date 256

db2cluster 147

dd 275

df 87

dsmc backup /gpfs-ib/ 382

dsmmigfs 365

entstat 241, 246

errpt -a 277

ethtool 240–241

exportfs -ra 66

fdisk 297

gpfs.snap 256–257

gpfsperf 93

gunzip 385

gzip 385

ifconfig 86

installp 194

iostat 18, 56, 270

ip 239

iptrace 275

ln 324

lparstat 283

lparstat -i 176

ls 66, 267

ls -l 235

lsattr 240, 289

lscfg 288

lshwres 288

lsmmap 285–286, 288, 302, 308

lsnports 302

lspath 290, 292

lssam 147

lsslot 288

mexportfs 218
 mkdir 267
 mkinitrd 296
 mkvdev 285–286
 mmaddcallback 74, 84, 224–225
 mmadddisk 204, 214, 321
 mmapplypolicy 356
 mmapplypolicy gpfs-ib 354
 mmauth 154, 199, 207
 mmauth add 158, 160
 mmauth genkey new 154
 mmauth grant 158
 mmbackup 325, 376
 mmbackupconfig 325–326, 382
 mmchattr 64, 236
 mmchattr -P 322
 mmchcluster 45, 185
 mmchconfig 32, 47, 50, 57, 66, 156, 167, 206–207, 235, 248, 271
 mmchconfig adminMode=allToall 44
 mmchconfig adminMode=central 44
 mmchconfig cipherList=AUTHONLY 154–155
 mmchconfig designation=nonquorum 395
 mmchdisk 174, 186, 203
 mmchdisk ... start 395
 mmchfs 56
 mmchfs -Q 319
 mmchfs -V compat 193
 mmchfs -V full 193
 mmchnode 185, 206, 227
 mmchnode --nonquorum 182–183
 mmchnsd 203, 220, 222
 mmchpolicy 326, 328
 mmchpolicy -l test 328
 mmcrcluster 45, 164
 mmcrfs 60, 62–63, 82, 168, 210, 233, 321
 mmcrnsd 48, 54–55, 80, 111, 167, 321
 mmcrsnapshot 271, 318
 mmcrsnapshot gpfs-ib snap1 348
 mmcrvsd 55
 mmdefragfs 222
 mmdefragfs -i 223
 mmdegragfs 233
 mmdelcallback 224
 mmdeldisk 322
 mmdelfileset 271
 mmdelnsd 54
 mmdelsnapshot 271
 mmdf 198, 215, 254, 357
 mmdf -P 321
 mmdiag 250, 258
 mmdiag --network 270
 mmdiag --trace 260
 mmdiag --waiters 250, 259
 mmdsh 256
 mmexportfs 217, 222
 mmfsadm 250, 258
 mmfsadm cleanup 271
 mmfsadm dump 258
 mmfsadm dump all 260, 267
 mmfsadm dump fs 278
 mmfsadm dump tscomm 270
 mmfsadm dump waiters 259
 mmfsctl 163, 184–185, 393
 mmfsctl ... exclude 395
 mmfsctl exclude 182
 mmfsctl suspend 271
 mmfsenv -u 194, 268
 mmgetstate 139, 170–171, 173, 177, 180, 186
 mmgetstate -a 267
 mmgetstate -aL 199
 mmgetstate -aLs 157
 mmimportfs 222
 mmimports 217
 mmlinkfileset 323
 mmlsattr 205, 236, 354
 mmlsattr -L 323
 mmlscallback 84, 224, 358
 mmlscluster 151–152, 199
 mmlscluster | grep quorum 267
 mmlsconfig 169
 mmlsdisk 172, 175, 181, 321
 mmlsfileset 325
 mmlsfs 233
 mmlsfs -P 321
 mmlslicense 166
 mmlsmgr 270
 mmlsmount 86, 88, 139, 172, 175, 178, 181, 183
 mmlsnd -F 203
 mmlsnd 19, 168
 mmlspolicy 328
 mmlssnapshot 318
 mmmount 56, 169
 mmmount all -a 93
 mmount 210
 mmpmon 66, 251
 mmremotecluster add 207
 mmremotecluster 161
 mmremotecluster show all 159
 mmremotefs 56, 159, 161, 210
 mmremotefs add 207
 mmremotefs show all 159
 mmrestore 377
 mmrestoreconfig 325–326, 382
 mmstorefs 319
 mmrestripefile 322
 mmrestripefile -p 322
 mmrestripefs 62, 85, 175, 216, 322, 351
 mmrestripefs -R 216
 mmshutdown 171, 183, 194, 275
 mmshutdown -a 153, 206
 mmstartup 173, 184–185, 275
 mmstartup -a 157, 395
 mmtrace 260
 mmtracectl 260
 mount 56, 159
 mv 267, 324
 netstat 275
 netstat -D 270
 netstat -s 270

- nmon 66
- no 242
- oem_setup_env 311
- ping 242
- rcp 199
- rmvdev 287
- rpm -Uh 194
- scp 155, 199
- ssh nodename date 202
- ssh-keygen -t 227
- ssh-keygen -t rsa 199, 201
- sysctl 243
- tcpdump 275
- topas 270
- unmount 276
- update_all 194
- vfcmap 302–303
- Completely Fair Queuing (CFQ) 249
- configuration change 185, 248, 255

D

- daemon
 - dsmrecalld 365
 - mmfs 258
 - mmfsd 254
- data path failover 21
- DB2 cluster
 - file system 130
 - service 137, 147
- db2 process 140, 146
- DB2 pureScale 120, 122, 133, 137, 139, 147
 - cluster 122, 137
 - feature 120
 - main features 120
 - technology 120
- DB2 Setup wizard 131–132, 135
- dd command 355, 365, 367, 373–374, 377
- dev lnx_hdisk1 286
- df command 87
- direct attach disk 203
- Direct I/O 236–237
- disaster recovery (DR) 162–164, 393, 396
- dmapiFileHandleSize 32 95, 114, 138, 151–152, 156, 167, 169, 190, 219, 278
- dump tscomm 270, 274–275
 - Pending messages 274
 - stanza 274–275

E

- elevator algorithm 249
- england-gpfs 162, 164, 167–173, 175, 177–178, 180, 182, 184, 186
- entry field 136, 194
- entstat command 241, 246
- errpt -a 263
- ethtool command 241
- event lowDiskSpace 358–359, 372, 374–375
- example Mypool 321
- export-import function 217

F

- failure group 74, 79, 163, 167, 395
- Fibre Channel 280, 283, 289, 300–304
- Fibre Channel Protocol (FCP) 8
- file set 322–327, 363, 377, 379, 382, 384
- file set name 205, 237, 354, 378, 382, 384, 388, 391
- file system (FS) 69, 73–75, 77, 81–83, 85–86, 88–89, 93, 98, 109, 112, 114–117, 120, 130, 137–139, 142, 146–147, 149–150, 155–160, 163, 168–170, 172, 174–175, 178, 181–186, 191, 220, 223, 230–236, 247–248, 250–251, 254, 318–326, 328–329, 347–351, 353, 355, 358–359, 363–365, 367–369, 371–373, 375–380, 382–385, 387, 395
 - actual block size 233
 - authorization 209
 - bin extension 353
 - bin/echo running mmapplypolicy 358, 373
 - block size 232
 - cache configuration 236
 - change 220, 378
 - completed creation 82, 93, 112, 169
 - completed deletion 212
 - configuration 217, 222, 325, 382
 - consistent copy 318
 - creation 233, 257, 321
 - creation rate 233
 - data utilization 366
 - db2fs1 139, 147, 158
 - definition 217, 221
 - descriptor 74
 - descriptor quorum 163, 182, 184–185, 395
 - down disks 84
 - enough free space 376
 - existing snapshots 318
 - fragmentation 222–223, 233
 - fragmentation increase 233
 - fragmentation low 233
 - GPFS snapshots 319
 - gpfs1 89, 112, 115, 169, 172, 175, 181, 184, 186
 - gpfs-ib 158, 348, 350–351, 382–383
 - group multiple storage systems 319
 - I/O behavior 251
 - I/O performance 234
 - I/O utilization 251
 - import 220
 - large number 226
 - list storage pools 321
 - manager 170, 225, 325, 349, 355
 - metadata 174–175, 186, 197, 216
 - metadata replication 74
 - name 318
 - object 324
 - old NSD 197
 - online 351
 - parametrization 231
 - performance 231, 233
 - root 319
 - root directory 319
 - separate storage pool 321
 - size 319

- system storage pool 320
- testexports 217–218, 220–222
- testmigr 197, 215, 276
- testmigr usa 262, 265, 272–273, 275–278
- tree 318
- unmount point 278
- unmounts 277
- utilization 347, 369
- version 138, 218
- file system descriptor (FSDesc) 74
- first in first out (FIFO) 249
- force unmount 114–115, 145, 153, 171, 183, 191
- france-gpfs 162, 164, 167–169, 171, 173, 175, 177–178, 180–183, 185–186
- free disc 81, 112, 168, 196, 198, 204, 213, 351
- free inodes 196–198, 205, 214–216, 221, 223, 347–348, 358
- Number 217

G

- GARP VLAN Registration Protocol (GVRP) 309
- gathering data live 257
- General Parallel File System
 - See GPFS
- General Public License (GPL) 76, 81, 91
- generic segmentation offload (GSO) 239–240
- germany 191–192, 206, 242, 272–273, 276
- ghana-gpfs 162, 164, 167–169, 171, 173, 175, 177–178, 180–183, 185–186
- GiB 88, 113
- Gigabit Ethernet 70, 74, 85–86, 149
- GigE Lan 74, 97, 149
- global events 225
- GPFS 2, 194
 - architecture 230
 - configuration 74
 - level 191
 - older version 193
 - performance 229
 - restart daemon 146
 - scenarios 69
 - three-node 73, 77
- GPFS 3.3 194, 224
- GPFS 3.4 74, 91, 97, 149, 151, 258
- GPFS cache
 - consideration 234
 - policy 236
- GPFS cluster 85, 92, 189–191, 207–208, 225–226, 253–254, 258, 319–321, 346, 349, 363, 365
 - diagram 122
 - hierarchical storage manager 363
 - List node 165
- GPFS code 81
- GPFS daemon 113, 116, 146, 157, 170–171, 191, 235
 - interface 86
- GPFS device 141–144
- GPFS file
 - set 324
 - system 73, 115, 120, 138–139, 163, 191, 196–197, 205, 217, 221, 230, 256, 259, 265, 267, 269, 318, 326,

- 347, 363, 367, 385, 395
- system definition 221
- GPFS GPL layer binary 81
- GPFS level 159
- GPFS network 71
 - adapter failure 86
 - configuration 207
 - interface 206
- GPFS policy
 - engine 385
 - management tool 367
- GPFS problem 253, 257
 - determination process 257
- GPFS rpm 76
- GPFS Server
 - Manpages 191–192, 194–195
 - Message 191–192, 194–195
- GPFS startup
 - event 84
 - script 364
- gpfs.snap script 263, 267
- greece-gpfs 92, 96, 149, 152, 157, 159
- grep gpfs 76, 169, 192, 194–195, 202
- gunzip.sh file 389

H

- Hardware Management Console (HMC) 284, 288, 313–315
- health-check information 230
- Hierarchical Storage Manager (HSM) 363
 - client 363, 367, 371, 373, 376
- high availability (HA) 306–307, 309
- honduras-gpfs 109
- host bus adapter (HBA) 249
- Host Ethernet Adapter 280, 306, 313
- hypervisor 299–300, 306–307, 313

I

- I/O completion 250, 270, 273
- I/O load 288, 291
- I/O request 248, 284, 289
- I/O time 264
- IBM GPFS
 - support Web site 193
 - Windows version 108
- IBM service 256, 258, 261–262, 270
- ifconfig command 87–88
- inet6 addr 86–88
- information lifecycle management (ILM) 3, 318
- Inode File 82, 93, 112, 169, 211–212, 380
- Inode Information 196–198, 205, 214–216, 221, 223, 347–348, 352, 357
- inode update 142, 144–145, 251
- inodes 196–198, 205, 214, 216, 218, 221, 223–224, 323, 326, 347–348, 354, 356–357, 359–360, 362, 366, 368, 374–375, 387, 389
- Inodes scan 362
- Integrated Virtualization Manager (IVM) 284
- Interior Gateway Routing Protocol (IGRP) 306

- internal disc 73–76, 83, 85
- IP address 74, 78, 113, 118, 136–137, 151–152, 165, 206, 271
- ip command 245
- IP network 230–231
- IP over Infiniband (IPoB) 8
- ITSO laboratory 70
 - diagram 70
- itso-fileset file set
 - same files 382

J

- japan-gpfs 162, 164, 167–169, 171–173, 175, 177–178, 180–181, 184–186

K

- kernel extension 194, 268
- KGN site 163, 171
 - complete failure 171
 - GPFS daemons 185
- KVM node2 71–73

L

- Level 1.0 345, 349, 365–366, 369, 378, 381
- Link Aggregation Control Protocol (LACP) 10
- Linux
 - Delaware 71–72
 - node 71–73, 90, 117, 217, 243
 - public/private SSH key 77
 - server license 118
 - node1 71–73
 - node2 71–73
- Linux/AIX client 97
- local cluster 153, 259, 269, 273
- local events 225
- local node 83, 111, 115–116, 119, 139, 146, 157–158, 170, 172–173, 177–178, 181, 183, 186
- log
 - directory specification 336
 - mmfs.logs 257
- logical unit numbers (LUNs) 13
- lowDiskSpace event 355, 358–359, 372–373, 375, 387
- LPAR 176, 281, 283–284, 288, 301, 303, 306–307, 314–315
- LPAR name 176
- LUN number 287
- usr/lpp/mmfs/samples/debugtools/fsstruct 265

M

- management information base (MIB) 227
- master snapshot 263
- max 2 205, 237, 354, 378, 382, 384, 388, 391
- Max brk (MB) 327, 345, 363, 381
- Maximum disk size 196–198, 205, 212, 214–217, 221, 223, 347–348, 350, 352, 356–357, 367, 369–370, 373, 375
- maximum files to cache (MFTC) 236
- Maximum Number 136, 138, 196–198, 205, 214,

- 216–217, 221, 223, 248, 299, 347–348, 358
- maximum statistical cache (MSC) 236
- Maximum Transmission Unit (MTU) 239
- maxMBpS parameter 248
- media speed 240–241, 247
- metanode 44
- mexico-gpfs 162, 164, 167–169, 171–173, 177–178, 180–182, 184–186
- Mgmt IP 74, 98, 121, 162
- MGSRS 236
- MIGRATE/DELETE/EXCLUDE rule 354, 359, 362, 371, 375, 387
- migration policy 327, 329, 350, 353–356, 358, 372–374, 376
- minReleaseLevel 3.4.0.0 95, 114, 152, 167, 169, 278
- mmaddisk command 197, 215
- mmapplypolicy command 354–355, 359, 361, 385
- mmapplypolicy gpfs-ib 354, 356–357, 362, 368, 387, 389
- mmbackup command 377–379, 384
- mmchconfig cipherList 154, 156
- mmchconfig command 167, 170, 219, 227
- mmchlicense command 78, 92, 110, 117, 165, 202
- mmchlicense server 79, 92, 110, 118, 166
- mmchnsd command 220, 222
- mmchpolicy gpfs-ib 352–353, 371
- mmcommon preunmount 276–277
- mmcrfs 81–83, 88–89, 93, 112, 168, 210–211
- mmcrnsd command 80, 167, 204, 211
- mmdiag 250, 258, 261–262
- mmdash command 256
- mmfsadm command 258–259
- mmgetstate 171
- mmgetstate command 110, 177
 - daemon status 110
- mmiscluster command 137, 165, 191
- mmisfs command 138, 234
- mmismount command 86
- mmremotefs 159, 161, 210
- mmrestripefs 85, 175, 216, 322, 351
- module mmfs26 153
- module mmfslinux 153
- module tracedev 153
- mounted file system, I/O statistics 251
- MTU size 240, 306
- multi-cluster configuration 149
- Multi-Path I/O (MPIO) 13
- multiple node 255, 260, 265, 271
- multiple VSCSI
 - adapter 288
 - server 288

N

- N_Port ID Virtualization (NPIV) 280, 282–283, 296, 300, 302, 305
- namespace 322–325
- net.core.netd ev_max_backlog 243, 245
- Network Shared Disk
 - See NSD
- node germany 197, 215, 228, 274, 276–277
 - SSH authorized_keys configuration 228

- node name 78, 83, 109–110, 113–114, 116, 118–119, 137, 139, 146, 149, 151–152, 157–158, 162, 165–166, 170–171, 173, 176–178, 180, 182, 186, 256, 343, 345–347, 349, 378, 381
- node quorum
 - algorithm 47
 - with tiebreaker disks algorithm 47
- NSD 79–80, 82, 84–85, 90, 93, 96–97, 111, 114, 150, 162, 167, 170, 172, 174–175, 181, 184, 187, 195–197, 214, 230–231, 233, 236, 244, 249, 254, 264, 266, 270, 273–274, 276, 319–320, 347, 351, 371, 380
 - definition 80, 163
 - I/O completion 270, 273
 - usa 262, 265, 273, 276–277
 - waiter 273
- name 203
- server 81, 112, 168, 196, 198, 203–204, 213, 218, 220, 222, 231, 270, 284
 - configuration 222
 - failure 231
 - GPFS Linux InfiniBand cluster 90
 - list 203
 - node 270
 - situation 220
 - volume 230, 233
- NSD_Desc file 80, 82
- NTFS TxF 42–43

O

- Object Data Manager (ODM) 300
- OLAP 25
- OLTP 25
- online analytical processing
 - See OLAP

Online IBM.Application

- ca_db2sdin1_0-rs 147
 - brazil 147
- cacontrol_db2sdin1_128_brazil
 - brazil 148
- cacontrol_db2sdin1_129_serbia
 - serbia 148
- db2_db2sdin1_0-rs 147
 - algeria 147
- db2_db2sdin1_1-rs 147
- db2mnt-db2sd_20100719132932-rs 147
 - algeria 147
 - brazil 147
- idle_db2sdin1_997_algeria-rs 147
 - algeria 147
- idle_db2sdin1_997_chile-rs 147
- idle_db2sdin1_998_algeria-rs 147
 - algeria 147
- idle_db2sdin1_998_chile-rs 147
- idle_db2sdin1_999_algeria-rs 147
 - algeria 147
- idle_db2sdin1_999_chile-rs 147
- instancehost_db2sdin1_algeria
 - algeria 148
- instancehost_db2sdin1_brazil
 - brazil 148

- instancehost_db2sdin1_serbia
 - serbia 148
- primary_db2sdin1_900-rs 148
 - brazil 148

- online transaction processing

- See OLTP

- OpenFabrics Enterprise Distribution (OFED) 94

- OS 76, 102, 196, 214, 264, 268

P

- pagepool 234–235, 271
- pagepool 256M 138, 151, 156
- Path MTU 243
- peer GPFS clusters 53
- Persistent Reserve (PR) 50, 285
- physical adapter 239, 300–304, 306, 310, 313
 - available ports 302
- Physical Host Ethernet Adapter (PHEA) 313
- placement policy 320–322, 326–327, 351, 353, 388
- POK site
 - accessibility 172
 - disks 163
 - workload running 172
- pool total 196–198, 205, 214–217, 221, 223, 347–348, 350, 352, 356–357, 368–370, 373, 375
- port_group 313
- previous PTF 268
- primary server 78, 113, 118, 137, 151–152, 165, 185, 191, 208, 346
- proper GPFS license designation 78, 92, 110, 117, 165, 202
- public key 155, 159, 161, 201, 227
- pureScale feature 120, 127, 134, 137
 - DB2 Enterprise Edition 127
 - DB2 v98 121

Q

- quality of service (QoS) 312
- quorum
 - definition of 46, 266
 - node 78, 83, 111, 115–117, 119, 139, 146, 157, 170, 172–173, 177–178, 181, 183, 186, 225, 266, 395

R

- Real Application Clusters (RAC) 237
- Redbooks website 399
 - contact us xi
- Redundant Disk Array Controller (RDAC) 14
- Release 2 345, 349, 365–366, 369, 378, 381
- Reliable Scalable Clustering Technology (RSCT) 121, 137, 147
- remote cluster 150–151, 193, 199, 207, 209, 220, 259, 267, 269, 273–274
- Remote Direct Memory Access (RDMA) 26, 296
- remote node 83, 111, 115–117, 119, 139, 146, 157–158, 170, 172–173, 177–178, 181, 183, 186, 193, 199, 259
- Remote Procedure Call
 - See RPC

- render farm 34
- rolling upgrade 190
- root environment 311
- root@argentina mmcrfs 82, 88–89
- root@argentina mmcnsd 79–81
- root@argentina src 81, 83
- root@australia gpfs-ib 377–379, 381–382, 384
- root@australia tmp 84–85
- root@australia work 364–365, 368
- root@denmark x86_64 84, 86–88
- root@italy ts3200 331
- root@nigeria perf 93, 96
- root@slovenia bin 250–251, 299–300
- root@slovenia slave 297–299
- root@spain work 92, 94, 361–362, 371–373, 380–381, 386–389, 391
- round trip time (RTT) 241–242
- RPC 258
- RPC wait 269–270, 273
- running GPFS cluster 190
 - rolling migration 190

S

- Scope Link 86–88
- SCP 191, 199, 207–208, 219, 221, 227
- SCSI RDMA Protocol (SDP) 8
- secondary server 78–79, 113, 118, 137, 151–152, 165, 185, 191, 208
- secondary waiter 269, 273
- Secure Sockets Layer
 - See SSL
- select value 136, 194
- server license 79, 110, 118, 166
- service 16.1 274
- Service Pack (SP) 120
- Shared Ethernet Adapter (SEA) 239, 306, 308, 312
- Shared Virtual Ethernet Adapter (SVEA) 308
- Signal 11 277
- single point of failure (SPOF) 231
- sites POK 163, 167
- size 26214400 KB 93, 168, 197, 204, 211–212, 215
- slovenia 200, 261–262, 272, 275–277
- snapshot, definition of 65
- Socket Direct Protocol (SRP) 8
- spain-gpfs 92, 346–347, 351, 359, 380
- SSH key 199, 201
- SSL 150
- SSL key 153, 155
- Stale NFS 87, 275
- storage area network (SAN) 257, 300
- storage pool 82, 93, 112, 169, 196–198, 203–205, 211, 214–217, 221, 223, 318–322, 324, 326–329, 342, 347–348, 350–352, 354–361, 363, 366–367, 369–373, 375–376, 378, 380, 382–384, 386, 388, 391
 - Allocation Map 82, 93, 112, 169
 - file placement 322
 - Flushing Allocation Map 351
 - free space 355
 - specific files 322
- storage pool, definition of 64

- Storage SAN Volume Controller (SVC) 15
- store method 294–295, 297, 303–304
- stub file 67
- Summary information 79, 83, 110, 115–116, 119, 139, 146, 157, 166, 170–171, 173, 177–178, 180, 183, 186
- symlink 94
- System Automation (SA) 121, 137, 147
- System Management Interface Tool (SMIT) 192
- System p
 - client node 97
 - cluster 91
 - Virtual I/O Server (VIOS) 239
- system pool 319–320, 328–329, 352–353, 355–356, 359, 371–374, 386, 388
- system storage pool 320–322, 326, 351, 372
 - multiple disks 322
- System x
 - client 90
 - node 149
 - server 70

T

- tape library
 - definition 343
 - driver 331
- TCP Segmentation Offload
 - Packet 247
 - Packet Error 247
- teaming 6
- Technology Level (TL) 120
- th 4 93, 96
- three-node cluster 75, 85
- tiebreaker site 163, 168, 172, 181, 394–395
- Time Sharing Option (TSO) 240
- Tivoli Storage Manager 321, 325, 328–330, 332–335, 339, 342–345, 349, 365–366, 369, 371, 376, 378–379, 381, 384
 - Backup Archive client 325
 - client 344–345
 - client backup function 345
 - code root directory 332
 - command 384
 - environment 329–330, 332
 - from GPFS 367
 - installation 329
 - itso-stg-pool 375
 - manual 332, 339, 349
 - move data to 372
 - server 325, 330, 342, 365
 - tool 384
 - unlink caution 325
 - user 384
 - V6.1 Technical Guide 330
 - Version 6.2 332
- total number 234, 345, 349–350, 378, 381
- Transactional NTFS (TxF) 42–43
- TRCBUFSIZE buffer 261–262, 267
- ttl 242
- TxF (Transactional NTFS) 42–43

U

UDP fragmentation offload (UFO) 240
unmount 264, 275–276, 278
use itso.com 98
User Datagram Protocol (UDP) 306
User Exit Script
 LOWDISK 359
 MIGDISK 374–375
UTC 354, 359, 362, 375

V

VERBS RDMA 96
 device ehca0 port 1 96
 library 96
Verifying GPFS 114, 154, 156, 167, 184, 206, 208
VIOC 285–286, 288, 299–300, 303–305
VIOS 13, 239, 280–290, 292, 294–296, 299–302,
304–312, 315
 Shared Ethernet Adapters 312
virtual Fibre Channel adapter 300–304
 specific WWPN 303
virtual Fibre Channel client adapter 304
virtual SCSI
 See VSCSI
virtualized environment 279, 281
VSCSI
 adapter 284–285, 290
 allocation map 285
 client adapter 284, 288
 driver 295
 initiator device 284
 multipath implementation 284
 slot 288

W

waiters 259
Windows node1 71–73, 109
Windows server 97–98, 100, 102, 109, 113, 115
 file systems 97
 GPFS cluster 109
 host name 109
 node quorum 97
Windows Server 2008 R2 98–100
Writebehind worker 273



Implementing the IBM General Parallel File System (GPFS) in a Cross-Platform Environment

(0.5" spine)
0.475" <-> 0.873"
250 <-> 459 pages



Implementing the IBM General Parallel File System (GPFS) in a Cross-Platform Environment



Describes features, use cases, and implementation scenarios of GPFS v3.4

Explains management, configuration, performance, and tuning factors

Gives problem determination and disaster recovery help

This IBM Redbooks publication provides a documented deployment model for IBM GPFS in a cross-platform environment with IBM Power Systems, Linux, and Windows servers. With IBM GPFS, customers can have a planned foundation for file systems management for cross-platform access solutions.

This book examines the functional, integration, simplification, and usability changes with GPFS v3.4. It can help the technical teams provide file system management solutions and technical support with GPFS, based on Power Systems virtualized environments for cross-platform file systems management.

The book provides answers to your complex file systems management requirements, helps you maximize file system availability, and provides expert-level documentation to transfer the how-to skills to the worldwide support teams.

The audience for this book is the technical professional (IT consultants, technical support staff, IT architects, and IT specialists) who is responsible for providing file system management solutions and support for cross-platform environments that are based primarily on Power Systems.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-7844-00

ISBN 0738435473