DS3700 Best Practices Guide
For Use in an HPC Environment with GPFS

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

6 Feb 12

version 3.3c

raypaden@us.ibm.com
512-286-7055

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 inquires, 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.

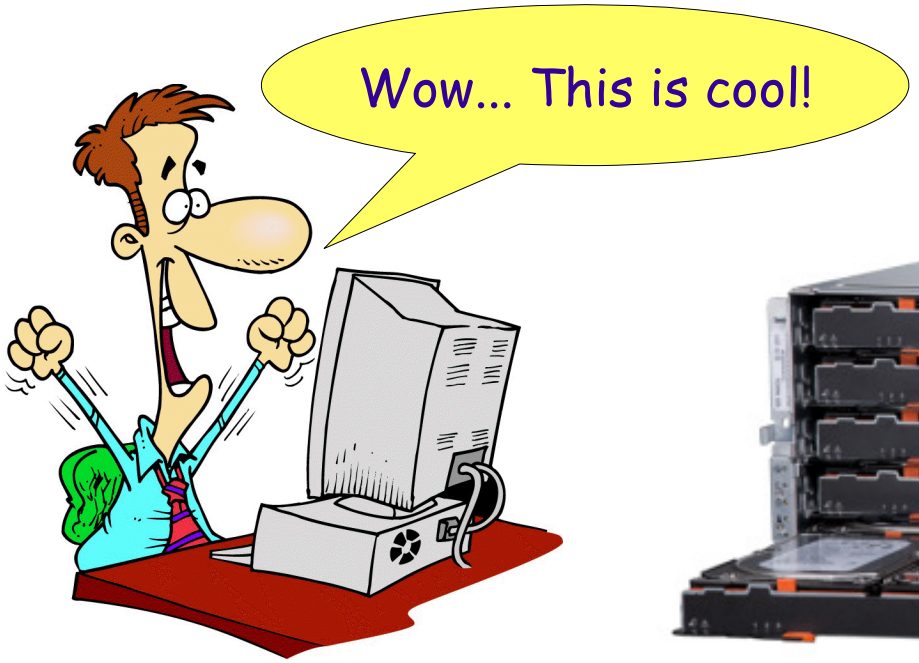
Other company, product and service names may be trademarks or service marks of others.

Outline

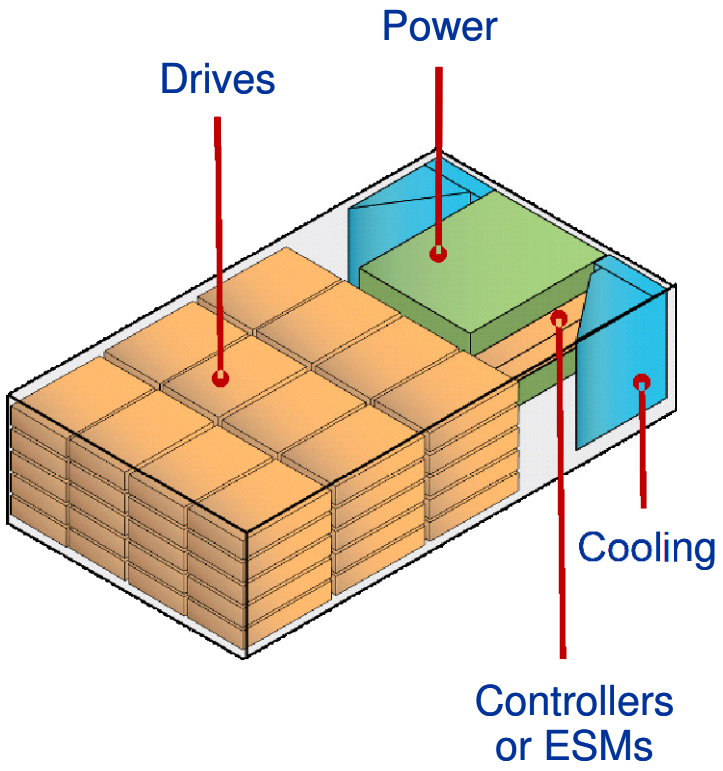
1. Product Overview: What is the DCS3700?
   Physical and Architectural Views
2. Solution Examples
   a. Solution Design Strategies and Performance Measurement
   b. Solution #1 – Maximum Capacity
   c. Solution #2 – Balanced Performance/Capacity
   d. Solution #3 – Optimizing IOP Performance
      A Multi-tier Solution Using the DCS3700 with the DS3524
   e. Component Availability and Selection
3. Using the DCS3700
   a. Benchmark configurations
      Phase 1
      Phase 2
   b. Example RAID 5 and RAID 6 configurations
   c. Using Dual Port 10 Gbit/s Ethernet (2xTbE)
   d. Multi-Pathing Architecture
   e. Installing and Configuring GPFS for a DCS3700
      Based on the Phase 1 configuration
   f. Selected Benchmark Results

DCS3700 – Product Overview

The following pages introduce the DCS3700, providing physical and architectural views and related details.

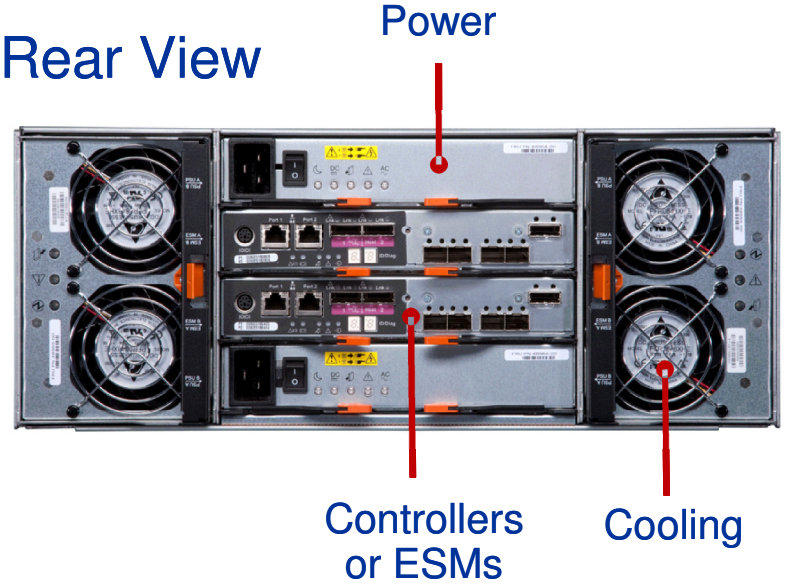


DCS3700 – Enclosure Views



Front View

Rear View



Model Numbers
- DCS3700 = 1818-80C
- DCS3700 Expansion Tray = 1818-80E
  This is sometimes called the DCS3700E

Notes:
- The DCS3700 uses the same RAID controllers as the DS3512 and DS3524.
- Prior to their acquisition by NetApp, the Enginio division of LSI was the OEM for the FAStT, DS3000, DS4000, DS5000 families of storage controllers.

DCS3700 – Enclosures, Trays, Drawers and Disks

Supported disks
- 2 TB, 3.5”, 7200 RPM
- 3 TB, 3.5”, 7200 RPM
- 300 GB, 2.5”, 15000 RPM
- 200 GB, 2.5”, SSD
- 400 GB, 2.5”, SSD

Front View without bezel

Drawer #1 →
Drawer #2 →
Drawer #3 →
Drawer #4 →
Drawer #5 →

Tray or Enclosure

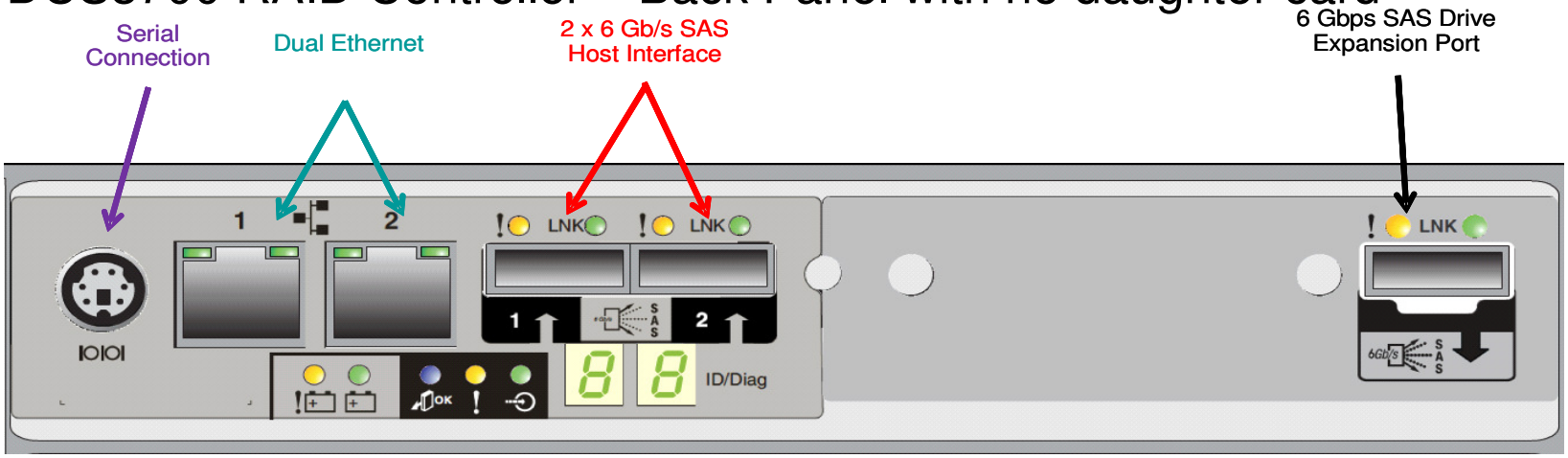
Enclosures are called “trays”:
- a “tray” contains 5 x “drawers”
- a “drawer” contains 12 x disks

Rear View showing an enclosure with dual RAID controllers+

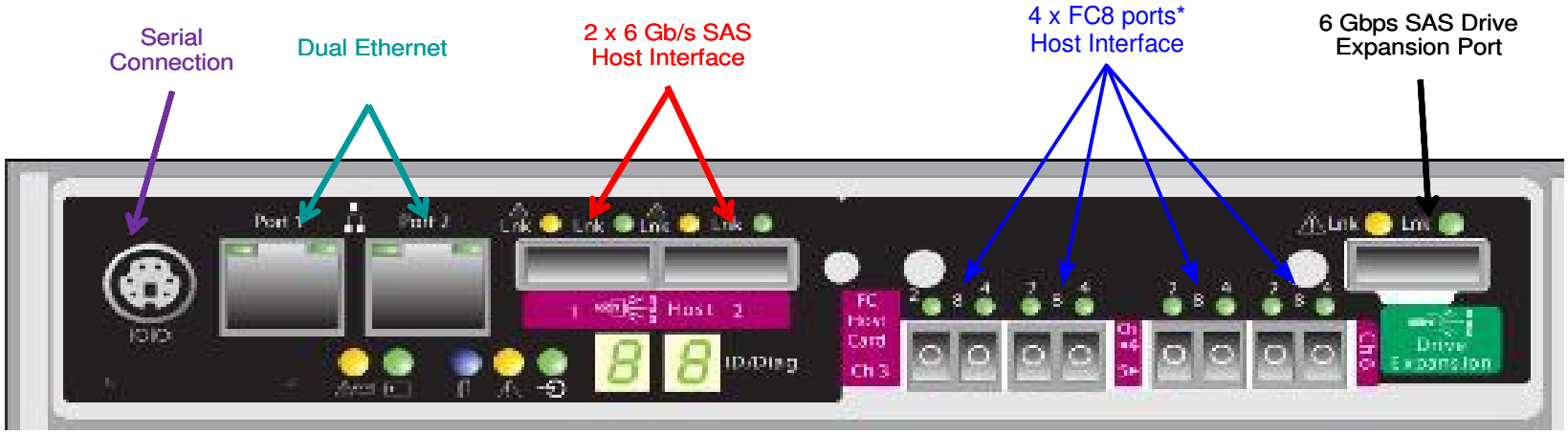
+ The DCS3700 is an *enclosure* containing dual *RAID controllers* (as shown here) plus 5 disk *drawers* (n.b., “internal disks”). An *expansion tray* is an *enclosure* containing dual *ESMs (Electronic Service Modules)* plus 5 disk *drawers* (n.b., external disks). The *ESMs* only provide connectivity from the *expansion trays* to the *RAID controllers* while the *RAID controllers* provide the “intelligence” controlling the storage functions. The *ESMs* and *RAID controllers* fit in the same physical slot. A DCS3700 is simply called a “*controller*” and contains the 2 *RAID controllers*.

DCS3700 – RAID Controller View

DCS3700 RAID Controller – Back Panel with no daughter card

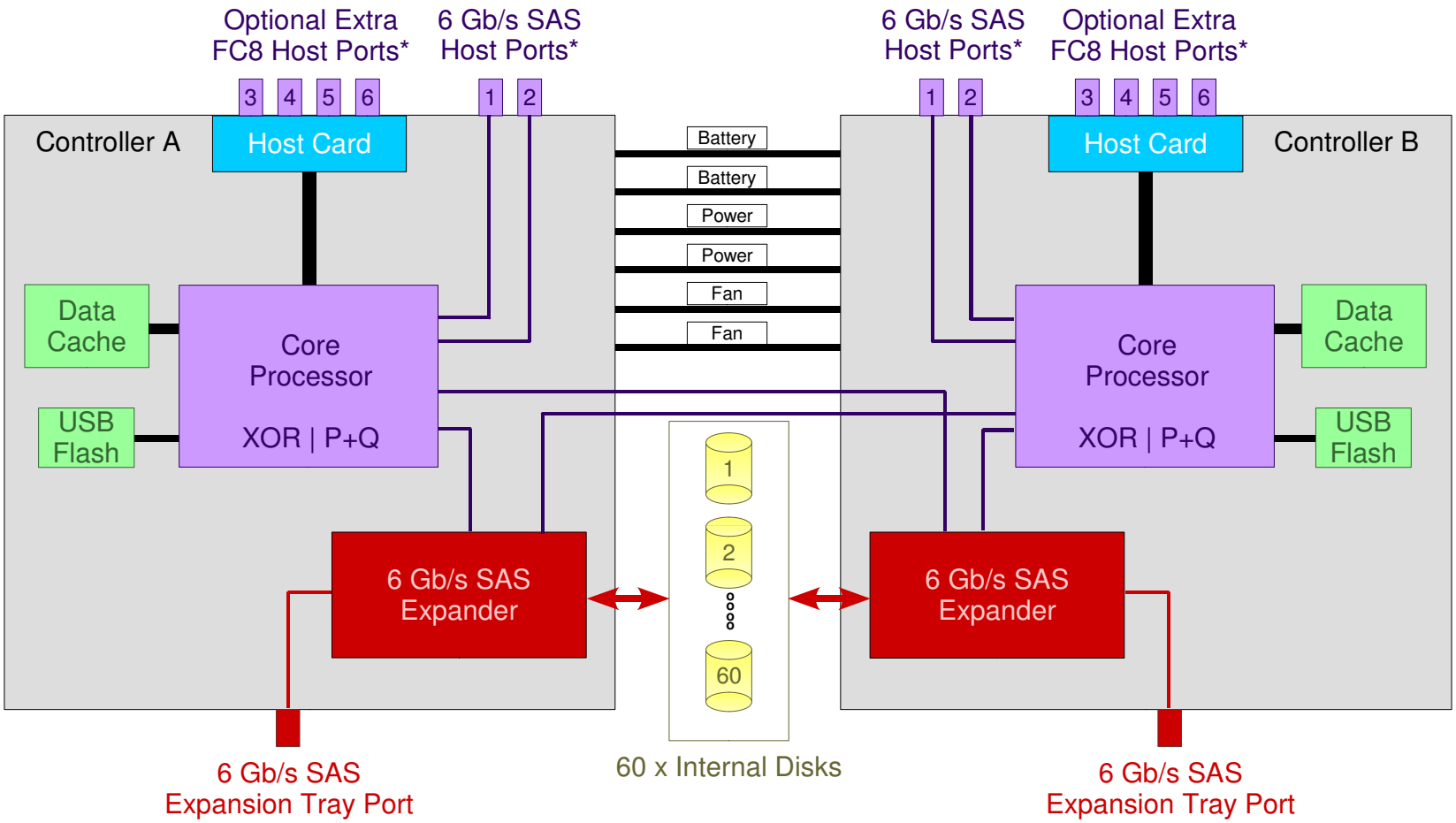


DCS3700 RAID Controller – Back Panel with Fibre Channel daughter card*



* The Fibre Channel daughter card is optional and not always necessary since the dual SAS port HBAs that come standard with the DCS3700 are sufficient to harvest it's full bandwidth. However, under some circumstances, the FC8 daughter may be useful, or even necessary. Following slides provide more details.

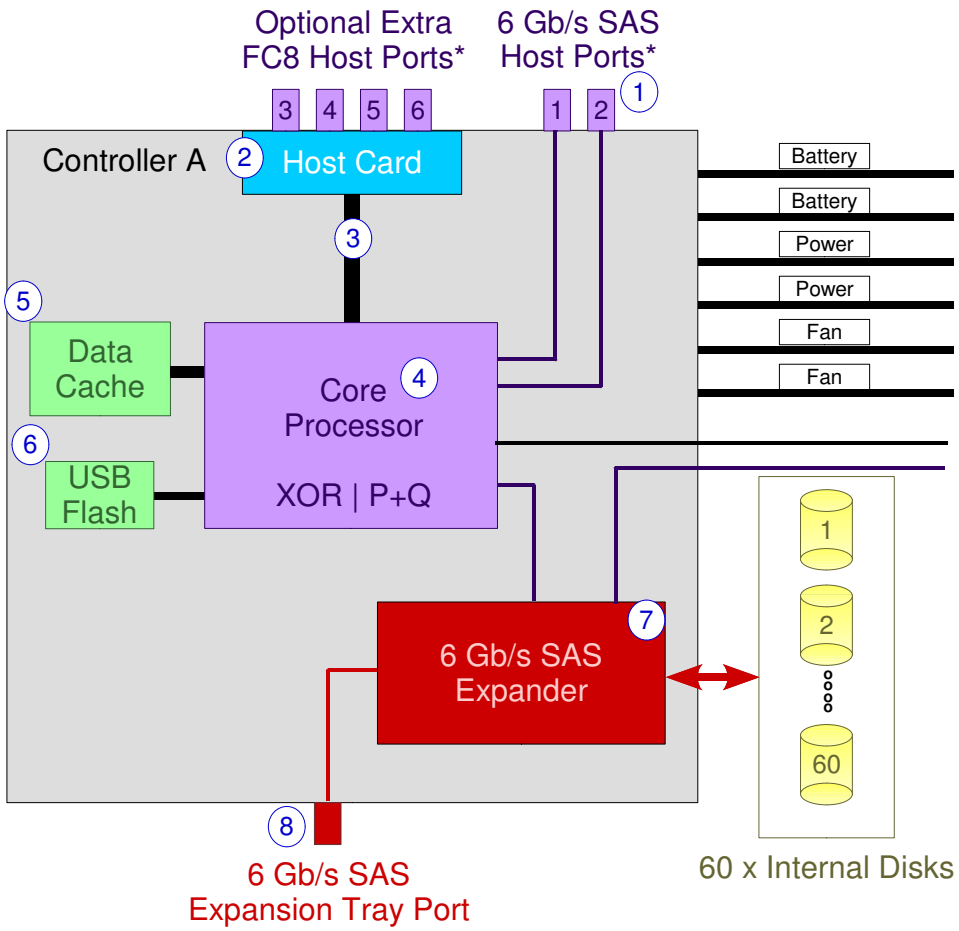
DCS3700 – Dual RAID Controller Architecture



* **Host Connections**

The DCS3700 supports 2 x 6 Gb/s SAS connections (via built in adapters) or 4 x FC8 connections (via an optional "daughter card") per RAID controller. It is necessary to use one type of connection or the other, but not both. The peak BW for the DCS3700 is the sum of the ports used from both RAID controllers. See the following slide for more details.

DCS3700 – RAID Controller Components



1. 2 x Gb/s SAS Host ports (standard)
2. Extra host ports*; optional daughter card supporting 2 alternatives:
   - 2 x 6 Gb/s SAS ports
   - 4 x 8 Gb/s Fibre Channel Ports
3. PCI-E 2.0 x8 buses interface between ASIC and external interfaces providing 4 GB/s
4. 6 Gb/s SAS Power PC 440 ROC 800 MHz
   - Super scalar core, 1.9x MIPS mulitplier = 1.5 GHz x-scale
   - Performs both XOR and P+Q calculations
5. 1 or 2 GB data cache memory
6. Cache destage to flash upon a power loss
7. SAS port expander for internal disks
8. Single 6 Gb/s SAS drive expansion port

*** Host Connections**

The DCS3700 supports 2 types of host connections.

1. The standard connection is a built in, dual port, 6 Gb/s SAS adapter capable of sustaining peak performance.
2. An optional daughter card provides 4 x FC8 ports (i.e., 8 Gb/s fibre channel).

Each 6 Gb/s SAS port consists of “4 lanes” for an aggregate line rate = 24 Gb/s. Thus 1 x SAS port = 3 x FC8 ports. For this reason, SAS host connections are preferred and recommended in this document. However, if cable lengths longer than those available for SAS connections are required or if multi-port server SAS HBAs are not easily available, then FC8 connections may be used. However, while it is common practice to use 2 x FC8 ports per RAID controller, in disk configurations capable of sustaining > 3 GB/s more FC8 ports may be needed (e.g., 120 x 15000 RPM disks).

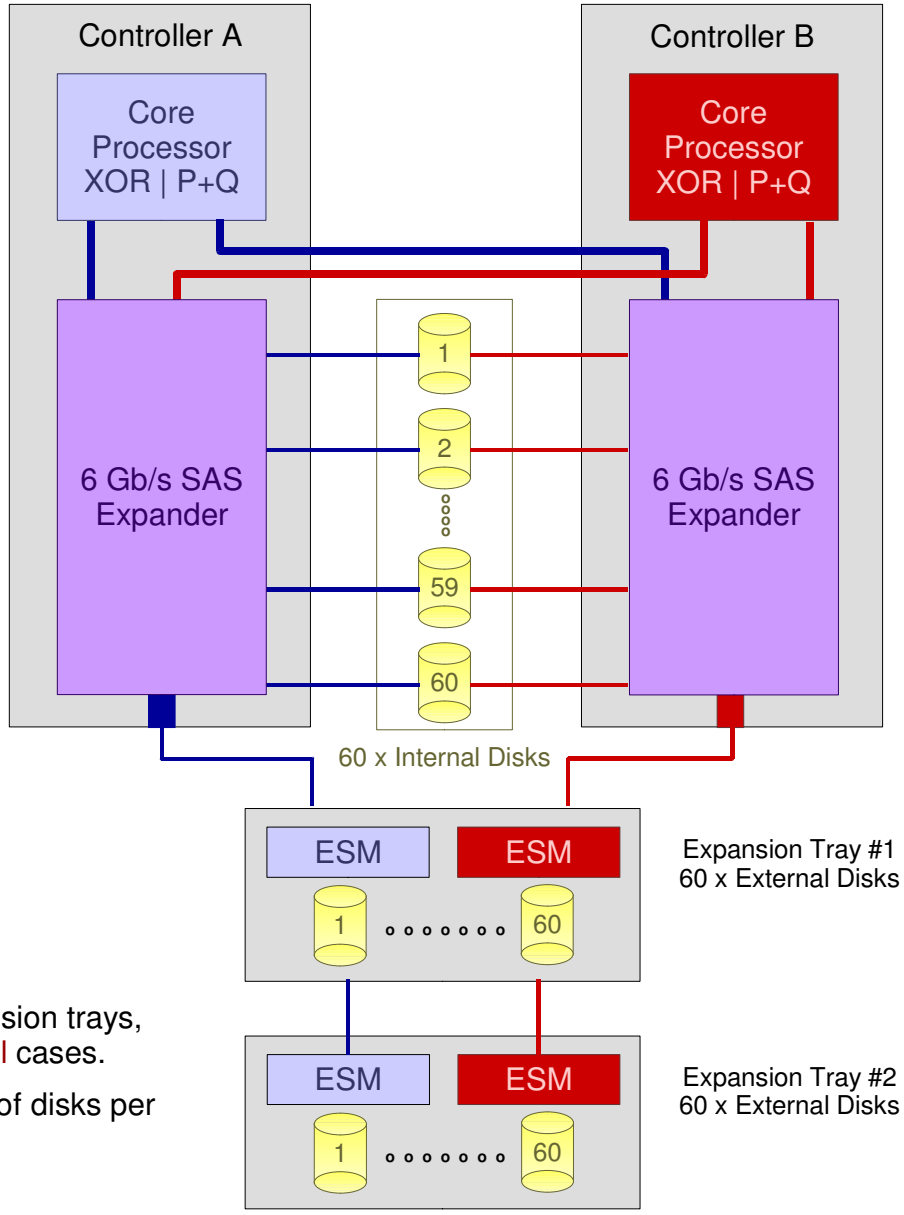
DCS3700 – Disks, Expansion Trays and Drive Channels

Connectivity

- Fully redundant pathing from host to drive
- Each controller can access all drive ports
- Each drive chip can access every drive

Disks and Expansion Trays

- Solid State Disk, 2.5”
  200 GB, 400 GB
- 15000 RPM, 2.5”, SAS
  300 GB
- 7200 RPM, 3.5”, Near Line SAS
  2 TB, 3 TB
- Supports 60 internal disks
- Supports up to 180 disks by adding up to 2 expansion trays (60 disks per tray).



COMMENT: Adding Expansion Trays

The DCS3700 can increase capacity by adding expansion trays, but this will not significantly improve performance in all cases.

When including expansion trays, be sure the number of disks per tray (both internal and external disks) is the same.

DCS3700 – Solution Examples

You've got a problem!

The following pages discuss solution strategies and example solutions. Most of them are based on 2 TB NL SAS disk.

You have requirements...
- Data rate
- Data capacity
- Disk technology
- LAN
- Servers
- Clients
- Cost

Now you need to put them into a solution!

Solution Examples Emphasize 2 TB, 7200 RPM Disk

1. **Disk Technology**
   The current version of this slide set emphasizes solutions using 7200 RPM NL-SAS disks which is the most common case. While most of the solutions presented can also use 2.5” 15000 RPM SAS disk as well as SSD, the performance and capacity calculations will be very different as well as the number of disks needed to achieve balance.

   A future version of this slide set will include solutions optimized for the use faster, lower capacity disks.

   However, the spreadsheet associated with this slide set documents the results and configuration information for tests using 2.5” 15000 RPM SAS disk.

2. **Capacity Calculations**
   Unless stated otherwise, capacity calculations for the current version of this slide set are based on 2 TB per disk. If the recently announced 3 TB NL SAS disks are adopted, raw and usable capacities will increase 50%.

   A future version of this slide set will include calculations based on 3 TB disks. However, it is important to keep in mind that as disk sizes in increase, the number disks needed to satisfy capacity requirements may be insufficient to meet performance requirements; *i.e.*, as capacity per disk increases, the performance to capacity ratio decreases.

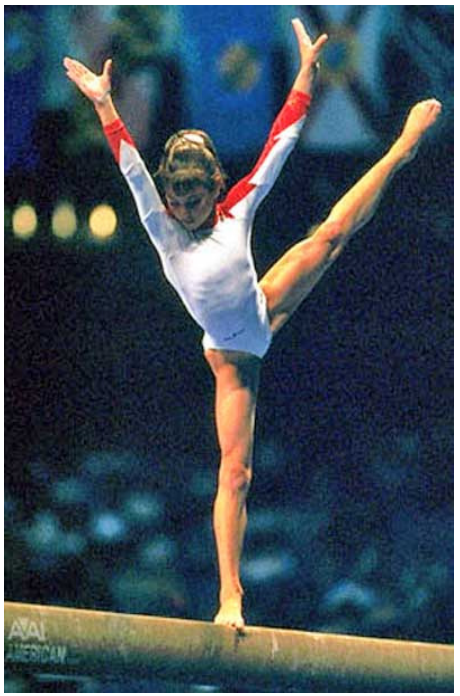
Solution Design Strategies: Balance

Ideally, an I/O subsystem should be balanced. There is no point in making one component of an I/O subsystem fast while another is slow. Moreover, overtaxing some components of the I/O subsystem (e.g., HBAs) may disproportionately degrade performance.

However, this goal cannot always be perfectly achieved. A common imbalance is when capacity takes precedence over bandwidth; then the aggregate bandwidth based on the number of disks may exceed the aggregate bandwidth supported by the controllers and/or the number of storage servers.

This presentation illustrates solutions achieving balance in varying degrees.

1. Solutions providing *maximum capacity* balance the number of servers and network adapters with the number of controllers. However, the number of disks per controller has greater potential bandwidth than can be consumed by their managing controller.
2. Solutions providing *balanced performance/capacity* balance the number of disks per controller as well as the number of servers and network adapters with the number of controllers.



Performance is inversely proportional to capacity.
- Todd Virnoche

Solution Design Strategies: Building Blocks

A useful design strategy for storage solutions is to define a "storage building block" which is the smallest increment of storage, servers and networking by which a storage system can grow. The GPFS file system supports this architecture and is used in the solutions illustrated on the following pages.

Using this architecture, a storage solution consists of 1 or more storage building blocks. This allows customers to conveniently expand their storage solution in increments of storage building blocks (i.e., "build as you grow" strategy).

COMMENTS:

- **Linear Scaling**: GPFS readily supports this architecture since it scales linearly in the number of disks, storage controllers, NSD servers, GPFS clients, and so forth.
- **Granularity**: One of the advantages of the DCS3700 is that it allows for solutions with “smaller” building blocks. This allows greater flexibility in scaling out compared to controllers requiring a larger number disks to achieve full bandwidth (*c.f.*, DS5300, DCS9900).

Solution Design Strategies: Building Block Components

The building blocks presented in this document are based on the following components:

- Storage: primarily the DCS3700
- Servers: x3650 M3
- Networking
  LAN: IB QDR, TbE
  SAN: primarily SAS
- System software
  File system: GPFS
  OS: Linux
  Driver: RDAC-MPP

Comment on Units

- Units for performance and capacity[1] are generally given in units of 2^n with the following prefixes[2]: K = 2^{10}, M = 2^{20}, G = 2^{30}, T = 2^{40}, P = 2^{50}

Footnotes:

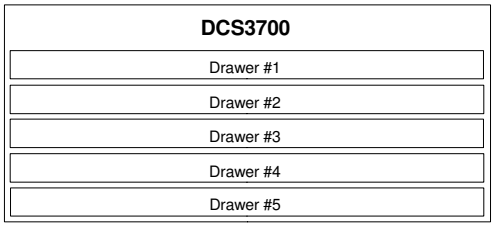
1. The unit prefix for raw capacity is ambiguous; it is simply the value assigned by the OEM.
2. Alternatively, the International System of Units (SI) recommends the prefixes Ki, Mi, Gi, Ti, Pi
   See http://physics.nist.gov/cuu/Units/binary.html

Solution Design Strategies: Capacity *vs.* Performance

Capacity *vs.* Performance[1] (using 2TB 7200[2] RPM Disks)

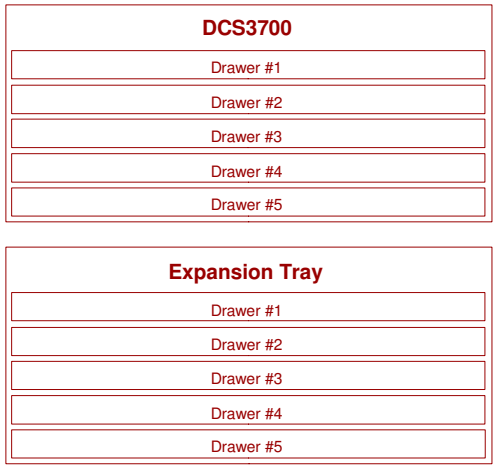
Since 60 x 7200 RPM disks *nearly* achieves their full *streaming*[3] bandwidth on a single DCS3700, then 2 x DCS3700s is *nearly* 2X faster than 1 x DCS3700 + 1 x Expansion tray. Therefore, solutions using multiple DCS3700s are preferable over solutions using multiple expansion trays when it is necessary to optimize performance.

1 x DCS3700



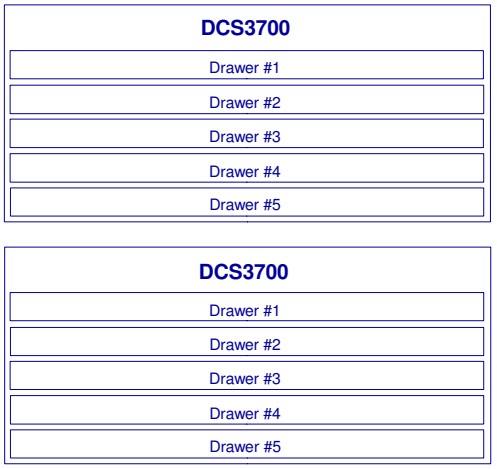
Raw Capacity = 120 TB
Max Streaming Bandwidth
- write < 1.6 GB/s
- read < 2.0 GB/s

1 x DCS3700 + 1 x Expansion Tray



Raw Capacity = 240 TB
Max Streaming Bandwidth
- write < 2.2 GB/s
- read < 2.4 GB/s

2 x DCS3700



Raw Capacity = 240 TB
Max Streaming Bandwidth
- write < 3.2 GB/s
- read < 4.0 GB/s

FOOTNOTE:

1. This observation does not apply to random IOP** work loads when using “spinning disks” with slow seek times (e.g., 3.5” near line SAS disk @ 7200 RPM). Independent of the of disks, a DCS3700 can sustain up to 12,500 write IOP/s and 40,000 read IOP/s (*n.b.*, these are “to media” rates, not merely cache rates). Since typical IOP workloads are randomly distributed, 60 of these near line disks can sustain approximately 1200 write IOP/s and 2000 read IOP/s; therefore even with the maximum number of disks supported by the DCS3700 (*n.b.*, 180), a typical IOP workload would only sustain 3600 write IOP/s and 6000 read IOP/s which falls far below the capability of a DCS3700. By contrast, a streaming workload using between 60 to 80 near line SAS disks will saturate a DCS3700 which leads to this observation.
2. The trade-off for 15000 RPM disk is not as clear cut. See the spreadsheet accompanying this document for performance measurements.
3. Streaming and IOP workloads are carefully defined below.

Observations Regarding Performance

Performance Projections

Performance projections used in these sample solutions are based on benchmarks using production environment parameters under GPFS best practice guidelines measured by HPC I/O benchmarks. Higher data rates are easily achievable using benchmark codes, but are not reflective of typical production environments. Stated rates are least upper bounds, reproducible generally within 10% using standard HPC benchmark codes (e.g., gpfsperf, ibm.v4c, IOR, XDD) with appropriate parameters in a properly configured production environment. Actual performance using a typical mixture of application codes are unlikely to exceed the projected data rates. Benchmark configurations and associated measurements used for these performance projections are included toward the end of this document and in an associated spread sheet.

Streaming *vs.* IOP Performance

- *Streaming performance* is based on *logically* sequential access patterns (*i.e.*, tasks access disjoint file records with seek offsets increasing monotonically in the seek offset space[1]). However, file records will not necessarily be physically contiguous on the disks. If they are, streaming performance is improved, but due to entropy and thread scheduling, it is very unlikely that file records will be accessed in a physically sequential access pattern. Streaming access patterns are typically measured in MB/s or GB/s.
- *IOP performance* is based on small (*i.e.*, 10's of KB or less), randomly distributed file transactions. They can be small files which will be randomly distributed over the disks physically or they can be small records accessed randomly in a file. Generally, cache locality is very poor for IOP work loads. IOP workloads are typically measured in IOP/s, transactions/s or files/s.
- IOP workloads are much more challenging regarding performance than streaming workloads. For streaming workloads, larger transactions are amortized over each seek arm movement[2] while for IOP workloads, small transactions are amortized over each seek arm movement[2]. Therefore, *it is imperative* to use disks[2] supporting faster seek times for IOP workloads! (*n.b.*, it is *not* considered a best practice to use 3.5” near line SAS disks for IOP dominate workloads.

FOOTNOTES:

1 .Start at the beginning of a file and go to the end of the file.

2 .Seek times: 2.5” 15,000 RPM disk ~= 2 milliseconds, 3.5” 15,000 RPM disk ~= 3.6 milliseconds, 3.5” 7200 RPM disk ~= 9 milliseconds.

Solution #1: Overview

Solution #1 provides 2 similar capacity optimized building blocks. Capacity is optimized by deploying the maximum number of disks per DCS3700.

Given the lower performance to capacity ratio of these solutions, TbE generally provides adequate LAN performance (though IB QDR can easily be used in its place). However, one building block solution is proposed useing the higher performance of IB QDR to reduce the number of servers.

A secondary goal of this solution is optimize rack utilization.

2 solutions are illustrated.

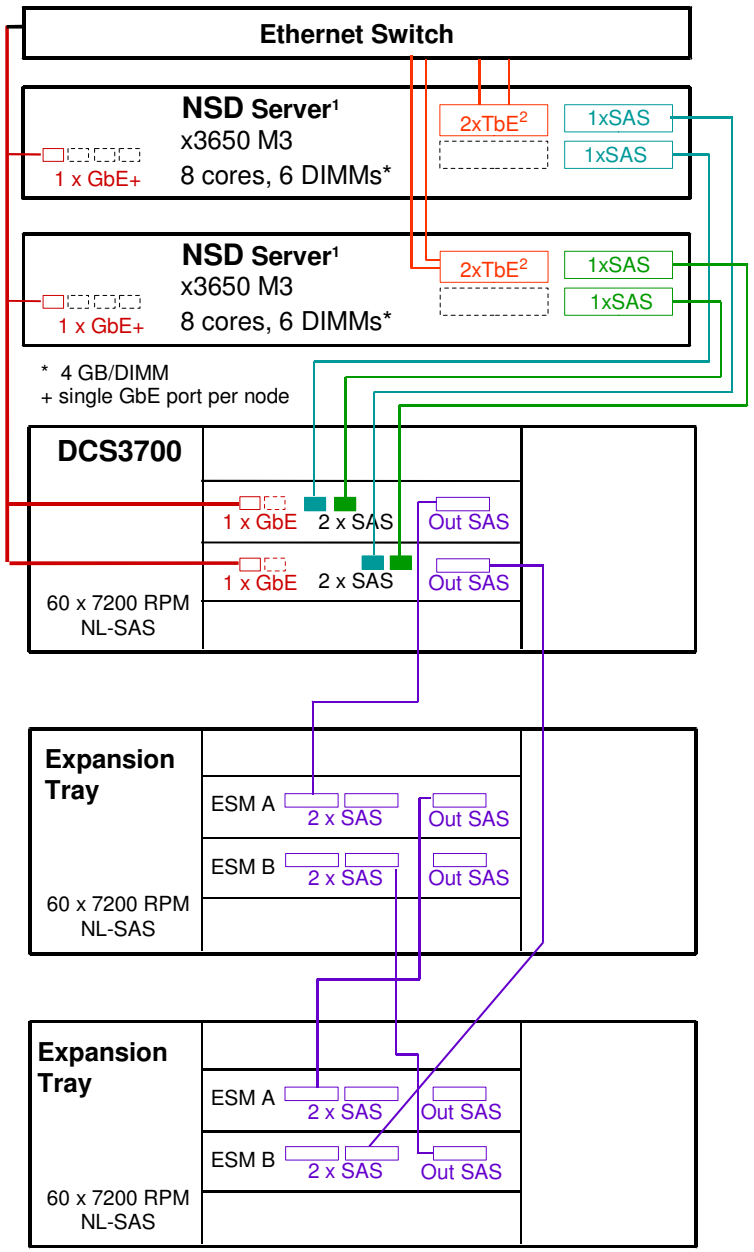
1a: Basic Building Block using 2xTbE

1. Solution using standard components
2. Denser solution using slightly different components

1b: Basic Building Block using IB QDR

LAN: 2xTbE
n.b., IB QDR can easily replace TbE.

Solution #1a: Basic Building Block



Analysis

NSD Server

- Effective BW per NSD server < 1.4 GB/s
  - x3650 M3 with 8 cores and 6 DIMMs (4 GB per DIMM)
  - 1 x GbE < 80 MB/s
  - 2 x TbE[2] < 1.4 GB/s
  - 2 x single port 6 Gb/s SAS adapters

1 x DCS3700 Turbo with 2 x EXP3560 trays

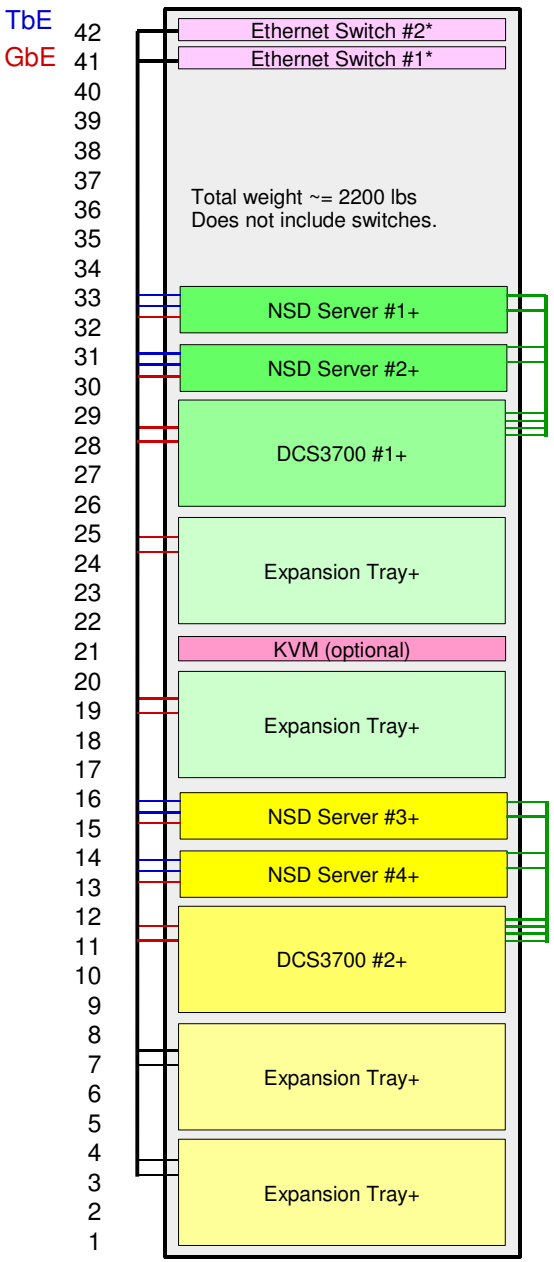
- 180 x 2 TB near line SAS disks
- 18 x 8+2P RAID 6 arrays
- Capacity: raw = 360 TB, usable < 262 TB[3]
- Performance
  - Streaming rate: write < 1.6 GB/s[4], read < 2.0 GB/s[4]
  - IOP rate (random 4K transactions): write < 3600 IOP/s[5], read < 6000 IOP/s[5]

FOOTNOTES:

1. The x3650 M3 can be replaced with an x3550 M3 if a single dual port port SAS HBA in place of 2 single port SAS HBAs.
2. An IB QDR HCA can replace the dual port TbE adapter; performance will not increase.
3. The DCS3700 provides a capacity of 14.55 TB per RAID 6 array for the file system to use.
4. The stated streaming rates are least upper bounds (LUB); these rates are based on GPFS/DCS3700 benchmarks using 60 x 7200 RPM Near Line SAS disks. Extrapolating from other tests, greater LUB rates may be expected (e.g., write < 1.7 GB/s and read < 2.4 GB/s using at least 80 of these disks).
5. These rates are extrapolated from actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk. These tests assume completely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly.

LAN: 2xTbE
n.b., IB QDR can easily replace TbE.

Solution #1a.1: Physical View



COMPONENTS

4 x NSD servers (x3650 M3) each with the following components:
- 2 x quad core westmere sockets, 6 x DIMMs (2 GB or 4 GB per DIMM)
- 1 x GbE, 2 x TbE or 1 x IB QDR, 2 x single port SAS (6 Gb/s)

2 x DCS3700, each with the following components
- 2 x Expansion Tray
- 180 x 2 TB, 7200 RPM Near Line SAS disks as 6 x 8+P+Q RAID 6 arrays
- Capacity: raw = 360 TB, usable < 262 TB
- 4 x SAS host ports @ 6 Gb/s; n.b., 2 SAS host ports per RAID controller

Switches: Provide Ethernet and IB switches as needed.

Comment: This configuration consists of 2 building blocks. Adding additional building blocks will scale performance and capacity linearly.

AGGREGATE STATISTICS

Disks
- 360 x 2 TB, 7200 RPM Near Line SAS disks
- 36 x 8+P+Q RAID 6 arrays, 1 LUN per array

Capacity
- raw = 720, usable < 524 TB[1]

Performance
- streaming rate : write < 3.2 GB/s, read < 4 GB/s
- IOP rate (random, 4K transactions): write < 7200 IOP/s, read < 12,000 IOP/s[2]

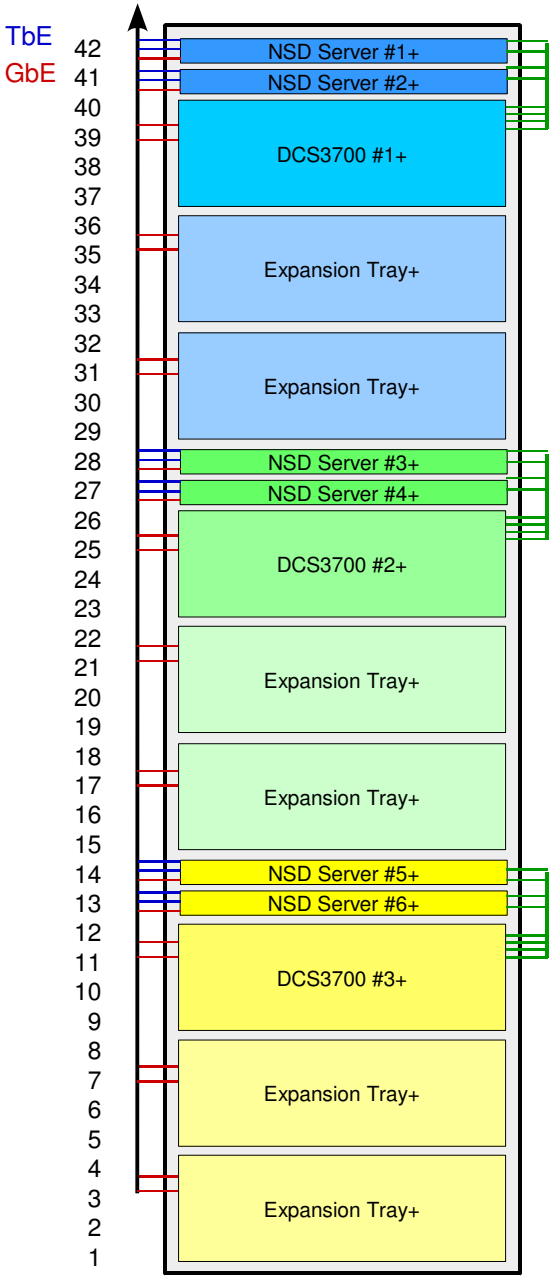
COMMENT: Maintaining good streaming performance requires careful attention being given to balance. Alterations disrupting balance (e.g., inconsistent number of disks or expansion trays per DCS3700) will compromise performance.

FOOTNOTES:
1.Usable capacity is defined as the storage capacity delivered by the controller to the file system. Additionally, file system metadata requires a typically small fraction of the usable capacity. For the GPFS file system, this is *typically* < 1.5% (~= 8 TB in this case). With a very large number (e.g., billions) of very small files (e.g., 4 KB), the metadata capacity *may* be much larger (e.g., > 10%). Metadata overhead in this case is application environment specific and difficult to project.
2.These numbers are based purely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly. (n.b., These rates are based on actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk.)

* These switches are included in this diagram for completeness. If the customer has adequate switch ports, then these switches may not be needed.
\+ Due to SAS cable lengths (3M is recommended) it is necessary to place the NSD servers in the same rack as the controllers.

LAN: 2xTbE
n.b., IB QDR can easily replace TbE.

Solution #1a.2: Physical View



COMPONENTS

4 x NSD servers (**x3550 M3**) each with the following components:
- 2 x quad core westmere sockets, 6 x DIMMs (2 GB or 4 GB per DIMM)
- 1 x GbE, 2 x TbE or 1 x IB QDR, **1 x dual port SAS (6 Gb/s)**

3 x DCS3700, each with the following components
- 2 x Expansion Tray
- 180 x 2 TB, 7200 RPM Near Line SAS disks as 6 x 8+P+Q RAID 6 arrays
- Capacity: raw = 360 TB, usable = 262 TB
- 4 x SAS host ports @ 6 Gb/s; n.b., 2 SAS host ports per RAID controller

Switches: Provide Ethernet switches as needed.

Comment: This configuration consists of 3 building blocks. Adding additional building blocks will scale performance and capacity linearly.

AGGREGATE STATISTICS

Disks
- 540 x 2 TB, 7200 RPM Near Line SAS disks
- 54 x 8+P+Q RAID 6 arrays, 1 LUN per array

Capacity
- raw = 1080, usable = 786 TB[1]

Performance
- streaming rate : write < 4.8 GB/s, read < 6 GB/s
- IOP rate (random, 4K transactions): write < 10,800 IOP/s, read < 18,000 IOP/s[2]

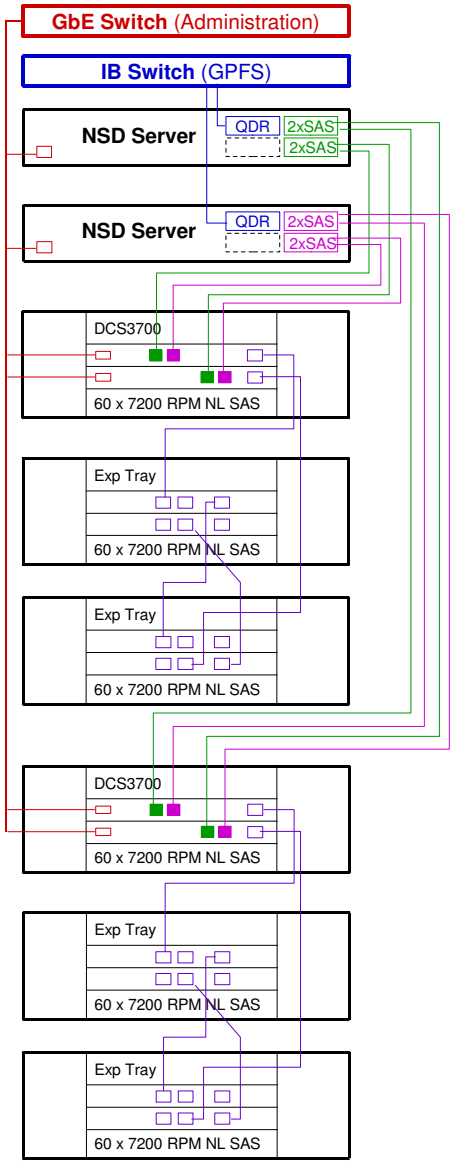
COMMENT: Maintaining good streaming performance requires careful attention being given to balance. Alterations disrupting balance (e.g., inconsistent number of disks or expansion trays per DCS3700) will compromise performance.

FOOTNOTES:
1.Usable capacity is defined as the storage capacity delivered by the controller to the file system. Additionally, file system metadata requires a typically small fraction of the usable capacity. For the GPFS file system, this is *typically* < 1.5% (~= 12 TB in this case). With a very large number (e.g., billions) of very small files (e.g., 4 KB), the metadata capacity *may* be much larger (e.g., > 10%). Metadata overhead in this case is application environment specific and difficult to project.
2.These numbers are based purely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly. (n.b., These rates are based on actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk.)

COMMENT: This solution is similar to #1a.1, but uses using slightly different components in the building block (*n.b.*, x3550 M3, dual port SAS HBA). This allows 3 building blocks to fit into a single rack.

* These switches are included in this diagram for completeness. If the customer has adequate switch ports, then these switches may not be needed.
\+ Due to SAS cable lengths (3M is recommended) it is necessary to place the NSD servers in the same rack as the controllers.

Solution #1b: Basic Building Block



Analysis

NSD Server

- Effective BW per NSD server < 3.0 GB/s
  - x3650 M3 with 8 cores and 6 DIMMs (4 GB per DIMM)
  - 1 x GbE < 80 MB/s
  - 1 x IB QDR[2] < 3.0 GB/s
  - 2 x single port 6 Gb/s SAS adapters

2 x DCS3700 Turbo each with 2 expansion trays

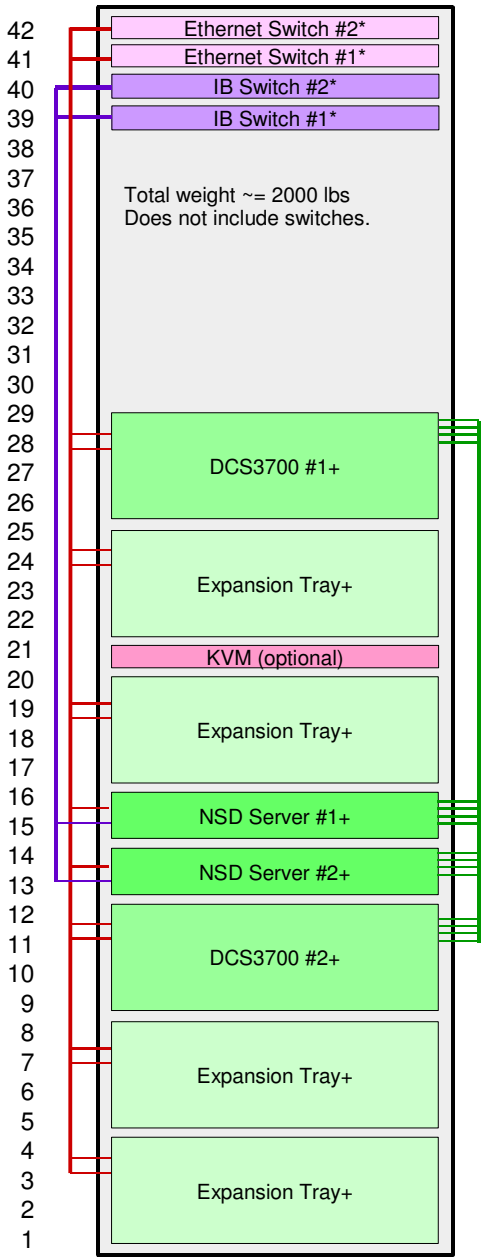
- 360 x 2 TB near line SAS disks
- 36 x 8+2P RAID 6 arrays
- Capacity: raw = 720 TB, usable = 524 TB[3]
- Performance
  - Streaming rate: write < 3.2 GB/s[4], read < 4.0 GB/s[4]
  - IOP rate (random 4K transactions): write < 3600 IOP/s[5], read < 6000 IOP/s[5]

FOOTNOTES:

1.The x3650 M3 can be replaced with an x3550 M3 if a single dual port port SAS HBA in place of 2 single port SAS HBAs.
2.An IB QDR HCA is essential to this solution in order to harvest the full BW potential of the DCS3700s.
3.The DCS3700 provides a capacity of 14.55 TB per RAID 6 array for the file system to use.
4.The stated streaming rates are least upper bounds (LUB); these rates are based on GPFS/DCS3700 benchmarks using 60 x 7200 RPM Near Line SAS disks. Extrapolating from other tests, greater LUB rates may be expected (e.g., write < 1.7 GB/s and read < 2.4 GB/s using at least 80 of these disks).
5.These rates are extrapolated from actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk. These tests assume completely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly.

COMMENT: This building block is similar to #1a except that by using IB QDR for the LAN, it can double the capacity and performance of the building block while reducing the number of NSD servers. However, when scaling out, building block increments are quite large.

Solution #1b: Physical View



COMPONENTS

4 x NSD servers (x3650 M3) each with the following components:
- 2 x quad core westmere sockets, 6 x DIMMs (2 GB or 4 GB per DIMM)
- 1 x GbE, 2 x TbE or 1 x IB QDR, 2 x single port SAS (6 Gb/s)

Switches: Provide Ethernet and IB switches as needed.

Comment: This configuration consists of 1 building blocks. Adding additional building blocks will scale performance and capacity linearly.

AGGREGATE STATISTICS

2 x DCS3700 Turbo each with 2 x EXP3560 trays
- 360 x 2 TB near line SAS disks
- 36 x 8+2P RAID 6 arrays
- Capacity: raw = 720 TB, usable = 524 TB[3]
- Performance

  Streaming rate: write < 3.2 GB/s[4], read < 4.0 GB/s[4]

  IOP rate (random 4K transactions): write < 3600 IOP/s[5], read < 6000 IOP/s[5]

FOOTNOTES:

1.The x3650 M3 can be replaced with an x3550 M3 if a single dual port port SAS HBA in place of 2 single port SAS HBAs.
2.An IB QDR HCA is essential to this solution in order to harvest the full BW potential of the DCS3700s.
3.The DCS3700 provides a capacity of 14.55 TB per RAID 6 array for the file system to use.
4.The stated streaming rates are least upper bounds (LUB); these rates are based on GPFS/DCS3700 benchmarks using 60 x 7200 RPM Near Line SAS disks. Extrapolating from other tests, greater LUB rates may be expected (e.g., write < 1.7 GB/s and read < 2.4 GB/s using at least 80 of these disks).
5.These rates are extrapolated from actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk. These tests assume completely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly.

COMMENT: Maintaining good streaming performance requires careful attention being given to balance. Alterations disrupting balance (e.g., inconsistent number of disks or expansion trays per DCS3700) will compromise performance.

Solution #2: Overview

Solution #2 provides several similar building block alternatives for solutions offering a balance between performance and capacity. Performance is optimized by using the fewest number of disks needed to saturate (or least come close to saturating) controller streaming performance. Capacity is optimized by using high capacity disks (e.g., 2 TB/disk). This solution is balanced between the extremes of performance and capacity since high capacity drives are slower (e.g., 7200 RPM vs. 15000 RPM).

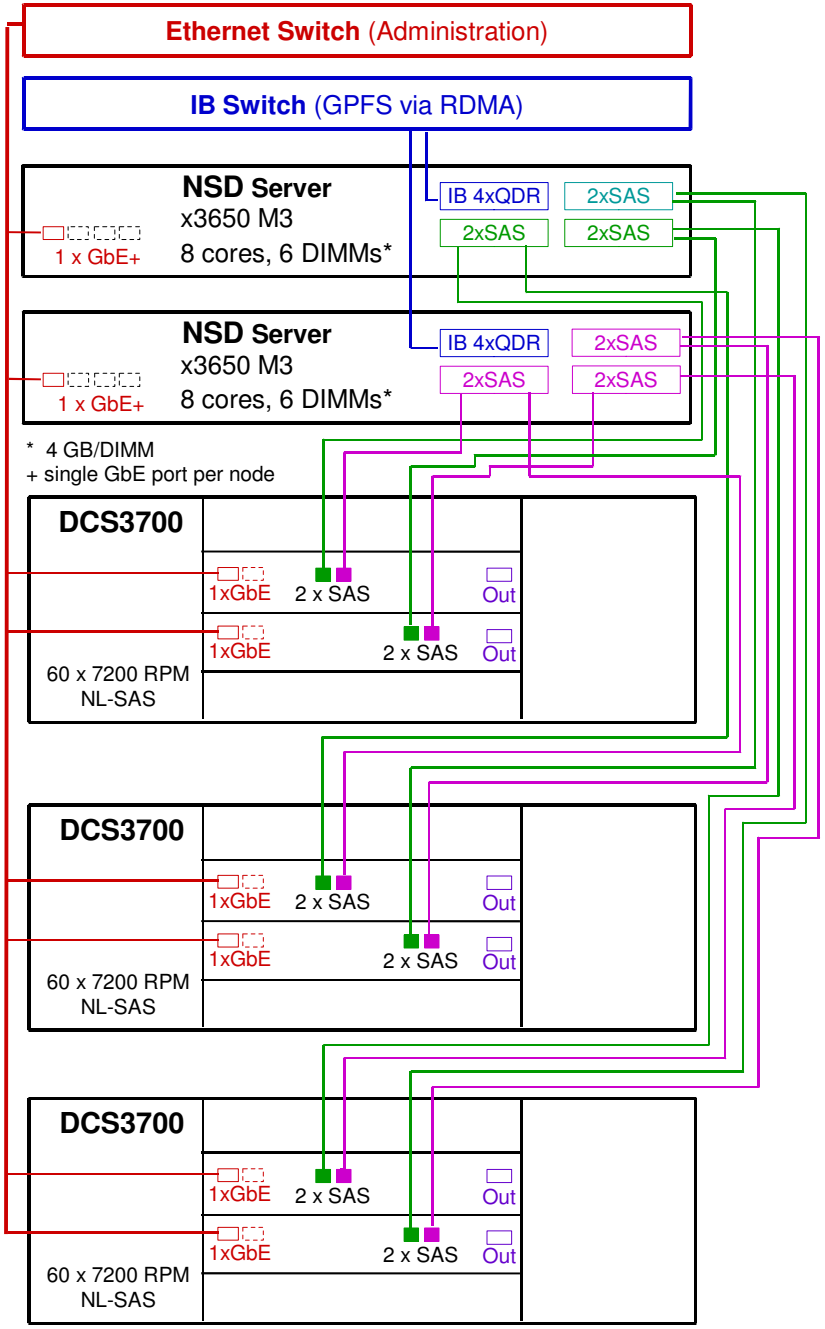
Most of these variations work better with IB QDR than TbE though one solution designed specifically for TbE and another can substitute TbE for IB QDR with only a moderate loss in performance.

A secondary goal of this solution is optimize rack utilization.

Four solutions are illustrated:

2a: Basic Building Block using IB QDR, 2 per rack
2b: Basic Building Block using 2xTbE, 2 per rack
2c: Smaller Building Block using IB QDR or TbE, 4 per rack
    1. Single Rack Solution
    2. Dual Rack Solution
2d: Building Block for a Larger Frame using IB QDR, 3 per rack

Solution #2a: Building Block



Analysis

NSD = *Network Storage Device*
These are the storage servers for GPFS.

NSD Server

- Effective BW per NSD server < 3 GB/s
  - x3650 M3 with 8 cores and 6 DIMMs (4 GB per DIMM)
  - 1 x GbE < 80 MB/s
  - 1 x IB QDR < 3 GB/s (n.b., using RDMA)
  - 2 x dual port 6 Gb/s SAS adapters[1]

DCS3700 Turbo

- 60 x 2 TB near line SAS disks
- 6 x 8+2P RAID 6 arrays
- Capacity: raw = 120 TB, usable = 87.3 TB[2]
- Performance
  - Streaming rate: write < 1.6 GB/s, read < 2.0 GB/s
  - IOP rate (random 4K transactions): write < 1200 IOP/s, read < 2000 IOP/s[4]
  - IOP rate (mdtest): find mdtest results below[5]

Aggregate Building Block Statistics

- 2 x NSD servers
- 3 x DCS3700 Turbo
- 180 x 2 TB near line SAS disks as 18 x 8+2P RAID 6 arrays
- Capacity: raw = 360 TB, usable = 262 TB[2]
- Performance
  - Streaming rate: write < 4.8 GB/s, read < 5.5 GB/s[3]
  - IOP rate (random 4K transactions): write < 3600 IOP/s, read < 6000 IOP/s[4]
  - IOP rate (mdtest): scaling tests remain to be completed[5]

FOOTNOTES:

1.Wire speed for a 1 x 6 Gb/s port < 3 GB/s (n.b., 4 lanes @ 6 Gb/s per lane); with 6 ports per node, the potential SAS aggregate BW is 18 GB/s! The 6 ports are needed for redundancy, not performance.

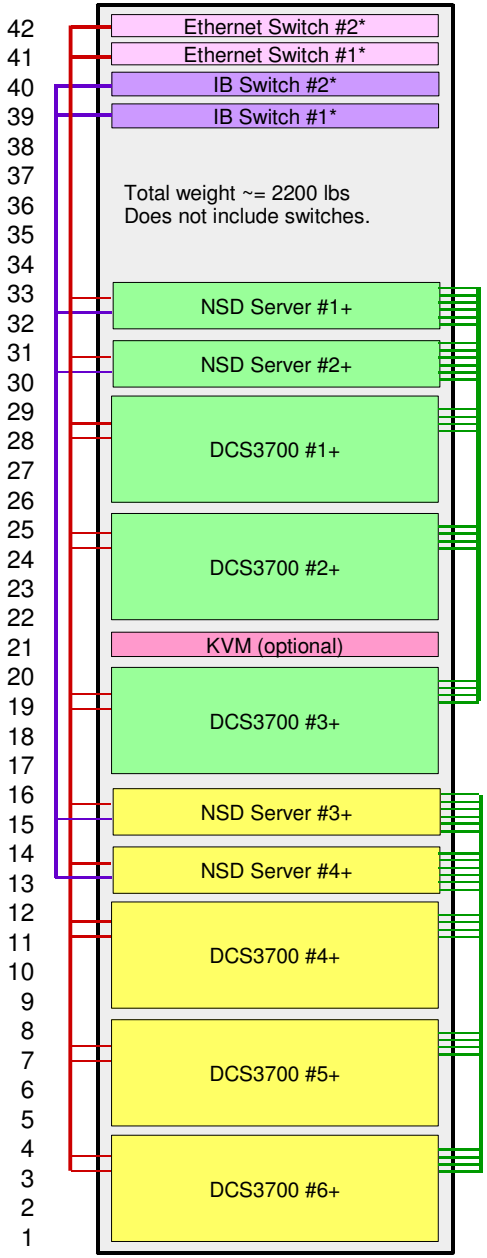
2.The DCS3700 provides a capacity of 14.55 TB per RAID 6 array for the file system to use.

3.Theoretically, this solution should be able to deliver 6 GB/s; however, this requires pushing performance to IB QDR limit. While this may be feasible, performance expectations are being lowered as a precaution.

4.These rates are extrapolated from actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk. These tests assume completely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly.

5.These limited scale tests are included to show the impact that file system optimization can have on small transaction rates; i.e., the random 4K random transaction test is a worst possible case.

Solution #2a: Physical View



COMPONENTS

4 x NSD servers (x3650 M3) each with the following components:
- 2 x quad core westmere sockets, 6 x DIMMs (2 GB or 4 GB per DIMM)
- 1 x GbE, 1 x IB QDR, 2 x dual port SAS (6 Gb/s)

6 x DCS3700, each with the following components
- 60 x 2 TB, 7200 RPM Near Line SAS disks as 6 x 8+P+Q RAID 6 arrays
- Capacity: raw = 120 TB, usable = 87.3 TB
- 4 x SAS host ports @ 6 Gb/s; n.b., 2 SAS host ports per RAID controller

Switches: Provide IB and Ethernet switches as needed.

Comment: This configuration consists of 2 building blocks. Adding additional building blocks will scale performance and capacity linearly.

AGGREGATE STATISTICS

Disks
- 360 x 2 TB, 7200 RPM Near Line SAS disks
- 36 x 8+P+Q RAID 6 arrays, 1 LUN per array

Capacity
- raw = 720, usable = 522 TB[1]

Performance
- streaming rate : write < 9.6 GB/s, read < 11 GB/s[2]
- IOP rate (random, 4K transactions): write < 7200 IOP/s, read < 12,000 IOP/s[3]
- IOP rate (mdtest): scaling tests remain to be completed[4]

COMMENT: Maintaining good streaming performance requires careful attention being given to balance. Alterations disrupting balance (e.g., replacing controllers with expansion trays, or indiscriminately adding expansion trays) will compromise performance.

FOOTNOTES:

1.Usable capacity is defined as the storage capacity delivered by the controller to the file system. Additionally, file system metadata requires a typically small fraction of the usable capacity. For the GPFS file system, this is *typically* < 1.5% (~= 8 TB in this case). With a very large number (e.g., billions) of very small files (e.g., 4 KB), the metadata capacity *may* be much larger (e.g., > 10%). Metadata overhead in this case is application environment specific and difficult to project.

2.Theoretically, this solution should be able to deliver 12 GB/s; however, this requires pushing performance to IB QDR limit. While this may be feasible, performance expectations are being lowered as a precaution.

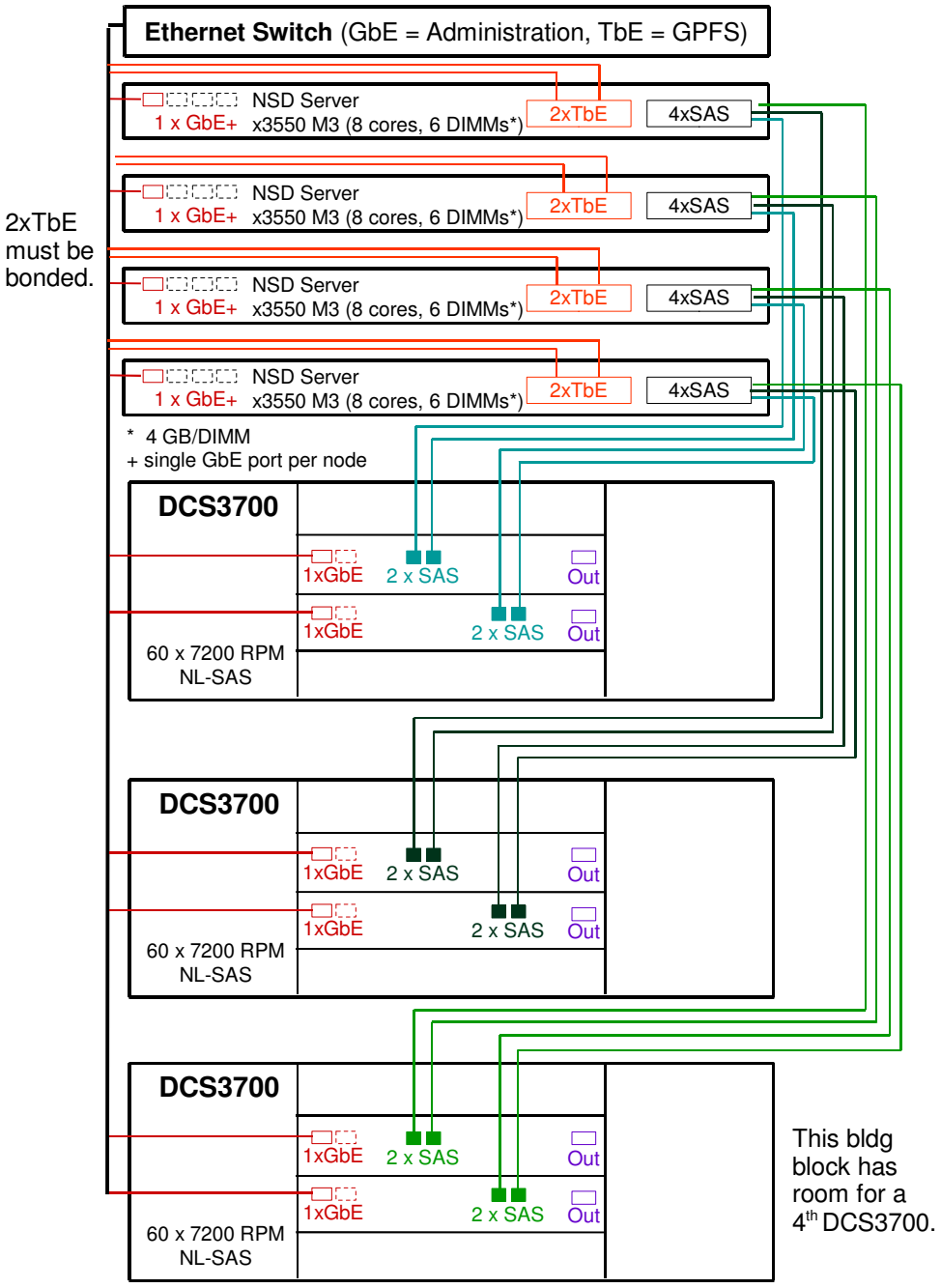
3.These numbers are based purely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly. (n.b., These rates are based on actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk.)

4.Limited scale mdtest results are included on the previous page to show the impact that file system optimization can have on small transaction rates; i.e., the random 4K random transaction test is a worst possible case.

* These switches are included in this diagram for completeness. If the customer has adequate switch ports, then these switches may not be needed.
+ Due to SAS cable lengths (3M is recommended) it is necessary to place the NSD servers in the same rack as the controllers.

LAN: 2xTbE
Since 2xTbE is only ½ as fast as IB QDR, twice as many NSD servers are needed.

Solution #2b: Building Block



Analysis

NSD = *Network Storage Device*
These are the storage servers for GPFS.

NSD Server

- Effective BW per NSD server < 1.4 GB/s
  - x3550 M3 with 8 cores and 6 DIMMs (4 GB per DIMM)
  - 1 x GbE < 80 MB/s
  - 1 x dual port TbE < 1.4 GB/s
  - 1 x quad port 6 Gb/s SAS adapter[1]

DCS3700 Turbo

- 60 x 2 TB near line SAS disks
- 6 x 8+2P RAID 6 arrays
- Capacity: raw = 120 TB, usable = 87.3 TB[2]
- Performance
  - Streaming rate: write < 1.6 GB/s, read < 2.0 GB/s
  - IOP rate (random 4K transactions): write < 1200 IOP/s, read < 2000 IOP/s[4]
  - IOP rate (mdtest): find mdtest results below[5]

Aggregate Building Block Statistics

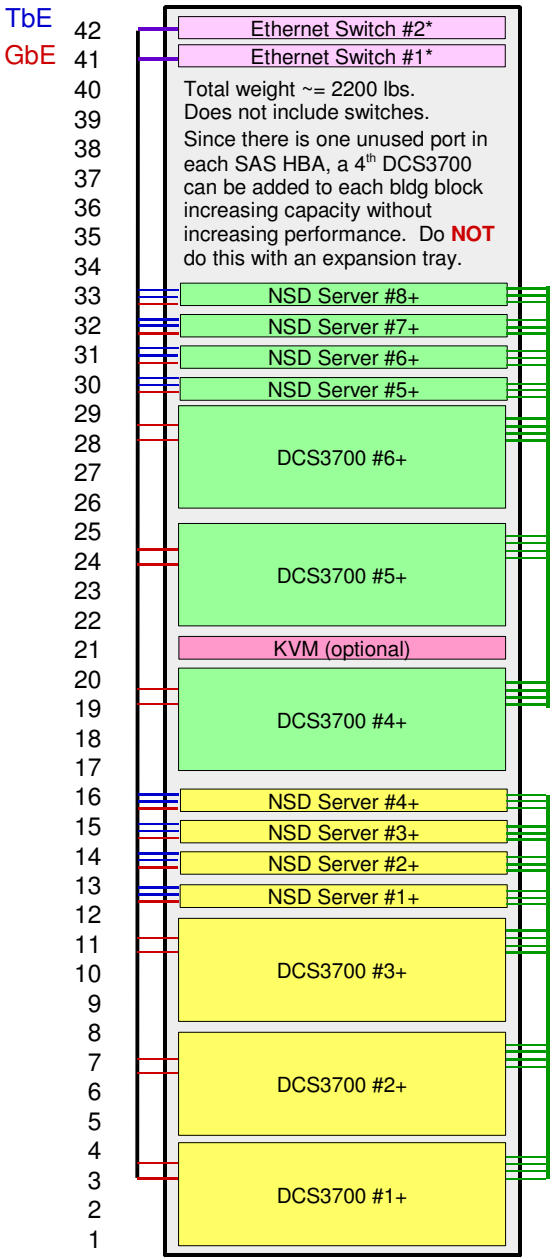
- 4 x NSD servers
- 3 x DCS3700 Turbo
- 180 x 2 TB near line SAS disks as 18 x 8+2P RAID 6 arrays
- Capacity: raw = 360 TB, usable = 262 TB[2]
- Performance
  - Streaming rate: write < 4.8 GB/s, read < 5.5 GB/s[3]
  - IOP rate (random 4K transactions): write < 2400 IOP/s, read < 4000 IOP/s[4]
  - IOP rate (mdtest): scaling tests remain to be completed[5]

FOOTNOTES:

1. Wire speed for a 1 x 6 Gb/s port < 3 GB/s (n.b., 4 lanes @ 6 Gb/s per lane); by using 3 out of 4 ports, the potential SAS aggregate BW is 9 GB/s! The 3 ports in use are needed for redundancy, not performance.
2. The DCS3700 provides a capacity of 14.55 TB per RAID 6 array for the file system to use.
3. Theoretically, this solution should be able to deliver 6 GB/s; however, this is slightly constrained by the 2xTbE.
4. These rates are extrapolated from actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk. These tests assume completely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly.
5. These limited scale tests are included to show the impact that file system optimization can have on small transaction rates; i.e., the random 4K random transaction test is a worst possible case.

LAN: 2xTbE
Since 2xTbE is only ½ as fast as IB QDR, twice as many NSD servers are needed.

Solution #2b: Physical View



COMPONENTS

8 x NSD servers (x3550 M3) each with the following components:
- 2 x quad core westmere sockets, 6 x DIMMs (2 GB or 4 GB per DIMM)
- 1 x GbE, 1 x dual port TbE, 1 x quad port SAS (6 Gb/s)

6 x DCS3700, each with the following components
- 60 x 2 TB, 7200 RPM Near Line SAS disks as 6 x 8+P+Q RAID 6 arrays
- Capacity: raw = 120 TB, usable = 87.3 TB
- 4 x SAS host ports @ 6 Gb/s; n.b., 2 SAS host ports per RAID controller

Switches: Provide Ethernet switches as needed.

Comment: This configuration consists of 2 building blocks. Adding additional building blocks will scale performance and capacity linearly.

AGGREGATE STATISTICS

Disks
- 360 x 2 TB, 7200 RPM Near Line SAS disks
- 36 x 8+P+Q RAID 6 arrays, 1 LUN per array

Capacity
- raw = 720, usable = 522 TB[1]

Performance
- streaming rate : write < 9.6 GB/s, read < 11 GB/s[2]
- IOP rate (random, 4K transactions): write < 7200 IOP/s, read < 12,000 IOP/s[3]
- IOP rate (mdtest): scaling tests remain to be completed[4]

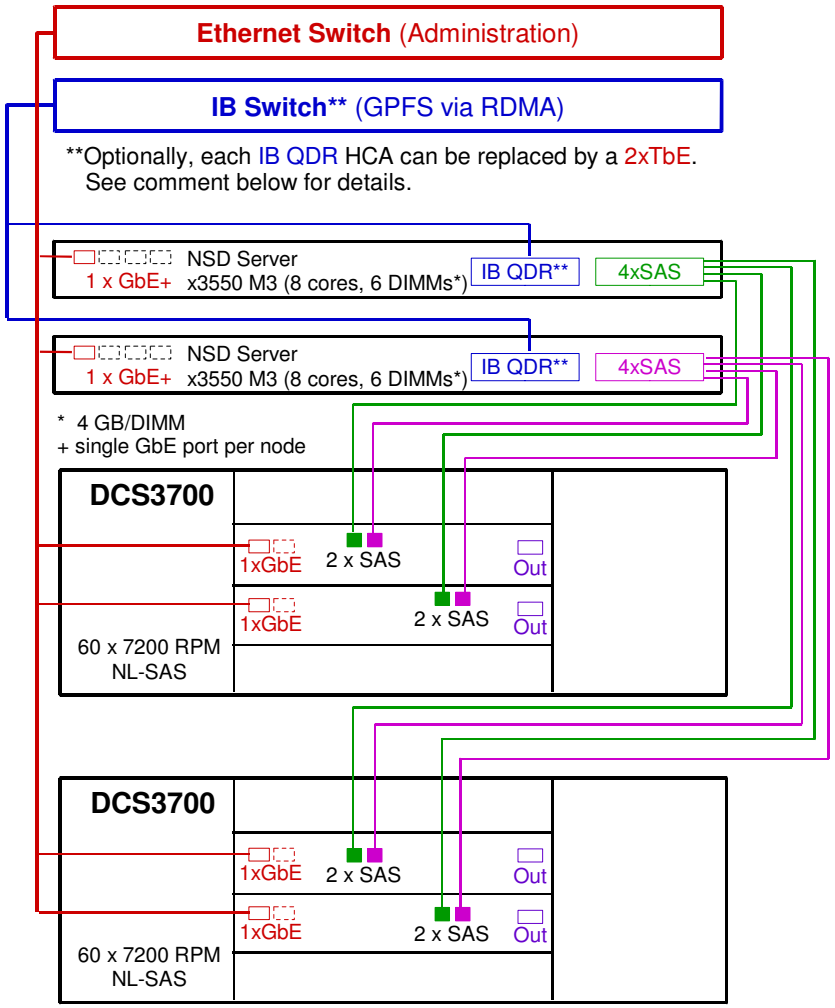
COMMENT: Maintaining good streaming performance requires careful attention being given to balance. Alterations disrupting balance (e.g., replacing controllers with expansion trays, or indiscriminately adding expansion trays) will compromise performance.

FOOTNOTES:
1.Usable capacity is defined as the storage capacity delivered by the controller to the file system. Additionally, file system metadata requires a typically small fraction of the usable capacity. For the GPFS file system, this is *typically* < 1.5% (~= 8 TB in this case). With a very large number (e.g., billions) of very small files (e.g., 4 KB), the metadata capacity *may* be much larger (e.g., > 10%). Metadata overhead in this case is application environment specific and difficult to project.
2.Theoretically, this solution should be able to deliver 12 GB/s; however, this is slightly constrained by the 2xTbE.
3.These numbers are based purely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly. (n.b., These rates are based on actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk.)
4.Limited scale mdtest results are included below to show the impact that file system optimization can have on small transaction rates; i.e., the random 4K random transaction test is a worst possible case.

* These switches are included in this diagram for completeness. If the customer has adequate switch ports, then these switches may not be needed.
\+ Due to SAS cable lengths (3M is recommended) it is necessary to place the NSD servers in the same rack as the controllers.

LAN: IB QDR
IB QDR can be replaced by 2xTbE. Footnotes and comments describe the impact.

Solution #2c: Building Block



COMMENT:
The LAN fabric can meaningfully be replaced by Ethernet using bonded 2xTbE adapters in place of the IB QDR adapter on each node. However, the peak streaming performance will drop to an aggregate of 2.8 GB/s for write or read when using 2xTbE. The impact upon writes are less significant compared with reads.

	IB QDR	2xTbE
Write:	3.2 GB/s	2.8 GB/s
Read:	4.0 GB/s	2.8 GB/s

Analysis

NSD Server

- Effective BW per NSD server < 3 GB/s
  - x3550 M3 with 8 cores and 6 DIMMs (4 GB per DIMM)
  - 1 x GbE < 80 MB/s
  - 1 x IB QDR < 3 GB/s -or- 1 x dual port TbE < 1.4 GB/s
  - 1 x quad port 6 Gb/s SAS adapter[1]

DCS3700 Turbo

- 60 x 2 TB near line SAS disks
- 6 x 8+2P RAID 6 arrays
- Capacity: raw = 120 TB, usable = 87.3 TB[2]
- Performance using IB QDR
  - Streaming rate: write < 1.6 GB/s, read < 2.0 GB/s
  - IOP rate (random 4K transactions): write < 1200 IOP/s, read < 2000 IOP/s[3]
  - IOP rate (mdtest): find mdtest results below[4]

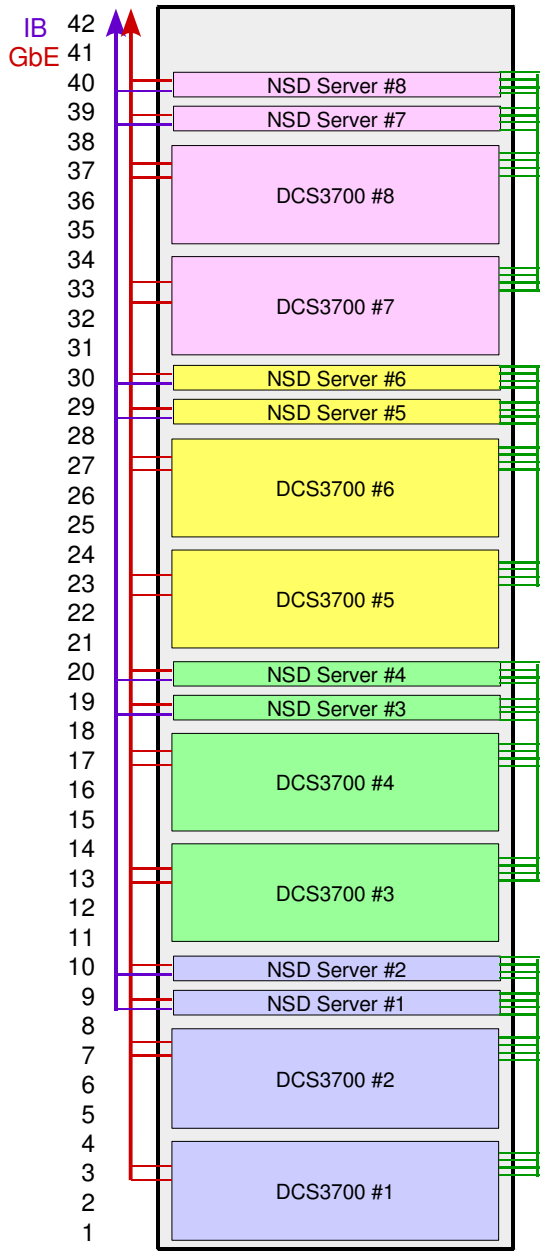
Aggregate Building Block Statistics

- 2 x NSD servers
- 2 x DCS3700 Turbo
- 120 x 2 TB near line SAS disks as 12 x 8+2P RAID 6 arrays
- Capacity: raw = 240 TB, usable = 174.6 TB[2]
- Performance using IB QDR
  - Streaming rate: write < 3.2 GB/s, read < 4.0 GB/s
  - IOP rate (random 4K transactions): write < 2400 IOP/s, read < 4000 IOP/s[3]
  - IOP rate (mdtest): scaling tests remain to be completed[4]

FOOTNOTES:
1.Wire speed for a 1 x 6 Gb/s port < 3 GB/s (n.b., 4 lanes @ 6 Gb/s per lane); with 4 ports, the potential SAS aggregate BW is 12 GB/s! The 4 ports are needed for redundancy, not performance.
2.The DCS3700 provides a capacity of 14.55 TB per RAID 6 array for the file system to use.
3.These rates are extrapolated from actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk. These tests assume completely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly.
4.These limited scale tests are included to show the impact that file system optimization can have on small transaction rates; i.e., the random 4K random transaction test is a worst possible case.

LAN: IB QDR
IB QDR can be replaced by 2xTbE. Footnotes and comments describe the impact.

Solution #2c.1: Physical View



COMPONENTS

8 x NSD servers (x3550 M3) each with the following components:
- 2 x quad core westmere sockets, 6 x DIMMs (2 GB or 4 GB per DIMM)
- 1 x GbE, 1 x IB QDR or 1 x dual port TbE, 1 x quad port SAS (6 Gb/s)

8 x DCS3700, each with the following components
- 60 x 2 TB, 7200 RPM Near Line SAS disks as 6 x 8+P+Q RAID 6 arrays
- Capacity: raw = 120 TB, usable = 87.3 TB
- 4 x SAS host ports @ 6 Gb/s; n.b., 2 SAS host ports per RAID controller

Switches: Provided externally

Comment: This configuration consists of 4 building blocks. Adding additional building blocks will scale performance and capacity linearly.

AGGREGATE STATISTICS

Disks
- 480 x 2 TB, 7200 RPM Near Line SAS disks
- 48 x 8+P+Q RAID 6 arrays, 1 LUN per array

Capacity
- raw = 960, usable = 698.4 TB[1]

Performance using IB QDR[2]
- streaming rate : write < 12.8 GB/s[2], read < 16 GB/s[2]
- IOP rate (random, 4K transactions): write < 9600 IOP/s, read < 16,000 IOP/s[3]
- IOP rate (mdtest): scaling tests remain to be completed[4]

COMMENT: Maintaining good streaming performance requires careful attention being given to balance. Alterations disrupting balance (e.g., replacing controllers with expansion trays, or indiscriminately adding expansion trays) will compromise performance.

FOOTNOTES:

1.Usable capacity is defined as the storage capacity delivered by the controller to the file system. Additionally, file system metadata requires a typically small fraction of the usable capacity. For the GPFS file system, this is *typically* < 1.5% (~= 10.5 TB in this case). With a very large number (e.g., billions) of very small files (e.g., 4 KB), the metadata capacity *may* be much larger (e.g., > 10%). Metadata overhead in this case is application environment specific and difficult to project.

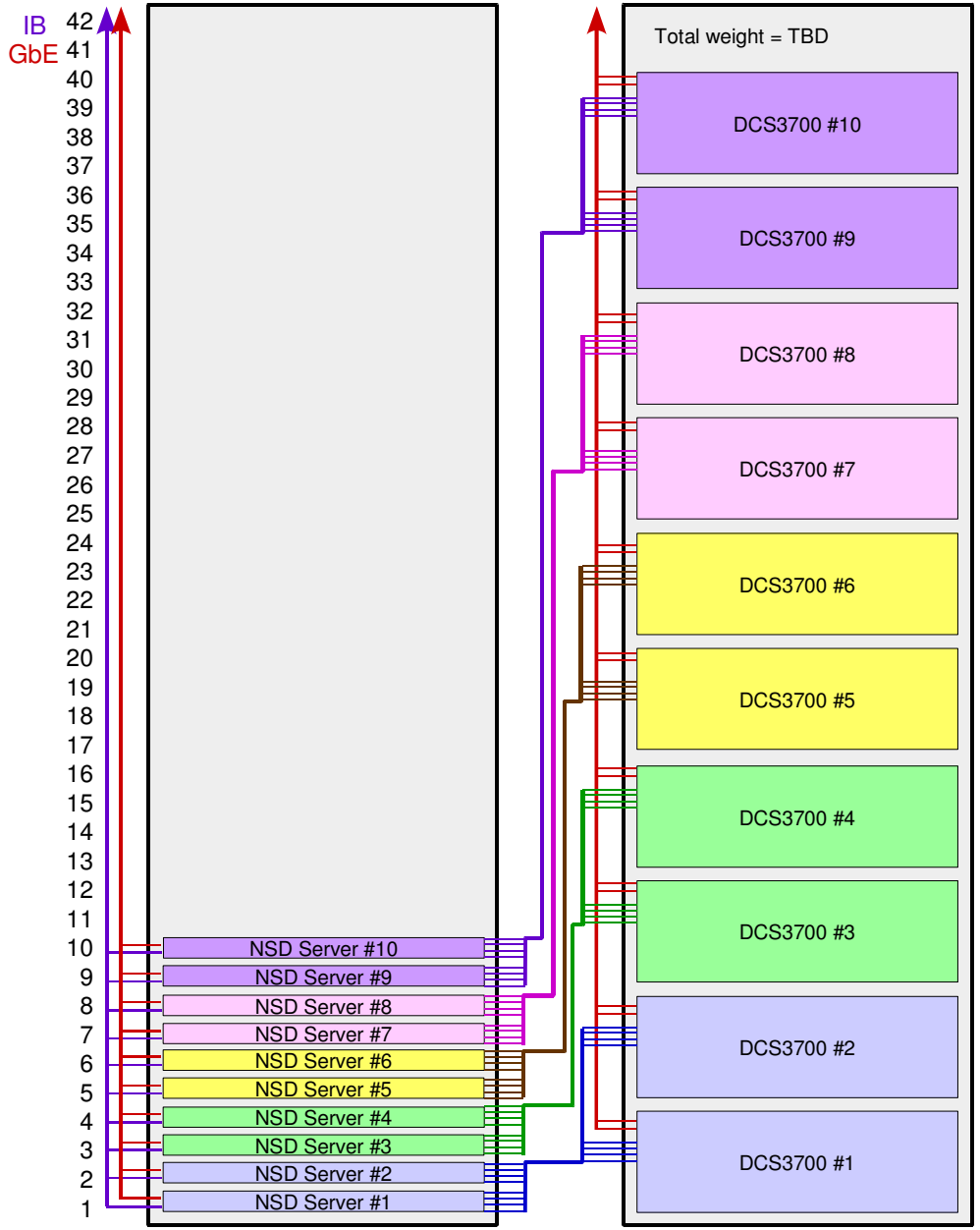
2.If the IB QDR HCAs are replaced with 2xTbE adapters, aggregate streaming are: write < 11 GB/s, read < 11 GB/s. IOP rates should not be impacted by the choice of LAN adapter.

3.These numbers are based purely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly. (n.b., These rates are based on actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk.)

4.Limited scale mdtest results are included below to show the impact that file system optimization can have on small transaction rates; i.e., the random 4K random transaction test is a worst possible case.

LAN: IB QDR
IB QDR can be replaced by 2xTbE. Footnotes and comments describe the impact.

Solution #2c.2: Physical View



COMPONENTS

10 x NSD servers
- x3550 M3
- 2 x quad core westmere sockets, 6 x DIMMs (2 GB or 4 GB per DIMM)
- 1 x GbE, 1 x IB QDR, 1 x quad port SAS (6 Gb/s)

10 x Storage Controllers
- DCS3700
- 60 x 2 TB, 7200 RPM NL-SAS disks as 6 x 8+P+Q RAID 6 arrays
- Capacity: raw = 120 TB, usable = 87.3 TB
- 4 x SAS host ports @ 6 Gb/s; n.b., 2 SAS host ports per controller

Comment: This configuration consists of 5 building blocks. Adding more building blocks will scale performance and capacity linearly.

AGGREGATE STATISTICS

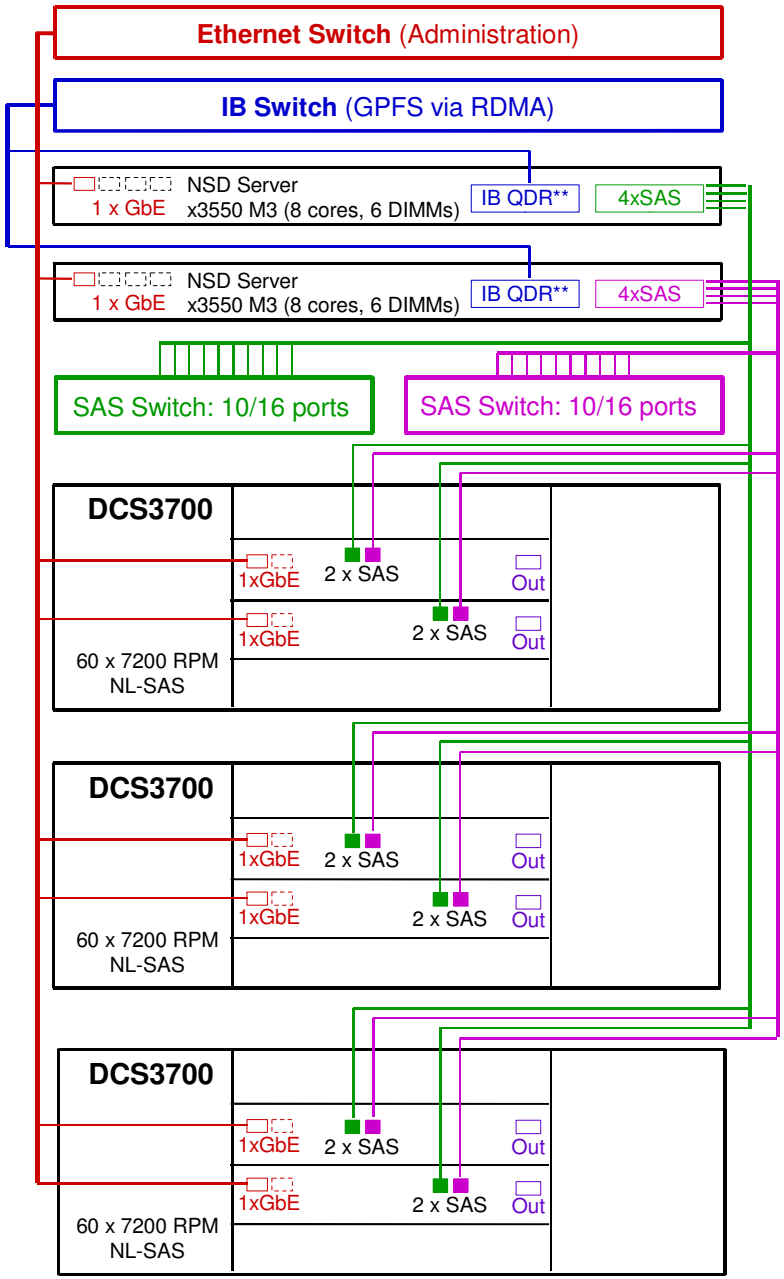
Disks
- 600 x 2 TB, 7200 RPM Near Line SAS disks
- 60 x 8+P+Q RAID 6 arrays, 1 LUN per array

Capacity
- raw = 1200 TB, usable = 873.4 TB

Performance using IB QDR
- streaming rate : write < 16 GB/s, read < 20 GB/s
- IOP rate (random, 4K transactions):
  write < 12,000 IOP/s,
  read < 20,000 IOP/s

COMMENTS:
This solution is nearly identical #2c.1, except that it packages the components in such a way as to support 600 disks in a single frame. This requires placing the NSD servers in a separate frame. This configuration is intended to compete with other products that support 600 disks in a single frame, though it may only be expansion trays requiring the controllers to be contained in the server frame.
CAUTION: This solution requires longer SAS cables (e.g., at least 6M); these cables are nonstandard. Alternatively, the DCS3700s can be configured with a FC daughter card where longer cables are standard.

Solution #2d: Building Block



Analysis

NSD Server

- Effective BW per NSD server < 3 GB/s
  - x3550 M3 with 8 cores and 6 DIMMs (4 GB per DIMM)
  - 1 x GbE < 80 MB/s
  - 1 x IB QDR < 3 GB/s
  - 1 x quad port 6 Gb/s SAS adapter[1]

DCS3700 Turbo

- 60 x 2 TB near line SAS disks as 6 x 8+2P RAID 6 arrays
- Capacity: raw = 120 TB, usable = 87.3 TB[2]
- Performance using IB QDR
  - Streaming rate: write < 1.6 GB/s, read < 2.0 GB/s
  - IOP rate (random 4K transactions): write < 1200 IOP/s, read < 2000 IOP/s[4]
  - IOP rate (mdtest): find mdtest results below[5]

Aggregate Building Block Statistics

- 2 x NSD servers
- 3 x DCS3700 Turbo
- 180 x 2 TB near line SAS disks as 18 x 8+2P RAID 6 arrays
- Capacity: raw = 360 TB, usable = 262 TB[2]
- Performance
  - Streaming rate: write < 4.8 GB/s, read < 5.5 GB/s[3]
  - IOP rate (random 4K transactions): write < 3600 IOP/s, read < 6000 IOP/s[4]
  - IOP rate (mdtest): scaling tests remain to be completed[5]

FOOTNOTES:

1.Wire speed for a 1 x 6 Gb/s port < 3 GB/s (n.b., 4 lanes @ 6 Gb/s per lane); with 4 ports, the potential SAS aggregate BW is 12 GB/s! The 4 ports are needed for redundancy, not performance.

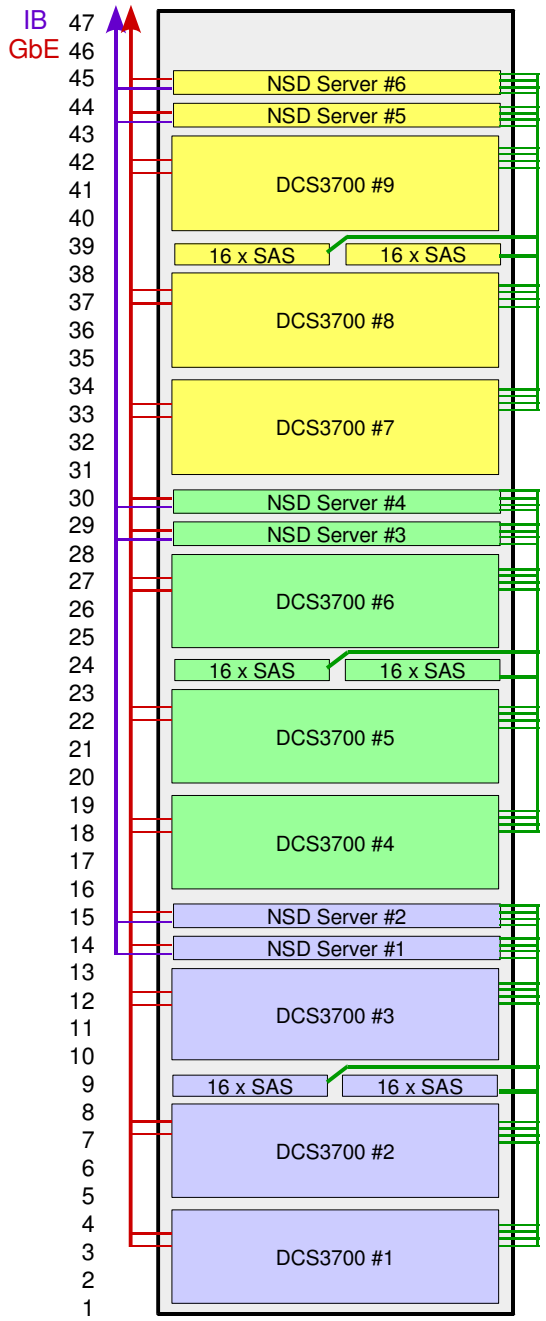
2.The DCS3700 provides a capacity of 14.55 TB per RAID 6 array for the file system to use.

3.Theoretically, this solution should be able to deliver 6 GB/s; however, this requires pushing performance to IB QDR limit. While this may be feasible, performance expectations are being lowered as a precaution.

4.These rates are extrapolated from actual tests using 15000 RPM disk assuming seek rates on 7200 RPM disk < 33% of 15000 RPM disk. These tests assume completely random 4K transactions (n.b., no locality) to raw devices (n.b., no file system). Instrumented code accessing random 4K files will measure a lower IOP rate since they can not measure the necessary metadata transactions. Favorable locality will increase these rates significantly.

5.These limited scale tests are included to show the impact that file system optimization can have on small transaction rates; i.e., the random 4K random transaction test is a worst possible case.

Solution #2d: Physical View



COMPONENTS

6 x NSD servers

- x3550 M3
- 2 x quad core westmere sockets, 6 x DIMMs (2 GB or 4 GB per DIMM)
- 1 x GbE, 1 x IB QDR, 1 x quad port SAS (6 Gb/s)

9 x Storage Controllers

- DCS3700
- 60 x 2 TB, 7200 RPM NL-SAS disks as 6 x 8+P+Q RAID 6 arrays
- Capacity: raw = 120 TB, usable = 87.3 TB
- 4 x SAS host ports @ 6 Gb/s; n.b., 2 SAS host ports per controller

Comment: This configuration consists of 3 building blocks. Adding more building blocks will scale performance and capacity linearly.

AGGREGATE STATISTICS

Disks

- 540 x 2 TB, 7200 RPM Near Line SAS disks
- 54 x 8+P+Q RAID 6 arrays, 1 LUN per array

Capacity

- raw = 1080 TB, usable = 785.7 TB

Performance using IB QDR

- streaming rate : write < 14.4 GB/s, read < 16.5 GB/s[2]
- IOP rate (random, 4K transactions):
  write < 10,800 IOP/s,
  read < 18,000 IOP/s

COMMENTS:

This solution uses a building block similar to building block #2a. It modifies it by using 1u servers so that 3 building blocks can fit in a single frame. Because these servers do not have as many PCI-E slots, SAS switches are added to compensate for a lack of SAS ports on the servers. This configuration is intended to compete with other products that support 600 disks in a single frame; while it only supports 540 disks, the servers fit in the same frame. If a solution supporting 600 disks is desired, see solution #2c.2.

CAUTION

A 47u rack and SAS switches are recommended for this solution. But keep in mind that the 47u rack can not shipped pre-loaded from the factory and that the SAS switches are not available from IBM. If this solution is adopted, the LSI SAS6160 is recommended.

Solution #3: Overview

Optimizing IOP Performance

A Multi-tier solution Using the DS3524 with the DCS3700

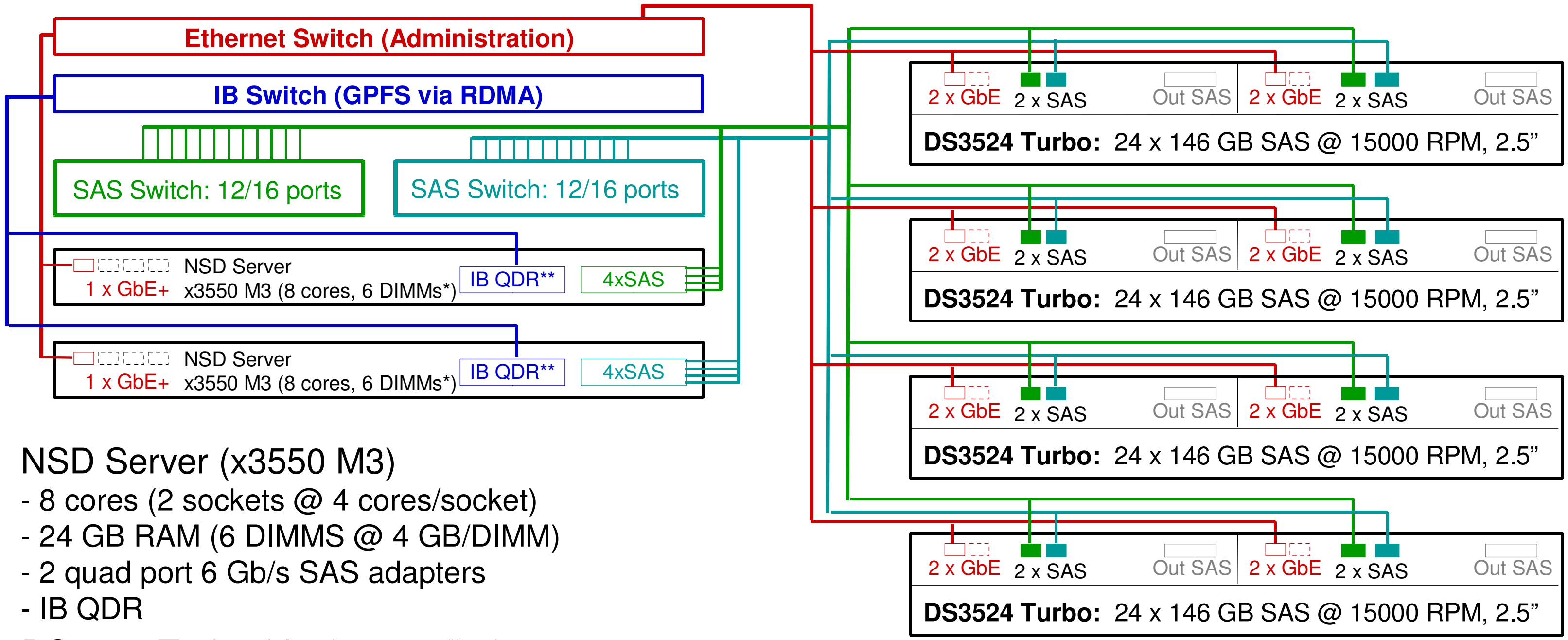
The previous examples in this document are based on the use of 7200 RPM near line SAS disks. These disks can be configured in the DCS3700 to provide either a capacity optimized solution or a balanced capacity/streaming performance optimized solution, but as noted earlier they are not well suited for optimizing IOP performance.

In order to optimize IOP performance, an alternative configuration is to use a multi-tiered solution (under the GPFS ILM feature) deploying the DS3524[1] with 15,000 RPM SAS disks optimizing IOP rates and the DCS3700 optimizing capacity.

The following pages illustrate this solution.

Footnote:
1. The DS3524 uses the same RAID controllers as the DCS3700, but in a different form factor.

Solution #3: Basic Building Block



NSD Server (x3550 M3)
- 8 cores (2 sockets @ 4 cores/socket)
- 24 GB RAM (6 DIMMS @ 4 GB/DIMM)
- 2 quad port 6 Gb/s SAS adapters
- IB QDR

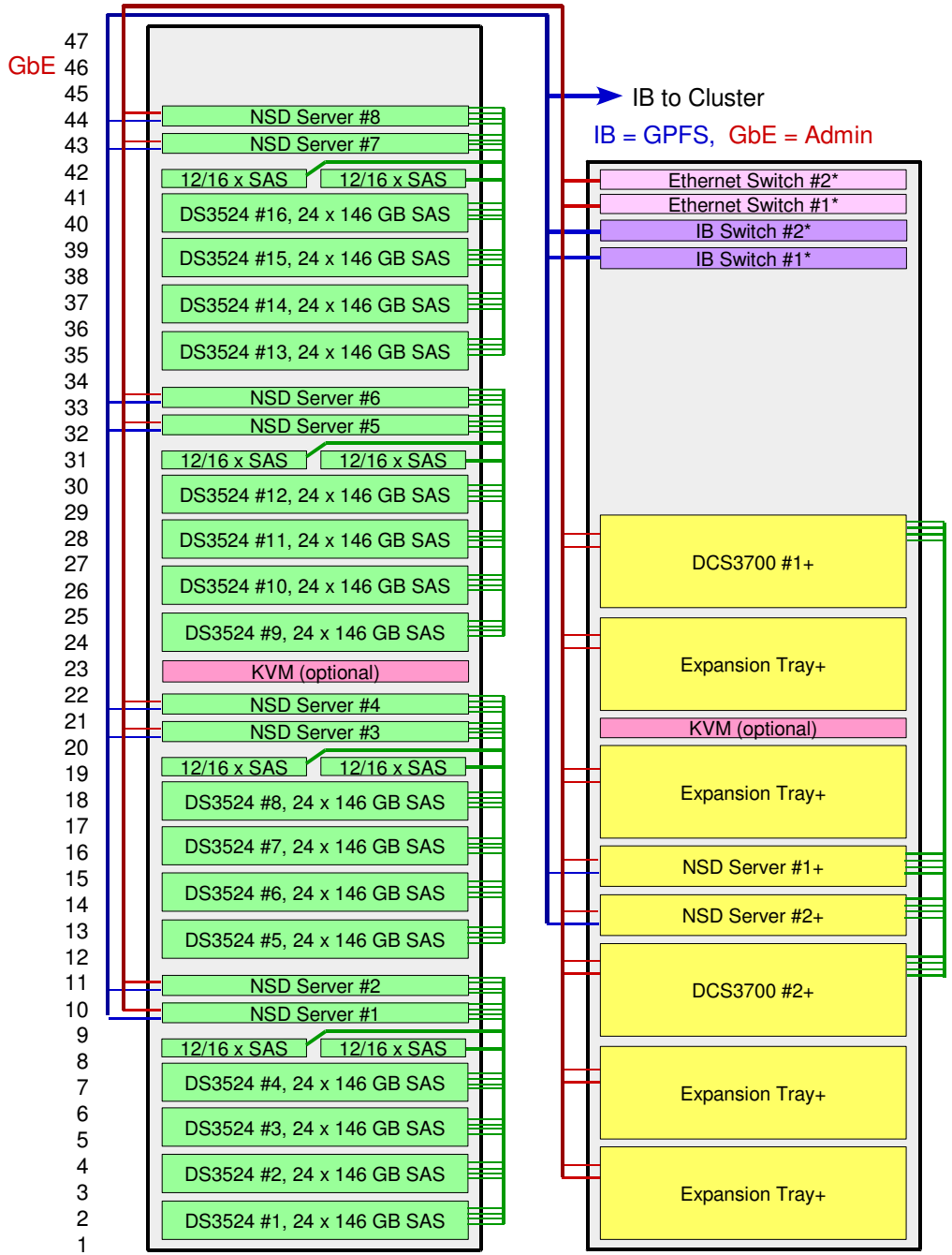
DS3524 Turbo (dual controller)
- 2 SAS ports per controller

Disk per DS3524
- 24 x 146 GB SAS disks @ 15000 RPM
- 6 x 2+2 RAID 10 Arrays
- Capacity: raw ~= 3.5 TB, usable < 1.7 TB

Expected Disk Performance per DS3524
- Streaming write rate[1] < 500 MB/s
- Streaming read rate[1] < 800 MB/s
- IOP write rate:[2] 3000 to 4500 IOP/s
- IOP read rate:[2] 4500 to 10,000 IOP/s

FOOTNOTES: Data rates are based on theoretical calculations for a GPFS file system spanning 24 disks in a single DS3524 configured as described using -j scatter. Validation testing is recommended.
1. Assumes sequential access pattern measured by well written instrumented code.
2. Assumes 4K "to media" transactions measured by the controller. The lower bound assumes random 4K transactions while the upper bound assumes good locality. These rates include both GPFS data and metadata transactions. Instrumented code not measuring metadata transactions will measure lower IOP rates.

Multi-tier Solution Using Building Blocks #1b and #3



Tier #1 – IOP Optimized Storage

- 4 Building Blocks
- Aggregate Statistics

  Capacity: raw = 56 TB, usable < 28 TB

  Streaming

  write < 8 GB/s

  read < 13 GB/s

  IOP rate

  write: 48,000 to 72,000 IOP/s,

  read < 72,000 to 160,000 IOP/s

Tier #2 – Capacity Optimized Storage

- 1 Building Block
- Aggregate Statistics

  Capacity: raw = 720 TB, usable < 524 TB

  Streaming

  write < 3.2 GB/s

  read < 4 GB/s

COMMENTS:

The general idea behind this solution is to provide a tier of storage supporting high transaction rates combined with a second tier of cost effective storage. The GPFS file system provides a “policy engine” that manages these 2 tiers of storage.

A 47u rack is recommended for Tier #1 as it can hold 4 building blocks. But if this frame is infeasible, a 42u frame easily be used instead holding 3 building blocks. This solution also requires SAS switches, but these are not available from IBM. If this solution is adopted, the LSI SAS6160 is recommended.

Miscellaneous Observations
Component Availability and Selection

1. DCS3700 Rack Support
   a. 4u, 19” rack mount enclosure
   b. There are several rack height options:
      - 36u: standard rack size which is generally shipped pre-loaded
      - 42u: common choice[1] for most system x clusters
        can be loaded on-site or shipped pre-loaded under a services contract
      - 47u: must be loaded on-site

2. Host connections
   a. DCS3700 supports both SAS and FC host connections
      - SAS host connections are standard and preferred where possible. However, multi-port SAS host bus adapters (HBAs) may be difficult to procure.
      - FC8 connections may optionally used if necessary. Earlier slides provide details.
   b. SAS switches are generally not needed nor recommended. However, some solutions requiring more ports than provided by the x3650 M3 recommend a SAS switch.[2] These switches must be purchased from non-IBM vendors.

3. LAN Server Connections
   In order to reduce the number of storage servers, high speed LAN interfaces are commonly used for them. Today, GPFS clusters commonly use Infiniband (IB) or 10 Gbit/s Ethernet (TbE). Aggregated, multi-port TbE adapters (e.g., 2xTbE) are an emerging technology being used to improve the effective data rate of servers. While this document encourages their use, it should be noted that best practices vary considerably between vendors for properly configuring them. Specific cases for 2 configurations are included as examples toward the end of this document.

Footnotes:
1. Most examples are shown using 42u racks because it is by far the most common choice in a system x cluster.
2. Alternatively, a server with more ports may be used (e.g., x3850 X5; since it is 4U, the solution is not as dense).

Using the DCS3700

The following pages provide a potpourri of usage notes and guidelines for using the DCS3700 with GPFS. These include benchmark configurations, DCS3700 configuration guidelines for these configurations, detailed GPFS installations notes for one of the benchmark configurations, a sample* of benchmark results, etc. These pages are intended to be used as an example.

Footnote:
* Much more thorough benchmark results can be found in the spread sheet associated with this slide set.

Using the DCS3700
Benchmark Configurations

Benchmarking was done in 2 phases.

Phase 1*
- Cluster of 6 GPFS Nodes
  x3650 M3
- LAN: IB DDR2 for GPFS, GbE for admin
- GPFS 3.4.0.2
- RHEL 5.6 (2.6.18-164, x86_64)
- RDAC MPP: 09.03.0C00.0450
- SAS Driver: 01.101.00.00
- SAS FW: 3.00.00

- 1 x DCS3700
- 60 x 7200 RPM Disks
- DCS3700 FW: 07.77.13.00
- SMclient: 90.77.G0.10

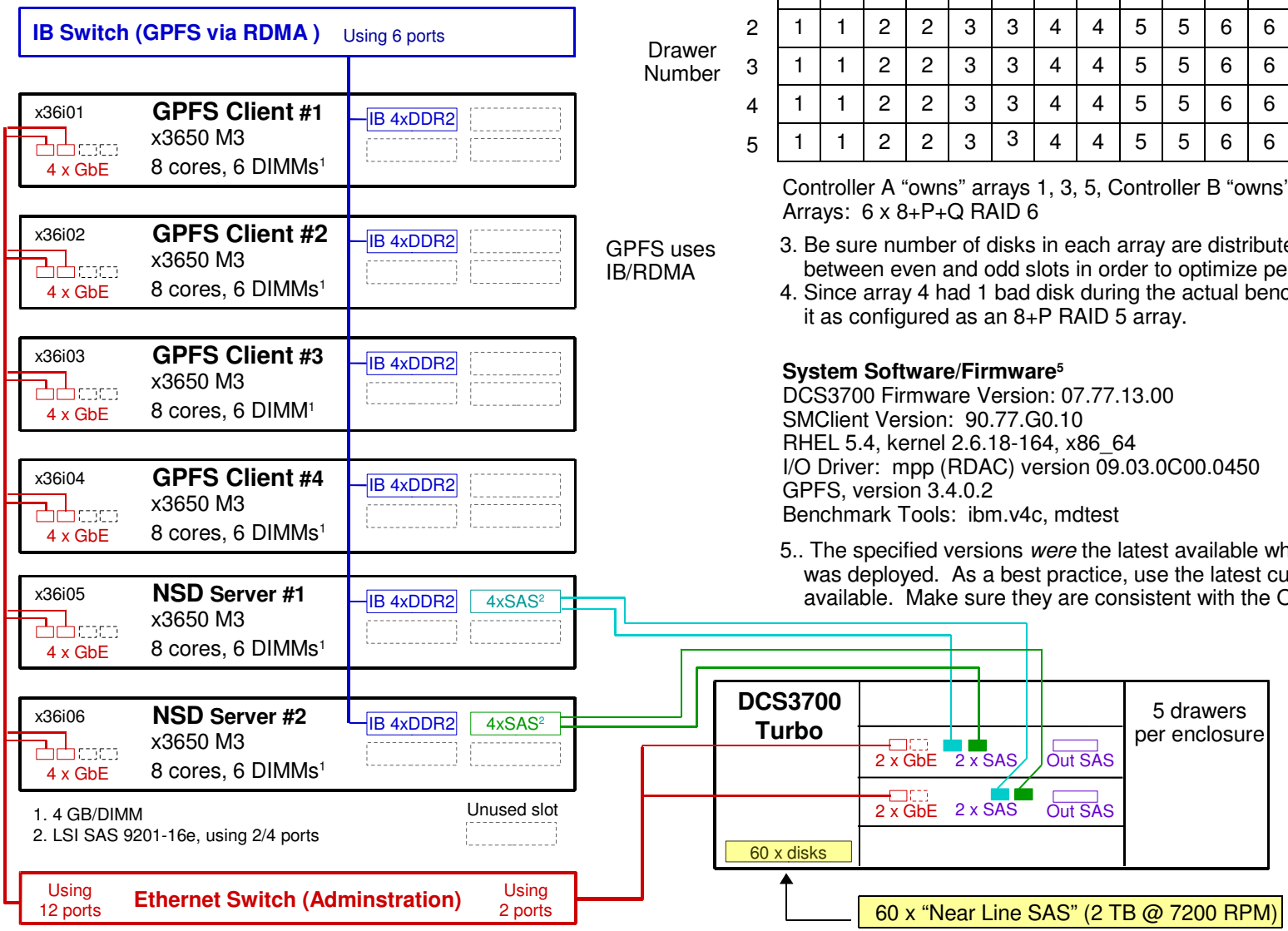
Phase 2*
- Cluster of 6 GPFS Nodes
  x3650 M3
- LAN: IB DDR2 for GPFS, GbE for admin
- GPFS 3.4.0.6
- RHEL 5.6 (2.6.18-164, x86_64)
- RDAC MPP: 09.03.0C00.0450
- SAS Driver: 10.00.00.00
- SAS FW: 10.00.02

- 1 x DCS3700 with 1 x Expansion Tray
- 2 disk options
  120 x 7200 RPM
  120 x 15000 RPM
- DCS3700 FW: 07.80.55.00
- SMclient: 10.80.G5.48

Footnote:
* A manufacturing prototype of the DCS3700 was used in Phase 1, while a customer version of the DCS3700 was used in Phase 2.

Using the DCS3700
Phase 1 - Hardware Configuration

Performance figures quoted in this document are based on benchmarks using the following system:



Disk → Array Mapping[3]

Drawer Number	Disk Number											
	1	2	3	4	5	6	7	8	9	10	11	12
1	1	1	2	2	3	3	4	4	5	5	6	6
2	1	1	2	2	3	3	4	4	5	5	6	6
3	1	1	2	2	3	3	4	4	5	5	6	6
4	1	1	2	2	3	3	4	4	5	5	6	6
5	1	1	2	2	3	3	4	4	5	5	6	6

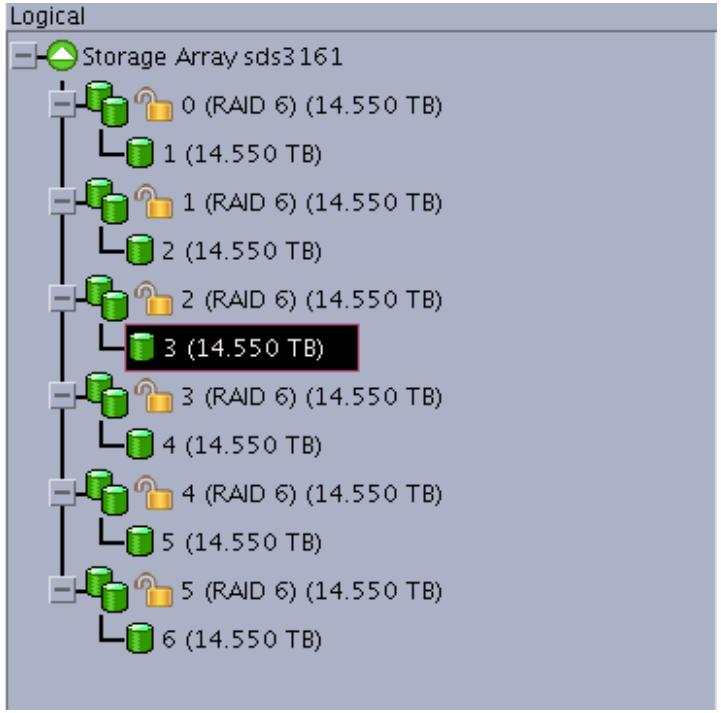
Controller A “owns” arrays 1, 3, 5, Controller B “owns” arrays 2, 4, 6
Arrays: 6 x 8+P+Q RAID 6

3. Be sure number of disks in each array are distributed uniformly between even and odd slots in order to optimize performance.
4. Since array 4 had 1 bad disk during the actual benchmark runs, it as configured as an 8+P RAID 5 array.

System Software/Firmware[5]
DCS3700 Firmware Version: 07.77.13.00
SMClient Version: 90.77.G0.10
RHEL 5.4, kernel 2.6.18-164, x86_64
I/O Driver: mpp (RDAC) version 09.03.0C00.0450
GPFS, version 3.4.0.2
Benchmark Tools: ibm.v4c, mdtest

5.. The specified versions *were* the latest available when the system was deployed. As a best practice, use the latest current versions available. Make sure they are consistent with the OS version.

Using the DCS3700
Phase 1 - Hardware Configuration



This system is configured with are 60 disks divided into 6 arrays @ 8+P+Q RAID 6.

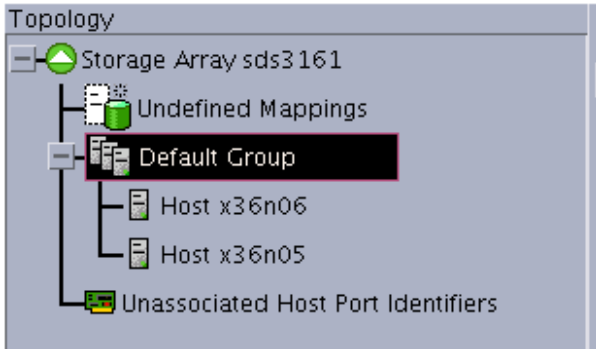
Each array contains a single volume (i.e., LUN).

LUN to SCSI device mapping:*

LUN 1 → `/dev/sdc`
LUN 2 → `/dev/sdd`
LUN 3 → `/dev/sde`
LUN 4 → `/dev/sdf`
LUN 5 → `/dev/sdg`
LUN 6 → `/dev/sdh`

* While there are no guarantees, generally LUN discovery by the host (i.e., NSD server) leads to consistent SCSI device mapping between hosts, however, this is not required by GPFS. For example, if 1 of the hosts has 2 internal disks and while the other only has 1, the mapping will not be consistent. In that case, use the SCSI device name of the primary NSD server in the mmcrnsd command.

All volumes are contained in a single storage partition mapped to the default group.



Defined Mappings

Volume Name	Accessible By	LUN	Volume Capacity	Type
1	Default Group	0	14.550 TB	Standard
2	Default Group	1	14.550 TB	Standard
3	Default Group	2	14.550 TB	Standard
4	Default Group	3	14.550 TB	Standard
5	Default Group	4	14.550 TB	Standard
6	Default Group	5	14.550 TB	Standard
Access	Default Group	7		Access

Using the DCS3700
Phase 1 - Hardware Configuration

The following is list of DCS3700 tuning parameters used for this configuration.

1. Host type = Linux
2. All volumes are contained in a single storage partition mapped to the default group
3. Cache block size = 32 KB
4. If the RAID architecture is 8+P+Q RAID 6, let the segment size = GPFS blocksize / 8
   Here are some common examples
   - if blocksize = 4M, then segment size = 512K
   - if blocksize = 1M, then segment size = 128K
   - if blocksize = 256K, then segment size = 32K

   All of the parameters on this page can be configured using either the SMclient GUI or script. See the sample script provided with this document.
5. Cache settings
   - write cache: do only 1 of the following 2 options[1]
     write caching = disabled
     write caching = enabled and write caching with miroring = enabled
   - read cache = { enabled or disabled }[2]
   - dynamic cache read prefetch = { enabled or disabled }[2]
   - media scan[3]
     background media scan = enabled[3]
     media scan with redundancy check = disabled[3]
   - pre-read redundancy check = { enabled or disabled }[4]

FOOTNOTES:

1. The best performance is achieved using write cache=enabled, and write caching with mirroring = disabled, but this is considered unsafe since if the cache is compromised due to a RAID controller failure, the file system will likely be lost. If you use this option, be prepared to manage the risk and keep faithful backup.
2. GPFS caching renders the impact of controller read caching inert; i.e., these settings neither improve or hurt performance.
3. The 2 media scan options run in the background. The shorter their duration, the greater impact they have on performance. The duration is set at the same time the parameter is enabled or disabled. e.g., If the duration = 30 days, overhead is a few % for streaming workloads. The media scan options do the following tasks:
   background media scan: validate that the disk sectors are readable
   media scan with redundancy check: do a parity check at the same time disk sectors are validated for readability
4. The pre-read redundancy check evaluates parity when sectors are read. This is a foreground process with significant overhead. e.g., Enabling this option decreases the streaming read rate up to 50%.

Using the DCS3700
Phase 1 - Hardware Configuration

Setting the Transfer Size

In Linux, it's necessary to set the following parameters for *each LUN* on *each server* to ensure getting full block transfers.
- max_sectors_kb
- read_ahead_kb

As a best practice, set max_sectors_kb and read_ahead_kb >= GPFS blocksize, keeping in mind that these values can not exceed max_hw_sectors_kb.

A simple way to do this is to execute a script like this on each server (i.e., NSD servers) with LUNs from the DCS3700:

```
for i in sdc sdd sde sdf sdg sdh
   echo 4096 > /sys/block/$i/queue/max_sectors_kb    # default value = 512, SAS max value = 4096
   echo 4096 > /sys/block/$i/queue/read_ahead_kb     # default value = 512, SAS max value = 4096
done
```

Unfortunately, if no action is taken, these parameters are reset to their default values during a reboot (which are generally too small); therefore, it is necessary to re-run this script every time a node with these LUNs is rebooted.

Alternatively, these values can be properly set during a reboot using either the /etc/rc.d/rc.local script or the udev utility.

Here is simple udev example to illustrate the idea. It assumes that only the DCS3700 is using mpp (i.e., RDAC). Carefully test this script* before putting it into production.

```
> cat etc/udev/rules.d/90-tune-dcs3700.rules
KERNEL=="sd*", SUBSYSTEM=="block", RUN+="/etc/udev/scripts/tune_dcs3700 %k"

> cat /etc/udev/scripts/tune_dcs3700
#!/bin/bash
DEV=`/opt/mpp/lsvdev | grep $1 | awk '{print $4}'`
if [ $DEV ]; then
  logger $(date) Setting xfer size for /dev/$1
  echo 4096 > /sys/block/$1/queue/max_sectors_kb
  echo 4096 > /sys/block/$1/queue/read_ahead_kb
else
 logger $(date) $1 is not a dcs3700 device
fi
```

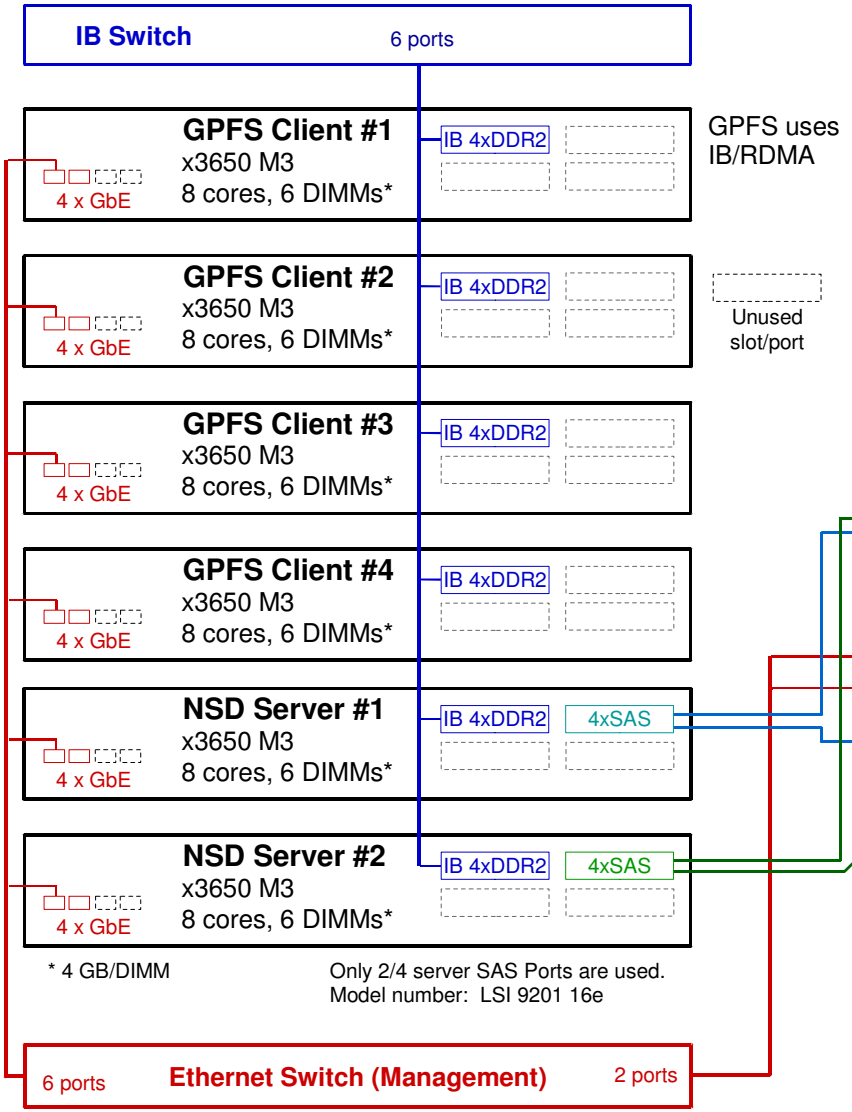
* This udev example was developed under RHEL 5 but does not appear to work under RHEL 6

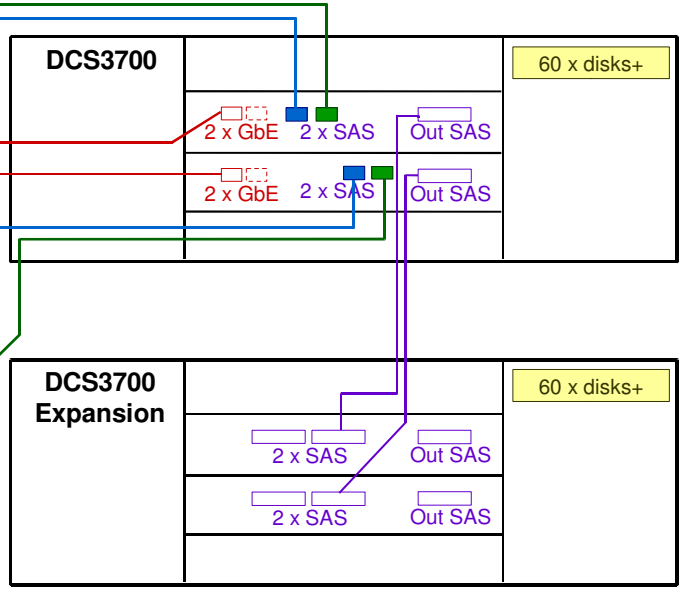
COMMENT: The units for max_sectors_kb and read_ahead_kb are Kilobytes (2^10). Setting these parameters to 4096 allows 4 MB transfers. If it is OK if this value is greater than the GPFS blocksize. For FC drivers, the max value for these parameters is 32768.

See the following link for more information on the udev utility.

http://en.wikipedia.org/wiki/Udev

Using the DCS3700
Phase 2 - Hardware Configuration





COMMENT:
Notes regarding transaction sizes, cache settings and other DCS3700 configuration parameters, and SMclient views are similar to Phase 1 and are not reproduced here for Phase 2.

Legend
GbE = Gigabit Ethernet; TbE = Ten gigabit Ethernet
4xSAS = LSI 9201-16e (quad port SAS adapter)
2xFC8 = dual port, fibre channel HBA
IB 4xDDR2 = single port Infiniband adapter, 4 lanes, double rate

Footnote:
+ The configuration of the trays (i.e., controllers and disks) are modified to meet testing priorities. They can be configured with the following disks:
  - 120 x 7200 RPM, 3.5", 2 TB, Near Line SAS disks
  - 120 x 15000 RPM, 2.5", 300 GB, SAS disks

Using the DCS3700
Phase 2 – Optimal RAID 5 Disk to Array Layout

Tray 0[1] – Controller

Drawers[2]												
	1	A1	B2	A3	B4	A5	B6	A7	B8	A9	B10	A11	B12
	2	B2	A1	B4	A3	B6	A5	B8	A7	B10	A9	B12	A11
	3	A1	B2	A3	B4	A5	B6	A7	B8	A9	B10	A11	B12
	4	B2	A1	B4	A3	B6	A5	B8	A7	B10	A9	B12	A11
	5	A1	B2	A3	B4	A5	B6	A7	B8	A9	B10	A11	B12

Tray 1[1] – Expansion Unit

Drawers[2]												
	1	A13	B14	A15	B16	A17	B18	A19	B20	A21	B22	A23	B24
	2	B14	A13	B16	A15	B18	A17	B20	A19	B22	A21	B24	A23
	3	A13	B14	A15	B16	A17	B18	A19	B20	A21	B22	A23	B24
	4	B14	A13	B16	A15	B18	A17	B20	A19	B22	A21	B24	A23
	5	A13	B14	A15	B16	A17	B18	A19	B20	A21	B22	A23	B24

Arrays: 24 x 4+P RAID 5
Segment size = { 512K | 256K | 128K | 64K | 32K }
Cache page size = 32K
Controller A LUNs: 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23
Controller B LUNs: 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24

Best Practice:
- Use 4+P RAID 5 or 8+P+Q RAID 6 for 15000 RPM SAS disks.
- There should be 1 LUN per array.

Optimizing the Disk to Array Mapping
- Be sure the number of disks in each array are distributed as uniformly *as possible* between even and odd slots.

Footnotes:
1. Each tray either contains a set of RAID controllers or a set of ESMs.
   The tray with the RAID controllers is *the* controller (i.e., DCS3700). The RAID controllers provide the disk semantics.
   The tray with the ESMs is an expansion unit where the ESMs are SAS switches.
   A controller can have 0, 1 or 2 expansion units.
2. Each tray (i.e., DCS3700 and its expansion tray) contains 5 drawers where each drawer has 12 disks.

Using the DCS3700
Phase 2 – Optimal RAID 6 Disk to Array Layout

Tray 0[1] – Controller

Drawers[2]												
	1	A1	A1	B2	B2	A3	A3	B4	B4	A5	A5	B6	B6
	2	A1	A1	B2	B2	A3	A3	B4	B4	A5	A5	B6	B6
	3	A1	A1	B2	B2	A3	A3	B4	B4	A5	A5	B6	B6
	4	A1	A1	B2	B2	A3	A3	B4	B4	A5	A5	B6	B6
	5	A1	A1	B2	B2	A3	A3	B4	B4	A5	A5	B6	B6

Tray 1[1] – Expansion Unit

Drawers[2]												
	1	A7	A7	B8	B8	A9	A9	B10	B10	A11	A11	B12	B12
	2	A7	A7	B8	B8	A9	A9	B10	B10	A11	A11	B12	B12
	3	A7	A7	B8	B8	A9	A9	B10	B10	A11	A11	B12	B12
	4	A7	A7	B8	B8	A9	A9	B10	B10	A11	A11	B12	B12
	5	A7	A7	B8	B8	A9	A9	B10	B10	A11	A11	B12	B12

Arrays: 12 x 8+P+Q RAID 6
Segment size = { 512K | 256K | 128K | 64K | 32K }
Cache page size = 32K
Controller A LUNs: 1, 3, 5, 7, 9, 11
Controller B LUNs: 2, 4, 6, 8, 10, 12

Best Practices:
- Use 8+P+Q RAID 6 for 7200 RPM near line SAS disks.
- There should be one LUN per array.

Optimizing the Disk to Array Mapping
- Be sure the number of disks in each array are distributed as uniformly *as possible* between even and odd slots.

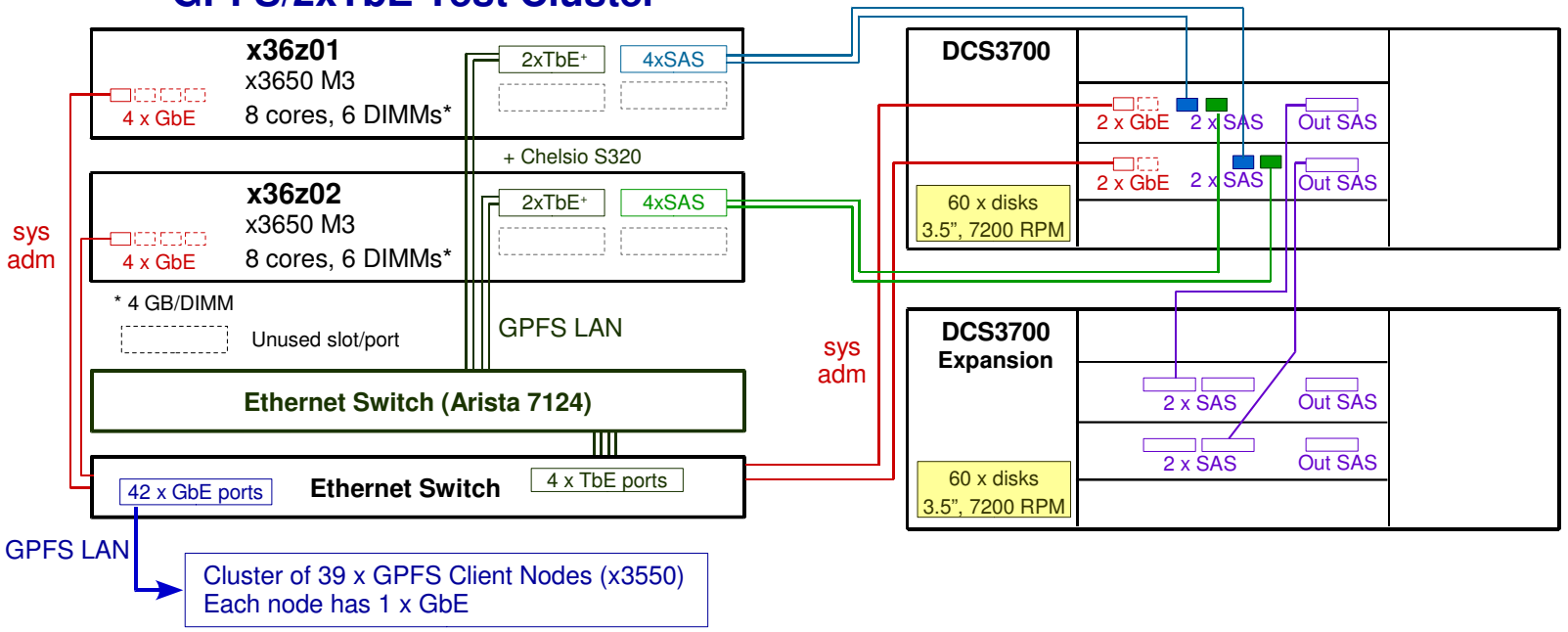
Footnotes:
1. Each tray either contains a set of RAID controllers or a set of ESMs.
   The tray with the RAID controllers is *the* controller (e.g., DCS3700). The RAID controllers provide the disk semantics.
   The tray with the ESMs is an expansion unit where the ESMs are SAS switches.
   A controller can have 0, 1 or 2 expansion units.
2. Each tray (e.g., DCS3700 and its expansion tray) contains 5 drawers where each drawer has 12 disks.

Modified Phase 2 Configuration

2xTbE = Dual Port, 10 GbE

Using the DCS3700
Effectively Using 2xTbE with GPFS

GPFS/2xTbE Test Cluster



WARNING: Managing these issues vary from one environment to another. There is no cookbook answer on how to do this. This page should be seen as an example of **one** way to do it and thereby establishing the validity of the concept. The following pages provide case studies from 2 different customers illustrating this.

Software Versions
RHEL 5.4
- Kernel: 2.6.18-164.el5
GPFS 3.4.0.6

Issue
- Since server BW is much greater than TbE BW, use multi-port TbE adapters to better utilize server BW.
  - rate (TbE)[1] < 725 MB/s
  - rate (x3650 M3)[2] < 10 GB/s
- Must manage "many to few" relationship between GPFS clients and NSD servers[3]
  - Default switch/TCP parameters may be inadequate.

Solution
- Since GPFS requires single IP address, aggregate multiple TbE adapters (e.g., channel aggregation)
- Manage flow control to improve read (2 alternatives):
  - Best practice: tune flow control at level 2 (MAC layer)
  - Alternative solution: tune flow control at level 4 (TCP layer)
- Increase nsdMaxWorkerThreads to improve write:
  - set nsdMaxWorkerThreads > 3 * number of LUNs

Example using illustrated configuration
- Modify TCP layer[4]

```
sysctl -w net.ipv4.tcp_wmem="4096 16384 4194304"
sysctl -w net.ipv4.tcp_rmem="4096 16384 4194304"
```

- Modify nsdMaxWorkerThreads on NSD servers[5]

```
mmchconfig nsdMaxWorkerThreads=144 nsdnodes
```

- Result using streaming benchmark with GPFS
  Write rate < 1800 MB/s, read rate < 2800 MB/s

Read performance is close to theoretical peak. Can write performance still be improved?

Footnotes:
1. This data rate is measured by an application code using GPFS over TbE; its what's leftover after overhead is subtracted.
2. This is theoretical performance using Gen 2 PCI-E adapters; as best practice, never size system for more than 70% of this rate.
3. Tools like iperf do not properly emulate GPFS LAN traffic. Must use nsdperf to demonstrate this networking problem.
4. Since I did not have authorized access to the GbE switch, I merely "tweaked" TCP parameters to control the flow.
5. This value is probably set to high. Lower values *should* work.

Using the DCS3700
Effectively Using 2xTbE with GPFS – Case Study #1

Since there are many ways to solve the problem of effectively using 2xTbE, emails describing 2 independent solutions are included for the reader. The solution[1] on this page used nsdperf for the process of discovery. It was then used for the GPFS solution shown on the previous page.

---

1. nsdperf mode: nwrite[2]

a. The nwrite is implemented in your code as "read by server", i.e. for each Nwrite request received from a client, the server allocates a Worker thread which issues an Read back to the client. This Worker thread is blocked waiting for the Read data to come back. By default, there are only 10 worker threads in the nsdperf code which means the server can only handle 10 nwrite requests at any given time. Given there are 40 clients in your setup and each client can issue multiple nwrite requests, this 10 worker threads significantly limit the system performance.

b. The solution is to launch the server with more Worker threads. In this case, I set "–w 200" to start 200 Worker threads that seems to have solved the problem. If a GPFS cluster has more client nodes, you may need more Worker threads.

2. nsdperf mode: read[2]

a. I found the low Read performance was caused by packet drop in the path from servers to clients that triggered TCP retransmits. These slow start timeout retransmits significantly hurt the performance.

b. The solution is to correctly apply flow control to avoid packet drops. There are two types of flow control that run at different levels of network protocol stack.
- Level 2: MAC level flow control. This requires the NIC adapters on both client and server as well as each switch in the cluster enable the Ethernet flow control. If this flow control works properly, it can effectively avoid packet loss. However, some switches do not support level 2 flow control, or they do not support jumbo frames and flow control both on. In this case, we have to turn to higher level flow controls. By default, Chelsio NIC cards all have this level 2 flow control turned on.
- Level 4: TCP level flow control. If any one equipment between clients and servers does not support level 2 flow control, we have to use TCP flow control to avoid packet drops. This can be done by setting smaller TCP socket buffer size so that a connection will have a smaller TCP window size. Each NIC card and switch has certain buffers. If the TCP socket buffer size is smaller than the total of buffers in the path, no packet will be dropped. However, these buffers in NIC and switches are shared by all connections, so it's hard to guarantee an individual connection can have the buffer it needs. At the meantime, too small of TCP window size will hurt performance too especially when the buffer size is smaller than the product of bandwidth and round-trip latency. The TCP window sizes can be set by the following examples:

```
sysctl -w   net.ipv4.tcp_wmem="4096  16384  4194304"
sysctl -w   net.ipv4.tcp_rmem="4096  32768  4194304"
```

Footnotes:
1. This solution was worked out with help of engineers at Chelsio Communications.
2. The nsdperf test types nwrite and read most faithfully emulate GPFS LAN access patterns.

Using the DCS3700
Effectively Using 2xTbE with GPFS – Case Study #2

Since there are many ways to solve the problem of effectively using 2xTbE, emails describing 2 independent solutions are included for the reader.

The GPFS cluster built for this project is now rock solid and the bench marking scripts complete successfully in 1/3 the stipulated time.

On the 8 GPFS NSD servers we ended up making the following changes from the Network perspective:

Each node has two 10 Gigabit Chelsio adapters and two 1 GB adapters. Two 10 Gb cards (as well as two 1 Gb cards) have been bonded to provide availability and increased bandwidth. Several Bond modes can be used as discussed under:

http://www.cyberciti.biz/howto/question/static/linux-ethernet-bonding-driver-howto.php

This customer was using 802.3ad or 4 mode. Under heavy load, the ports on the network switch would simply flap. That would result in non-functional 10 Gb network (Daemon network) and 1 Gb network (Admin network). The loss of network connectivity would result in GPFS to go down. I was executing a simple ping test between the GPFS NSD servers (on all the eight nodes) with the output being captured to a log file. Consistently when the benchmark script would be kicked off and I/O load was considerably high, the logs clearly indicated that connectivity between nodes on would be lost (both Daemon network as well as Admin Network)

The network team changed the bond mode from 4 (802.3ad) to 0 (balance-rr) Reboot all the nodes for the new bond configuration would take into effect. Multiple runs of all the four bench marking scripts completed successfully in far less time allocated for each run.

From the GPFS perspective the following things were done:

1) Increased the token manager node count from 3 to 8
2) Executed mmfsck - two of them went through several cleaning process. and eventually they showed up as clean.
3) Altered the value of tokenMemLimit mmchconfig tokenMemLimit=1G
4) Altered the value of sharedMemLimit mmchconfig sharedMemLimit=512M
5) Altered the value of pagepool to 512M

The solution seems to hold good and perform very well.

Using the DCS3700
Multi-Pathing Drivers

RDAC MPP vs. Linux Device Mapper (MPIO) Drivers

Following best practices, the benchmark configurations provide 2 levels of failover.

1. HBA Failover (i.e., failover between multiple SAS ports in a server) ← This is the focus of this slide.
2. NSD Server Failover (i.e., failover between multiple servers)

Dual RAID controller architectures generally support 1 of 2 forms of failover semantics:

1. Dual active (e.g., DCS3700)
   Both RAID controllers are active, but each LUN is managed by only one of the RAID controllers at any given time. Under nominal conditions, each RAID controller manages half of the LUNs guaranteeing a uniform distribution of the workload. In a failover situation, the other RAID controller takes over management for the LUNs from the failed RAID controller. The driver coordinates with the controllers to select the proper path to the LUN.
2. Active/Active (e.g., DCS9900)
   Both RAID controllers are active and each LUN is accessible by each RAID controller. Under nominal conditions, the driver determines a unique path from a server to a LUN balancing access across the paths. In a failover situation, the driver will use an active alternative path to access the LUN.

The *best practice* choice of failover driver depends on the controller failover semantics

1. Dual active: RDAC MPP
   See http://www.lsi.com/sep/Pages/rdac/ds4000.aspx
2. Active/Active: Device Mapper Multi-Path (aka, MPIO)
   Device Mapper (DM) generally comes pre-installed with Linux, but MPIO is a separate RPM that does not come pre-installed. MPIO works within the DM rubric.
3. Comments
   a. While MPIO *can* work with dual active failover semantics, RDAC MPP is still the best practice recommendation.
   b. MPIO and RDAC MPP can not both be installed on the same server; you must install one or the other.

Using the DCS3700
Multi-Pathing Drivers

RDAC MPP vs. Linux Device Mapper (MPIO) Drivers

Following best practices, the benchmark configurations provide 2 levels of failover.

1. HBA Failover (i.e., failover between multiple SAS ports in a server) ← This is the focus of this slide.
2. NSD Server Failover (i.e., failover between multiple servers)

Dual RAID controller architectures generally support 1 of 2 forms of failover semantics:

1. Dual Active (e.g., DCS3700)
   Both RAID controllers are active, but each LUN is owned by only one of the RAID controllers. Under nominal conditions, the preferred path from server to LUN is via the owner of the LUN; the LUN can not be accessed via the path through the other controller. In a failover situation (e.g., bad cable or GBIC), ownership of the LUN transfers to the other controller; this path then becomes the preferred and only path. Once the failover condition is resolved, ownership returns to the original owner.
2. Active/Active (e.g., DCS9900)
   Both RAID controllers are active and each LUN is simultaneously accessible by each RAID controller; none-the-less, each LUN has an owner and best performance is achieved by accessing the LUN via the owner. In a failover situation (e.g., bad cable or GBIC), the driver selects the alternate path to the LUN. Once the failover condition is resolved, the driver restores access to the LUN via the original path.
3. Comment
   Properly configured, the workload for Dual Active and Active/Active controllers can be uniformly distributed across both RAID controllers.

The *best practice* choice of failover driver depends on the controller failover semantics

1. Dual active: RDAC MPP
   See http://www.lsi.com/sep/Pages/rdac/ds4000.aspx
2. Active/Active: Device Mapper Multi-Path (aka, MPIO)
   Device Mapper (DM) generally comes pre-installed with Linux, but MPIO is a separate RPM that does not come pre-installed. MPIO works within the DM rubric.
3. Comments
   a. While MPIO *can* work with dual active failover semantics, RDAC MPP is still the best practice recommendation.
   b. MPIO and RDAC MPP can not both be installed on the same server; you must install one or the other.

The example on this page is based on Phase 1 configuration, but applies to Phase 2 configuration.

Using the DCS3700
Multi-Pathing Drivers - How RDAC MPP Works

Assume: There are 2 x NSD servers: x36n05 and x36n06.
There are 6 LUNs (0..5) on the DCS3700.
Since the LUNs are "twin tailed" (i.e., mounted on both servers), all 6 LUNs appear as SCSI devices in /dev on both servers...

```
[root@x36n05 gpfs_install]# ls /dev/sd*
/dev/sdc  /dev/sdd  /dev/sde  /dev/sdf  /dev/sdg  /dev/sdh
[root@x36n06 gpfs_install]# ls /dev/sd*
/dev/sdc  /dev/sdd  /dev/sde  /dev/sdf  /dev/sdg  /dev/sdh
```

The 2 internal SAS disks are not included in the ls listings.

Under RDAC MPP, lsvdev shows the LUN mappings. For example, LUN 0 maps to sdc on x36n05 and x36n06.[1]

```
[root@x36n05 gpfs_install]# /opt/mpp/lsvdev
  Array Name      Lun    sd device
  -------------------------------------
  sds3161         0     -> /dev/sdc
  sds3161         1     -> /dev/sdd
  sds3161         2     -> /dev/sde
  sds3161         3     -> /dev/sdf
  sds3161         4     -> /dev/sdg
  sds3161         5     -> /dev/sdh
```

```
[root@x36n06 gpfs_install]# /opt/mpp/lsvdev
  Array Name      Lun    sd device
  -------------------------------------
  sds3161         0     -> /dev/sdc
  sds3161         1     -> /dev/sdd
  sds3161         2     -> /dev/sde
  sds3161         3     -> /dev/sdf
  sds3161         4     -> /dev/sdg
  sds3161         5     -> /dev/sdh
```

Without using a multi-pathing driver (e.g., RDAC MPP), the following would happen on either node...

```
[root@x36n05 gpfs_install]# ls /dev/sd*
/dev/sdc  /dev/sde  /dev/sdg  /dev/sdi  /dev/sdk  /dev/sdm
/dev/sdd  /dev/sdf  /dev/sdh  /dev/sdj  /dev/sdl  /dev/sdn
```

There are 2 mappings for each LUN while only one of them will be active. Moreover, there will be no failover.

The RDAC MPP driver guarantees that only one mapping to each LUN on each node exists (as shown by the lsvdev command).
It also distributes the mappings uniformly across both SAS ports so that both ports get used equally.
If a SAS connection fails (e.g., bad GBIC or cable), then LUNs presented over that port failover to the other port on that node.
Therefore, if a port fails, the LUN is still accessible from that node.

To configure this, install RDAC MPP on both nodes (x36n05 and x36n06).
Create the LUNs on the DCS3700 under the default group which includes both nodes.
Issue the command mppBusRescan on both nodes.
The lsvdev command will verify that the LUNs are properly configued.

The default group is the set of nodes connected "out of band" (i.e., via the SAS host connections) to the DCS3700. See the SMclient screen dump on an earlier slide. Note that the default group can have more than 2 nodes.

Footnote:
1. The LUN mappings are not guaranteed to be the same on both nodes. There are various means to compensate for this in GPFS.

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

The following pages provide an actual example of installing and configuring GPFS under Linux for the benchmark configuration. The Phase 1 configuration was used for this example, but it can be easily adapted to the Phase 2 configuration.

This example can be used as a hands on guide.

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

green arial font is used for miscellaneous comments and other annotations

COMMENT: This example is based on GPFS 3.4, but the steps are similar for GPFS 3.2 and GPFS 3.3.

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

Outline of Steps to Install and Configure GPFS

1. Establish administrative control and scope
   Enable ssh with passwordless root access to designated nodes.
2. Install the GPFS code
   a. Base version (*e.g.*, 3.4.0.0) It is always necessary to start by installing the base version.
   b. PTF version (*e.g.*, 3.4.0.5)
3. Build portability RPM It is only necessary to build the portability layer under Linux.
4. Configure a GPFS File System
   a. Create cluster
   b. Declare client and server licenses
   c. Change global GPFS parameters and start the GPFS daemon
   d. Create the NSDs
   e. Create and Mount the file system

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

Steps to Install the GPFS Code under Linux

Installing GPFS under AIX and Windows is quite different. See the note below for references.

1. Create an NFS mounted installation directory for extracting the RPMs
   - Alternatively, copy the RPMs to all nodes
   - Example installation directory: /gpfs_install/gpfs_3.4.0.0
2. Copy base version RPMs to the installation directory and extract the RPMs on all nodes in the GPFS cluster.
   - Sample RPM names:
     - gpfs.base-3.4.0-0.x86_64.rpm
     - gpfs.docs-3.4.0-0.noarch.rpm
     - gpfs.gpl-3.4.0-0.noarch.rpm
     - gpfs.msg.en_US-3.4.0-0.noarch.rpm

   gpfs.docs... contains the man pages. They will be installed in /usr/share/man/
3. Download the latest update package. This comes as a tar/gzip file from
   - https://www14.software.ibm.com/webapp/set2/sas/f/gpfs/download/home.html
   - gpfs-3.4.0-5.x86_64.update.tar.gz
     - This file contains a different version of the same RPMs as the base version.
4. Copy this file to the installation directory, then gunzip/untar this file, and extract the RPMs on all nodes in the GPFS cluster.
   - Example installation directory: /gpfs_install/gpfs_3.4.0.5

The steps for doing this are more thoroughly documented in chapter 5 of the *GPFS Concepts, Planning and Installation Guide*
The steps for installing the GPFS code under AIX and Windows are also documented in this guide. It can be found at
http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=%2Fcom.ibm.cluster.gpfs.doc%2Fgpfs_faqs%2Fgpfsclustersfaq.html

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

Steps to Install the GPFS Code under Linux

5. Select a node in the cluster and do the following:
   - cd /usr/lpp/mmfs/src
   - make Autoconfig
   - make World
   - make InstallImages
   - make rpm

     sample rpm name: gpfs.gplbin-2.6.32-71.el6.x86_64-3.4.0-5.x86_64.rpm
6. Copy portability rpm to all nodes and extract.
7. Warnings and Caveates
   - If a cluster has mixed architectures and/or kernel levels, it is necessary build a portability rpm for each instance and copy it to like nodes.
   - Required Linux patches for GPFS can be found at:

     http://www.ibm.com/developerworks/opensource/

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

```
[root@x36n06 gpfs_install]# export PATH=$PATH:/usr/lpp/mmfs/bin

[root@x36n06 gpfs_install]# cat node_spec.lst
x36i01
x36i02
X36i03
x36i04:quorum-manager
x36i05:quorum-manager
x36i06:quorum-manager
```

The remaining nodes must be licensed as clients.

Administrative domain spans all nodes (i.e., the traditional security model).

The manager-quorum nodes must be licensed as servers.

```
[root@x36n06 GPFS_install]#  mmcrcluster -n node_spec.lst
-p x36i01 -s x36i02 -R /usr/bin/scp -r /usr/bin/ssh
Tue May 31 21:56:06 EDT 2011: mmcrcluster: Processing node x36i01
Tue May 31 21:56:06 EDT 2011: mmcrcluster: Processing node x36i02
Tue May 31 21:56:06 EDT 2011: mmcrcluster: Processing node x36i03
Tue May 31 21:56:07 EDT 2011: mmcrcluster: Processing node x36i04
Tue May 31 21:56:07 EDT 2011: mmcrcluster: Processing node x36i05
Tue May 31 21:56:08 EDT 2011: mmcrcluster: Processing node x36i06
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.
```

There are 2 LAN interfaces for each node in this cluster; the IB interface has hostnames x36**i**<int> and the GbE interface has hostnames x36**n**<int>. This GPFS cluster is defined over the IB LAN. Thus all GPFS communication (data and administration) will traverse the IB LAN, while the GbE LAN is used for traditional system administration.

This is an informational message explaining that 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)

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

```
[root@x36n06 GPFS_install]#  mmlscluster
```

"List" the cluster to verify that the cluster is created as intended

```
GPFS cluster information
========================
  GPFS cluster name:         x36i01.pbm.ihost.com
  GPFS cluster id:           9306696057255737353
  GPFS UID domain:           x36i01.pbm.ihost.com
  Remote shell command:      /usr/bin/ssh
  Remote file copy command:  /usr/bin/scp

GPFS cluster configuration servers:
-----------------------------------
  Primary server:    x36i01.pbm.ihost.com
  Secondary server:  x36i02.pbm.ihost.com

 Node  Daemon node name             IP address       Admin node name     Designation
-------------------------------------------------------------------------------------
   1   x36i01.pbm.ihost.com    129.40.6.193   x36i01.pbm.ihost.com
   2   x36i02.pbm.ihost.com    129.40.6.194   x36i02.pbm.ihost.com
   3   x36i03.pbm.ihost.com    129.40.6.195   x36i03.pbm.ihost.com
   4   x36i04.pbm.ihost.com    129.40.6.196   x36i04.pbm.ihost.com      quorum-manager
   5   x36i05.pbm.ihost.com    129.40.6.197   x36i05.pbm.ihost.com      quorum-manager
   6   x36i06.pbm.ihost.com    129.40.6.198   x36i06.pbm.ihost.com      quorum-manager
```

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

```
root@x36n06 GPFS_install]#  cat license_server.lst
x36i04
x36i05
x36i06

[root@x36n06 GPFS_install]# mmchlicense server --accept -N license_server.lst

The following nodes will be designated as possessing GPFS server licenses:
        x36i04.pbm.ihost.com
        x36i05.pbm.ihost.com
        x36i06.pbm.ihost.com
mmchlicense: Command successfully completed
mmchlicense: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.

[root@x36n06 GPFS_install]#  cat license_client.lst
x36i01
x36i02
x36i03

[root@x36n06 GPFS_install]# mmchlicense client --accept -N license_client.lst
        x36i01.pbm.ihost.com
        x36i02.pbm.ihost.com
        x36i03.pbm.ihost.com
mmchlicense: Command successfully completed
mmchlicense: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

mmchlicense parameters
server: server license type
client: client license type
--accept: suppress the license prompt (this implies you accept license terms)
-N: list of nodes for a given license type

It is necessary to explicitly declare both license types.

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

List a summary of the GPFS licenses for this cluster.

```
[root@x36n06 gpfs_install]# mmlslicense

 Summary information
---------------------
Number of nodes defined in the cluster:                         6
Number of nodes with server license designation:                3
Number of nodes with client license designation:                3
Number of nodes still requiring server license designation:     0
Number of nodes still requiring client license designation:     0
```

Here is an example from another cluster showing the utility of the mmlslicense -L command line option.
-L: Displays the license type for each node, using an * to designate nodes with licenses out of compliance.

```
[root@node-01 gpfs_install]# mmlslicense -L
Node name Required license Designated license
--------------------------------------------------------------------
node01.acme.com        server          server
node02.acme.com        server          server
node03.acme.com        server          client *
node04.acme.com        client          client
node05.acme.com        client          node *

Summary information
---------------------
Number of nodes defined in the cluster: 5
Number of nodes with server license designation: 2
Number of nodes with client license designation: 2
Number of nodes still requiring server license designation: 1
Number of nodes still requiring client license designation: 1
```

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

```
[root@x36n06 gpfs_install]# mmchconfig maxMBpS=3000,maxblocksize=4m,pagepool=1G,
autoload=yes,adminMode=allToAll
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.
```

Change selected global parameters.

```
[root@x36n06 gpfs_install]# mmlsconfig
Configuration data for cluster x36i01.pbm.ihost.com:
----------------------------------------------------
clusterName x36i01.pbm.ihost.com
clusterId 9306696057255737353
autoload yes
minReleaseLevel 3.4.0.0
dmapiFileHandleSize 32
maxMBpS 3000
maxblocksize 4m
pagepool 1G
adminMode allToAll

File systems in cluster x36i01.pbm.ihost.com:
---------------------------------------------
(none)
```

List global parameters that have been changed.

The mmchconfig parameters are
maxMBpS: Limit the LAN BW per node. To get peak rate, set it ~= 2X the desired BW; do NOT set it excessively large.
maxblocksize: Maximum file system block size allowed. This parameter can not be easily changed.
pagepool: Size of GPFS cache.
autoload: yes -> start mmfsd when a node is rebooted
adminMode: allToAll -> all nodes allow passwordless root access
client -> only a subset of nodes allow passwordless root access

COMMENT:
mmchconfig parameters can be set differently on different nodes using the -N option. There are many more mmchconfig parameters possible, most of which are undocumented.

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

GPFS can use IB (Infinband) in 2 ways: IPoIB (IP over IB) or RDMA. RDMA generally gives higher performance. IPoIB is the default option; to use it requires no further action after issuing the mmcrcluster command over an IB LAN via IP addresses. But to use RDMA, it must be explicitly enabled for GPFS.

Use the ibstatus command to find the IB device names and active IB ports.

```
[root@x36n01 gpfs_install]# ibstatus
Infiniband device 'mlx4_0' port 1 status:
        default gid:     fe80:0000:0000:0000:0002:c903:0003:f9ed
        base lid:        0x2
        sm lid:          0x1
        state:           4: ACTIVE
        phys state:      5: LinkUp
        rate:            20 Gb/sec (4X DDR)

Infiniband device 'mlx4_0' port 2 status:
        default gid:     fe80:0000:0000:0000:0002:c903:0003:f9ee
        base lid:        0x0
        sm lid:          0x0
        state:           1: DOWN
        phys state:      2: Polling
        rate:            10 Gb/sec (4X)
```

Enable RDMA and inform GPFS which IB device name and port number to use via the mmchconfig command.

```
[root@x36n06 gpfs_install]# mmchconfig verbsRdma=enable,verbsPorts=mlx4_0/1
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
   affected nodes.  This is an asynchronous process.
```

In this configuration, GPFS administration uses the IP LAN via IPoIB while file data transfers between the GPFS clients to NSD servers uses the IB LAN via RDMA/IB. Alternatively, if the GPFS cluster is defined over the GbE LAN (using GbE hostnames in the mmcrcluster command), GPFS can still use the IB LAN for GPFS client to NSD server communication via RDMA/IB enabling it via mmchconfig as shown above. In this way, GPFS administration can be confined to the GbE LAN along with other system administration.

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

```
[root@x36n06 gpfs_install]# mmstartup -a
Wed Jun  1 13:28:19 EDT 2011: mmstartup: Starting GPFS ...
```

Issue mmstartup -a to start the GPFS daemon (aka, mmfsd) on all nodes in the cluster.

```
[root@x36n06 gpfs_install]# mmgetstate -a

 Node number  Node name        GPFS state
------------------------------------------
       1      x36i01           active
       2      x36i02           active
       3      x36i03           active
       4      x36i04           arbitrating
       5      x36i05           arbitrating
       6      x36i06           arbitrating
```

Issue mmgetstate -a to be sure GPFS is active on all nodes before proceeding.

If you issue mmgetstate -a *immediately* after starting GPFS on all nodes, you may find many nodes still in an arbitrating state trying acquire quorum. This is especially true on larger clusters. By repeating the mmgetstate -a command a few a seconds later, all of the nodes should be in an active state.

```
[root@x36n06 gpfs_install]# mmgetstate -a

 Node number  Node name        GPFS state
------------------------------------------
       1      x36i01           active
       2      x36i02           active
       3      x36i03           active
       4      x36i04           active
       5      x36i05           active
       6      x36i06           active
```

GPFS states
active: GPFS is ready for operations.
arbitrating: a node is trying to acquire a quorum
down: GPFS daemon is not running on the node or is recovering from an internal error.
unknown: an indeterminate state indicating that a node can not be accessed.

```
[root@x36n06 gpfs_install]# mmshutdown -a
```

Issue mmshutdown -a to stop the GPFS daemon (aka, mmfsd) on all nodes in the cluster. Since the output is long and tedious, it is not included in this exampe.

Selected mmstartup, mmshutdown and mmgetstate command line options
-a all nodes in the GPFS cluster
-N <file_name> selected list of GPFS nodes (host names or IP addresses)

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

```
[root@x36n06 gpfs_install]# cat disk.lst
/dev/sdc:x36i05,x36i06::::dcs3700_c
/dev/sdd:x36i06,x36i05::::dcs3700_d
/dev/sde:x36i05,x36i06::::dcs3700_e
/dev/sdf:x36i06,x36i05::::dcs3700_f
/dev/sdg:x36i05,x36i06::::dcs3700_g
/dev/sdh:x36i06,x36i05::::dcs3700_h
```

Create a NSD specification file.

The input version of the NSD specification file for mmcrnsd.

The format of this file is...
f1:f2:f3:f4:f5:f6:f7:
where
- f1 = scsi device
- f2 = comma seperate NSD server list
  there can be upto 8 NSD servers
- f3 = NULL (retained for legacy reasons)
- f4 = usage
- f5 = failure group
- f6 = NSD name
- f7 = storage pool name

Fields left blank are filled with default value

Back up this specifications since its an input/output file for the mmcrnsd.

```
[root@x36n06 gpfs_install]# cp disk.lst disk.lst.orig
```

```
[root@x36n06 gpfs_install]# mmcrnsd -F disk.lst -v no
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: 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
-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)

```
[root@x36n06 gpfs_install]# cat disk.lst
# /dev/sdc:x36i05,x36i06::::dcs3700_c
dcs3700_c:::dataAndMetadata:4005::
# /dev/sdd:x36i06,x36i05::::dcs3700_d
dcs3700_d:::dataAndMetadata:4006::
# /dev/sde:x36i05,x36i06::::dcs3700_e
dcs3700_e:::dataAndMetadata:4005::
# /dev/sdf:x36i06,x36i05::::dcs3700_f
dcs3700_f:::dataAndMetadata:4006::
# /dev/sdg:x36i05,x36i06::::dcs3700_g
dcs3700_g:::dataAndMetadata:4005::
# /dev/sdh:x36i06,x36i05::::dcs3700_h
dcs3700_h:::dataAndMetadata:4006::
```

This is the output version of the NSD specification file.
It is used as input for mmcrfs.

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

```
[root@x36n06 gpfs_install]# mmlsnsd
```

Verify that the NSDs were properly created.

```
 File system   Disk name    NSD servers
---------------------------------------------------------------------------
 gpfs1         dcs3700_c    x36i05.pbm.ihost.com,x36i06.pbm.ihost.com
 gpfs1         dcs3700_d    x36i06.pbm.ihost.com,x36i05.pbm.ihost.com
 gpfs1         dcs3700_e    x36i05.pbm.ihost.com,x36i06.pbm.ihost.com
 gpfs1         dcs3700_f    x36i06.pbm.ihost.com,x36i05.pbm.ihost.com
 gpfs1         dcs3700_g    x36i05.pbm.ihost.com,x36i06.pbm.ihost.com
 gpfs1         dcs3700_h    x36i06.pbm.ihost.com,x36i05.pbm.ihost.com
```

```
[root@x36n06 gpfs_install]# mmlsnsd -X
```

Verify that the NSDs were properly created showing extended attributes.

```
 Disk name    NSD volume ID      Device       Devtype  Node name                Remarks
-------------------------------------------------------------------------------------------
 dcs3700_c    81286FE54DE676EA   /dev/sdc     generic  x36i05.pbm.ihost.com     server node
 dcs3700_c    81286FE54DE676EA   /dev/sdc     generic  x36i06.pbm.ihost.com     server node
 dcs3700_d    81286FE64DE676EB   /dev/sdd     generic  x36i05.pbm.ihost.com     server node
 dcs3700_d    81286FE64DE676EB   /dev/sdd     generic  x36i06.pbm.ihost.com     server node
 dcs3700_e    81286FE54DE676EC   /dev/sde     generic  x36i05.pbm.ihost.com     server node
 dcs3700_e    81286FE54DE676EC   /dev/sde     generic  x36i06.pbm.ihost.com     server node
 dcs3700_f    81286FE64DE676EC   /dev/sdf     generic  x36i05.pbm.ihost.com     server node
 dcs3700_f    81286FE64DE676EC   /dev/sdf     generic  x36i06.pbm.ihost.com     server node
 dcs3700_g    81286FE54DE676ED   /dev/sdg     generic  x36i05.pbm.ihost.com     server node
 dcs3700_g    81286FE54DE676ED   /dev/sdg     generic  x36i06.pbm.ihost.com     server node
 dcs3700_h    81286FE64DE676ED   /dev/sdh     generic  x36i05.pbm.ihost.com     server node
 dcs3700_h    81286FE64DE676ED   /dev/sdh     generic  x36i06.pbm.ihost.com     server node
```

Best Practice: The DCS3700 supports a "dual active" architecture rather than "active:active". While it can function using the Linux Device Mapper driver, the Linux MPP (aka RDAC) driver is best suited to exploit the "dual active" architecture.

"generic" is the device type for RDAC driver.
"dmm" is the device type for Linux Device mapper driver.

These are the SCSI device names used by RDAC. If device name changes, it will not imact GPFS as it relies on the NSD volume ID.
"/dev/dm-<int> is used for the Linux Device Mapper driver.

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

```
[root@x36n06 gpfs_install]# mmcrfs /gpfs1 gpfs1 -F disk.lst -A yes -B 4m -v no -n 32 -j scatter
The following disks of gpfs1 will be formatted on node x36n01.pbm.ihost.com:
    dcs3700_c: size 15623913472 KB
    dcs3700_d: size 15623913472 KB
    dcs3700_e: size 15623913472 KB
    dcs3700_f: size 15623913472 KB
    dcs3700_g: size 15623913472 KB
    dcs3700_h: size 15623913472 KB
Formatting file system ...
Disks up to size 142 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/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
-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)
-n: estimated number of nodes that will mount this file system (see **note** below).

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, 128K, 256k. 512k, 1M, 2M, 4M.

The /etc/fstab is automatically updated by this command.

COMMENT:

- Do **not** forget to set the -n parameter. Since it provides an estimate for the number of nodes that will mount the file system, try estimate future growth without wildly overestimating. While it can be off quite a bit with minimal impact, after it crosses a certain threshold performance can be severely impacted (e.g., performance will be impacted when it is off by an order of magnitude and the file system is over 70% capacity) and this parameter can not be easily changed.
- If you configure GPFS with a SAN topology on a cluster that you anticipate will exceed 32 nodes, seek technical assistance from IBM.

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

Verify file system status.

```
[root@x36n06 gpfs_install]# mmlsfs gpfs1
flag                value                    description
------------------- ------------------------ -----------------------------------
 -f                 131072                   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                 scatter                  Block allocation type
 -D                 nfs4                     File locking semantics in effect
 -k                 all                      ACL semantics in effect
 -n                 32                       Estimated number of nodes that will mount file system
 -B                 4194304                  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      Wed Jun  1 13:30:20 2011 File system creation time
 --fastea           yes                      Fast external attributes enabled?
 --filesetdf        no                       Fileset df enabled?
 --inode-limit      134217728                Maximum number of inodes
 -P                 system                   Disk storage pools in file system
 -d                 dcs3700_c;dcs3700_d;dcs3700_e;dcs3700_f;dcs3700_g;dcs3700_h  Disks in file system
 -A                 yes                      Automatic mount option
 -o                 none                     Additional mount options
 -T                 /gpfs1                   Default mount point
 --mount-priority   0                        Mount priority
```

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

Verify disk status.

```
[root@x36n06 gpfs_install]# mmlsdisk gpfs1
disk         driver   sector failure holds    holds                            storage
name         type       size   group metadata data  status        availability pool
------------ -------- ------ ------- -------- ----- ------------- ------------ ---------
dcs3700_c    nsd         512    4005 yes      yes   ready         up           system
dcs3700_d    nsd         512    4006 yes      yes   ready         up           system
dcs3700_e    nsd         512    4005 yes      yes   ready         up           system
dcs3700_f    nsd         512    4006 yes      yes   ready         up           system
dcs3700_g    nsd         512    4005 yes      yes   ready         up           system
dcs3700_h    nsd         512    4006 yes      yes   ready         up           system
```

status
- ready: normal status
- suspended: indicates that data is to be migrated off this disk
- being emptied: transitional status in effect while a disk deletion is pending
- replacing: transitional status in effect for old disk while replacement is pending
- replacement: transitional status in effect for new disk while replacement is pending

availabity
- up: disk is available to GPFS for normal read and write operations
- down: no read and write operations can be performed on this disk
- recovering: an intermediate state for disks coming up
  - during this state GPFS verifies and corrects data
  - read operations can not be performed (since recovering data may be stale)
  - write operations can be performed
- unrecovered: the disks was not successfully brought up

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

```
[root@x36n06 gpfs_install]# mmmount /gpfs1 -a
Wed Jun  1 13:31:03 EDT 2011: mmmount: Mounting file systems …

[root@x36n06 gpfs_install]# chmod 777 /gpfs1
[root@x36n06 gpfs_install]# touch /gpfs1/test_file.1
[root@x36n06 gpfs_install]# ls -l /gpfs1
total 0
-rw-r--r-- 1 root root           0 Jun 14 05:37 test_file.1

[root@x36n06 gpfs_install]# time dd if=/dev/zero of=/gpfs1/test_file.2 bs=4096k count=4096
4096+0 records in
4096+0 records out
17179869184 bytes (17 GB) copied, 12.2174 seconds, 1.4 GB/s
real    0m12.263s
user    0m0.005s
sys     0m3.651s

[root@x36n06 gpfs_install]# cat /etc/fstab
/dev/rootvg/rootlv      /                       ext3    defaults        1 1
/dev/rootvg/homelv      /home                   ext3    defaults        1 2
/dev/rootvg/varlv       /var                    ext3    defaults        1 2
/dev/rootvg/optlv       /opt                    ext3    defaults        1 2
/dev/rootvg/usrlv       /usr                    ext3    defaults        1 2
/dev/rootvg/tmplv       /tmp                    ext3    defaults        1 2
x36n01:/scratch         /scratch                nfs     defaults        0 0
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/rootvg/swaplv      swap                    swap    defaults        0 0
/dev/gpfs1          /gpfs1                  gpfs       rw,mtime,atime,dev=gpfs1,noauto 0 0
```

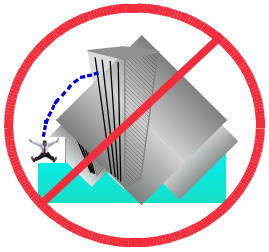
Permissions propogate to mount points on all node
Sanity check.

Use dd as another sanity check,
NOT a benchmark!

mmcrfs automatically adds a GPFS stanza to the /etc/fstab file

Using the DCS3700 with GPFS
Installing and Configuring GPFS for the DCS3700

What if I screw up?
Clean up and start over again!



Don't jump!
It's easier the second time.

Option #1 The proper way to do it.

1.Unmount the GPFS file system

```
[root@x36n06 gpfs_install]# mmunmount /gpfs1 -a
```

2.Delete GPFS file system

```
[root@x36n06 gpfs_install]# mmdelfs /gpfs1 -p
```

3.Delete GPFS NSDs

```
[root@x36n06 gpfs_install]# for i in c d e f g h
> do
> mmdelnsd dcs3700_$i
> done
```

4.Shutdown GPFS daemons

```
[root@x36n06 gpfs_install]# mmshutdown -a
```

5.Delete the GPFS cluster

```
[root@x36n06 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@x36n06 gpfs_install]# mmunmount /gpfs1 -a
[root@x36n06 gpfs_install]# mmfsadm cleanup
```

2.Delete selected configuration files on all nodes

```
[root@x36n06 gpfs_install]# rm -f /var/mmfs/etc/mmfs.cfg
[root@x36n06 gpfs_install]# rm -f /var/mmfs/gen/*
[root@x36n06 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.

Using the DCS3700 with GPFS
GPFS Monitoring and Performance Tools

The mmdiag command displays diagnostic information about the internal GPFS state on the current node. See the mmdiag man page for more details.

```
[root@x36n06 gpfs_install]# mmdiag --help

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
```

Use the mmdiag as a safer alternative to the mmfsadm command. The mmfsadm command is intended for use by trained service personnel. IBM suggests you do not run this command except under the direction of such personnel.

Using the DCS3700 with GPFS
Monitoring and Performance Tools

Performance Monitoring: mmpmon

- display I/O statistics per mounted file system for each file system on a node
- display aggregate I/O statistics from multiple nodes using the `nlist` option
- display latency measurements in a histogram format

Command syntax for with output in human readable form

- mmpmon -i command_file
  - commands specifying what is displayed and how they are displayed is done by a command file
  - see chapter 6 in the GPFS 3.4 *Advanced Administration Guide* for details

Miscellaneous

- mmpmon requires root access
- Up to 5 instances of mmpmon can be run on 1 node at one time
- mmpmon can be configured to collect statistics from multiple nodes at the same time

Using the DCS3700 with GPFS
Monitoring and Performance Tools

Latency Histogram Using mmpmon

```
[root@x36n01 gpfs_install]# cat cmd.1                <-- Define the histogram
rhist nr 64k;256k;1m;4m; 1;10;30;100                      Form: rhist nr <packet sizes> <space> <time intervals>
[root@x36n01 gpfs_install]# mmpmon -i cmd.2
mmpmon node 129.40.6.193 name x36i01 rhist nr 64k;256k;1m;4m; 1;10;30;100 OK
[root@x36n01 gpfs_install]# cat cmd.2
rhist on
[root@x36n01 gpfs_install]# mmpmon -i cmd.2          <-- enable histogram
mmpmon node 129.40.6.193 name x36i01 rhist on OK
[root@x36n01 gpfs_install]# cat cmd.3
rhist s                                              <-- display histogram data for non-empty bins
[root@x36n01 gpfs_install]# mmpmon -i cmd.3
mmpmon node 129.40.6.193 name x36i01 rhist s OK read timestamp 1308054574/52992
size range          262145 to    1048576 count       2048
  latency range        0.0 to        1.0 count       1649
  latency range        1.1 to       10.0 count         26
  latency range       10.1 to       30.0 count         66
  latency range       30.1 to      100.0 count        247
  latency range      100.1 to        0.0 count         60
size range         1048577 to    4194304 count       2048
  latency range        0.0 to        1.0 count        118
  latency range        1.1 to       10.0 count        501
  latency range       10.1 to       30.0 count        100
  latency range       30.1 to      100.0 count        550
  latency range      100.1 to        0.0 count        779
mmpmon node 129.40.6.193 name x36i01 rhist s OK write timestamp 1308054574/53031
size range         1048577 to    4194304 count       2048
  latency range        0.0 to        1.0 count         14
  latency range        1.1 to       10.0 count        537
  latency range       10.1 to       30.0 count        497
  latency range       30.1 to      100.0 count        543
  latency range      100.1 to        0.0 count        457
[root@x36n01 gpfs_install]# cat cmd.4
rhist off
[root@x36n01 gpfs_install]# mmpmon -i cmd.4          <-- disable histogram
mmpmon node 129.40.6.193 name x36i01 rhist off OK
```

To use mmpmon, do the following
1. Login as root on node X
2. Login as user on node X
3. Execute cmd.1, cmd.2
4. Run application(s)
5. Execute cmd.3
6. Repeat steps 4, 5 as needed
7. Execute cmd.4

mmpmon can be configured to collect and display histogram data from multiple nodes at the same time.

Using the DCS3700 with GPFS
Monitoring and Performance Tools

Network Performance: nsdperf

- nsdperf is a tool designed to assess network performance independent of GPFS
- nsdperf easily does “many to many” tests emulating GPFS behavior, but without using GPFS*

Miscellaneous

- Found in /usr/lpp/mmfs/samples/net
- See README file for documentation

* GPFS is very sensitive to the LAN; if there are LAN problems, they will appear as “symptoms” when using GPFS. The nsdperf tool was designed to help determine whether a performance problem in GPFS was merely the result of a networking issue or a systemic GPFS issue. If you suspect that there is a network issue affecting GPFS, you can run nsdperf; since it does not use any GPFS code and if it performs poorly, it implies that the network is the fundamental cause of the problem rather than GPFS.

Using the DCS3700 with GPFS
Monitoring and Performance Tools

Assessing Network Performance using nsdperf

```
[root@x36n06 net]# g++ -O2 -DRDMA -o nsdperf -lpthread -lrt -libverbs nsdperf.C
[root@x36n06 net]# cat doit.x
for i in 1 2 3 4 5 6
do
   ssh x36i0$i '/scratch/gpfs_install/net/nsdperf -s </dev/null >/dev/null 2>&1 &'
Done
[root@x36n06 net]# cat cmd.6
rdma on
client x36i01 x36i02 x36i03 x36i04
server x36i05 x36i06
[root@x36n06 net]# ./doit.x
[root@x36n06 net]# ./nsdperf -i cmd.6
RDMA is now on
Connected to x36i01
Connected to x36i02
Connected to x36i03
Connected to x36i04
Connected to x36i05
Connected to x36i06
Nsdperf> ttime 30
Test time set to 30 seconds
nsdperf> test write read
4-2 write 3930 MB/sec (938 msg/sec), cli 1% srv 1%, time 30, buff 4194304, RDMA
4-2 read 3920 MB/sec (934 msg/sec), cli 1% srv 1%, time 30, buff 4194304, RDMA
nsdperf> test nwrite
4-2 nwrite 3930 MB/sec (938 msg/sec), cli 1% srv 1%, time 30, buff 4194304, RDMA
nsdperf> test rw
4-2 rw 7160 MB/sec (1710 msg/sec), cli 1% srv 1%, time 30, buff 4194304, RDMA
nsdperf> killall
Connection to 129.40.6.193 broken
Connection to 129.40.6.194 broken
Connection to 129.40.6.195 broken
Connection to 129.40.6.196 broken
Connection to 129.40.6.197 broken
Connection to 129.40.6.198 broken
nsdperf> quit
[root@x36n06 net] #
```

Hostnames specify IB LAN

The default is 10 seconds

Find nsdperf in /usr/lpp/mmfs/samples/net

Test types:

write	Clients write round-robin to all servers
read	Clients read round-robin from all servers
nwrite	Same as write test, but use NSD-style writing
swrite	Each tester thread writes to only one server
sread	Each tester thread reads from only one server
rw	Half of the tester threads read and half write

Using the DCS3700 with GPFS
Benchmark Results – Streaming #1

Performance *vs.* Transaction Size

Transaction Size[1]	4 MB	2 MB	1 MB	512 KB	256 KB
wc_off	1455.3	1518.6	519.7	279.0	145.8
wc_on, mirror_off	1859.1	1755.6	xxx	454.1	266.6
wc_on, mirror_on	1653.8	1568.4	747.8	324.4	216.3
rc_off	2008.4	1790.3	692.6	367.1	197.1
rc_on, dynamic prefetch_off	2024.6	1807.3	666.6	339.3	182.7
rc_on, dynamic prefetch_on	2038.5	1811.1	673.3	349.6	189.7

LEGEND
- K = 2^{10}, M = 2^{20}
- wc = write cache, rc = read cache

NOTES & CAVEATS
- Benchmark Code: ibm.v4b
- Data rates are given in MB/s. They should be used as a *realistic* least upper bound. Different HPC I/O benchmark tools will yield slightly different results *generally* varying within 10% of these results. A more thorough benchmark analysis can be found in the associated spread sheet.
- Each job consists of 48 MPI tasks with 8 tasks per per node.
- File system spans 6 x LUNs (n.b., 1 LUN[2] per array) where each array is 8+P+Q RAID 6
- GPFS block allocation map type = scatter (*i.e.*, -j scatter)
- Other GPFS parameters: pagepool = 1024 MB, maxMBpS = 3000
- max_sectors_kb = read_ahead_kb = 4096

FOOTNOTES
1. The application buffer size = GPFS block size = driver buffer size = DCS3700 stripe width
2. Each LUN spans the entire array (*n.b.*, there is no “short stroking”)

Using the DCS3700 with GPFS
Benchmark Results – Streaming #2

Performance *vs*. Number of LUNs

Transaction Size[1]	6	4	2	1
wc_off	1455.3	1042.2	503.4	230.1
wc_on, mirror_off	1859.1	1281.6	667.5	356.9
wc_on, mirror_on	1653.8	1237.6	630.4	346.1
rc_off	2008.4	1473.6	795.8	434.5
rc_on, dynamic prefetch_off	2024.6	1503.5	865.3	473.8
rc_on, dynamic prefetch_on	2038.5	1516.0	863.1	468.3

LEGEND
- K = 2^{10}, M = 2^{20}
- wc = write cache, rc = read cache

NOTES & CAVEATS
- Data rates are given in MB/s. They should be used as a *realistic* least upper bound. Different HPC I/O benchmark tools will yield slightly different results *generally* varying within 10% of these results. A more thorough benchmark analysis can be found in the associated spread sheet.
- Each job consists of 48 MPI tasks with 8 tasks per per node.
- File system spans the specified number of LUNs (n.b., 1 LUN[2] per array) where each array is 8+P+Q RAID 6
- GPFS block allocation map type = scatter (*i.e.*, -j scatter)
- Other GPFS parameters: pagepool = 1024 MB, maxMBpS = 3000
- max_sectors_kb = read_ahead_kb = 4096

FOOTNOTES
1. The application buffer size = GPFS block size = driver buffer size = DCS3700 stripe width = 4 MB
2. Each LUN spans the entire array (*n.b.*, there is no “short stroking”)

Using the DCS3700 with GPFS
Benchmark Results – mdtest*

	option:-i	option:-N	option:-u	option:-n	# of nodes+	
Row 1	3	1	1	8000	4	np=4, -n 8000
Row 2	3	1	1	4000	4	np=8, -n 4000
Row 3	3	1	1	2000	4	np=16, -n 2000
Row 4	3	1	1	1000	4	np=32, -n 1000

	Directory creation	Directory stat	Directory removal	File creation	File stat	File removal	Tree creation	Tree removal
Row 1	3788.934	2009.88	1418.258	8344.002	1978.817	2097.736	66.907	12.596
Row 2	7778.261	3425.214	3700.784	8409.497	3464.765	2529.623	34.858	11.098
Row 3	6506.285	5055.528	6096.262	8665.716	4909.534	2310.092	19.452	8.244
Row 4	5944.136	6098.31	8294.777	6592.982	4793.371	2149.819	7.769	3.174

Benchmark Configuration

- The job was run on 4 GPFS client nodes
- File system spans 6 x LUNs (n.b., 1 LUN[2] per array) where each array is 8+P+Q RAID 6 where each LUN spans the entire array.
- Block allocation map type = scatter (*i.e.*, -j scatter)
- Other GPFS parameters: pagepool = 1024 MB, maxMBpS = 3000

FOOTNOTES:

* mdtest is an MPI-coordinated metadata benchmark test that performs open/stat/close operations on files and directories and then reports the performance. mdtest is available at http://mdtest.sourceforge.net/

+ mdtest was executed only on the GPFS client nodes

Using the DCS3700 with GPFS
Benchmark Results – File Create, List, and Remove

Test Notes

- Tests were single threaded and done on one of the NSD servers.
- Uncached operations were done after unmounting and remounting the file system.
- File system spans 6 x LUNs (n.b., 1 LUN[2] per array) where each array is 8+P+Q RAID 6 where each LUN spans the entire array. Disks were 7200 RPM.
- Block allocation map type = scatter (*i.e.*, -j scatter)
- Other GPFS parameters: pagepool = 1024 MB, maxMBpS = 3000

```
20K files @ 32 KB per file:
          uncached        cached
Create:   16.1 seconds    N/A
ls -l:    31.7 seconds    19.2 seconds
rm -R:    10.2 seconds     6.1 seconds

50K files @ 32 KB per file
          uncached        cached
Create:   34.1 seconds    N/A
ls -l:    60.5 seconds    93.6 seconds
rm -R:     8.7 seconds    22.5 seconds

100K files @ 32 KB per file
          uncached         cached
Create:    86.5 seconds    N/A
ls -l:    172.9 seconds    163.7 seconds
rm -R:     50.0 seconds     46.0 seconds
```

Using the DCS3700 with GPFS
Installing and Configuring GPFS - Errata

GPFS is officially tested with RHEL and SUSE Linux *only,* but it *generally* works with other RHEL like distributions.

- CentOS is the most common example.

Non-RHEL distributions may require some "tweaking".

- e.g., **`make Autoconfig`** fails when building the portability layer

CentOS 5.4 example.

- Must change the Red Hat version identifier

```
# echo "Red Hat Enterprise Linux Server release 5.4 (Tikanga)" > /etc/redhat-release
```

- Very occasionally other things need tweaking; e.g., "broken" symbolic link
  - The link may appear as follows...

```
# ls -la /lib/modules/2.6.18-164.el5/build*
lrwxrwxrwx 1 root root 46 Feb  2 16:14 build -> ../../../usr/src/kernels/2.6.18-164.el5-x86_64
```

  - It should look like this...

```
# ls -la /lib/modules/2.6.18-164.el5/build*
lrwxrwxrwx 1 root root 44 Feb  4 13:08 build -> /usr/src/kernels/2.6.18-164.11.1.el5-x86_64/
```

  - Fix it with the following steps...

```
# unlink /lib/modules/2.6.18-164.el5/build
# ln -s /usr/src/kernels/2.6.18-164.11.1.el5-x86_64/ /lib/modules/2.6.18-164.el5/build
```

Questions and Answers