Dell™ HPC NFS Storage Solution

A Solution Guide

Li Ou, Scott Collier, Onur Celebioglu
Dell HPC Storage Solutions
Engineering Team

Dell HPC Storage Solutions
www.dell.com/hpc
www.dellhpcsolutions.com

September 2010

Contents

Introduction

Cluster computing has become the most popular architecture for high performance computing (HPC) today. In all clusters, the storage I/O system is one of the most important components that impacts how efficiently the cluster runs. With multiple compute nodes accessing data simultaneously, parallel file systems gained popularity on cluster computing platforms. A great example is the Dell | Terascala HPC Storage Solution (DT-HSS), that offers high throughput Lustre-based storage as an appliance.

While a DT-HSS system offers great flexibility to satisfy parallel I/O needs for performance, capacity, scalability and ease of use, not all HPC applications are heavily I/O dependent. For clusters running applications with lower I/O needs, simplicity, reliability and cost may be the primary design factors as opposed to performance and scalability. Furthermore, even in clusters with higher I/O requirements, scratch storage needs can be met with a product such as DT-HSS but administrators may prefer a secondary storage repository to keep users' home directories, application storage and longer term storage of the application data. For these needs, a Network File System (NFS) provides a robust solution. NFS is available with virtually all Linux distributions: therefore, it is cost effective and is easy to configure, deploy, and maintain.

The Dell NFS Storage Solution (NSS) is a unique new storage solution providing cost-effective NFS storage as an appliance. Designed to scale from 20 TB installations up to 80 TB of usable space, the NSS is delivered as a fully configured, ready-to-go storage solution and is available with full hardware and software support from Dell. The solution is highly flexible, available in configurations that can plug into either an Ethernet fabric via its 10GbE adaptor, or into any existing InfiniBand® network through a Quad Data Rate (QDR) IB HCA. The system is tuned and optimized to fully utilize the underlying hardware it is built on, delivering, for example, a write throughput of up to 1480MB/s with the IP over InfiniBand (IPoIB) protocol.

This guide describes the Dell HPC NFS Storage Solution which delivers all the benefits of a NFS file system-based storage solution in a simple-to-use, cost-effective appliance.

Figure 1. HPC NFS Storage Solution

Dell HPC NFS Storage Solution

- Dell developed/validated reference configurations to achieve optimal performance and reliability
- Scales up to 80 TB usable space under a single namespace
- Up to 1.4 GB/s throughput
- 10Gb Ethernet or QDR connectivity
- Fully supported from Dell that is easy to acquire, deploy and manage

NSS Overview

The Dell HPC NFS Storage Solution (NSS) is available in three configurations. These configurations are small (20TB useable space), medium (40TB useable space) and large (80TB useable space). The purpose of this appliance is to provide an NFS storage solution configured following Dell-developed best practices, and is easy to implement and manage. The NSS is also easy to install and integrate into an existing cluster. Based on testing performed in HPC Engineering Lab, detailed tuning and best practices guidance was developed to achieve optimum performance and reliability.

The NSS solution has two parts: The NFS gateway server and direct-attached storage (DAS) enclosures. The NFS gateway server is a Dell PowerEdge™ R710. For all three configurations, the server is configured with Intel® 5600 series six-Core processors. In small and medium configurations, the R710 contains 24 GB of 1333MHz DDR3 memory. While the large configuration, the server has 48 GB, but supports configurations with up to 192 GB of RAM. Internal storage for the OS on the R710 is configured with two HDs set up in a RAID 1 configuration with a global hot spare. There are also two additional drives configured as a RAID 0 for scratch space to help with operations such as file system checks and repairs. The R710 also has four PCIe G2 slots (two x4 and two x8).

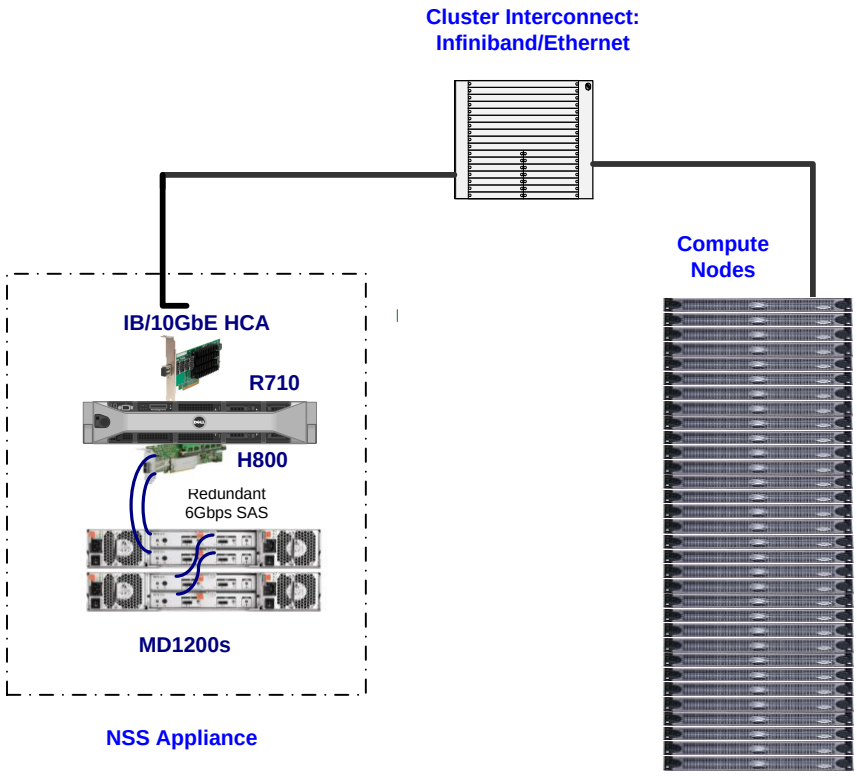
The Dell NSS leverages the Dell PowerVault™ MD1200 enclosure for the external storage. The MD1200 holds up to 12 hot-pluggable, 3.5 inch hard drives in a single 2U rack enclosure. All NSS configurations use 2TB, 7200 RPM NearLine SAS drives for user data. The MD1200 also supports removing and replacing redundant components without disruption to operation. For example, there are two Enclosure Management Modules (EMMs) that provide redundant enclosure management capability, as well as hot pluggable hard drives and two fully redundant power supplies.

All NSS configurations use the Dell PERC H800 RAID controller with two x4 external mini-SAS ports, a transportable battery backup unit and support for a PCIe 2.0 x8 host interface. The PERC H800 enables 6 Gb/s SAS end-to-end high performance and scalable solution for Dell PowerEdge servers and PowerVault MD1200 expansion arrays. The H800 has 512MB (or 1GB) of battery-backed cache memory and supports RAID levels 0, 1, 5, 6, 10, 50 and 60. The PERC H800 provides the logic to govern everything that occurs within the MD1200 arrays by performing the necessary calculations and controlling the I/O performance operations. Multiple MD1200s can be daisy chained from a single H800 with a current maximum of 96 3.5” drives. Each Dell NSS configuration utilizes both ports going from the H800s to the MD1200 enclosures to increase redundancy and improve performance. Both ports are used when daisy chaining from one MD1200 to the next MD1200. There is no additional configuration required to utilize both ports. The MD1200 EMMs can see all the LUNs and, if there is an enclosure management module failure, it is transparent to the NFS server as the logic and paths are handled by the PERC H800.

The Dell NSS is available with either 10GigE Ethernet or QDR InfiniBand connectivity. The NSS can connect and integrate with your QDR IB fabric whether the cluster is brand new or existing. The NFS server can have a Mellanox ConnectX QDR InfiniBand card to connect to the InfiniBand fabric.

See Figure 1 for more information on the hardware included in the NSS. Figure 2 shows an example high performance compute cluster using a NSS appliance as the back-end storage. Either Ethernet or InfiniBand can be used for connectivity depending on the existing cluster configuration.

Figure 2. NSS Architecture Overview



How To Order

Dell HPC NFS Storage Solution is available in the standard configurations listed in Table 1. Contact your Dell Sales Representative to discuss which offering would suit the best in your environment and the ordering process. Customers can order any of the six pre-configured solutions or customization can be made to fit specific needs. Based on the customization, some of the best practices discussed in this document may not apply.

Table 1. NSS Server Hardware

	Small	Medium	Large
NFS Gateway Server Configuration			
Server Model	PowerEdge R710	PowerEdge R710	PowerEdge R710
Processor	Dual Intel Xeon 5520	Dual Intel Xeon 5520	Dual Intel Xeon 5520
Memory	24GB (6x4GB 1333 MHz Dual Ranked RDIMMs)	24GB (6x4GB 1333 MHz Dual Ranked RDIMMs)	48GB (6x8GB 1333 MHz Dual Ranked RDIMMs)
Optional InfiniBand HCA	Mellanox ConnectX-2 QDR InfiniBand HCA	Mellanox ConnectX-2 QDR InfiniBand HCA	Mellanox ConnectX-2 QDR InfiniBand HCA
Optional 10GigE NIC	Intel X520 DA 10Gb Dual Port SFP+ Adapter	Intel X520 DA 10Gb Dual Port SFP+ Adapter	Intel X520 DA 10Gb Dual Port SFP+ Adapter
Local Storage and RAID Controller	PERC H700 with (5) 146GB 10K SAS hard disks	PERC H700 with (5) 146GB 10K SAS hard disks	PERC H700 with (5) 146GB 10K SAS hard disks
External RAID Controller	(1) PERC H800	(1) PERC H800	(2) PERC H800
Direct Attached Storage Configuration			
Storage Enclosure	(1) PowerVault MD1200	(2) PowerVault MD1200	(4) PowerVault MD1200
Hard Disk Drives	(12) 2TB NL SAS	(24) 2TB NL SAS	(48) 2TB NL SAS
Usable Capacity	20TB	40TB	80TB
Software and Services			
Software	Red Hat Enterprise Linux 5.5	Red Hat Enterprise Linux 5.5	Red Hat Enterprise Linux 5.5
	Red Hat Scalable File System (XFS)	Red Hat Scalable File System (XFS)	Red Hat Scalable File System (XFS)
	Dell OpenManage™ Server Administrator	Dell OpenManage Server Administrator	Dell OpenManage Server Administrator
Services	Three-year Dell ProSupport with Mission Critical, 4hr Onsite Service, 24x7	Three-year Dell ProSupport with Mission Critical, 4hr Onsite Service, 24x7	Three-year Dell ProSupport with Mission Critical, 4hr Onsite Service, 24x7

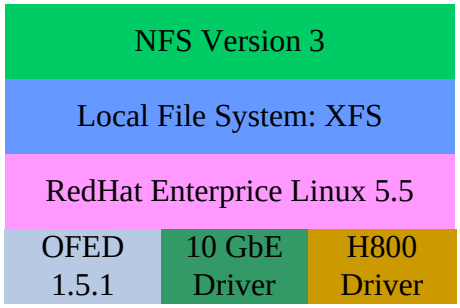
NSS Solution Details

Customers have different requirements with regard to the I/O subsystem used for the cluster. The NSS is appropriate for customers who need an industry standard storage solution that is easy to administer, is reliable and has very good performance within certain boundaries. This section provides an overview of the software and hardware required to build a NSS, as well as the recommended performance optimizations.

Software

The NSS Software stack is illustrated in Figure 3 The software stack is completely supported and all NSS configurations come standard with 3 years of 24x7 support. The subsections below describe each of the components of the software stack.

Figure 3. NSS Software Stack



Operating System

The Dell HPC NSS solution uses Red Hat Enterprise Linux® 5.5 installed on the R710 NFS gateway server.

Systems Management

Dell OpenManage Systems Administrator (OMSA) provides a comprehensive systems management solution to manage PowerEdge servers and direct-attached PowerVault storage. OMSA simplifies both server and storage management with a web-based management GUI or CLI, provides up-to-date inventory and health information including alerts from server and storage components and is used to configure storage internal and directly attached to PowerEdge servers.

NFS

NFS is an industry standard network file system that is commonly used in Linux environments. The NFS packages are native to most Linux distributions including Red Hat Enterprise Linux. NFS provides an access protocol that easy to setup, configure and manage and that is commonly used in HPC environments. The version of NFS used in the NSS solutions is version 3 and used TCP as the network protocol. NFS provides a great solution when the I/O requirements of the cluster can be met by a single server.

Red Hat Scalable File System and LVM

The Red Hat Scalable File System or XFS file system was originally created by SGI (Silicon Graphics Inc.) to run on IRIX. XFS was ported over to Linux and is now available on most Linux distributions. XFS was chosen for the NSS because XFS is capable of scaling beyond 16 TB and provides good performance for a broad range of applications. At the time of the paper, ext3 and ext4 are both limited to less than 16 TB per single file system.

Some of the key benefits of XFS are:

- Scalability in capacity – XFS is 64-bit file system that can theoretically scale to a many Exabytes.
- Journaling to guarantee quick file system recovery.
- Internally partitioned file system scheme called allocation groups. Allocation groups provide scalability and parallelism, particularly for file system checking and recovery.
- Striped allocation which ensures the file system is aligned with the underlying RAID hardware thus ensuring optimized performance.

Red Hat provides XFS via a Red Hat Network (RHN) channel called Red Hat Scalable File System. There are 3 packages in the RH SFS channel: `xfsprogs`, `xfsprogs-devel` and `xfsdump`. The version of XFS used in the NSS is 2.10.2-7.

Logical Volume Manager(LVM) is a flexible storage management tool that provides many features for managing the storage subsystem. Some of these features include snapshots, resizing logical volumes and concatenating or striping LUNs together into a single LUN. For example, LVM is used on the NSS large configuration to stripe two 40TB LUNs together so that the NFS server can present a single namespace to the clients. See Table 2 for more information on the driver and OS kernel version used in the testing presented in this whitepaper.

Table 2. NSS Server Software Versions

NSS Server Software Versions							
	OFED	Intel 10GigE NIC	Broadcom NIC	H800	Kernel	XFS	OMSA
NFS Server	1.5.1	2.0.44-k2	2.0.2	00.00.04.17-RH1	2.6.18-194.el5	2.10.2-7	6.3.0

NSS Configurations

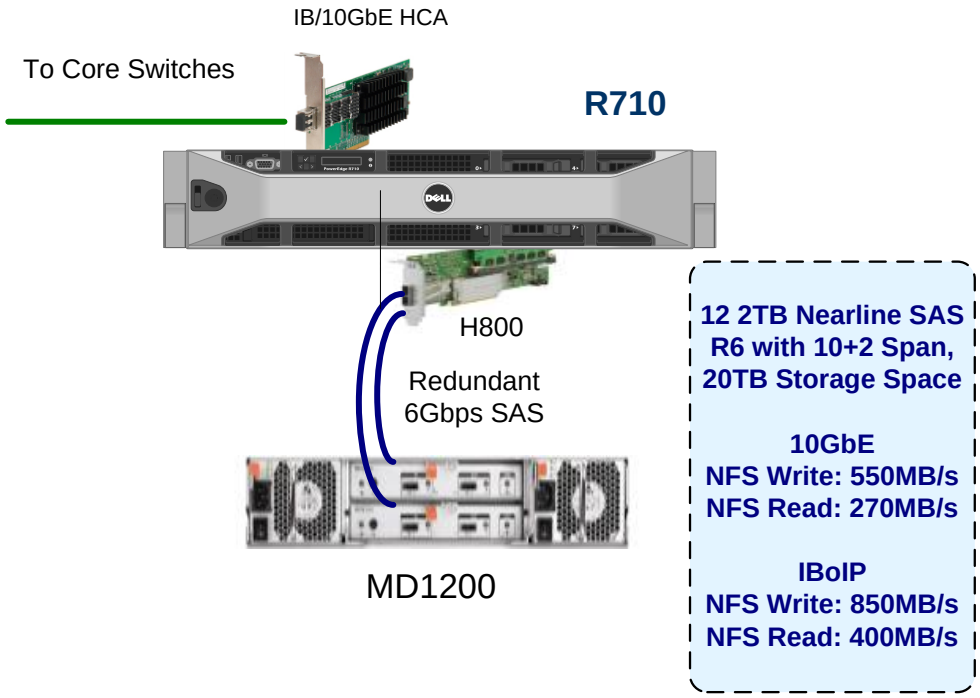
The NSS is available in three storage configurations to allow the customer to choose which NSS is right for the environment based on the current capacity and performance needs. The choices are:

- Small (20 TB usable)
- Medium (40 TB usable)
- Large (80TB usable)

Each size is available with a 10GigE interface or an InfiniBand QDR interface for a total of six standard configurations.

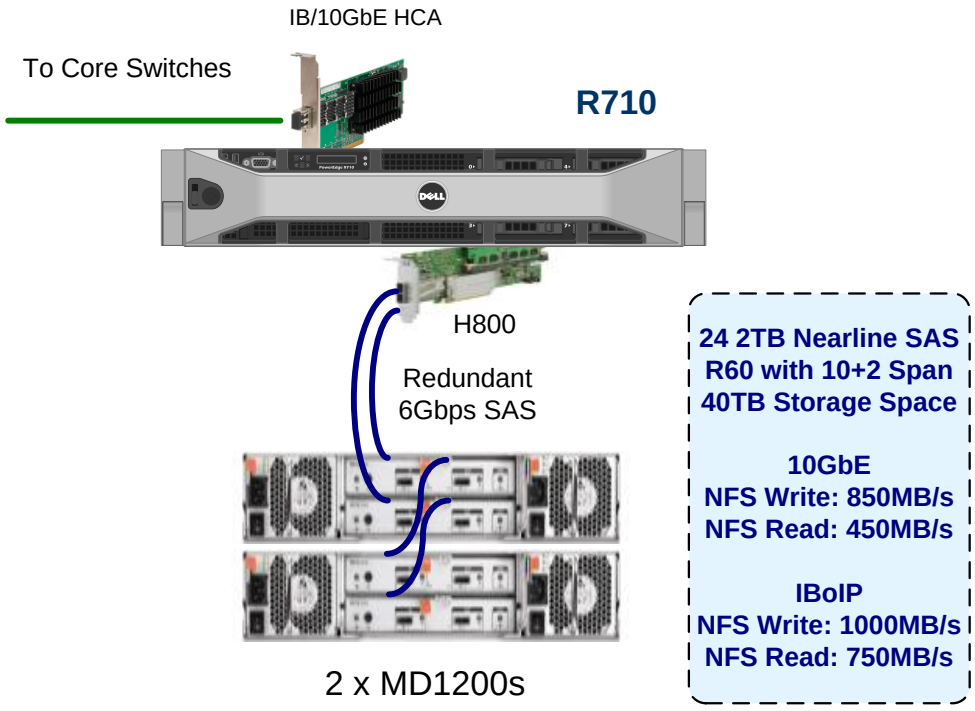
The first storage configuration, shown in Figure 4 is the NSS Small which includes a PowerEdge R710 server attached to a single PowerVault MD1200 via a single PERC H800. The MD1200 has a RAID 6 (10+2) configuration that provides both a safe level of redundancy as well as optimal performance. The stripe element size (sometimes known as “chunk size”) on the RAID 6 or 60 is 512 KB. The total raw disk capacity is 24 TB with a useable capacity of 20 TB.

Figure 4. NSS Small



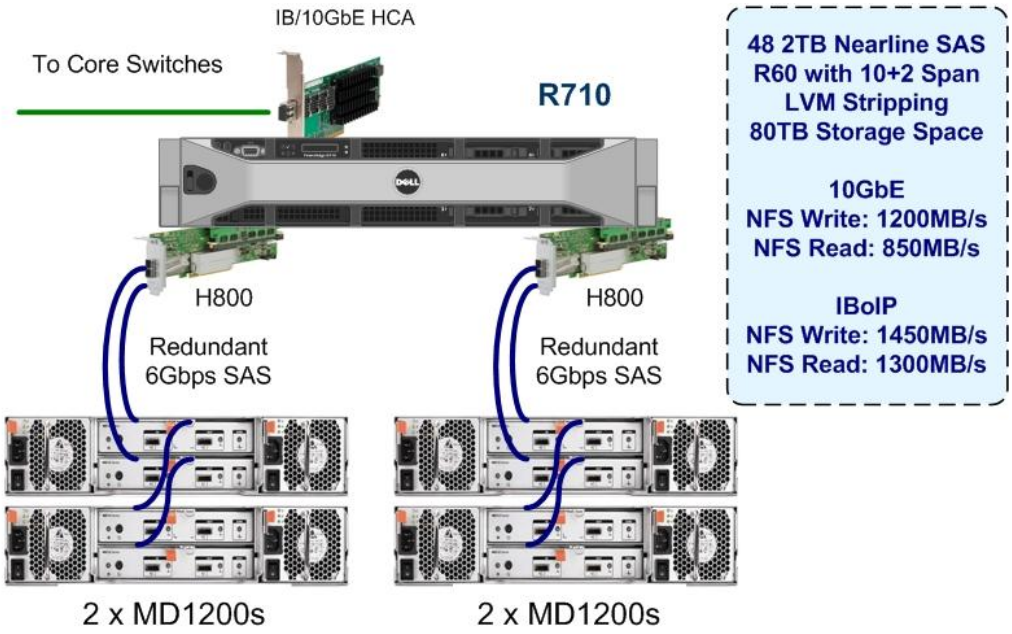
The second storage configuration shown in Figure 5 is the NSS Medium that includes a PowerEdge R710 server attached to a PowerVault MD1200 and has an additional MD1200 enclosure daisy-chained into the configuration. The NSS Medium has a RAID 60 (10 + 2, 2 Spans) volume with a 512 KB stripe element size. The NSS Medium provides 48 TBs of raw disk space and 40 TBs of usable disk space doubling the capacity from the NSS Small as well as improving performance.

Figure 5. NSS Medium



The third storage configuration shown in Figure 6 is the NSS Large which has two PERC H800s; each H800 has two MD1200 enclosures attached. The MD1200 enclosures on each controller are configured in a RAID 60 (10 + 2, 2 Spans) with a 512 KB stripe element size. The LUNs are then striped together via LVM with a 1 MB stripe size. This provides 48 total drives with a raw capacity of 96 TB and a usable capacity of 80 TB effectively doubling the capacity of the NSS Medium and considerably improving performance. The NSS Large is recommended for customers who have larger capacity and performance requirements.

Figure 6. NSS Large



In all three NSS configurations, each R710 server comes with two hard drives configured in a RAID 1 for the operating system, two hard drives configured in a RAID 0 for swap space and a single disk drive for a global hot spare. For all the configurations, the stripe element size is optimized per a prior performance study conducted by the Dell HPC Engineering team[4].

Performance Optimizations

While the default settings may be suitable for many environments, there are opportunities to change some of the configurations and improve performance. This section of the paper covers how to optimize NFS, XFS, and the underlying storage.

NFS Optimizations

The first set of optimizations presented in this paper is specific to NFS – particularly the server. Based on a previous study conducted in Dell HPC Engineering lab, it was found that 256 NFS threads (from a default of 8 threads) provide a good balance between reads and writes on the system. Increasing the number of threads also increases the NFS read-ahead cache. On the NFS server, exporting the LUN with the `async` mount option in `/etc/exports` allows the NFS server to cache data in a write-back fashion, which improves performance because the NFS server can continue servicing requests before a commit to disk is finished. Using the `async` option increases the risk of data loss in the event of a system failure. However, the overall performance is increased because the server tells the client that the data is in the storage even if it is still in cache. For customers where performance is not as important as data integrity, the default mount option of `sync` is recommended.

Another optimization that was made for the NFS server is change the I/O scheduler to `deadline` from the default I/O scheduler, `cfq`. CFO is not optimized for the type of I/O that a NFS server generates.

The `cfq` scheduler is the default because it handles several I/O patterns quite efficiently by distributing the available I/O equally among all the I/O requests. However, when using the `cfq` scheduler in a NFS environment, process starvation is introduced because of the increased disk seeks incurred with NFS I/O and then, the NFS server becomes overloaded and non-responsive. However, the `deadline`[2] I/O scheduler attempts to minimize I/O latency by re-ordering requests by block number to improve performance. In a NFS environment, the algorithm that `deadline` uses is better suited for HPC type workloads. Both `cfq` and `deadline` I/O schedulers were tested and the `deadline` scheduler performed better. For details on how to configure the I/O scheduler and NFS options see Appendix B.

Red Hat Scalable File System (XFS)

XFS[3] is used as the file system and this section describes the different `xfs` format options used to tune performance. By default, `xfs` tries to query the underlying storage device and optimize the settings accordingly. In the case of using LVM, this works fine; however, when presenting a raw LUN to `xfs`, it is important to specify the stripe unit (`su`) and stripe width (`sw`). The stripe unit is basically the stripe element size with which the LUN was formatted. The stripe width tells `xfs` how many data drives are in the LUN.

By default, there are 8 log buffers and the `-l size` option tells `xfs` how large the log buffers can become. This can improve metadata performance; however, the larger the log, the longer it may take to recover a file system that was not unmounted cleanly. For example, the `mkfs.xfs` command looks like the following:

Small Configuration:

```
mkfs.xfs -d su=512k,sw=10 -l size=128m /dev/sdX
```

Medium Configuration:

```
mkfs.xfs -d su=512k,sw=20 -l size=128m /dev/sdX
```

Large Configuration (with LVM):

```
mkfs.xfs -l size=128m /dev/VolGroup/LogVol
```

In this configuration, `xfs` queries LVM, so no `su/sw` is needed.

When mounting an `xfs` file system, there are options to optimize the file system as well. The options used for the NSS are `noatime,allocsize=1g,nobarrier,inode64,logbsize=262144`. The list below explains these options.

- The `noatime` option tells the file system not to update the inode access time. This can be helpful if a lot of small files are being modified on the system. However, you also lose the ability to check when a file was last accessed (used) which is helpful if files need to be moved or removed.
- The `allocsize=1g` option tells `xfs` to speculatively extend allocation past the end of the file by up to 1 GB during delayed allocation. This reduces the amount of file fragmentation that occurs by ensuring large files do not interleave small extents when doing concurrent write back. The default speculative preallocation range is 64 KB.
- `nobarrier` tells `xfs` to disable barriers. Barrier support flushes of write-back cache at appropriate times. The PERC H800 has battery-backed write cache; therefore, the risk of data

loss when write barriers are disabled is minimized. In this case, barrier is disabled to increase performance and let the RAID controller handle the cache.

- There are two methods to configuring inodes in an `xfs` file system -- `inode32` and `inode64`. By default an `xfs` file system is mounted with `inode32`. In the `xfs` file system, the inodes reflect their location on disk, meaning that when the file system is mounted with `inode32`, all the 32-bit inodes will be in the first terabyte of space on a LUN. The `inode64` mount option allows `xfs` to create 64-bit inodes. Considering the large size of drives available today, when the inodes are at the beginning of the disk and the data blocks are spread across the rest of the disk, there is not much locality between the inodes and the data, thus reducing performance because of disk seeks. When the file system is mounted with `inode64`, `xfs` is able to place the inodes closer to the data, thus improving performance by reducing disk seeks. The drawback to mounting an `xfs` file system with `inode64` is that some backup programs and applications do not recognize 64-bit inodes which can cause issues when performing NFS sub-directory mounts of an `xfs` file system.
- The `logbsize` option can improve metadata performance as well by increasing the size of the log buffers. For details on how to configure `xfs`, see Appendix B.

LVM

In the case of the NSS Large configuration, two PERC H800 controllers each have two MD1200 enclosures attached. For each H800, the LUNs are set up in a RAID 60 (2 Span, 10+2) configuration. Then, LVM is used to create a logical volume that is striped across both LUNs to improve performance. By striping the data across both LUNs, the I/O can be distributed across both RAID controllers and all the disks. The LVM stripe size is set to 1024 k and the number of stripes is configured as 2. For instructions on how to configure LVM, see Appendix B.

TCP/IP

The networking stack has been tuned on the client side by increasing the default size of the TCP receive (`tcp_rmem`) buffers to 2.5 times the default `rsize` of the NFS mount. You can find the default `rsize` of the NFS mount by issuing `cat /proc/mounts` on a compute node (NFS client). The default size is 1048576 so the TCP receive buffers should be increased to 2621440 on the compute nodes. The increase helps improve read performance. For clients with 10GigE cards, the increase also enables jumbo frames and set the maximum transmission unit (MTU) value to 9000. If IPoIB is being used, the defaults for MTU are used. For instructions on how to configure `tcp_rmem` and jumbo frames, see Appendix B.

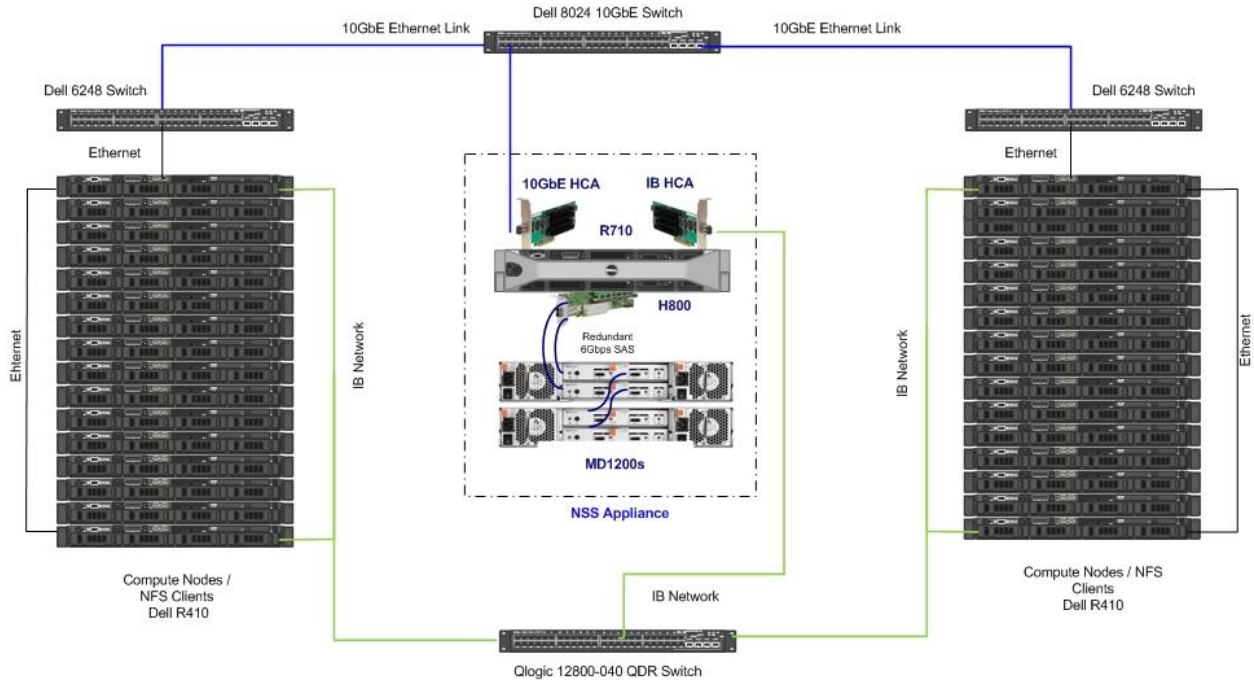
Performance Evaluation

This section focuses is on testing performance for metadata operations, random I/O Operations per Second (IOPs), large sequential reads and large sequential writes for each configuration. The performance of each configuration varies because of the number of disks and RAID controllers involved. The intent is to run the same tests on all three configurations to analyze how each configuration can handle the same workloads. The configurations are:

- NSS Small
- NSS Medium
- NSS Large

The cluster used in this study consists of 64 PowerEdge R410 servers as compute nodes that were used as NFS clients. See Figure 7 for a diagram of the test cluster used for this study. Note that the NSS Appliances were changed out to perform the comparisons. Tests were performed over both 10GigE and InfiniBand fabrics.

Figure 7. Cluster Configuration



Network Configuration

Ethernet

The NFS gateway server has an Intel X520 10 GB/s Dual Port SFP+ Ethernet adapter over which the NFS traffic flows. The NIC is plugged directly into the Dell PowerConnect™ 8024 10 GB/s Ethernet switch. There are two PowerConnect 6248 switches which are 48 port 1 GB/s Ethernet switches. The PowerConnect 6248 switches have 4 10 GB/s uplink modules so there is virtually no network oversubscription to the central core switch. All four of these uplink modules are connected to the PowerConnect 8024 and then aggregated together with 802.3ad. The switches also have their ports configured at the maximum MTU (jumbo frames). The NFS server and all compute nodes also have jumbo frames enabled with MTU size of 9000.

InfiniBand

In the InfiniBand configurations, the NFS gateway server has a Mellanox ConnectX-2 InfiniBand HCA. The Open Fabrics Enterprise Distribution (OFED) stack tested on both clients and NFS server is OFED 1.5.1. The clients also have Mellanox ConnectX-{1,2} QDR InfiniBand HCAs. All the compute nodes and the NFS server are plugged into a Qlogic 12800-040 QDR InfiniBand switch. For customers with an existing InfiniBand fabric, InfiniBand using IPoIB is an excellent alternative for NFS because it provides better performance than 1 GB/s or
10 GB/s Ethernet fabrics.

Compute Nodes
There are 64 R410 servers functioning as compute nodes and they are also the NFS clients. The compute nodes run PCM 2.0.1 Dell Edition which is based off of RHEL 5.5. The compute nodes have a 1 GB Ethernet connection to the PowerConnect 6248 switches. Each compute node also has a Mellanox ConnectX InfiniBand HCA that is used for IPoIB testing. All the compute nodes have 24 GB of memory.

Performance Benchmarks

The NSS configurations were tested to see how well they could handle the following I/O patterns:

- Large sequential writes
- Large sequential reads
- Random writes IOPs
- Random reads IOPs
- Metadata operations

The tests above were generated by using two industry standard tools: IOzone and mdtest. IOzone is used to perform the large sequential reads and writes as well as the random read IOPs and write IOPs. The version of IOzone used for this study is 3.327.

The large sequential reads and large sequential writes are conducted using 25 GB files and a request size of 1024 KB to ensure that the NFS server and client cache is saturated. The write patterns are N-N, meaning that each thread writes to its own file and there is no parallel I/O. IOzone was executed in clustered mode and one thread was launched on each compute node.

The IOPS testing was done with 4 KB record sizes since this is the size corresponding to most IOPS testing. It was found that the InfiniBand and 10GigE interfaces resulted in almost the same random IOPS performance (both read and write). Therefore, only the InfiniBand random read IOPS and random write IOPS are shown in this whitepaper.

The metadata tests were performed with `mdtest` version 1.8.3. The metadata tests include file and directory creates, stats, and removals. While these benchmarks do not cover every I/O pattern, they help to characterize what types of applications could benefit from the different NSS configurations.

InfiniBand Sequential Reads and Writes

Figure 8 shows that for large sequential writes running over InfiniBand (IPoIB), the greatest level of performance is achieved using the NSS Large configuration. The peak write throughput is about 1.45 GB/s and the large configuration sustains write performance better than the NSS Small and Medium configurations. The reason for this is because more disks and a second H800 were added for the Large configuration. The additions provide more processing power and more spindles to handle the load. The data shows that in all configurations, there is a drop in performance after a certain number of nodes. This is due to the fact that the MD1200's are populated with 2 TB NL SAS drives and the nature of the NFS concurrent write traffic causes the disks to become seek bound causing the decline in performance.

Figure 8. NSS IPoIB Sequential Writes – All 3 Configurations

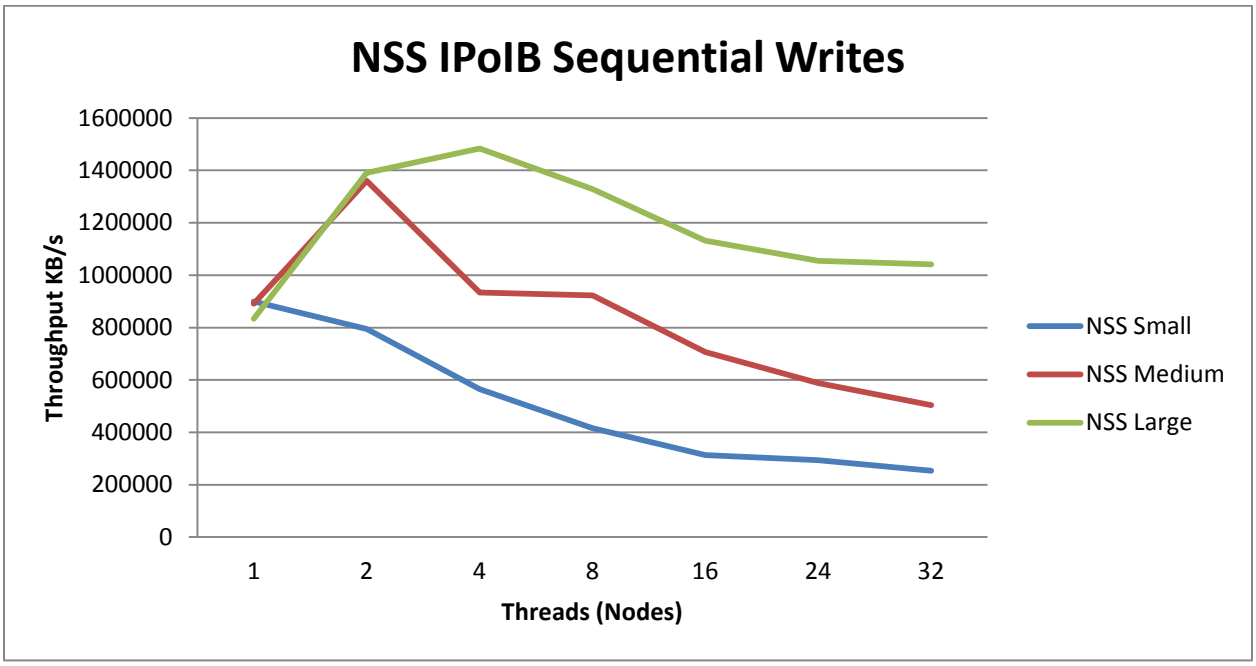
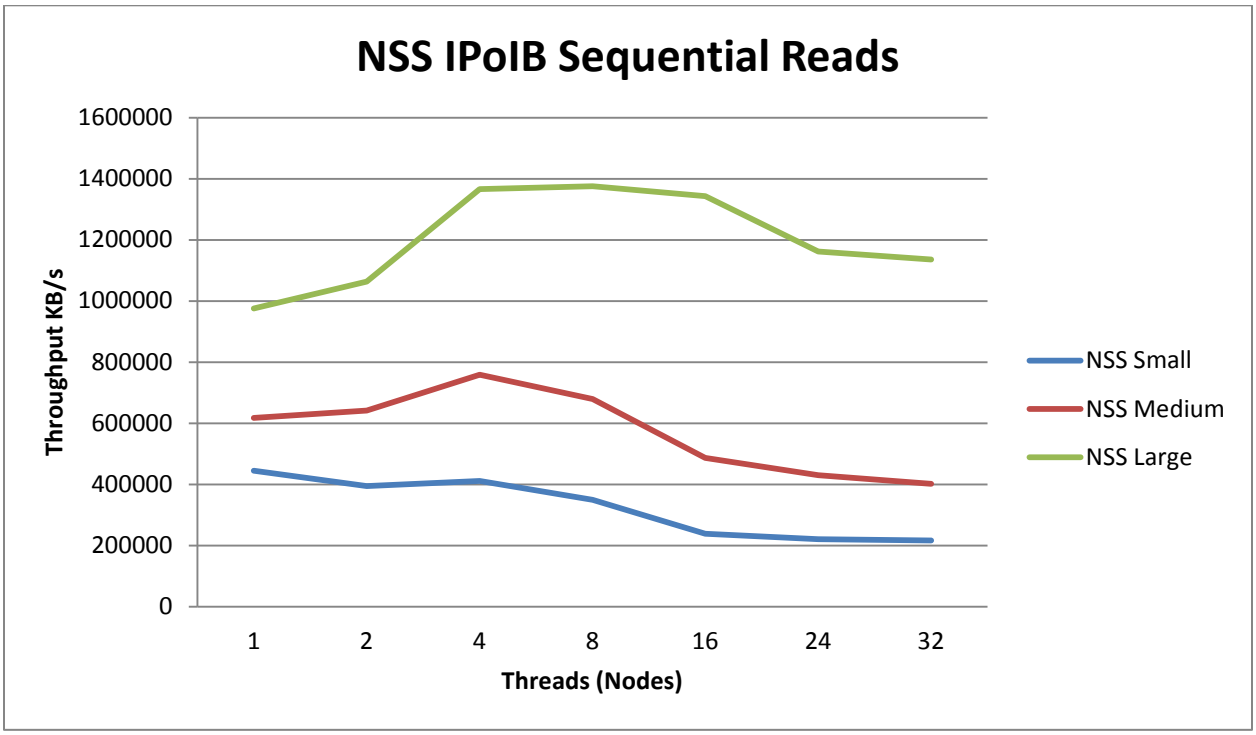


Figure 9 shows the large sequential read performance running over InfiniBand for each configuration for a range of nodes (NFS clients). The read performance drops slower than the writes because of the challenging nature of NFS write operations. This is a clear demonstration that as more disks are added to the solution the higher the peak read performance and a better level of sustained throughput is obtained.

Figure 9. NSS IPoIB Sequential Reads



10GigE Sequential Reads and Writes

Figure 11 shows that NSS Small can achieve a peak sequential write performance of approximately 550 MB/s and that the Large configuration peaks at approximately 1.2 GB/s. Furthermore, all three configurations show a decline in performance after a certain number of compute nodes – this was described during the review of the InfiniBand IPoIB writes. However, it is also clear that by increasing the number of drives and adding an additional controller, the peak write performance and the ability to sustain that performance is improved. Notice that the peak takes longer to achieve on 10GigE than on InfiniBand because of the increased number of nodes required to saturate the 10GigE fabric. The compute nodes in this test are using GigE connections instead of the IB connections in the IPoIB tests.

Figure 10. NSS 10GigE Sequential Writes

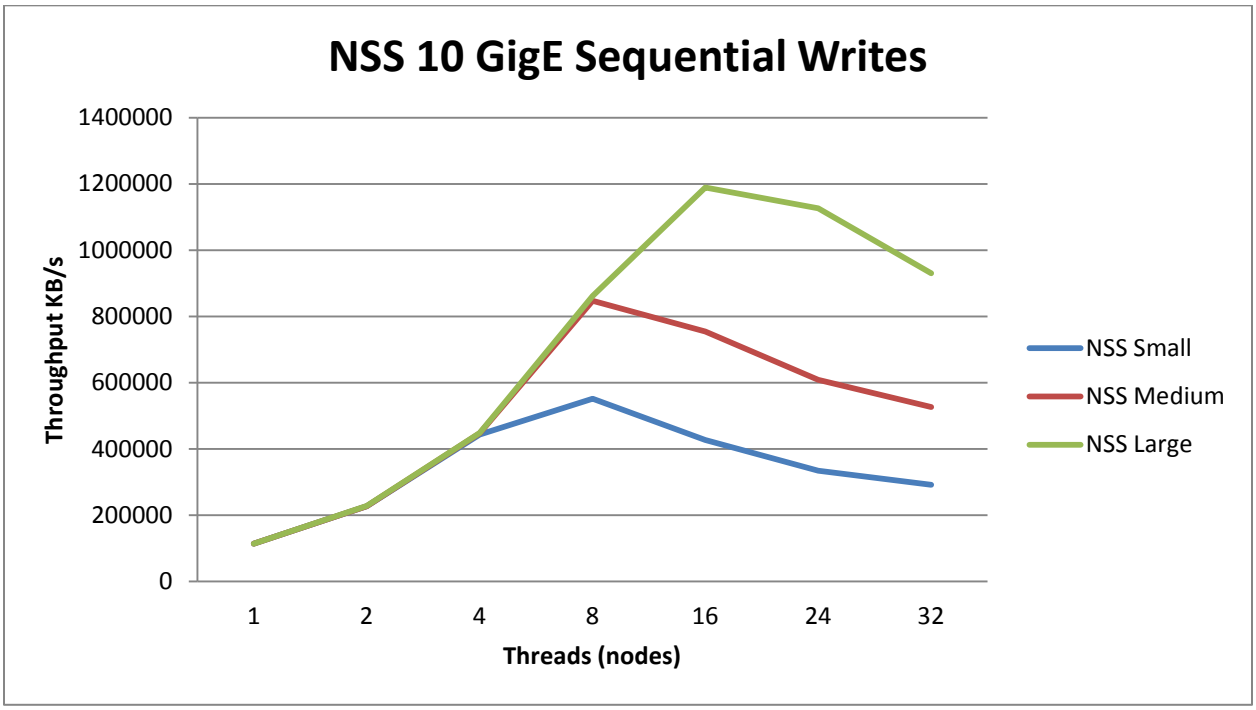
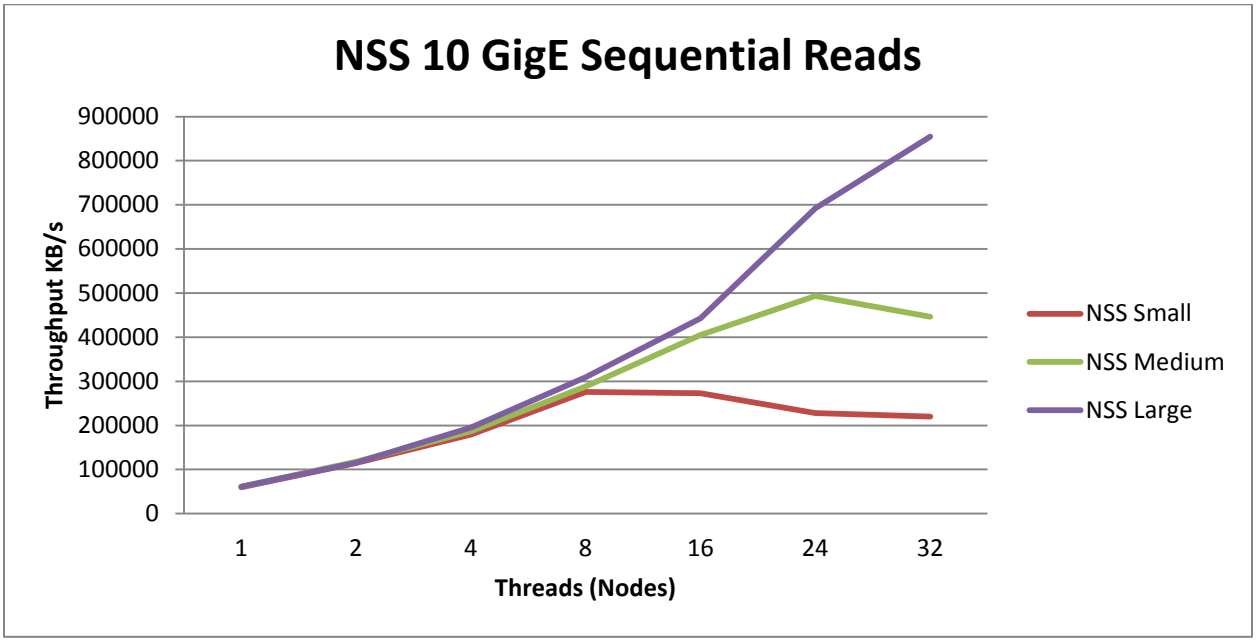


Figure 11 shows that the peak read performance and saturation point of NFS reads for the NSS Small configuration is approximately 270 MB/s. By adding one more MD1200 array creating the NSS Medium configuration, the peak performance and saturation point are approximately 500 MB/s. Finally, by adding an additional PERC H800 controller and 2 more arrays for creating the NSS Large configuration, IOzone is unable to show peak read performance or saturation point while running on only 32 nodes; more NFS clients would be needed to find this read performance saturation point.

Figure 11. NSS 10GigE Sequential reads



The write performance is higher than the read performance (shown in Figure 11) because of the cache involved in the NSS solution. The clients have 24 GB of RAM and the NFS server has 24 GB of RAM as well as 512 MB–1 GB of RAM on the PERC H800. When the clients issue the writes to the NFS server, the NFS server cache is able to take advantage of all of the memory and aggregates the writes into large sequential write and flushes that to disk in an optimized fashion and consequently, the performance improves. With a similar concept, sequential read is highly dependent on read ahead; however, the NFS service limits read ahead cache size such that the sequential read performance is reduced because the requests have to be satisfied from disk more frequently.

IPoIB Random Read IOPS and Random Write IOPS

Figure 12 shows the results of running IOzone to measure the random read IOPS for each configuration. For these tests, the request size was set to 4 k. To more accurately measure these IOPs, the server and client cache was minimized by using the NFS mount options described in Appendix B. Depending on the file system that is being tested this parameter may change. In this study, both InfiniBand IPoIB and 10GigE were tested and it was noticed that the results presented here are similar, so only the InfiniBand results are shown. Figure 12 shows that by adding another H800 controller and more disks, the peak performance of the NSS Medium configuration and the NSS Large configuration cannot be saturated with 32 nodes.

Figure 12. NSS IPoIB Random Read IOPS

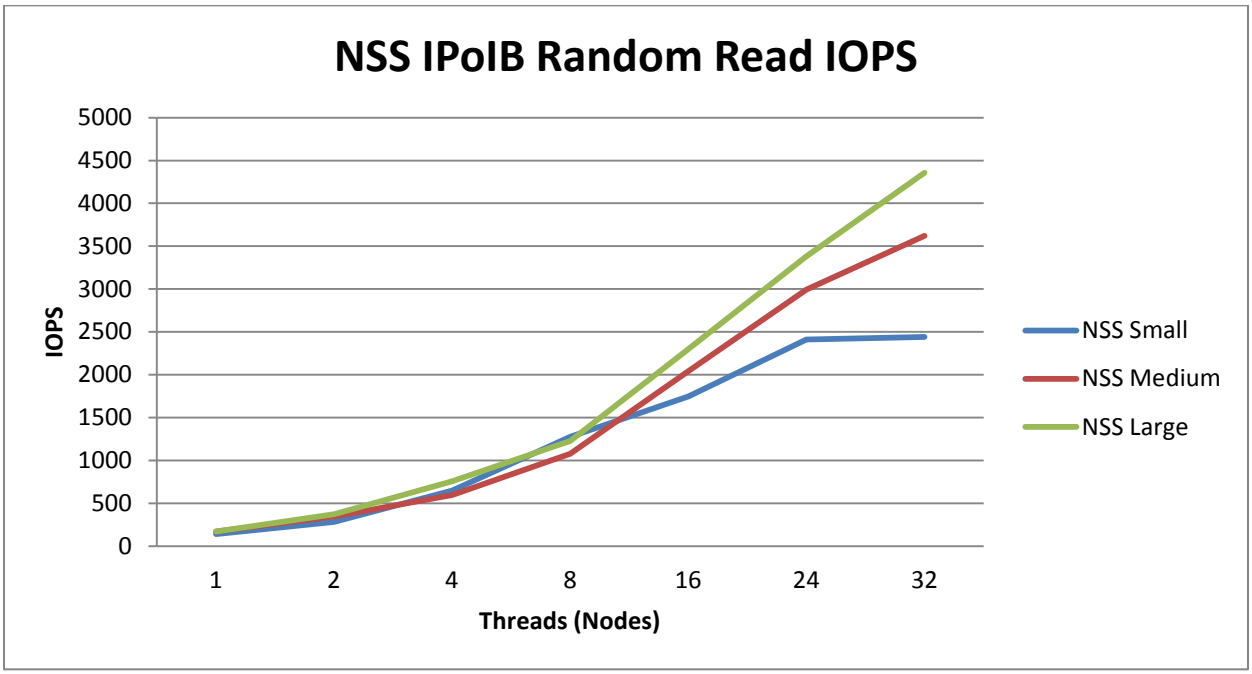
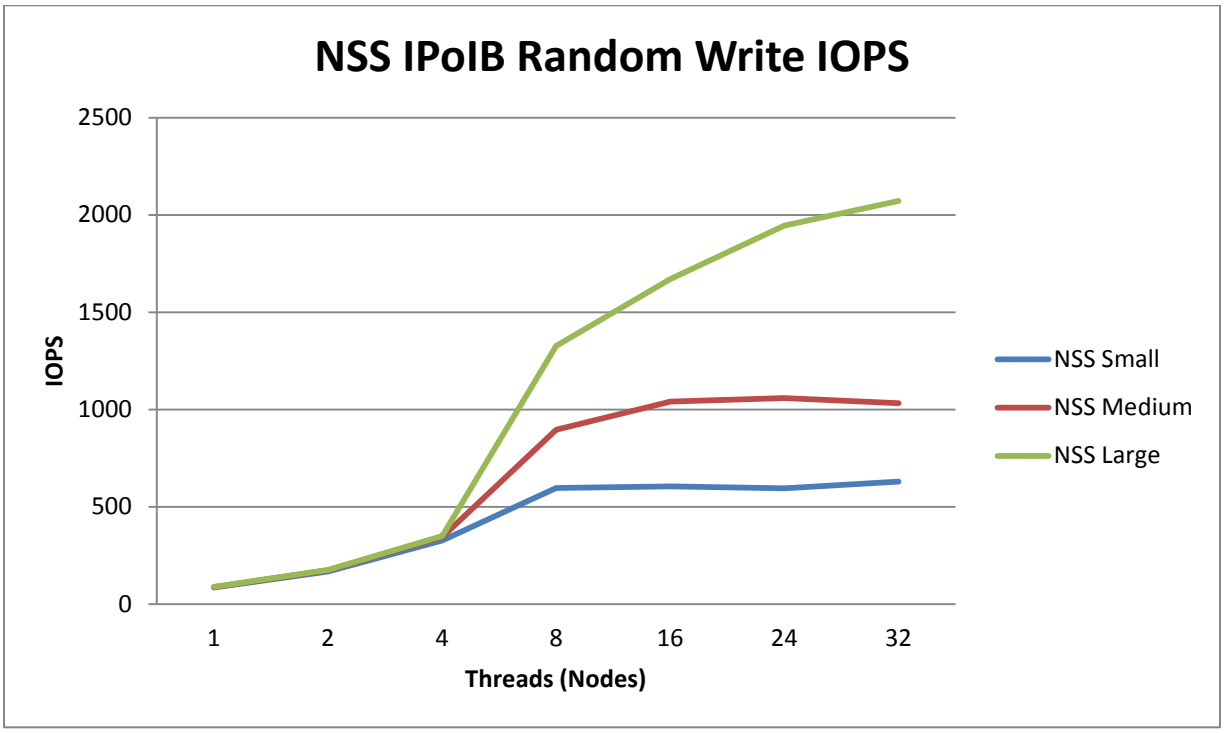


Figure 13 shows that for the random-write IOPS by adding an additional H800 controller and additional drives, the number of IOPs that can be generated doubles from NSS Medium configuration to the NSS Large configuration. This is due to the additional processing power of the extra controller and the extra disks.

When comparing Figures 12 and Figure 13, it is observed that the random access writes are slower than the random access reads. For this type of random workload, the server cache does not play as big a role in the performance as seen with sequential access.

Figure 13. NSS IPoIB Random Write IOPS – All 3 Configurations



IPoIB Metadata Tests

Figures 14 to 16 show the results of running `mdtest` to measure the metadata IOPs for each configuration, including file create operations, file stat operations and file unlink operations. One million files were created, stated and unlinked concurrently from multiple NFS clients on the NFS server. Between each metadata operation, both the client caches and the server cache were purged for accurate measurement. In this study, both InfiniBand IPoIB and 10GigE were tested and it was noticed that the metadata operations are not network bound, but disk bound. Thus, only the IPoIB results are presented in this paper. The results show that all three metadata operations are highly scalable in terms of both the number of NFS clients and the various configurations of backend storages. Specifically, adding additional MD1200 arrays and another H800 controller improves metadata performances under a high volume of concurrent requests, especially from eight clients or more.

Figure 14. NSS IPoIB Metadata - File Create

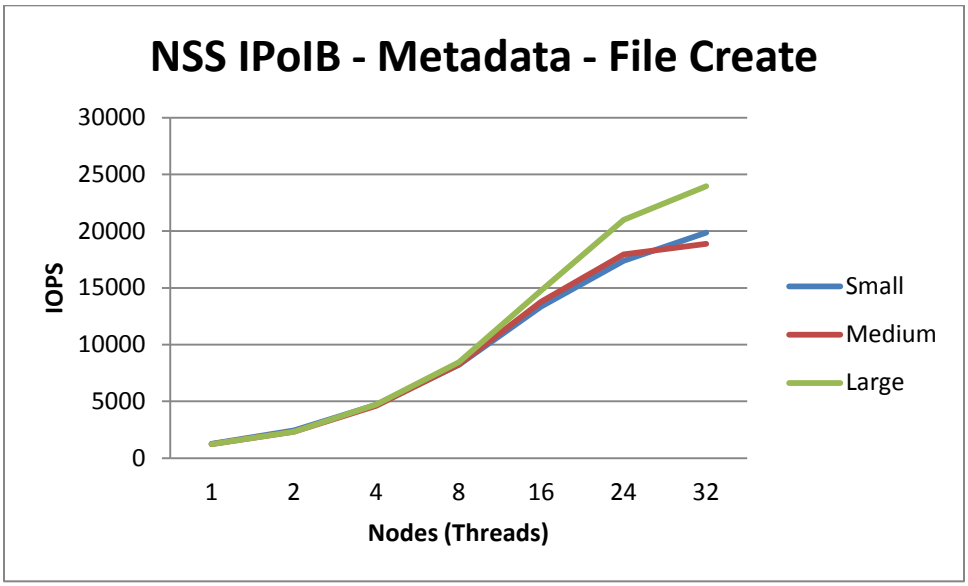


Figure 15. NSS IPoIB Metadata - File Stat

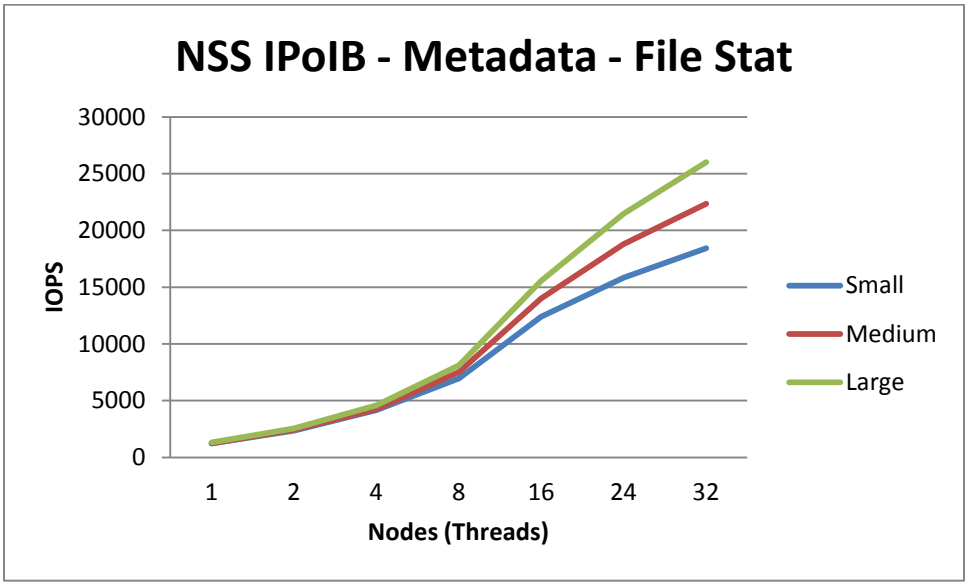
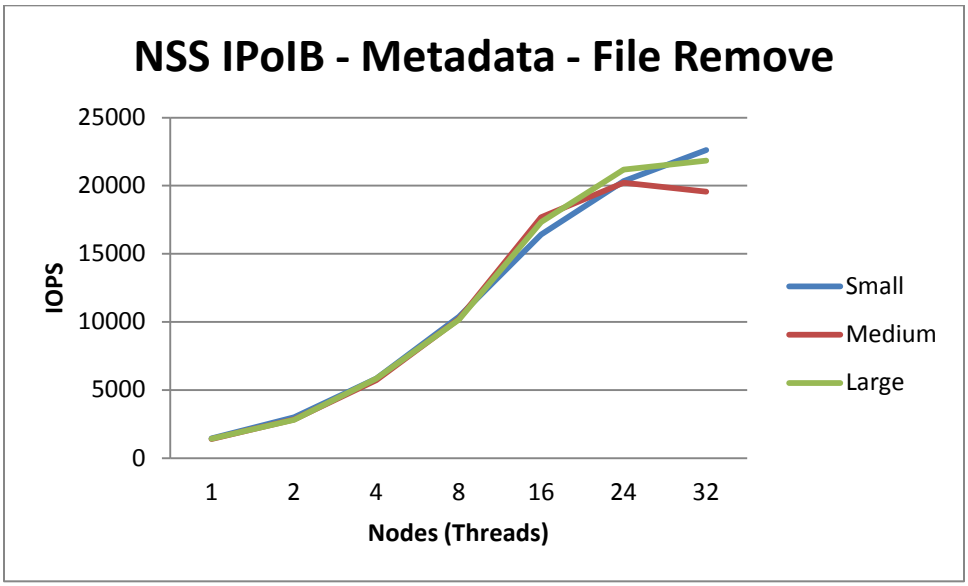


Figure 16. NSS IPoIB Metadata File Remove



Conclusion

This guide provides a reference point for customers who want to take advantage of Dell's expertise in architecting NFS storage solutions for HPC environments or to build their NFS solutions with Dell hardware using tested methodologies. The guidelines presented in this document provide information on how to choose Dell hardware and software components, configure the storage infrastructure, and then test it to make sure the NSS solution is optimized and integrated into the HPC (clustered) or standard Linux (standalone) environment. By combining the knowledge of the I/O characteristics of the application in use and the capabilities of the NSS solutions presented in this paper, making a decision about how much storage you need to perform at a certain level has never been easier.

The NSS solutions provide an easy way to deploy NFS configurations that can run on 10GigE or InfiniBand. With throughput ranging from 400 MB/s to 1.4 GB/s, the Dell HPC NFS Storage Solution provides a cost effective storage solution using industry standard hardware and software.

Appendix A. References

[1] Dell NFS Whitepaper demonstrating NFSD thread best practices:
http://i.dell.com/sites/content/business/solutions/whitepapers/en/Documents/hpc-pv-md1200-nfs.pdf

[2] Deadline Scheduler:

http://www.linuxjournal.com/article/6931?page=0,2

[3] XFS File System:

http://xfs.org/index.php/Main_Page

Appendix B. NSS Integration

Integrating Dell NSS into a High Performance Cluster

There are a few ways to deploy a NSS solution:

1. Configure from scratch.
2. Purchase server with OS pre-installed.
3. Provision NFS server with cluster middleware.

Required software:

- RHEL
  - `nfs-utils` (server and client)
  - `xfsprogs`
  - `xfsprogs-devel`
  - `xfsdump`
- OMSA 6.3
- OFED 1.5.1

Configuring the NFS Server (In a Stand Alone Environment)

To configure a NSS, here are the high-level steps with more detailed steps in the sections that follow.

For the Server:

1. Confirm network and H800 PCI cards are in the appropriate slots.
2. Insert drives for global hot spare and RAID 0.
3. Install the operating system.
4. Install the Red Hat Scalable File System XFS packages.
5. Install Dell OpenManage Server Administrator Version 6.3.
6. Configure global hot spare and RAID 0 for swap.
7. Configure the operating system.
8. Cable the MD1200 storage arrays to the R710.
9. Configure the LUN's on the MD1200 storage array and present to host.
10. Prepare the file system for export.

For the Compute Nodes (NFS Clients):

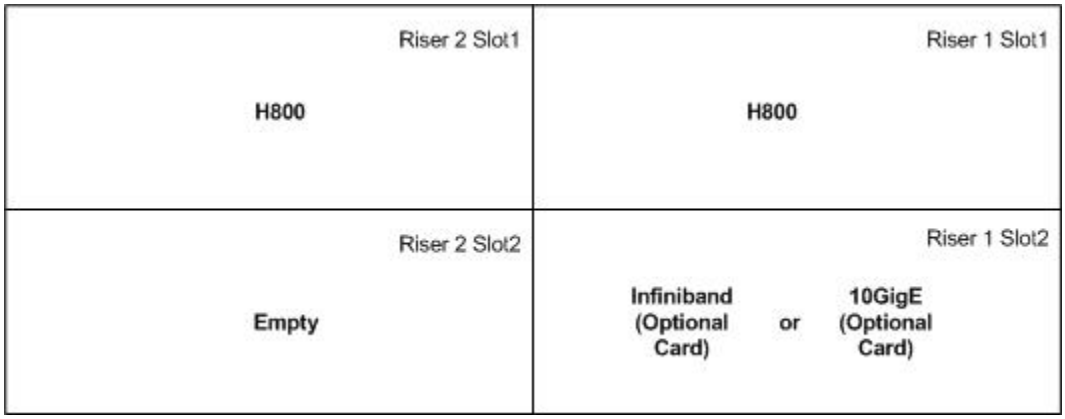
1. Configure the operating system.
2. Mount NFS share.

Confirming the PCI Slot Location
The PowerEdge R710 has 4 PCI slots. Riser 1 (Center Riser) has 2 x4 links and Riser 2 (Left Riser) has 2 x8 links.

- For NSS Small and Medium configurations:
  - H800 – Riser 1, Slot 1

- InfiniBand HCA Riser 1, Slot 2 OR
- Intel 10GigE NIC Riser 1, Slot 2

- For NSS Large Configuration:
  - H800 (1) – Riser 1, Slot 1
  - H800 (2) – Riser 2, Slot 1
  - InfiniBand HCA Riser 1, Slot 2 OR
  - Intel 10GigE NIC Riser 1, Slot 2



Inserting drives for OS, global hot spare and RAID 0

1. Upon delivery, there are two drives installed as a RAID 1 for the operating system. If the drives are not pre-installed, please configure drives in slot 0 and 1 as RAID1 for the operating system.
2. There should be 3 additional drives:
   - 2 for RAID 0 (OS Swap)
   - 1 for Global Hot Spare
3. Place the 2 drives for RAID 0 (OS Swap) in slots 2 and 3.
4. Place the drive for the global hot spare in slot 4.
5. These drives will be configured later with OMSA.

Installing the Operating System

1. Only required if the OS did not come pre-installed.
2. For RHEL, please see the installation guide for more information.

www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.5/html/Installation_Guide/index.html

Installing the Red Hat Scalable File System Packages

1. You will need an account on **rhn.redhat.com.**
2. Register the RHSFS key.
3. Use `yum` to install the RHSFS packages.

```
# yum -y install xfsprogs xfsprogs-devel xfsdump
```

Installing Dell OpenManage Server Administrator Version 6.3

1. Download OMSA from **support.dell.com** and place on the NFS server.
2. Extract the OMSA tarball.

```
# tar xzvf OM-SrvAdmin-Dell-Web-LX-6.3.0-2075_A00.7.tar.gz
```

3. Launch the install script.

```
# cd linux/supportscripts/
# ./srvadmin-install.sh
```

4. Accept the license.
5. Choose the appropriate components.

```
Components for Server Administrator Managed Node Software:

[x] 1. Server Administrator Web Server
[x] 2. Server Instrumentation
[x] 3. Storage Management
[ ] 4. Remote Enablement
[ ] 5. Remote Access Core Component
[ ] 6. Remote Access SA Plugin Component
[ ] 7. All

Enter the number to select a component from the above list.
Enter c to copy selected components to destination folder.
Enter i to install the selected components.
Enter r to reset selection and start over.
Enter q to quit.
```

6. Choose `I` to install.
7. Enter `y` to start the services

For more information on installing OMSA and its features, please refer to the documentation at: http://support.dell.com/support/edocs/software/svradmin/6.3/index.htm

Configuring Global Hot Spare and RAID 0 for Swap

The configuration of swap and global hot spare will be the same for all three configurations: NSS Small, Medium and Large.

1. To configure a global hot spare, view the current controllers, virtual disks and physical disks.
   Get the controller IDs

```
# omreport storage controller
```

   Get the current virtual disk layout

```
# omreport storage vdisk
```

   Get the IDs for the physical disks

```
# omreport storage pdisk controller=2
```

2. To create the RAID 0 swap:

```
# omconfig storage controller action=createvdisk controller=2 raid=r0
size=max pdisk=0:0:2,0:0:3
```

3. To create the global hot spare:

```
# omconfig storage pdisk action=assignglobalhotspare controller=2
pdisk=1:0:4 assign=yes
```

For more information on using OMSA CLI, please refer to
http://support.dell.com/support/edocs/software/svradmin/6.3/en/CLI/HTML/index.htm

Configuring the Swap Partition in OS

1. Look at current swap space.

```
# free -m
```

2. Create a swap space on the RAID 0.

```
# mkswap /dev/sdX
```

3. Turn on swap.

```
# swapon /dev/sdc
```

4. Look at new swap space.

```
# free -m
```

5. Place a swap entry in /etc/fstab (note the backslash below because the line wraps).

```
# cp /etc/fstab{,.orig}
# echo "/dev/sdc        swap         swap  defaults \    0 0" >>
/etc/fstab
```

Configuring the Operating System

Configuring the Scheduler

On the NFS server, change the scheduler from `cfq` to `deadline` globally for all devices by modifying `/etc/grub.conf` so it is persistent across reboots.

1. Make a backup of `/etc/grub.conf`.

```
# cp /etc/grub.conf{,.orig}
```

2. Add this to the end of the kernel line in `/etc/grub.conf`.

```
elevator=deadline
```

3. Reboot for this to take effect.

Configuring Networking for TCP/IP

If the NFS server has a 10GigE network card, modify the MTU setting to 9000 (replace “X” with the Interface number).

1. Configure the MTU.

```
echo "MTU=9000" >> /etc/sysconfig/network-scripts/ifcfg-ethX
```

2. Restart the networking services.

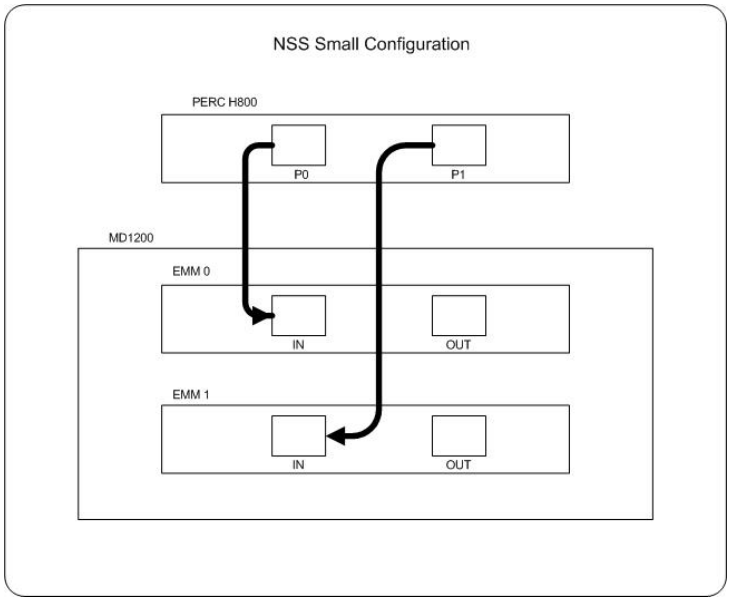
```
# service network restart
```

Cabling the MD1200 Storage Arrays to R710

NSS Small Configuration

1. Cable Port 0 on the H800 to the IN port on EMM 0 on the MD1200.
2. Cable Port 1 on the H800 to the IN port on EMM 1 on the MD1200.

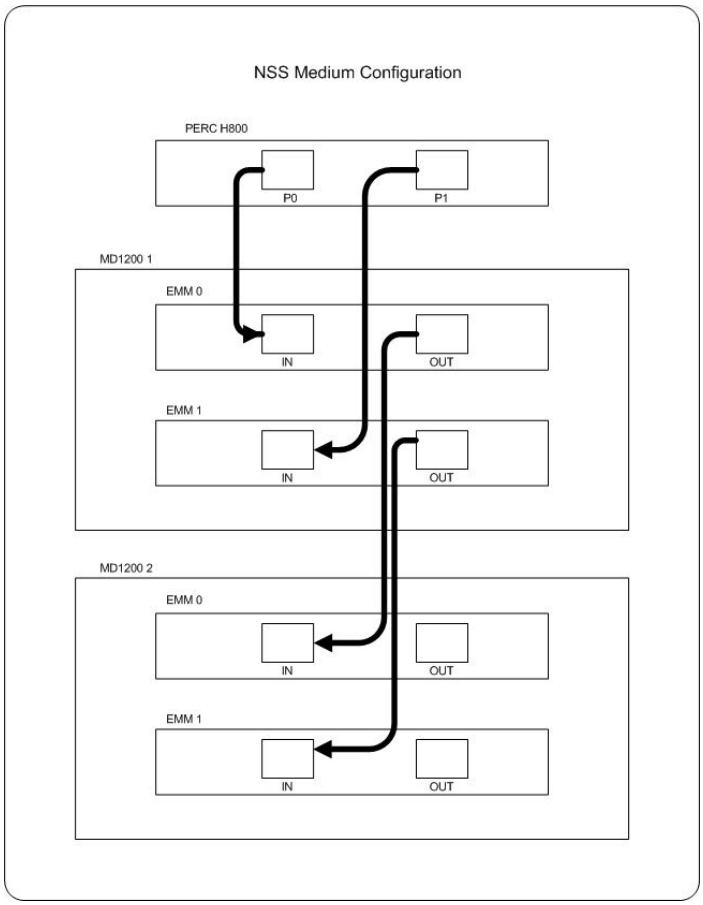
Figure 17. NSS Small Cabling



NSS Medium Configuration

1. Cable Port 0 on the H800 to the IN port on EMM 0 on the MD1200 1.
2. Cable Port 1 on the H800 to the IN port on EMM 1 on the MD1200 1.
3. Cable the OUT port on EMM 0 to the IN port on EMM 0 on MD1200 2.
4. Cable the OUT port on EMM 1 to the IN port on EMM 1 on MD1200 2.

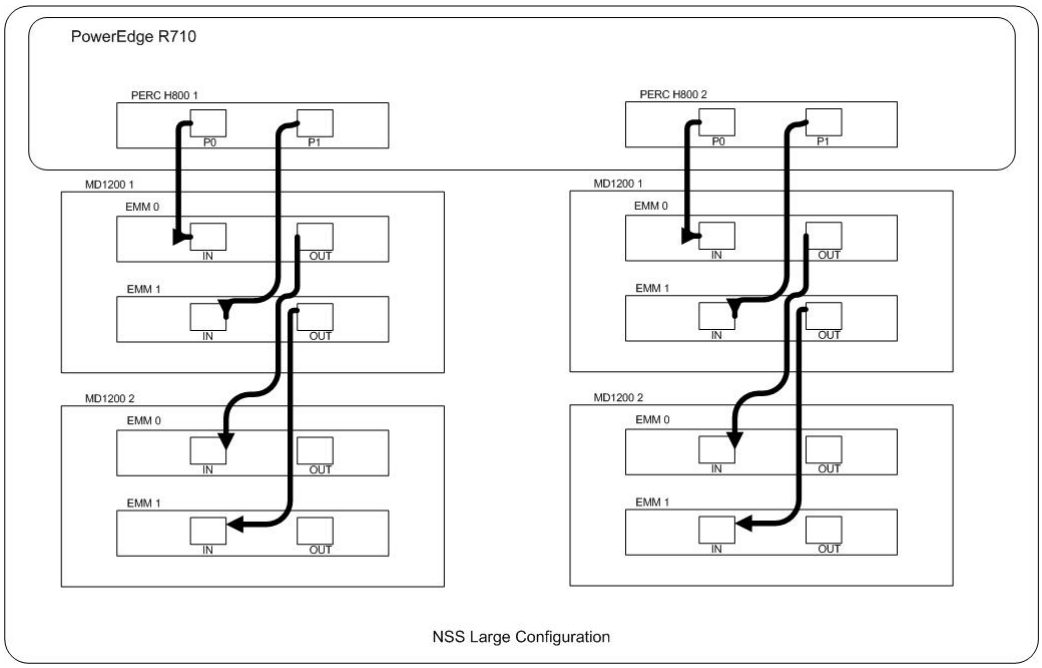
Figure 18. NSS Medium Cabling



NSS Large Configuration

1. Cable Port 0 on the H800 1 to the IN port on EMM 0 on the MD1200 1.
2. Cable Port 1 on the H800 1 to the IN port on EMM 1 on the MD1200 1.
3. Cable the OUT port on EMM 0 to the IN port on EMM 0 on MD1200 2.
4. Cable the OUT port on EMM 1 to the IN port on EMM 1 on MD1200 2.
5. Repeat steps 1 through 4 for the second H800 (H800 2).

Figure 19. NSS Large Cabling



Configuring the LUNs on the MD1200

To configure the LUNs on the MD1200 use `omconfig`.

1. NSS Small Configuration (R6, 10+2, Single Span)

```
omconfig storage controller action=createvdisk controller=2 raid=r6
size=max stripesize=512kb
pdisk=1:0:0,1:0:1,1:0:2,1:0:3,1:0:4,1:0:5,1:0:6,1:0:7,1:0:8,1:0:9,1:0:10
,1:0:11,1:0:12
```

2. NSS Medium Configuration (R60, 10+2, Two Span)

```
omconfig storage controller action=createvdisk controller=0 raid=r60
size=max stripesize=512kb
pdisk=0:0:0,0:0:1,0:0:2,0:0:3,0:0:4,0:0:5,0:0:6,0:0:7,0:0:8,0:0:9,0:0:10
,0:0:11,0:1:0,0:1:1,0:1:2,0:1:3,0:1:4,0:1:5,0:1:6,0:1:7,0:1:8,0:1:9,0:1:
10,0:1:11 spanlength=12
```

3. NSS Large Configuration (R60, 10+2, Two Span per Controller)

```
omconfig storage controller action=createvdisk controller=0 raid=r60
size=max stripesize=512kb
pdisk=0:0:0,0:0:1,0:0:2,0:0:3,0:0:4,0:0:5,0:0:6,0:0:7,0:0:8,0:0:9,0:0:10
,0:0:11,0:1:0,0:1:1,0:1:2,0:1:3,0:1:4,0:1:5,0:1:6,0:1:7,0:1:8,0:1:9,0:1:
10,0:1:11 spanlength=12

omconfig storage controller action=createvdisk controller=1 raid=r60
size=max stripesize=512kb
pdisk=0:0:0,0:0:1,0:0:2,0:0:3,0:0:4,0:0:5,0:0:6,0:0:7,0:0:8,0:0:9,0:0:10
```

```
,0:0:11,0:1:0,0:1:1,0:1:2,0:1:3,0:1:4,0:1:5,0:1:6,0:1:7,0:1:8,0:1:9,0:1:
10,0:1:11 spanlength=12
```

Preparing the File System for Export

LVM for NSS Large Configuration

Set up LVM with the following options:

1. Create the physical volumes.

```
pvcreate /dev/sd{b,c}
```

2. Create the volume group.

```
vgcreate VGName /dev/sdb /dev/sdc
```

3. Create the logical volume

```
lvcreate -i 2 -I 1024 -l 100%FREE VGName
```

Formatting the File System with XFS

1. Small Configuration – Format the LUN with xfs

```
mkfs.xfs -d su=512k,sw=10 -l size=64m /dev/sdX
```

2. Medium Configuration – Format the LUN with xfs

```
mkfs.xfs -d su=512k,sw=20 -l size=64m /dev/sdX
```

3. Large Configuration – Format the LUN with xfs

```
mkfs.xfs -l size=128m /dev/VGName/lvol0
```

Mount the File System

1. Mount the file system with the appropriate options. Create a mount point, if needed.

```
# mount -o noatime,allocsize=1g,nobarrier,inode64 /dev/sdX
/xfs/mnt/point
```

2. Backup `/etc/fstab` and then make the `xfs` mount options permanent by adding to `/etc/fstab`

```
# cp /etc/fstab{,.orig}

# echo "/dev/sdb /mnt/nfstop-r6-512-12-drives_1 xfs
rw,noatime,attr2,nobarrier,inode64,allocsize=1048576k,
logbsize=262144,noquota 0 0" >> /etc/fstab
```

Exporting the File System

1. Modify the exports file.

```
# cp /etc/exports{,.orig}

# echo "/path/to/share/ *(rw,async,no_root_squash)" >> /etc/exports

# service nfs restart
```

Configuring the Compute Nodes

Configuring the Operating System on the Client

1. Set the MTU on the client to 9000 and modify the `tcp_rmem` settings.

```
echo "MTU=9000" >> /etc/sysconfig/network-scripts/ifcfg-ethX
```

2. Add the following to `/etc/sysctl.conf`.

```
# increasing the default TCP receive memory size

net.ipv4.tcp_rmem = 4096 2621440 16777216
```

3. Activate the changes with `sysctl`.

```
# sysctl -p
```

Mount the NFS share

To configure the clients to persistently mount the NFS share:

1. Edit the `/etc/fstab` on the clients to make sure the mount is persistent on reboot (note the backslash in the following command because it wraps).

```
# echo "X.X.X.X:/path/to/nfs/share nfs
defaults,noac,rw,noatime,hard,intr 0 \ 0" >> /etc/fstab
```

Appendix C. Benchmarks Command Reference

IOzone

This section describes the commands used to benchmark the Dell NSS. IOzone can be downloaded from http://www.iozone.org/ and installed on the NFS server and all the compute nodes. For the random access patterns, this procedure was followed to minimize cache effects:

1. Unmount NFS share on clients.
2. Stop NFS on NFS server.
3. Unmount LUN on NFS server.
4. Mount LUN on NFS server.
5. Start NFS on NFS server.
6. Mount NFS Share on clients.

The IOzone tests were run from 1-32 nodes in clustered mode.

IOzone Sequential Writes

```
# /usr/sbin/iozone -i 0 -c –e –w –r 1024k –s 25g –t 32 -+n -+m ./clientlist
```

IOzone Sequential Reads

```
# /usr/sbin/iozone -i 1 -c –e –w –r 1024k –s 25g –t 32 -+n -+m ./clientlist
```

IOzone IOPs Random Access (Reads and Writes)

```
# /usr/sbin/iozone -i 2 –w -r 4k -I -O -w -+n -s 2G –t 32 -+m ./clientlist
```

Description of command line arguments:

IOzone Command Line Arguments	Description
`-i 0`	Write test
`-i 1`	Read test
`-i 2`	Random Access test
`-+n`	No retest
`-c`	Includes close in the timing calculations
`-t`	Number of Threads
`-e`	Includes flush in the timing calculations
`-r`	Records size
`-s`	File size
`-t`	Number of Threads

+m	Location of clients to run IOzone on when in clustered mode
-w	Does not unlink (delete) temporary file
-I	Use O_DIRECT, bypass client cache
-+n	No retests selected

By using -c and -e in the test, IOzone provides a more realistic view of what a typical application is doing. The O_Direct command line parameter allows us to bypass the cache on the compute node on which we are running the IOzone thread.

mdtest

mdtest can be downloaded from http://sourceforge.net/projects/mdtest/ and compiled and installed on a NFS share that is accessible from all compute nodes. mdtest is launched with mpirun so the MPI environment will need to be set up appropriately. For these tests, OpenMPI from the Mellanox OFED 1.5.1 kit was used. OFED was pushed out to the compute nodes with Platform PCM 2.0.1 middleware tools. As with the IOzone random access patterns, this procedure was followed to minimize cache effects during the metadata testing:

1. Unmount NFS share on clients.
2. Stop NFS on NFS server.
3. Unmount LUN on NFS server.
4. Mount LUN on NFS server.
5. Start NFS on NFS server.
6. Mount NFS Share on clients.

Metadata file and directory creation

```
# mpirun -np 32 --nolocal --hostfile ./hosts /nfs/share/mdtest -d
/nfs/share/filedir -i 6 -b 320 -z 1 -L -I 3000 -y -u -t -C
```

Metadata file and directory stat

```
# mpirun -np 32 --nolocal --hostfile ./hosts /nfs/share/mdtest -d
/nfs/share/filedir -i 6 -b 320 -z 1 -L -I 3000 -y -u -t -R -T
```

Metadata file and directory removal

```
# mpirun -np 32 --nolocal --hostfile ./hosts /nfs/share/mdtest -d
/nfs/share/filedir -i 6 -b 320 -z 1 -L -I 3000 -y -u -t -r
```

Description of command line arguments:

mdtest Command Line Arguments	**Description**
-np	Number of Processes
--nolocal	Instructs mpirun not to run locally
--hostfile	Tells mpirun where the hostfile is

-d	What directory `mdtest` should run in
-i	The number of iterations the test will run
-b	Branching factor of directory structure
-z	Depth of the directory structure
-L	Files only at leaf level of tree
-I	Number of files per directory tree
-y	Sync the file after writing
-C	Create files and directories
-R	Randomly stat files
-T	Only stat files and directories
-r	Remove files and directories left over from run