Configuring the GPFS/DCS9900 Storage System
Best Practices

Raymond L. Paden, Ph.D.
HPC Technical Architect
High Performance Technical Computing

23 Sep 09
Version 5.2
Special Notices from IBM Legal

This presentation was produced in the United States. IBM may not offer the products, programs, services or features discussed herein in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the products, programs, services, and features available in your area. Any reference to an IBM product, program, service or feature is not intended to state or imply that only IBM's product, program, service or feature may be used. Any functionally equivalent product, program, service or feature that does not infringe on any of IBM's intellectual property rights may be used instead of the IBM product, program, service or feature.

Information in this presentation concerning non-IBM products was obtained from the suppliers of these products, published announcement material or other publicly available sources. Sources for non-IBM list prices and performance numbers are taken from publicly available information including D.H. Brown, vendor announcements, vendor WWW Home Pages, SPEC Home Page, GPC (Graphics Processing Council) Home Page and TPC (Transaction Processing Performance Council) Home Page. IBM has not tested these 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.

IBM may have patents or pending patent applications covering subject matter in this presentation. The furnishing of this presentation does not give you any license to these patents. Send license inquiries, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 USA.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. Contact your local IBM office or IBM authorized reseller for the full text of a specific Statement of General Direction.

The information contained in this presentation has not been submitted to any formal IBM test and is distributed "AS IS". While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. The use of this information or the implementation of any techniques described herein is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. Customers attempting to adapt these techniques to their own environments do so at their own risk.

IBM is not responsible for printing errors in this presentation that result in pricing or information inaccuracies.

The information contained in this presentation represents the current views of IBM on the issues discussed as of the date of publication. IBM cannot guarantee the accuracy of any information presented after the date of publication.

IBM products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

Any performance data contained in this presentation was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements quoted in this presentation may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Some measurements quoted in this presentation may have been estimated through extrapolation. Actual results may vary. Users of this presentation should verify the applicable data for their specific environment.

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

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

LINUX is a registered trademark of Linus Torvalds. Intel and Pentium are registered trademarks and MMX, Itanium, Pentium II Xeon and Pentium III Xeon are trademarks of Intel Corporation in the United States and/or other countries.

Other company, product and service names may be trademarks or service marks of others.
The following pages provide a very brief overview to what GPFS is and how it works via two generic configuration architectures.

1. LAN Configuration
   This is a client/server configuration. The LUNs are mounted in /dev on the servers and accessed by the clients via the LAN.

2. SAN Configuration
   GPFS is not a SAN file system but it can run in a SAN centric mode. No distinction is made between client and server. All LUNs are mounted in /dev on all nodes in the cluster. The storage controller may connect directly to the nodes or via a SAN fabric.
What is GPFS?

General Parallel File System

All of GPFS's rivals do some of these things, none of them do all of them!

- **General**: supports wide range of applications and configurations
- **Cluster**: from large (4000+ in a multi-cluster) to small (only 1 node) clusters
- **Parallel**: user data and metadata flows between all nodes and all disks in parallel
- **HPC**: supports high performance applications
- **Flexible**: tuning parameters allow GPFS to be adapted to many environments
- **Capacity**: from high (4+ PB) to low capacity (only 1 disk)
- **Global**: Works across multiple nodes, clusters and labs (i.e., LAN, SAN, WAN)
- **Heterogeneous**:
  - Native GPFS on AIX, Linux, Windows as well as NFS and CIFS
  - Works with almost any block storage device
- **Shared disk**: all user and meta data are accessible from any disk to any node
- **RAS**: reliability, accessibility, serviceability
- **Ease of use**: GPFS is not a black box, yet it is relatively easy to use and manage
- **Basic file system features**: POSIX API, journaling, both parallel and non-parallel access
- **Advanced features**: ILM, integrated with tape, disaster recovery, SNMP, snapshots, robust NFS support, hints
Local Area Network (LAN) Architecture
Clients Access Disks Through the Servers via the LAN

NSD
- SW layer in GPFS providing a "virtual" view of a disk
- virtual disks which correspond to LUNs in the NSD servers with a bijective mapping

LUN
- Logical Unit
- Abstraction of a disk
  - AIX - hdisk
  - Linux - SCSI device
- LUNs map to RAID arrays in a disk controller or "physical disks" in a server

Redundancy
Each server has 2 connections to the disk controller providing redundancy

No single points of failure
- primary/backup servers for each LUN
- controller/host connection fail over
- Dual RAID controllers

DCS9900 RAID sets are called "tiers".

Redundancy
Each LUN can have up to 8 servers. If a server fails, the next one in the list takes over.

Zoning
- Zoning is the process by which RAID sets are assigned to controller ports and HBAs
- GPFS achieves its best performance by mapping each RAID array to a single LUN in the host.

Twin Tailing
- For redundancy, each RAID array is zoned to appear as a LUN on 2 or more hosts.
SAN (Storage Area Network) Architecture
Client/Servers Access Disk via the SAN

LAN Fabric (e.g., Ethernet, IB)
tokens, heartbeat, etc.

All nodes act both as client and server.

All LUNs are mounted on all nodes.

GPFS is **not** a SAN file system; it merely can run in a SAN centric mode.

No single points of failure
- All LUNs mounted on all nodes
- SAN connection (FC or IB) fail over
- Dual RAID controllers

Multiple HBAs increase redundancy and cost.

Storage Controller

<table>
<thead>
<tr>
<th>RAID Controller</th>
<th>RAID Controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>A7</td>
</tr>
<tr>
<td>A2</td>
<td>A8</td>
</tr>
<tr>
<td>A3</td>
<td>A9</td>
</tr>
<tr>
<td>A4</td>
<td>A10</td>
</tr>
<tr>
<td>A5</td>
<td>A11</td>
</tr>
<tr>
<td>A6</td>
<td>A12</td>
</tr>
</tbody>
</table>

Zoning maps all RAID sets (LUNs) to all nodes.

CAUTION:

*Not* recommended for SANs with > 32 host ports. Scaling beyond this requires special tuning (e.g., set queue depth very small).
The following pages explain selected parameters and features and offer configuration guidelines of special interest for the GPFS/DCS9900 storage system.
A single couplet can support up to 1200 disks (i.e., 2 frames).
The DCS9900 is a RAID 6 configuration using 8 data disks and 2 parity disks (i.e., 8+2P)

Host side connections: 2 options
- Infiniband (IB) - 4xDDR2
- Fibre Channel (FC) - 8 Gb/s

Supported LUN block sizes
- 512, 1024, 2048, 4096 bytes
- GPFS only supports 512 bytes

Maximum number of disks
- 1200 disks

There are 8 host side connections.

Drive side FC Loops
2 x 10 FC @ 4 Gb/s

Tier 1
A B C D E F G H P P

Tier 2
A B C D E F G H P P

Tier 3
A B C D E F G H P P

8+2P RAID 6

COMMENTS
- Recommend creating only 1 LUN per tier for GPFS
- Supports byte striping
- Provides dual disk failure protection
- Parity is computed for each write I/O operation
- Parity is checked for each read I/O operation
RAID 6 on a DCS9900

- RAID 6 can either be an extension of RAID 3 or RAID 5
- The DCS9900 implements RAID 6 as a RAID 3 extension.

* Since RAID 6 group has 2 redundant disks, it is common to make the RAID group larger (e.g., 8+2p)
### DCS9900 Cache Organization

- There is 2.5 GB of cache per RAID controller for a total of 5 GB
- The cache page size is a configurable parameter
  - set this using the command "cache size=<int>"
    - valid choices are 64, 128, 256, 512, 1024, 2048, 4096 (units are in KB)
  - **Best Practice:** set the cache size to 1024, 2048 or 4096
    - optimum streaming performance occurs for any of these values
    - if GPFS blocksize is 2M or 4M, then cache size = 1024 or 2048 or 4096 gives same performance (n.b., a cache size smaller than the GPFS blocksize is OK!)

### Setting the OS transfer size

- Set cache page size >= OS transfer size
  - **Best Practice:** set them to be the same value

#### AIX
- `chdev -l fcs<int> -a max_xfer_size=<hex value>`
  - default = 0x100000 (i.e., 1 MB)

#### Linux
- set the `max_sectors_kb` parameter to the DCS9900 cache size
- located in `/sys/block/<SCSI device name>/queue/max_sectors_kb`
- typical SCSI device names are sdb, sdc, sdd, sde, ...
Configuration and Parameter Explanation and Guidelines

- **Write Caching: Write Back vs. Write Thru**
  - Enabling write back caching instructs the DCS9900 to write data blocks to cache and return control to the OS; data in the cache is actually written to disk later. If write thru caching is enabled, all data is written both to cache and disk before control is returned to the OS.
  - Enable write back caching using the command "cache writeback=on"
    - **warning:** if a controller fails, all data in its cache will be lost, possibly before it is written to disk
      - This can corrupt the file system since metadata can be lost.
      - Adopt proper risk management procedures if write back caching is enabled.
  - Enable write thru by setting "cache writeback=off"
    - **Best Practice:** disable write back caching
      - this will significantly degrade performance

- **Cache coherence**
  - The DCS9900 has the concept of "LUN ownership" or "LUN affinity". The controller in the couplet that created the LUN owns that LUN.
  - Both controllers in a DCS9900 couplet can both see a given LUN (even though only one of them created it) iff cache coherence is enabled
  - Cache coherence generally has a minimal performance degradation
  - **Best Practice:** Enable cache coherence using the command "dual coherency=on"
Read Caching: Prefetch

DCS9900 read caching uses a prefetch algorithm. The setting for this parameter is dependent on the GPFS block allocation map setting.

Best practice: set GPFS block allocation map to scatter
- assumes that number of nodes > 8 or number of LUNs > 8
- scatter vs. cluster
  - There are 2 block allocation map types for GPFS: scatter or cluster
  - scatter: randomly distribute file blocks over the LUN
  - cluster: write file data in clusters of contiguous disk blocks
  - COMMENT: There is no guarantee that contiguous file blocks on disk will be accessed in the same order that they are mapped to disk. This problem is exacerbated by increasing "disk entropy" through repeated create/delete cycles. Scatter guarantees uniform performance for multitask jobs accessing a common file.
- set MF bit using the command "cache mf=on"
- disable prefetching using the command "cache prefetch=0"
  - since file blocks are randomly distributed, prefetching hurts performance
Configuration and Parameter Explanation and Guidelines

User Data vs. Meta Data

- **Best Practice:** Segregate User Data and Meta Data
  - DCS9900 does streaming well, but randomly distributed small transactions not as well. Since meta data transactions are small, segregating user data and meta data can improve performance for meta data intensive operations.

Caveats and warnings

- Most beneficial in environments with significant meta data processing
- Must have enough dedicated metadata Only LUNs on controllers with good enough IOP rates to keep pace with DCS9900 LUNs.

FC Drivers

- **Linux**
  - GPFS uses the qla2400 driver for FC access to the DCS9900
    - this driver is AVT, not RDAC
  - Supports the DCS9900 active:active access model

- **AIX**
  - GPFS uses the MPIO driver in failover mode
  - Only supports an active:passive access model

Linux Multipathing

- While not officially supported, customers familiar with Linux multipathing report being able to get it work with GPFS and the DCS9900.
Summary of Selected DCS9900 Best Practice Settings
- 1 LUN per tier
- LUN block size = 512 (set interactively when the LUN is created)
- dual coherency=on
- cache size=1024 (assumes OS transfer size = 1MB)
- cache writeback=off
- cache mf=on
- cache prefetch=0
- ncq disabled

Summary of Selected GPFS Best Practice Settings
- pagepool >= 256M
- maxMBpS ~= 2X to 3X LAN connection data rate
- maxblocksize = 4096K
- blocksize = 4M (assumes objective is to optimize streaming access)
- if feasible, segregate user data and meta data
- set GPFS block allocation = scatter (i.e., mmcrfs -j scatter)
The logical configurations presented later in this report are based on the physical configuration described immediately on the following pages.
Physical Configuration

Performance Analysis
DCS9900 Performance
- Streaming data rate
  - write < 4.8 GB/s
  - read < 3.1 GB/s
- Noncached IOP rate
  - write < 47,000 IOP/s
  - read < 33,000 IOP/s
LAN: 4xDDR IB HCA (RDMA)
- Potential peak data rate per HCA < 1500 MB/s
- Required peak data rate per HCA < 1200 MB/s
SAN: 2xFC8 (dual port 8 Gbit/s Fibre Channel)
- Potential peak data rate per 2xFC8 < 1500 MB/s
- Required peak data rate per 2xFC8 < 1200 MB/s

Capacity Analysis
- SATA
  - 300 disks @ 1 TB/disk
  - 30 x 8+2P RAID 6 tiers
  - raw capacity < 300 TB
  - usable capacity < 240 TB
Physical Configuration
Drive Side Cabling

Front View

Rear View

COMMENT:
In order to avoid an overly "busy" diagram, cabling is shown only to the first and last disk enclosure; the reader can easily infer the cabling to the other disk enclosures.
The following pages document several configurations using GPFS with servers and the DCS9900 to illustrate best practices and the criteria by which to judge them.

**COMMENT:**
In order to simplify the diagrams, they only use 24 tiers (240 disks). Remember, best practices recommend using at least 160 x 15Krpm SAS disks or 300 x SATA disks which is the minimum needed to saturate DCS9900 bandwidth.
Logical Configuration Design Criteria
LUN to Port Mapping, Primary:Backup NSD Server Mapping and Cabling

The following pages examine VARIOUS schema for
- LUN to Port Mapping (i.e., zoning) on the DCS9900 couplet
- selecting primary and backup NSD servers for each LUN
- cabling

The LAN configurations are based on the following design criteria.
1. Guarantee that all controller ports and all HBA ports are uniformly active.
   - *n.b.*, the DCS9900 supports an active:active protocol

2. If a NSD server fails, its backup server can access its LUNs.

3. Consider LUNs associated with a given HBA or controller port. If an HBA or controller port fails, GPFS failover can access the associated LUNs over alternative paths from a backup NSD server for a given LUN *in a balanced manner* (*i.e.*, do not access all of the affected LUNs from a single NSD server).
   - This balance condition applies to performance under degraded conditions. It results in a slightly more complex logical configuration.

4. If one of the controllers in a couplet fails, the file system remains viable using backup NSD servers to access the LUNs of the failed controller over the other controller.
**Logical Configuration #1**

**LUN Zoning and NSD Configuration**

**RECOMMENDED** configuration when using GPFS over a LAN

**NSD Servers**

- **NSD Server N1**
  - Primary: LUNs 1, 2, 3, 4, 5, 6
  - Backup: LUNs 7, 8, 13, 14, 19, 20

- **NSD Server N2**
  - Primary: LUNs 7, 8, 9, 10, 11, 12
  - Backup: LUNs 1, 2, 15, 16, 21, 22

- **NSD Server N4**
  - Primary: LUNs 13, 14, 15, 16, 17, 18
  - Backup: LUNs 5, 6, 11, 12, 17, 18

- **NSD Server N3**
  - Primary: LUNs 19, 20, 21, 22, 23, 24
  - Backup: LUNs 3, 4, 9, 10, 23, 24

**Controller C1**

- **Ports Zoning**
  - P1: LUNs 1, 3, 5, 8, 14, 19, 20
  - P2: LUNs 7, 9, 11, 2, 16, 22

**Controller C2**

- **Ports Zoning**
  - P1: LUNs 2, 4, 6, 7, 13, 19
  - P2: LUNs 8, 10, 12, 1, 15, 21

**COMMENTS:**

- The LAN provides access to GPFS clients.
- LAN connections to the NSD servers require high speed connections (e.g., TbE or IB).
- Each LUN is zoned to be "twin-tailed" such that the LUN can be accessed from both controllers. One access path is primary and the other is secondary; the secondary path is accessed only under adverse conditions.
- This configuration completely satisfies all 4 design criteria.
- Topologically equivalent configurations can be achieved using different cabling and zoning schema.
- Requires TbE or other high speed LAN
- Requires cache coherence to be enabled
  - this is a DCS9900 parameter
- Controllers and drivers support "active:active" protocol

**DCS9900 couplet**

**LAN Connections**

- **TbE or IB**
- **IB or FC8**

**Accessed by primary server**

- **Accessed by secondary server**

**COMMENTS:**

- Requires TbE or other high speed LAN
- This configuration completely satisfies all 4 design criteria.
- Topologically equivalent configurations can be achieved using different cabling and zoning schema.
Logical Configuration #1
Primary:Backup NSD Mapping

Suppose each NSD server has 1 internal disk; it will likely be /dev/sda. The tables below list a likely SCSI device name in /dev for the LUNs on each server. The LUN number is the one assigned by the DCS9550 and it is unique; the SCSI device names are not unique (e.g., LUN 1 on N1 is sdb while LUN 1 on N2 is sdh). Given this LUN to SCSI device mapping, the disk.lst file to the right can be used with the mmcrnsd command as shown. It specifies for which SCSI devices that a given NSD server is primary or backup and assigns an NSD name that corresponds to the LUN number.

> cat disk.lst
/dev/sdb:N1:N2:::nsd_L1:
/dev/sdc:N1:N3:::nsd_L3:
/dev/sdd:N1:N4:::nsd_L5:
/dev/sdh:N1:N2:::nsd_L2:
/dev/sdj:N1:N3:::nsd_L4:
/dev/sdk:N1:N4:::nsd_L6:
/dev/sdc:N2:N1:::nsd_L7:
/dev/sdd:N2:N3:::nsd_L9:
/dev/sde:N2:N4:::nsd_L11:
/dev/sdi:N2:N1:::nsd_L8:
/dev/sdj:N2:N3:::nsd_L10:
/dev/sdk:N2:N4:::nsd_L12:
/dev/sdd:N3:N1:::nsd_L13:
/dev/sde:N3:N2:::nsd_L15:
/dev/sdf:N3:N4:::nsd_L17:
/dev/sdj:N3:N1:::nsd_L14:
/dev/sdk:N3:N2:::nsd_L16:
/dev/sdl:N3:N4:::nsd_L18:
/dev/sde:N4:N1:::nsd_L19:
/dev/sdf:N4:N2:::nsd_L21:
/dev/sdg:N4:N3:::nsd_L23:
/dev/sdk:N4:N2:::nsd_L20:
/dev/sdl:N4:N2:::nsd_L22:
/dev/sdm:N4:N3:::nsd_L24:
> mmcrnsd -F disk.lst -v no

COMMENT:
Since the SCSI device names are not unique between nodes, there can be some confusion in specifying the backup NSD server. The basic idea is to specify the backup NSD server on the same line as the primary server using the SCSI device name known to the primary server (GPFS will write unique identifiers on each LUN so that it can keep the LUN mappings straight). For example, LUN 1 is known as /dev/sdb on its primary NSD server, N1, but it is known as /dev/sdh on its backup NSD server, N2; nonetheless, the backup NSD server assignment is specified from the perspective of the primary server N1 where it is known as /dev/sdb. NSD device names are chosen to reflect the unique LUN ID.
Logical Configuration #1A
LUN Zoning and NSD Configuration: A Simpler Alternative

A SIMPLER alternative configuration when using GPFS over a LAN

NSD Servers

Controller C1
ports zoning
p1 LUNs 1 2 3 4 5 6
p2 LUNs 7 8 9 10 11 12
p3 LUNs 13 14 15 16 17 18
p4 LUNs 19 20 21 22 23 24

Controller C2
ports zoning
p1 LUNs 1 2 3 4 5 6
p2 LUNs 7 8 9 10 11 12
p3 LUNs 13 14 15 16 17 18
p4 LUNs 19 20 21 22 23 24

COMMENTS:
- Requires TbE or other high speed LAN
- This is configuration relaxes the balance requirement in design criterion #3 resulting in a simpler logical design. However, if a HBA or controller port fails, then all of the LUNs that are primary on that path will be serviced by the same backup node creating a bottleneck.
- Topologically equivalent configurations can be achieved using different cabling and zoning schema.
- Requires cache coherence to be enabled
  - this is a DCS9900 parameter
- Controllers and drivers support "active:active" protocol

LAN

IB or FC8 connections

Controller C1
ports zoning
p1 LUNs 1 2 3 4 5 6
p2 LUNs 7 8 9 10 11 12
p3 LUNs 13 14 15 16 17 18
p4 LUNs 19 20 21 22 23 24

Controller C2
ports zoning
p1 LUNs 1 2 3 4 5 6
p2 LUNs 7 8 9 10 11 12
p3 LUNs 13 14 15 16 17 18
p4 LUNs 19 20 21 22 23 24

COMMENTS:
- The LAN provides access to GPFS clients.
- LAN connections to the NSD servers require high speed connections (e.g., TbE or IB).
Logical Configuration #1A
Primary:Backup NSD Mapping

Suppose each NSD server has 1 internal disk; it will most likely be /dev/sda.

By issuing the following commands
> modprobe -r qla2400
> modprobe -i qla2400
the LUN to SCSI device mapping in the table immediately to the right is likely.

<table>
<thead>
<tr>
<th>LUN</th>
<th>SCSI Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sdb</td>
</tr>
<tr>
<td>2</td>
<td>sdc</td>
</tr>
<tr>
<td>3</td>
<td>sdd</td>
</tr>
<tr>
<td>4</td>
<td>sde</td>
</tr>
<tr>
<td>5</td>
<td>sdf</td>
</tr>
<tr>
<td>6</td>
<td>sdg</td>
</tr>
<tr>
<td>7</td>
<td>sdh</td>
</tr>
<tr>
<td>8</td>
<td>sdi</td>
</tr>
<tr>
<td>9</td>
<td>sdj</td>
</tr>
<tr>
<td>10</td>
<td>sdk</td>
</tr>
<tr>
<td>11</td>
<td>sdl</td>
</tr>
<tr>
<td>12</td>
<td>sdm</td>
</tr>
<tr>
<td>13</td>
<td>sdb</td>
</tr>
<tr>
<td>14</td>
<td>sdc</td>
</tr>
<tr>
<td>15</td>
<td>sdd</td>
</tr>
<tr>
<td>16</td>
<td>sde</td>
</tr>
<tr>
<td>17</td>
<td>sdf</td>
</tr>
<tr>
<td>18</td>
<td>sdg</td>
</tr>
<tr>
<td>19</td>
<td>sdh</td>
</tr>
<tr>
<td>20</td>
<td>sdi</td>
</tr>
<tr>
<td>21</td>
<td>sdj</td>
</tr>
<tr>
<td>22</td>
<td>sdk</td>
</tr>
<tr>
<td>23</td>
<td>sdl</td>
</tr>
<tr>
<td>24</td>
<td>sdm</td>
</tr>
</tbody>
</table>

> cat disk.lst
/dev/sdb:N1:N2:::nsd_L1:
/dev/sdc:N1:N2:::nsd_L2:
/dev/sdd:N1:N2:::nsd_L3:
/dev/sde:N2:N1:::nsd_L4:
/dev/sdf:N2:N1:::nsd_L5:
/dev/sdg:N2:N1:::nsd_L6:
/dev/sdh:N1:N2:::nsd_L7:
/dev/sdi:N1:N2:::nsd_L8:
/dev/sdj:N1:N2:::nsd_L9:
/dev/sdk:N2:N1:::nsd_L10:
/dev/sdl:N2:N1:::nsd_L11:
/dev/sdm:N2:N1:::nsd_L12:
/dev/sdn:N3:N4:::nsd_L13:
/dev/sdc:N3:N4:::nsd_L14:
/dev/sdd:N3:N4:::nsd_L15:
/dev/sde:N4:N3:::nsd_L16:
/dev/sdf:N4:N3:::nsd_L17:
/dev/sdg:N4:N3:::nsd_L18:
/dev/sdh:N3:N4:::nsd_L19:
/dev/sdi:N3:N4:::nsd_L20:
/dev/sdk:N3:N4:::nsd_L21:
/dev/sdk:N4:N3:::nsd_L22:
/dev/sdl:N4:N3:::nsd_L23:
/dev/sdm:N4:N3:::nsd_L24:

> mmcrnsd -F disk.lst -v no

COMMENT:
In this example the SCSI device names are the same on each NSD server for a given LUN; for example, LUN 1 is known as /dev/sdb on both nodes N1 and N2. This avoids the confusion associated with the previous logical configuration, but backup NSD server assignments are not balanced under degraded conditions (e.g., a FC port fails).
Logical Configuration #2
LUN Zoning and NSD Configuration

RECOMMENDED configuration when using GPFS over a SAN

NSD Servers

Controller C1
ports     zoning
p1        LUNs 1..12   p2        LUNs 13..24
p3        LUNs 1..12   p4        LUNs 13..24

Controller C2
ports     zoning
p1        LUNs 1..12   p2        LUNs 13..24
p3        LUNs 1..12   p4        LUNs 13..24

DCS9900 couplet

NSD Server N1
LUNs 1-12   P1
LUNs 13-24  P2
NSD Server N2
LUNs 13-24  P1
LUNs 1-12   P2
NSD Server N3
LUNs 1-12   P1
LUNs 13-24  P2
NSD Server N4
LUNs 13-24  P1
LUNs 1-12   P2

COMMENTS:
• C1 "owns" LUNs 1..12
• C2 "owns" LUNs 13..24
• Requires cache coherence to be enabled
  • this is a DCS9900 parameter
• Controllers and drivers support "active:active" protocol

COMMENTS:
• The LAN is only used for GPFS administration.
• There are no other GPFS clients.
• GbE is adequate.

COMMENTS:
• GbE provides adequate BW for this configuration.
• Topologically equivalent configurations can be achieved using different cabling and zoning schema.
• The design criteria for an NSD configuration do not generally apply to a SAN configuration; they are either irrelevant or naturally satisfied. The main criterion to pay attention to is to be sure that the zoning is set up so that all controller ports and all HBA ports are uniformly active.
Logical Configuration #2
NSD Mapping

In a SAN configuration, since all LUNs are zoned so that they are mounted on all nodes, there is no need to specify primary and backup servers. All nodes directly access the LUNs over the FC network and do not use the IP network to access user data from disk.

Suppose each NSD server has 1 internal disk; it will most likely be /dev/sda.

By issuing the following commands
> modprobe -r qla2400
> modprobe -i qla2400
the LUN to SCSI device mapping in the table immediately to the right is likely.

The disk.lst file to the far right can be used as the input file to the mmcrnsd command to designate the primary and secondary NSD servers for each LUN (i.e., SCSI device)

<table>
<thead>
<tr>
<th>LUN</th>
<th>SCSI Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sdb</td>
</tr>
<tr>
<td>2</td>
<td>sdc</td>
</tr>
<tr>
<td>3</td>
<td>sdd</td>
</tr>
<tr>
<td>4</td>
<td>sde</td>
</tr>
<tr>
<td>5</td>
<td>sdf</td>
</tr>
<tr>
<td>6</td>
<td>sdg</td>
</tr>
<tr>
<td>7</td>
<td>sdh</td>
</tr>
<tr>
<td>8</td>
<td>sdi</td>
</tr>
<tr>
<td>9</td>
<td>sdj</td>
</tr>
<tr>
<td>10</td>
<td>sdk</td>
</tr>
<tr>
<td>11</td>
<td>sdl</td>
</tr>
<tr>
<td>12</td>
<td>sdm</td>
</tr>
<tr>
<td>13</td>
<td>sdn</td>
</tr>
<tr>
<td>14</td>
<td>sdo</td>
</tr>
<tr>
<td>15</td>
<td>sdp</td>
</tr>
<tr>
<td>16</td>
<td>sdq</td>
</tr>
<tr>
<td>17</td>
<td>sdr</td>
</tr>
<tr>
<td>18</td>
<td>sds</td>
</tr>
<tr>
<td>19</td>
<td>sdt</td>
</tr>
<tr>
<td>20</td>
<td>sdu</td>
</tr>
<tr>
<td>21</td>
<td>sdv</td>
</tr>
<tr>
<td>22</td>
<td>sdw</td>
</tr>
<tr>
<td>23</td>
<td>sdx</td>
</tr>
<tr>
<td>24</td>
<td>sdy</td>
</tr>
</tbody>
</table>

> cat disk.lst
/dev/sdb:::dataOnly::nsd_L1:
/dev/sdc:::dataOnly::nsd_L2:
/dev/sdd:::dataOnly::nsd_L3:
/dev/sde:::dataOnly::nsd_L4:
/dev/sdf:::dataOnly::nsd_L5:
/dev/sdg:::dataOnly::nsd_L6:
/dev/sdh:::dataOnly::nsd_L7:
/dev/sdi:::dataOnly::nsd_L8:
/dev/sdj:::dataOnly::nsd_L9:
/dev/sdk:::dataOnly::nsd_L10:
/dev/sdl:::dataOnly::nsd_L11:
/dev/sdm:::dataOnly::nsd_L12:
/dev/sdn:::dataOnly::nsd_L13:
/dev/sdo:::dataOnly::nsd_L14:
/dev/sdp:::dataOnly::nsd_L15:
/dev/sdq:::dataOnly::nsd_L16:
/dev/sdr:::dataOnly::nsd_L17:
/dev/sds:::dataOnly::nsd_L18:
/dev/sdt:::dataOnly::nsd_L19:
/dev/sdu:::dataOnly::nsd_L20:
/dev/sdv:::dataOnly::nsd_L21:
/dev/sdw:::dataOnly::nsd_L22:
/dev/sdx:::dataOnly::nsd_L23:
/dev/sdy:::dataOnly::nsd_L24:
> mmcrnsd -F disk.lst -v no
The following pages provide an actual example of installing and configuring GPFS under Linux using 4 x3650 NSD servers and a DCS9550 storage controller and disk enclosures. The steps for doing this under AIX are very similar; differences are explained in the annotations.

Note the following:
- **red arial** font is used for annotations
- **blue courier** font is used to highlight commands and parameters
- **black courier** font is used for screen text
**EXAMPLE: Installing and Configuring GPFS**

The diagrams on this page illustrate the physical and logical configurations used in this example.

This example is based on Logical Configuration #1A presented earlier in this slide set.
EXAMPLE: Installing and Configuring GPFS

Outline of Steps to Install and Configure GPFS

1. Install the GPFS code
2. Build portability layer
3. Configure a GPFS File System
   a. Create cluster
   b. Change global GPFS parameters and start the GPFS daemon
   c. Create the NSDs
   d. Create and Mount the file system

It is only necessary to build the portability layer under Linux.
EXAMPLE: Installing and Configuring GPFS

Steps to Install the GPFS Code under Linux

1. Create an NFS mounted installation directory for extracting the RPMs
2. Copy base version RPMs to the installation directory and extract the RPMs on all nodes in the GPFS cluster.
   - gpfs.base-3.1.0-1.x86_64.rpm
   - gpfs.gpl-3.1.0-1.noarch.rpm
   - gpfs.msg.en_US-3.1.0-1.noarch.rpm
   - gpfs.docs-3.1.0-1.noarch.rpm
   It is necessary to purchase the base version. It is generally delivered via CD-ROM.
3. Download the latest update package. This comes as a tar/gzip file from
   - gpfs-3.1.0-12.x86_64.update.tar.gz
4. Copy this file to the installation directory, then gunzip/utar this file, and extract the RPMs on all nodes in the GPFS cluster.

The steps for doing this are more thoroughly documented in chapter 4 of the GPFS Concepts, Planning and Installation Guide
The steps for installing the GPFS code under AIX are documented in chapter 5 of the GPFS Concepts, Planning and Installation Guide
The GPFS Concepts, Planning and Installation Guide be found on the web at
EXAMPLE: Installing and Configuring GPFS

Build and Install the GPFS Portability Layer for Linux

1. It is necessary to repeat the following steps for each Linux distribution and version in the GPFS cluster.

2. Select a node in the cluster and do the following…
   - cd /usr/lpp/mmfs/src
   - export SHARKCLONEROOT=/usr/lpp/mmfs/src
   - make Autoconfig
     - This step "tweaks" a make file called site.mcr for the Linux installed distribution. If this step fails, manually edit the site.mcr file as described in /usr/lpp/mmfs/src/README.
   - make World
   - make InstallImages
   - echo $?

3. Copy the following modules that were just made to all other nodes in the cluster running under the same Linux distribution and version.
   - dumpconv, lxtrace, mmfslinux, mmfs26, tracedev

The steps for doing this are more thoroughly documented in chapter 4 of the GPFS Concepts, Planning and Installation Guide. It can be found on the web at
EXAMPLE:  Installing and Configuring GPFS

Create the GPFS Cluster

```
[root@gpfs1 gpfs_install]# cat node_spec.lst
server1:quorum
server2:quorum
server3:quorum
server4:quorum
```

A small number of nodes must be declared as quorum nodes. A GPFS cluster must have \( \frac{1}{2} \) of the quorum nodes + 1 running for the daemon to remain in active state.

```
[root@gpfs1 gpfs_install]# mmcrcluster -n node_spec.lst -p server1 -s server2 -R /usr/bin/scp -r /usr/bin/ssh
Thu May 31 12:26:59 CDT 2007: mmcrcluster: Processing node server1
Thu May 31 12:26:59 CDT 2007: mmcrcluster: Processing node server2
Thu May 31 12:27:00 CDT 2007: mmcrcluster: Processing node server3
Thu May 31 12:27:00 CDT 2007: mmcrcluster: Processing node server4
mmcrcluster: Command successfully completed
mmcrcluster: Propagating the cluster configuration data to all affected nodes.
```

This is an asynchronous process.

This is a common and routine message. GPFS is using ssh and scp to copy configuration information to all of the nodes asynchronously. It may be that the node on which the command is executed may complete before all of the other nodes are ready.

```
mmcrcluster parameters
-\( n \): list of nodes to be included in the cluster
-\( p \): primary GPFS cluster configuration server node
-\( s \): secondary GPFS cluster configuration server node
-\( R \): remote copy command (e.g., rcp or scp)
-\( r \): remote shell command (e.g., rsh or ssh)
```
Create the GPFS Cluster

[root@gpfs1 gpfs_install]# mmlscluster  "List" the cluster to verify that the cluster is created as intended

GPFS cluster information
===================================
GPFS cluster name: server1
GPFS cluster id: 9912599153855104355
GPFS UID domain: server1
Remote shell command: /usr/bin/ssh
Remote file copy command: /usr/bin/scp

GPFS cluster configuration servers:
------------------------------------
Primary server: server1
Secondary server: server2

<table>
<thead>
<tr>
<th>Node</th>
<th>Daemon node name</th>
<th>IP address</th>
<th>Admin node name</th>
<th>Designation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>server1</td>
<td>192.168.0.1</td>
<td>server1</td>
<td>quorum</td>
</tr>
<tr>
<td>2</td>
<td>server2</td>
<td>192.168.0.2</td>
<td>server2</td>
<td>quorum</td>
</tr>
<tr>
<td>3</td>
<td>server3</td>
<td>192.168.0.3</td>
<td>server3</td>
<td>quorum</td>
</tr>
<tr>
<td>4</td>
<td>server4</td>
<td>192.168.0.4</td>
<td>server4</td>
<td>quorum</td>
</tr>
</tbody>
</table>
EXAMPLE: Installing and Configuring GPFS

Change Global GPFS Parameters and Start the Daemon

[root@gpfs1 gpfs_install]# mmchconfig maxMBpS=2000,maxblocksize=4m,pagepool=256m,autoload=yes
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all affected nodes. This is an synchronous process.
[root@gpfs1 gpfs_install]# mmlsconfig  “List” the global parameters to verify that they are set as intended.
Configuration data for cluster server1:
-----------------------------------------------
clusterName server1
clusterId 9912599153855104355
clusterType lc
autoload yes
useDiskLease yes
maxFeatureLevelAllowed 913
maxMBpS 2000
maxblocksize 4m
pagepool 256m
[hofs518]
takeOverSdrServ no

File systems in cluster server1:
-----------------------------------------------

(root)

Start the GPFS daemon (aka, mmfsd)

[root@gpfs1 gpfs_install]# mmstartup -a
Thu May 31 12:29:38 CDT 2007: mmstartup: Starting GPFS ...
[...]
[root@gpfs1 gpfs_install]# mmgetstate -a
Be sure mmfsd is active on all nodes before proceeding.
EXAMPLE: Installing and Configuring GPFS

Create the NSDs

First create a file and back up file listing the NSD specifications (n.b., this is an input/output file to the mmcrnsd command).

The format for each line is as follows

f1:f2::f3:f4:f5:f6:

where

f1 = scsi device
f2 = comma separate NSD server list
f3 = usage
f4 = failure group
f5 = NSD name
f6 = storage pool name

Fields left blank are filled with default value

Under AIX, the mount points for the LUNs are called hdisks. For example, the line for /dev/sdf would look like this under AIX:

hdisk6:server2,server1::dataOnly::nsd_L5

NSD = Network Storage Device

In this context, you can think of an NSD as a “logical device”. In reality, it is a TCP socket allowing the GPFS clients to access the SCSI devices on the primary NSD server or the backup NSD server if the primary server is not available.

```
[root@gpfs1 gpfs_install]# cat disk.lst
/dev/sda9:server1::metadataOnly::nsd_meta1:
/dev/sda9:server2::metadataOnly::nsd_meta2:
/dev/sda9:server3::metadataOnly::nsd_meta3:
/dev/sda9:server4::metadataOnly::nsd_meta4:
/dev/sdb:server1,server2::dataOnly::nsd_L1:
/dev/sdc:server1,server2::dataOnly::nsd_L2:
/dev/sdd:server1,server2::dataOnly::nsd_L3:
/dev/sde:server2,server1::dataOnly::nsd_L4:
/dev/sdf:server2,server1::dataOnly::nsd_L5:
/dev/sdg:server2,server1::dataOnly::nsd_L6:
/dev/sdh:server1,server2::dataOnly::nsd_L7:
/dev/sdi:server1,server2::dataOnly::nsd_L8:
/dev/sdj:server1,server2::dataOnly::nsd_L9:
/dev/sdk:server2,server1::dataOnly::nsd_L10:
/dev/sdl:server2,server1::dataOnly::nsd_L11:
/dev/sdm:server2,server1::dataOnly::nsd_L12:
/dev/sdb:server3,server4::dataOnly::nsd_L13:
/dev/sdc:server3,server4::dataOnly::nsd_L14:
/dev/sdd:server3,server4::dataOnly::nsd_L15:
/dev/sde:server4,server3::dataOnly::nsd_L16:
/dev/sdf:server4,server3::dataOnly::nsd_L17:
/dev/sdg:server4,server3::dataOnly::nsd_L18:
/dev/sdh:server3,server4::dataOnly::nsd_L19:
/dev/sdi:server3,server4::dataOnly::nsd_L20:
/dev/sdj:server3,server4::dataOnly::nsd_L21:
/dev/sdk:server4,server3::dataOnly::nsd_L22:
/dev/sdl:server4,server3::dataOnly::nsd_L23:
/dev/sdm:server4,server3::dataOnly::nsd_L24:
[root@gpfs1 gpfs_install]# cp disk.lst disk.lst.orig
```
EXAMPLE: Installing and Configuring GPFS

Create the NSDs

[root@gpfs1 gpfs_install]# mmcrnsd -F disk.lst -v no
mmcrnsd: Processing disk sda9
mmcrnsd: Processing disk sda9
mmcrnsd: Processing disk sda9
mmcrnsd: Processing disk sda9
mmcrnsd: Processing disk sdb
mmcrnsd: Processing disk sdc
mmcrnsd: Processing disk sdd
mmcrnsd: Processing disk sde
mmcrnsd: Processing disk sdf
mmcrnsd: Processing disk sdg
mmcrnsd: Processing disk sdh
mmcrnsd: Processing disk sdj
mmcrnsd: Processing disk sdk
mmcrnsd: Processing disk sdl
mmcrnsd: Processing disk sdm
mmcrnsd: Processing disk sdb
mmcrnsd: Processing disk sdc
mmcrnsd: Processing disk sdd
mmcrnsd: Processing disk sde
mmcrnsd: Processing disk sdf
mmcrnsd: Processing disk sdg
mmcrnsd: Processing disk sdh
mmcrnsd: Processing disk sdj
mmcrnsd: Processing disk sdk
mmcrnsd: Processing disk sdl
mmcrnsd: Processing disk sdm
mmcrnsd: Propagating the cluster configuration data to all affected nodes. This is an asynchronous process.

The `mmcrnsd` parameters are

- **-F:** name of the NSD specification file (n.b., the file is changed by this command... keep a back up!)
- **-v:** check if this disk is part of an existing GPFS file system or ever had a GPFS file system on it (n.b., if it does/did and the parameter is yes, then mmcrnsd will not create it as a new NSD)
### Create the NSDs

```bash
[root@gpfs1 gpfs_install]# mmlsnsd
```

<table>
<thead>
<tr>
<th>File system</th>
<th>Disk name</th>
<th>Primary node</th>
<th>Backup node</th>
</tr>
</thead>
<tbody>
<tr>
<td>(free disk)</td>
<td>nsd_lun10</td>
<td>server2</td>
<td>server1</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun11</td>
<td>server2</td>
<td>server1</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun12</td>
<td>server2</td>
<td>server1</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun13</td>
<td>server3</td>
<td>server4</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun14</td>
<td>server3</td>
<td>server4</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun15</td>
<td>server3</td>
<td>server4</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun16</td>
<td>server4</td>
<td>server3</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun17</td>
<td>server4</td>
<td>server3</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun18</td>
<td>server4</td>
<td>server3</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun19</td>
<td>server3</td>
<td>server4</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun20</td>
<td>server3</td>
<td>server4</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun21</td>
<td>server3</td>
<td>server4</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun22</td>
<td>server4</td>
<td>server3</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun23</td>
<td>server4</td>
<td>server3</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun24</td>
<td>server4</td>
<td>server3</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun25</td>
<td>server1</td>
<td>server2</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun26</td>
<td>server1</td>
<td>server2</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun27</td>
<td>server1</td>
<td>server2</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun28</td>
<td>server1</td>
<td>server2</td>
</tr>
<tr>
<td>(free disk)</td>
<td>nsd_lun29</td>
<td>server1</td>
<td>server2</td>
</tr>
</tbody>
</table>

Verify that the NSDs were properly created.

The internal SCSI disks on the NSD servers will be used to store GPFS metadata. The metadata is mirrored on 2 levels. First, there are 2 system disks on each NSD server which are mirrored. Second, GPFS will mirror the data on 2 different nodes. This will create 4 copies of all metadata. Note that the GPFS mirroring guarantees that a node failure will not result in a loss of access to the file system since the metadata can be accessed on 2 different nodes.
EXAMPLE: Installing and Configuring GPFS

Create and Mount the File System

[root@gpfs1 gpfs_install]# mmcrfs /gpfs1 gpfs1 -F disk.lst -A yes -B 2048k -M 2 -m 2 -v no

The following disks of gpfs1 will be formatted on node gpfs1:

- nsd_meta1: size 58259691 KB
- nsd_meta2: size 58259691 KB
- nsd_meta3: size 58259691 KB
- nsd_meta4: size 58259691 KB
- nsd_lun1: size 3907041280 KB
- nsd_lun2: size 3907041280 KB
- nsd_lun3: size 3907041280 KB
- nsd_lun4: size 3907041280 KB
- nsd_lun5: size 3907041280 KB
- nsd_lun6: size 3907041280 KB
- nsd_lun7: size 3907041280 KB
- nsd_lun8: size 3907041280 KB
- nsd_lun21: size 3907041280 KB
- nsd_lun22: size 3907041280 KB
- nsd_lun23: size 3907041280 KB
- nsd_lun24: size 3907041280 KB

*** Lines for nsd_lun9 to nsd_lun20 are omitted so that everything fits on 1 page. ***

- Formatting file system ...
- Disks up to size 149 TB can be added to storage pool 'system'.
- Creating Inode File
- Creating Allocation Maps
- Clearing Inode Allocation Map
- Clearing Block Allocation Map
- 93 % complete on Thu May 31 12:32:18 2007
- 100 % complete on Thu May 31 12:32:19 2007
- Completed creation of file system /dev/gpfs1.
- mmcrfs: Propagating the cluster configuration data to all affected nodes. This is an asynchronous process.

Parameters for mmcrfs:

- /gpfs1: mount point
- gpfs1: device entry in /dev for the file system
- -F: output file from the mmcrnsd command
- -A: mount the file system automatically every time mmfsd is started
- -B: actual block size for this file system; it can not be larger than the maxblocksize set by the mmchconfig command
- -M: maximum number of metadata mirrors (can not be larger than 2)
- -m: actual number of metadata mirrors (can not be larger than –M)
- -v: check if this disk is part of an existing GPFS file system or ever had a GPFS file system on it (n.b., if it does/did and the parameter is yes, then mmcrfs will not include this disk in this file system)

The optimum value for the actual block size is both application and controller dependent. Experimentation is recommended to determine the best choice for this value. The options are 16k, 64k, 256k, 512k, 1024k, 2048k, 4096k.

The /etc/fstab is automatically updated by this command.
EXAMPLE: Installing and Configuring GPFS

Create and Mount the File System

Verify disk status..

```
[root@gpfs1 gpfs_install]# mmlsdisk gpfs1

<table>
<thead>
<tr>
<th>disk name</th>
<th>driver</th>
<th>sector</th>
<th>failure</th>
<th>holds</th>
<th>holds data</th>
<th>status</th>
<th>availability</th>
<th>storage pool</th>
</tr>
</thead>
<tbody>
<tr>
<td>nsd_meta1</td>
<td>nsd</td>
<td>512</td>
<td>4001</td>
<td>yes</td>
<td>no</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_meta2</td>
<td>nsd</td>
<td>512</td>
<td>4002</td>
<td>yes</td>
<td>no</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_meta3</td>
<td>nsd</td>
<td>512</td>
<td>4003</td>
<td>yes</td>
<td>no</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_meta4</td>
<td>nsd</td>
<td>512</td>
<td>4004</td>
<td>yes</td>
<td>no</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun1</td>
<td>nsd</td>
<td>512</td>
<td>4001</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun2</td>
<td>nsd</td>
<td>512</td>
<td>4001</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun3</td>
<td>nsd</td>
<td>512</td>
<td>4001</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun4</td>
<td>nsd</td>
<td>512</td>
<td>4002</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun5</td>
<td>nsd</td>
<td>512</td>
<td>4002</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun6</td>
<td>nsd</td>
<td>512</td>
<td>4002</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun7</td>
<td>nsd</td>
<td>512</td>
<td>4001</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun8</td>
<td>nsd</td>
<td>512</td>
<td>4001</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun9</td>
<td>nsd</td>
<td>512</td>
<td>4001</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun10</td>
<td>nsd</td>
<td>512</td>
<td>4002</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun11</td>
<td>nsd</td>
<td>512</td>
<td>4002</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun12</td>
<td>nsd</td>
<td>512</td>
<td>4002</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun13</td>
<td>nsd</td>
<td>512</td>
<td>4003</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun14</td>
<td>nsd</td>
<td>512</td>
<td>4003</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun15</td>
<td>nsd</td>
<td>512</td>
<td>4003</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun16</td>
<td>nsd</td>
<td>512</td>
<td>4004</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun17</td>
<td>nsd</td>
<td>512</td>
<td>4004</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun18</td>
<td>nsd</td>
<td>512</td>
<td>4004</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun19</td>
<td>nsd</td>
<td>512</td>
<td>4003</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun20</td>
<td>nsd</td>
<td>512</td>
<td>4003</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun21</td>
<td>nsd</td>
<td>512</td>
<td>4003</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun22</td>
<td>nsd</td>
<td>512</td>
<td>4004</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun23</td>
<td>nsd</td>
<td>512</td>
<td>4004</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
<tr>
<td>nsd_lun24</td>
<td>nsd</td>
<td>512</td>
<td>4004</td>
<td>no</td>
<td>yes</td>
<td>ready</td>
<td>up</td>
<td>system</td>
</tr>
</tbody>
</table>
```
EXAMPLE: Installing and Configuring GPFS

Create and Mount the File System

```
[root@gpfs1 gpfs_install]# mmmount /gpfs1 -a
Thu May 31 12:39:42 CDT 2007: mmmount: Mounting file systems ...

[root@gpfs1 gpfs_install]# for i in `seq 1 4`
> do > ssh server$i chmod 777 /gpfs1 > done

[root@gpfs1 gpfs_install]# touch /gpfs1/xxx
[root@gpfs1 gpfs_install]# ssh gpfs4 ls -l /gpfs1
  total 0
-rw-r--r--  1 root root 0 May 31 12:36 xxx
```

The GPFS mount command mounts the file system on multiple nodes in the cluster. The parameters are
/gpfs1: mount point
-a: mount it on all nodes in the cluster

"Quick and dirty" test to be sure the file system is up and mounted everywhere.
EXAMPLE: Installing and Configuring GPFS

What if I screw up?

Don't jump!

Clean up and start over again.

It's easier the second time!

Option #1  The proper way to do it.

1. Unmount the GPFS file system
   
   [root@gpfs1 gpfs_install]# mmunmount /gpfs1 -a
   
2. Delete GPFS file system
   
   [root@gpfs1 gpfs_install]# mmdelfs /gpfs1
   
3. Delete GPFS NSDs
   
   [root@gpfs1 gpfs_install]# for i in `seq 1 24` 
   > do 
   >   mmdelnsd nsd_lun$i 
   > done
   
4. Shutdown GPFS daemons
   
   [root@gpfs1 gpfs_install]# mmshutdown -a
   
5. Delete the GPFS cluster
   
   [root@gpfs1 gpfs_install]# mmdelnode -a

Properly deleting the file system ensures that the file system descriptors are deleted from the disks so that they will not create issues upon a subsequent file system creation attempt.

Properly deleting the NSDs ensures that the NSD descriptors are deleted so that they will not create issues upon a subsequent NSD creation attempt.

Option #2  But what if I really screw things up?

1. Unmount the GPFS file system and shutdown the GPFS daemons
   
   [root@gpfs1 gpfs_install]# mmunmount /gpfs1 -a
   
   [root@gpfs1 gpfs_install]# mmshutdown -a
   
2. Delete selected configuration files on all nodes
   
   [root@gpfs1 gpfs_install]# rm -f /var/mmfs/etc/mmfs.cfg
   
   [root@gpfs1 gpfs_install]# rm -f /var/mmfs/gen/*
   
   [root@gpfs1 gpfs_install]# rm -f /var/mmfs/tmp/*

WARNING: Use with extreme caution! Once the configuration files have been deleted, the GPFS cluster no longer exists and any data on the disks will likely be lost. Use this method only when Option #1 fails.
Overview

The following pages present sample building blocks for the DCS9900. They are based on best practice guidelines using balanced configurations; the number of storage servers (aka, NSD servers) are sufficient to harvest all available DCS9900 BW and that increasing the number of disks will not increase the performance of the building block.

2 disk options are illustrated with the following performance results.

- **Best Practice**: 160 x 15 Krpm SAS disks
  - 8+2P RAID 6
  - capacity using 450 GB/disk: raw ~ 72 TB, usable ~ 56 TB
  - streaming rate: write < 5.6 GB/s, read < 4.4 GB/s
  - IOP rate < 40,000 IOP/s

- **Best Practice**: 300 x SATA disks
  - 8+2P RAID 6
  - capacity using 1 TB/disk: raw ~ 300 TB, usable ~ 240 TB
  - streaming rate: write < 5.4 GB/s, read < 3.6 GB/s
    - This read rate is based on best practice tuning guidelines. Significantly higher read rates (e.g., 4.4 GB/s) are possible but require tuning not generally applicable (e.g., setting the allocation block map type to "cluster").
  - IOP rate < 40,000 IOP/s
**Building Block #1 (Ethernet)**

*Write Rate < 5.6 GB/s, Read Rate < 4.4 GB/s*  

---

**Ethernet Switch (TbE: GPFS, GbE: Administration)**

1. **NSD Server-01**
   - x3650 M2
   - 8 cores, 6 DIMMs
   - GbE, GbE, TbE

2. **NSD Server-02**
   - x3650 M2
   - 8 cores, 6 DIMMs
   - GbE, GbE, TbE

3. **NSD Server-03**
   - x3650 M2
   - 8 cores, 6 DIMMs
   - GbE, GbE, TbE

4. **NSD Server-04**
   - x3650 M2
   - 8 cores, 6 DIMMs
   - GbE, GbE, TbE

5. **NSD Server-05**
   - x3650 M2
   - 8 cores, 6 DIMMs
   - GbE, GbE, TbE

6. **NSD Server-06**
   - x3650 M2
   - 8 cores, 6 DIMMs
   - GbE, GbE, TbE

7. **NSD Server-07**
   - x3650 M2
   - 8 cores, 6 DIMMs
   - GbE, GbE, TbE

8. **NSD Server-08**
   - x3650 M2
   - 8 cores, 6 DIMMs
   - GbE, GbE, TbE

---

**DCS9900 (2U) RAID Controller**

1. **C1**
   - host ports
   - drive ports
   - 3
   - 4
   - GbE, GbE

2. **C2**
   - host ports
   - drive ports
   - 4
   - 4
   - GbE, GbE

---

**Disk trays**

- 60-Bay Disk Tray (4U) #1
- 60-Bay Disk Tray (4U) #5

**SAN switch**

- **Disk trays**
  - Minimum required to saturate DCS9900 BW
  - 160 x SAS disks: 72 TB
  - 300 x SATA disks: 300 TB

---

**COMMENT:**

- The FC8 HBAs can be replaced by 4xDDR IB HCAs using SRP, but sharing the LAN based IB switch is **not** recommended. The host ports can either be directly attached to the servers or separate IB switch can be used.

---

**COMMENT:**

- More disks (for a total of 1200) can be added to this solution but it will **not** increase performance.

---

**Peak sustained DCS9900 performance**

- Streaming data rate < 5.6 GB/s
- Noncached IOP rate < 40,000 IOP/s

**TbE (10 Gbit Ethernet Adapter)**

- Potential peak data rate per TbE < 725 MB/s
- Required peak data rate per TbE < 700 MB/s

**FC8 (single port 8 Gbit/s Fibre Channel)**

- Potential peak data rate per FC8 < 750 MB/s
- Required peak data rate per FC8 < 700 MB/s

---

*Based on benchmark using 160 x 15 Krpm SAS drives.*
Alternate Building Block #1 (Ethernet)
Write Rate < 5.6 GB/s, Read Rate < 4.4 GB/s

* Based on benchmark using 160 x 15 Krpm SAS drives.

Ethernet Switch (TbE: GPFS, GbE: Administration)

NSD Server-01
x3650 M2
8 cores, 6 DIMMs

NSD Server-02
x3650 M2
8 cores, 6 DIMMs

NSD Server-03
x3650 M2
8 cores, 6 DIMMs

NSD Server-04
x3650 M2
8 cores, 6 DIMMs

DCS9900 (2U) RAID Controller
C1
host ports
drive ports
host ports

DCS9900 (2U) RAID Controller
C2
host ports
drive ports
host ports

60-Bay Disk Tray (4U)

Disk trays
Minimum required to saturate DCS9900 BW

0
0
0

- 160 x SAS disks: 72 TB
- 300 x SATA disks: 300 TB

60-Bay Disk Tray (4U)

COMMENT:
More disks (for a total of 1200) can be added to this solution but it will not increase performance.

Peak sustained DCS9900 performance
- streaming data rate < 5.6 GB/s
- noncached IOP rate < 40,000 IOP/s

TbE (10 Gbit Ethernet Adapter)
- Potential peak data rate per 2 x TbE < 1450 MB/s
- Required peak data rate per 2 x TbE < 1400 MB/s

2xFC8 (dual port 8 Gbit/s Fibre Channel)
- Potential peak data rate per 2xFC8 < 1500 MB/s
- Required peak data rate per 2xFC8 < 1400 MB/s

COMMENTS:
- The 2xFC8 HBAs can be replaced by 4xDDR IB HCAs using SRP
- Unlike the previous configuration, this one uses 2 TbE adapters. The peak bandwidth of the x3650 M2 using Gen 1 PCI-E busses can easily sustain these data rates. However, the channel binding protocols needed for the 2 TbE adapters on each node do not always adequately balance the load. Validation testing is recommended before putting this configuration into production.
Building Block #2 (IB-RDMA)
Write Rate < 5.6 GB/s, Read Rate < 4.4 GB/s*

---

**Ethernet Switch (Administration)**

- **NSD Server-01**
  - x3650 M2
  - 8 cores, 6 DIMMs
- **NSD Server-02**
  - x3650 M2
  - 8 cores, 6 DIMMs
- **NSD Server-03**
  - x3650 M2
  - 8 cores, 6 DIMMs
- **NSD Server-04**
  - x3650 M2
  - 8 cores, 6 DIMMs

**IB Switch (GPFS)**

**DCS9900 (2U) RAID Controller**
- **C1**
  - 1 host port
  - 2 drive ports
  - 3 host ports
  - 4 host ports
- **C2**
  - 2 host ports
  - 3 drive ports
  - 4 host ports

- **60-Bay Disk Tray (4U)**
  - 160 x SAS disks: 72 TB
  - 300 x SATA disks: 300 TB

**Peak sustained DCS9900 performance**
- Streaming data rate < 5.6 GB/s
- Noncached IOP rate < 40,000 IOP/s

**4xDDR IB HCA (Host Channel Adapter)**
- Potential peak data rate per HCA < 1500 MB/s
- Required peak data rate per HCA < 1400 MB/s

**2xFC8 (dual port 8 Gbit/s Fibre Channel)**
- Potential peak data rate per 2xFC8 < 1500 MB/s
- Required peak data rate per 2xFC8 < 1400 MB/s

**COMMENT:**
More disks (for a total of 1200) can be added to this solution but it will **not** increase performance.

---

* Based on benchmark using 160 x 15 Krpm SAS drives.
Alternate Building Block #2 (IB-RDMA)

Write Rate < 5.6 GB/s, Read Rate < 4.4 GB/s*

* Based on benchmark using 160 x 15 Krpm SAS drives.

**COMMENTS - DCS9900 Host Connections**

- While IB 4xDDR (RDMA) can deliver rates up to 1500 MB/s over a LAN, in practice IB 4xDDR (SRP) delivers closer to 1300 MB/s over a SAN. The peak data rate for this solution may therefore be closer to 5.2 GB/s.
- Dual port IB 4xDDR HCAs may be used with all 8 host ports on the DCS9900 to provide failover redundancy. It may also provide a minor performance boost.

**Ethernet Switch (Administration)**

**DCS9900 (2U) RAID Controller**

**IB Switch (GPFS)**

**Disk trays**

- Minimum required to saturate DCS9900 BW
- 160 x SAS disks: 72 TB
- 300 x SATA disks: 300 TB

**Peak sustained DS5300 performance**

- Streaming data rate < 5.6 GB/s
- Noncashed IOP rate < 40,000 IOP/s

**LAN: 4xDDR IB HCA (RDMA)**

- Potential peak data rate per HCA < 1500 MB/s
- Required peak data rate per HCA < 1400 MB/s

**Host connections: 4xDDR IB HCA (SRP)**

- Potential peak data rate per HCA < 1500 MB/s
- Required peak data rate per HCA < 1400 MB/s
The following pages summarize benchmark tests for 2 different DCS9900 systems, one based on SATA and one based on 15Krpm SAS.
Benchmark #1 (SATA)
System Configuration

SOFTWARE:
AIX 5.3 with GPFS 3.2

IB Protocol: IBoIP/sp3
IB HCA: 4X DDR
2xTbE (LACP)

The IB and TbE ports were not used for the results reported below.
**Benchmark #1 (SATA)**

**Streaming Results**

**Common Parameter Settings**

- **AIX**
  - transfer size = 1 MB

- **DCS9900**
  - FW version = 5.03, SATA, 8+2P RAID 6, dual coherency=off, cache size=1024, cache mf=on

- **GPFS**
  - blocksize=4M, maxblocksize=4M, maxMBpS=4000, pagepool=4G, nsdMaxWorkerThreads=128, prefetchPct=60, nsdbufspace=70, worker1Threads=128, prefetchThreads=64
  - all LUNs are dataAndMetadata

- **Benchmark application (ibm.v4b)**
  - record size = 4M
  - access pattern = sequential
  - direct I/O = off
  - fize size >= 16 GB * number of nodes

<table>
<thead>
<tr>
<th>Tiers</th>
<th>nodes</th>
<th>tasks</th>
<th>write rate(^1) (MB/s)</th>
<th>read rate(^2) (MB/s)</th>
<th>write rate(^3) (MB/s)</th>
<th>read rate(^4) (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>277.2</td>
<td>222.3</td>
<td>376.5</td>
<td>392.3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>791.5</td>
<td>709.3</td>
<td>1067.0</td>
<td>1384.0</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>8</td>
<td>1429.0</td>
<td>1238.0</td>
<td>2354.0</td>
<td>3237.0</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>16</td>
<td>2740.0</td>
<td>1607.0</td>
<td>3676.0</td>
<td>4241.0</td>
</tr>
<tr>
<td>32</td>
<td>4</td>
<td>16</td>
<td>4842.0</td>
<td>2885.0</td>
<td>5234.0</td>
<td>5194.0</td>
</tr>
<tr>
<td>64</td>
<td>4</td>
<td>16</td>
<td>5411.0</td>
<td>3581.0</td>
<td>5395.0</td>
<td>4147.0</td>
</tr>
</tbody>
</table>

**Unique Parameter Settings for Each Column**

1. DCS9900 cache writeback=off, GPFS block allocation = scatter
2. DCS9900 cache prefetch=0, GPFS block allocation = scatter
3. DCS9900 cache writeback=off, GPFS block allocation = cluster
4. DCS9900 cache prefetch=1 GPFS block allocation = cluster
Benchmark #1 (SATA)
IOP Results

Common Parameter Settings

- **AIX**
  - transfer size = 1 MB
- **DCS9900**
  - FW version = 5.06, SATA, 8+2P RAID 6, dual coherency=on, cache writeback=off, cache mf=on, cache prefetch=0,
- **GPFS**
  - blocksize=256K, maxblocksize=4M, maxMBpS=4000, pagepool=1G, nsdMaxWorkerThreads=128, prefetchPct=60, nsdbufspace=70, worker1Threads=128, prefetchThreads=96, GPFS block allocation = scatter
  - all LUNs are dataAndMetadata
- **Benchmark application (ibm.v4b)**
  - record size = 4K
  - access pattern = sequential small file
    - 4 files each @ 4K, 8K, 12K, 16K per directory accessed over a binary director tree in preorder
  - direct I/O = off

<table>
<thead>
<tr>
<th>Tiers</th>
<th>nodes</th>
<th>tasks</th>
<th>write rate¹ (IOP/s)</th>
<th>read rate² (IOP/s)</th>
<th>write rate³ (IOP/s)</th>
<th>read rate⁴ (IOP/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>32</td>
<td>7522.5</td>
<td>3840.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>32</td>
<td>13459.0</td>
<td>5909.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>32</td>
<td>23641.0</td>
<td>12061.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>32</td>
<td>29852.0</td>
<td>16301.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>32</td>
<td>30381.0</td>
<td>27947.0</td>
<td>40978.0</td>
<td>32870.0</td>
</tr>
<tr>
<td>32</td>
<td>4</td>
<td>32</td>
<td>41128.0</td>
<td>33528.0</td>
<td>47658.0</td>
<td>32870.0</td>
</tr>
</tbody>
</table>

Unique Parameter Settings for Each Column
1. DCS9900 cache size=1024
2. DCS9900 cache size=1024
3. DCS9900 cache size=256
4. DCS9900 cache size=256
Benchmark #2 (SAS)  
System Configuration
Benchmark #2 (SAS)
Streaming Results

Common Parameter Settings

- **DCS9900**
  - SAS @ 15Krpm, 8+2P RAID 6, dual coherency=on, cache writeback=on, cache mf=on, cache prefetch=0

- **GPFS**
  - blocksize=4M, maxblocksize=4M, maxMBpS=4000, GPFS block allocation = scatter
  - all LUNs are dataAndMetadata

- **Benchmark application (xdd)**
  - access pattern = sequential large file
  - direct I/O = off

<table>
<thead>
<tr>
<th>Tiers</th>
<th>nodes</th>
<th>threads</th>
<th>record size (KB)</th>
<th>write rate (MB/s)</th>
<th>read rate (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>8</td>
<td>16</td>
<td>128</td>
<td>2506.16</td>
<td>4539.54</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>16</td>
<td>256</td>
<td>2486.71</td>
<td>4517.87</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>16</td>
<td>512</td>
<td>2485.51</td>
<td>4531.71</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>16</td>
<td>1024</td>
<td>2485.58</td>
<td>4512.11</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>16</td>
<td>2048</td>
<td>2480.43</td>
<td>4499.71</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>16</td>
<td>4096</td>
<td>5745.67</td>
<td>4453.76</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>16</td>
<td>8192</td>
<td>5749.74</td>
<td>4475.39</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>16</td>
<td>16384</td>
<td>5752.19</td>
<td>4487.19</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>16</td>
<td>32768</td>
<td>5743.40</td>
<td>4473.32</td>
</tr>
</tbody>
</table>
The following pages contain information on common NSD servers used with the DCS9900. These servers can generally be interchanged on a 1 to 1 basis, but there are nuanced differences.
The x3650 M2 is a common and cost effective storage server for GPFS in most System X environments (it can even be used with System P).

This diagram illustrates those features most useful to its function as a storage server.

**Data rates are based on the Gen 1 PCIe standard**.*

1. Listed bus rates are theoretical duplex rates assuming on 512 MB/s per link. Production data will be less.
2. Actual peak duplex rates for PCIe x8 adapter < 3.2 GB/s
3. Measured aggregate over 4 x PCIe x8 adapters < 7.3 GB/s

* See http://en.wikipedia.org/wiki/PCIe for details on the PCI Express standard
The x3550 may be a cost effective storage server for GPFS in some cases. It's main limitation is a lack of PCIe slots.

This diagram illustrates those features most useful to its function as a storage server.

Data rates are based on the *Gen 1 PCIe standard*.  
1. Listed bus rates are theoretical duplex rates assuming on 512 MB/s per link. Production data will be less.
2. Actual peak duplex rates for PCIe x8 adapter < 3.2 GB/s
3. Measured aggregate over 4 x PCIe x8 adapters < 7.3 GB/s

*See http://en.wikipedia.org/wiki/PCIe for details on the PCI Express standard*
**Overview**

The P6-p520 is a cost effective storage server for GPFS in most pSeries clusters using Ethernet. This diagram illustrates those features most useful to its function as a storage server.

**"DIRECT" GX Slot**
- IB cards are only supported in this slot.
- Card options:
  - dual port, IB 12xSDR @ 6:1 ratio (GX+)
  - dual port, IB 12xDDR @ 3:1 ratio (GX++)
  - RIO2 card @ 8:1 (GX+)
- 12x IB ports 1X and 4X cables
  - requires special "width changer" cable

- GX Bus width: 32 bits
- Rules of thumb:
  - Sustained simplex rates < 80% of simplex burst rate
  - Sustained duplex rates < 60% of duplex burst rate
  - single SDR "lane"
    - burst < 250 MB/s, sustained < 185 MB/s
  - single DDR "lane"
    - burst < 500 MB/s, sustained < 375 MB/s

**"Pass Thru" GX Slot**
- The pass thru GX slot occupies the same physical space as the 1st PCI-E slot. Therefore you cannot use both of these slots.
- Supports the RIO2 card @ 8:1 (GX+). It does **not** support IB card.

**Single PCI Adapter Data Rates**
- PCI-E 8x:
  - Simplex: Burst < 2000 MB/s, Sustained < 1400 MB/s
  - Duplex: Burst < 4000 MB/s, Sustained < 2100 MB/s
- PCI-X 2.0
  - Burst < 2000 MB/s, Sustained < 1400 MB/s (this is not a duplex protocol)

---

**NOTE:** Requires 4 core configuration in order to enable the "direct" GX slot.