Dell HPC NFS Storage Solution

High Availability Configurations

A Dell Technical White Paper

Garima Kochhar, Xin Chen, Onur Celebioglu

Dell HPC Engineering

Version 1.1 Updated May 2011

May 2011

Contents

Executive Summary (updated May 2011)

This solution guide describes the Highly Available configurations of the Dell HPC NFS Storage Solution (NSS). Guaranteeing high availability (HA) of user data is becoming a common requirement in HPC environments. The HA configurations of Dell NSS improve availability of data using a pair of NFS gateway servers in an active-passive configuration to provide data access to the HPC compute cluster. The goal is to improve service availability and data integrity in the presence of possible failures or faults, and to maximize performance in the failure-free case. Described here are the architecture, performance and best practices for building such solutions.

Version 1.1 of the solution guide, dated May 2011, includes updated 10 Gigabit Ethernet performance results.

1. Introduction

Clusters have become one of the most popular architectures for High Performance Computing (HPC) today.(1) The disparity between the time taken by the processor and network to read or write data and the slower speed of storage devices makes the storage subsystem of a cluster, in most cases, a bottleneck that negatively impacts the overall system performance. Furthermore, cost-effective clusters are usually built with commodity servers and storage devices and component failures or faults cannot be completely avoided in such systems. Maintaining high levels of reliability and availability in such an environment is an important issue. A well-designed cluster must balance computational and I/O needs and take reliability and availability factors into account as well.

Parallel file systems have gained popularity on high performance clusters, especially for clusters that require high end storage (>300 TB) and have intensive I/O demands. This requirement is met by the Dell | Terascala HPC Storage Solution (DT-HSS) that offers a high throughput scale-out storage appliance based on the Lustre file system and Dell PowerVault storage arrays. (2) For HPC users who do not have intensive I/O workloads with MPI-I/O oriented applications and the need for high scalability, the Dell NFS Storage Solution (NSS) is an attractive option. The NSS solution uses the NFS file system on top of the Red Hat Scalable File System (XFS) with Dell PowerVault storage to provide an easy to manage, reliable, and cost-effective solution for unstructured data. (3)

With this solution guide, Dell has updated the current NSS offerings to include highly available options (referred to as NSS-HA configurations in the rest of this document). The goal is to improve on the service availability and data integrity of NSS configurations in the presence of possible failures or faults, and to minimize the performance loss in the failure free case. For example, when a cluster component fails, the impact to cluster clients is minimized and both system availability and user data integrity are maintained.

This solution guide provides details on such NSS-HA solution offerings. The following sections describe the NSS-HA architecture in detail with special focus on the high availability capability. Subsequent sections discuss the performance of the NSS-HA solution offerings. An extensive appendix with detailed steps on configuring an NSS-HA environment is included towards the end of this document.

The Dell NFS Storage Solution is delivered as an all-inclusive storage solution and is available with deployment services and full hardware and software support from Dell. The solution is designed using standards-based HPC components and focus on ease of deployment and ease of use.

Technical details of the solution are provided in subsequent sections of this solution guide while a summary of the NSS options available from Dell is provided below.

1.1. NSS Solution Offerings From Dell

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

Details of the offerings without HA are covered in the solution guide *"Dell NFS Storage Solution for HPC (NSS)".* Details of NSS offerings with HA are provided in this solution guide.

Small	Medium	Medium-HA
• 20 TB of usable space. • QDR InfiniBand or 10Gb Ethernet connectivity. • NFS Gateway Server: Dell PowerEdge R710 running Red Hat Enterprise Linux 5.5 and XFS File System. • Storage: Dell PowerVault MD1200 with twelve 2TB NL SAS drives. • 3 years of Dell PRO Support for IT and Mission Critical 4HR 7x24 onsite pack. • Dell deployment service to speed up installation, optimize performance and integrate with your environment.	• 40 TB of usable space. • QDR InfiniBand or 10Gb Ethernet connectivity. • NFS Gateway Server: Dell PowerEdge R710 running Red Hat Enterprise Linux 5.5 and XFS File System. • Storage: Two Dell PowerVault MD1200 with twelve 2TB NL SAS drives. • 3 years of Dell PRO Support for IT and Mission Critical 4HR 7x24 onsite pack. • Dell deployment service to speed up installation, optimize performance and integrate with your environment.	• 40 TB of usable space. • QDR InfiniBand or 10Gb Ethernet connectivity. • NFS Gateway Servers: Two Dell PowerEdge R710 running Red Hat Enterprise Linux 5.5, XFS File System and Red Hat Cluster Suite. • Storage: One Dell PowerVault MD3200 and one Dell PowerVault MD1200 enclosures, each with twelve 2TB NL SAS drives. • PowerConnect 5424 and two switched PDUs to manage high availability • 3 years of Dell PRO Support for IT and Mission Critical 4HR 7x24 onsite pack.

Large	Large-HA
• 80 TB of usable space. • QDR InfiniBand or 10Gb Ethernet connectivity. • NFS Gateway Server: Dell PowerEdge R710 running Red Hat Enterprise Linux 5.5 and XFS File System. • Storage: Four Dell PowerVault MD1200 with twelve 2TB NL SAS drives. • 3 years of Dell PRO Support for IT and Mission Critical 4HR 7x24 onsite pack. • Dell deployment service to speed up installation, optimize performance and integrate with your environment.	• 80 TB of usable space. • QDR InfiniBand or 10Gb Ethernet connectivity. • NFS Gateway Servers: Two Dell PowerEdge R710 running Red Hat Enterprise Linux 5.5, XFS File System and Red Hat Cluster Suite. • Storage: One Dell PowerVault MD3200 and three MD1200 enclosures, each with twelve 2TB NL SAS drives. • One PowerConnect 5424 and two switched PDUs to manage high availability • 3 years of Dell PRO Support for IT and Mission Critical 4HR 7x24 onsite pack.

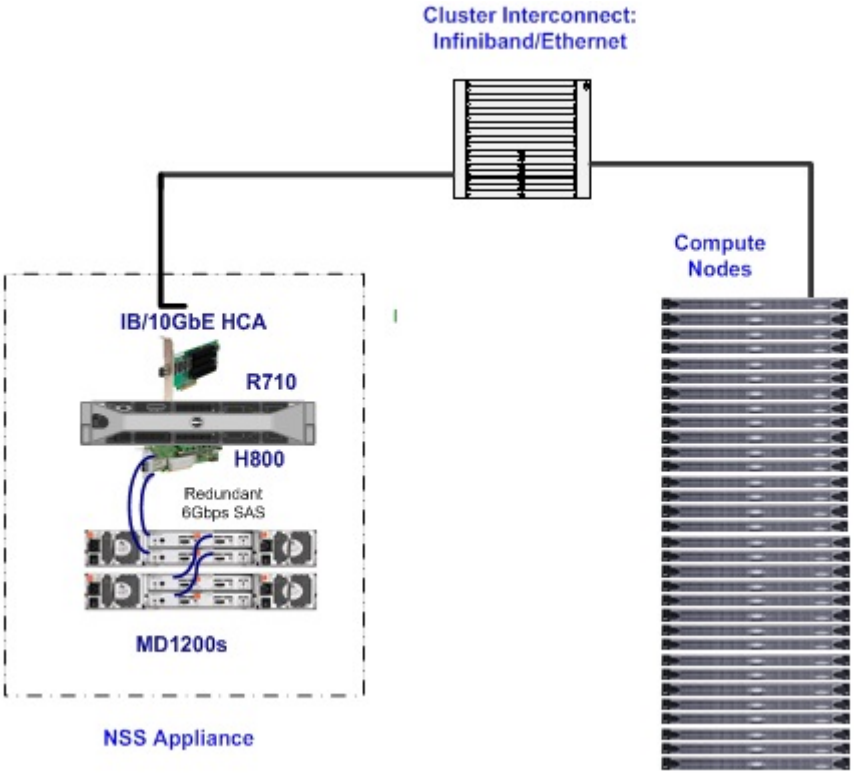
2. Dell NFS Storage Solution Technical Overview

This section provides a quick summary of the technical details of the Dell NFS Storage Solution (NSS) offerings without HA. Details on the solution's features, configuration and tuning guidance, performance characteristics and support are provided in the *Dell NFS Storage Solution for HPC (NSS)* solution guide [(3)].

The NSS offerings without HA consist of multiple PowerVault MD1200s attached to a PowerEdge R710 server. Each MD1200 storage array has twelve 2TB 7200rpm disks configured as a 10+2 RAID 6 virtual disk. The multiple RAID 6 LUNs are combined using RAID 60 and, optionally, Linux LVM, and then formatted as a Red Hat Scalable file system (XFS). This XFS file system is exported to the compute nodes via NFS. Compute nodes access the storage over the InfiniBand or 10 Gigabit Ethernet network.

The NSS is available in three configurations – Small, Medium and Large. These correspond to a 20TB, 40TB and 80TB solution respectively. Figure 1 shows an NSS Medium configuration with two PowerVault MD1200s.

Figure 1 - NSS-Medium Configuration



The NSS offerings with HA (NSS-HA) extend the NSS solution by introducing a high availability feature. It leverages the NSS building blocks (such as the server, software, storage and RAID configuration) and the performance tuning best practices as far as possible.

3. NFS Storage Solution with High Availability

The NFS Storage Solution with High Availability (NSS-HA) consists of a pair of Dell PowerEdge R710 servers that both have physical access to a shared PowerVault MD3200 storage enclosure that is extended with PowerVault MD1200 storage arrays. The servers are in active-passive configuration with only one server providing access to the data at any given time. Compute nodes as NFS clients access the data through the NSS-HA servers. The compute nodes are agnostic to which server owns the data path to the storage.

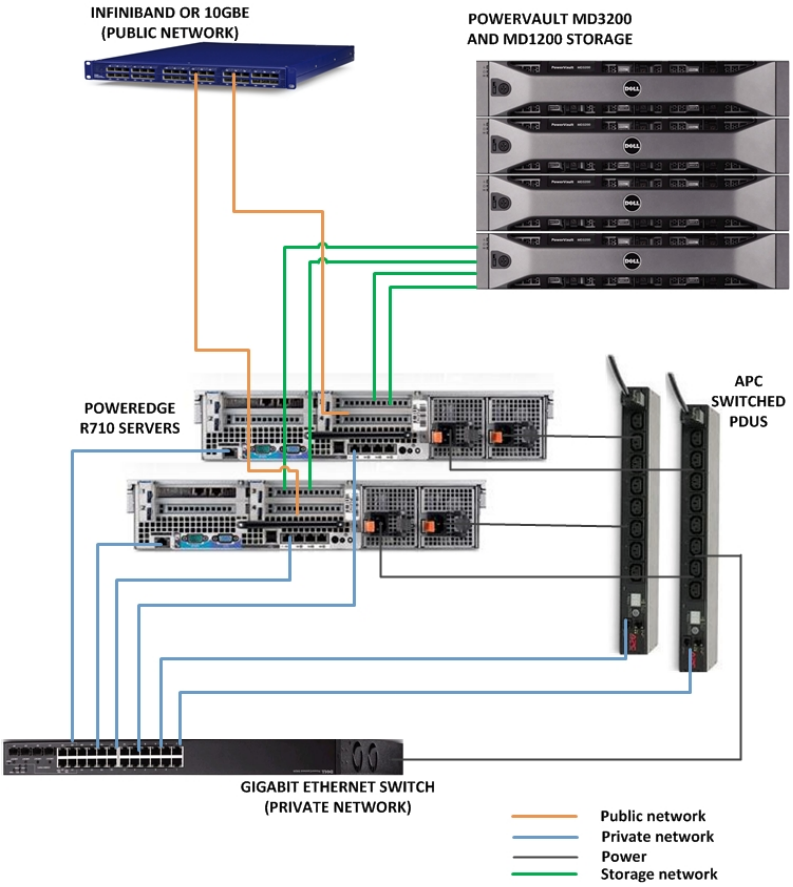
The HA functionality is provided through a combination of hardware and software. The following sections describe the architecture while *Appendix A: NSS-HA* provides detailed step by step instructions on setting up an NSS-HA environment.

3.1. Hardware

An NSS-HA configuration is shown in Figure 2. Hardware component redundancy is provided to achieve a high level of availability.

- Server Redundancy
  NSS-HA contains a pair of PowerEdge R710 servers. The two servers are configured in active/passive mode using the Red Hat Cluster Suite which will be described in later sections. In such a mode, when a server fails the other automatically takes over the service running on the failed server. Thus a single server failure does not cause loss of service, although a brief interruption (refer to Section 4.4) may occur while the failover is taking place.

Figure 2 - NSS-HA Configuration



Each PowerEdge R710 server has a Dell PERC H700 internal RAID controller and five local hard disks. Two disks are configured in RAID 1 with one disk designated as a hot spare for the operating system image. The two additional disks are configured in RAID 0 and used for swap space. Each server has a dual port Dell 6Gbps SAS HBA to connect to the external PowerVault MD3200 storage enclosure. Each server contains either an InfiniBand card or 10 Gigabit Ethernet card to connect to the compute nodes. The servers have an iDRAC enterprise management card for out-of-band systems management.

- Power redundancy

  Each server has dual power supplies. Each power supply in the server is connected to a different power bus via a power PDU, which avoids a single point of power supply failure. The configuration also includes two power PDUs for the two power supplies.

- Network redundancy

  The servers are connected to a Gigabit Ethernet switch that is used as the private network for communication between the active and passive servers. This network is used to monitor the heartbeat between the active and passive server. It is also used to communicate to the iDRAC and power PDUs.

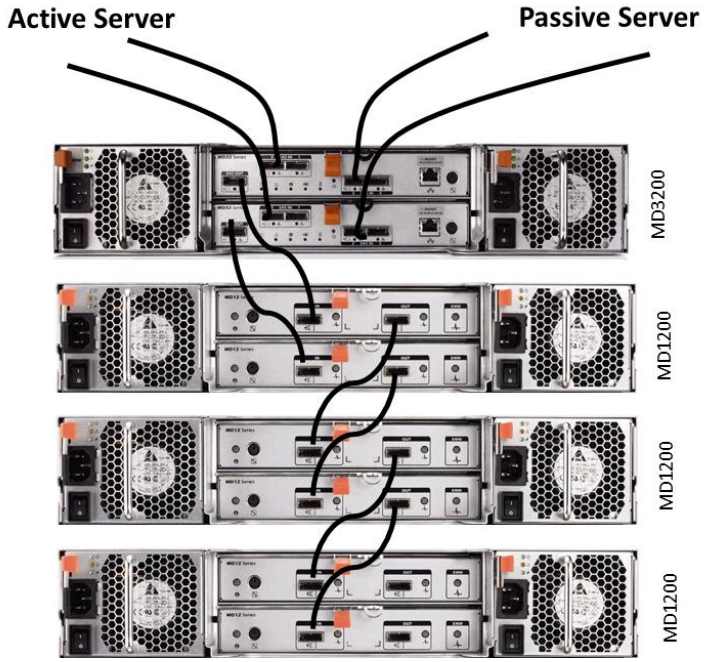
  The servers are also connected to an InfiniBand or 10 Gigabit Ethernet network. This is referred to as the public network. The compute cluster accesses the servers via this network. The public network switch is outside of the NSS-HA design. It is assumed that the reliability and configuration of the public network meets the high availability demands of the cluster. A single network card failure in one NFS server can be tolerated since there is a second server with an equivalent data path to the clients.

- Disk redundancy

  The storage for user data consists of a PowerVault MD3200 enclosure and one or more PowerVault MD1200 expansion arrays. Each array contains twelve 3.5" 2TB 7200rpm NearLine SAS disks. The disks in each array are set up in RAID 6, 10+2 configuration with 10 data disks and two parity disks. Such a RAID configuration provides sufficient redundancy to rebuild data when there is a disk failure. The segment size of each virtual disk is 512k. This gives each storage array a capacity of 20TB (10 data disks * 2TB per disk) usable space.

  The PowerVault MD3200 is configured to have read cache enabled, write cache enabled, write cache mirroring enabled and read prefetch enabled. The cache block size is set to 32k. Since the cache mirroring feature is enabled, the two RAID controllers in the MD3200 have mirrored caches. A single RAID controller failure can be tolerated with no impact to data availability.

Figure 3 - Example of PowerVault Storage Cabling



- I/O path redundancy

  Each server is connected to both controllers on the MD3200. Each server has two SAS cables directly connected to the MD3200, which eliminates a single point of server to storage I/O path failure. A redundant path from MD3200 to the MD1200 array is deployed to enhance the availability of the storage I/O path. The storage cabling is shown in Figure 3.

3.2. Software

The key software components in the solution are shown in Figure 4.

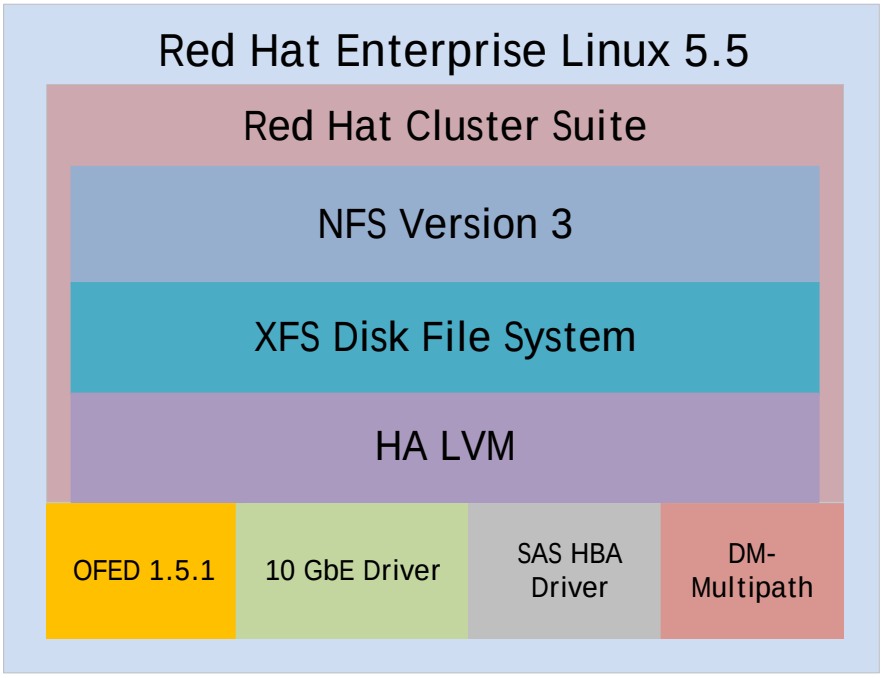
The NSS-HA runs on Red Hat Enterprise Linux 5.5 x86_64 (RHEL 5.5). The high availability component is provided by Red Hat Cluster Suite [4][5] which is an add-on feature available for RHEL 5.5.

The user data resides on a Red Hat XFS file system that is created on the shared storage provided by the Dell PowerVault storage arrays. The virtual disks on the shared storage are grouped using Linux Logical Volume Manager (LVM). This file system is exported to the clients or the compute nodes using the Network File System v3 (NFS).

The clients access the data over InfiniBand or 10 Gigabit Ethernet (10GbE). 10GbE drivers are native to RHEL 5.5. InfiniBand clusters need OFED 1.5.1 to be installed.

The PowerVault storage arrays are managed using the Dell Modular Disk Storage Manager (MDSM). The PowerEdge servers are managed using the Dell OpenManage Systems Administrator (OMSA).

Figure 4 - NSS-HA Software Components



3.3. High Availability Cluster

In the simple NSS solution, access to shared storage is provided through a single NFS server. A failure of the NFS server or a network failure can interrupt the client access to the shared storage

service impacting the availability of the entire system. In order to achieve better system availability, the NSS-HA solution extends NSS using the Red Hat Cluster Suite (RHCS).

RHCS-based clustering includes a high availability feature. A cluster service is configured such that a failure of any cluster member or cluster component does not interrupt the service provided by the cluster. Utilizing this high availability feature, a "HA cluster" is constructed in NSS-HA. The HA cluster consists of two PowerEdge R710 servers, and an HA service is defined which runs on one of the servers. To ensure data integrity, the HA service must run only on one cluster server at any time. If a failure occurs, the HA service will failover to the other PowerEdge R710 server while keeping the whole process transparent to the clients of the cluster as much as possible.

In the HA context, the word "cluster" refers to the pair of PowerEdge R710 servers and the RHCS software. This is distinct from the HPC cluster or the compute cluster which will be referred to as the clients. The word "server" refers to the PowerEdge R710 servers that make up the HA cluster.

An HA service is defined by a group of one or more cluster resources. All resources must be available for the cluster service to be up and running. When the service migrates from one server to another, all the resources migrate. Once defined and configured, the cluster resources are controlled solely by the HA cluster service and should not be manipulated outside of the HA cluster constructs.

In NSS-HA, the HA service comprises the following resources:

1) LVM - The LVM specifies the logical volume managed by the HA service. The virtual disks created on the storage arrays are grouped into a Linux logical volume. In NSS-HA, this LVM is configured with HA to ensure only one server has access to the LVM at a time.

2) File system – The LVM is formatted as an XFS file system. User data resides on this XFS file system. The HA service controls the mount options, mounting and unmounting of this file system.

3) NFS export – The NFS export resource ensures that NFS daemons are running. The XFS file system is exported to the clients over NFS. The HA service controls the NFS export options and client access.

4) Service IP – An IP address is associated with the HA service. In NSS-HA, clients access and mount the file system over NFS using this service IP. This IP is associated with the public network interface on the server currently running the cluster service.

5) Link monitoring – Link monitoring checks the status of the public network interface to which the service IP address is bound. A failed link will cause the cluster service to failover to the other server. This is an important component of the HA cluster since a failure on the public interface will prevent the clients from accessing the file system.

The cluster service can migrate or failover between the two servers in the HA cluster. But at any given time, only one server owns the HA service. Before a server (named "active") takes ownership of the HA service, it must determine that the second server (named "passive") is not running the service. This is to ensure that data is protected and there is never a situation when both servers are writing to the storage at the same time. The "active" server will start the HA service only if it

can first determine that the "passive" server is not providing the service. This is done by rebooting or "fencing" the "passive" server.

Since fencing is a critical component for the operation of the HA cluster, the NSS-HA solution includes two fence devices - the iDRAC and managed power distribution units (PDUs) – as previously described in the section on NSS-HA Hardware. When the "active" server is trying to fence the "passive", fencing is first attempted by logging into the "passive" server's iDRAC and rebooting the "passive" server. If that fails, the "active" server attempts to log into the APC PDUs and power cycle the power ports of the "passive" server. The "active" server tries these two fence methods in a loop till fencing is successful.

The active-passive HA philosophy is that it is better to have no server providing the HA service than to risk data corruption by having two active servers trying to access the same data volume. Therefore it is possible to have a situation when neither server is providing the cluster service. In this situation the system administrator will need to intervene and bring the cluster back to a healthy state where the clients can access the data.

3.4. Potential Failures and Fault Tolerance Mechanisms

The NSS-HA includes hardware and software components to build in HA functionality. The goal is to be resilient to several types of failures and transparently migrate the cluster service from one server to the other. This section discusses the NSS-HA response to potential failures. Detailed instructions on ***how*** to configure NSS-HA to tolerate these failures are provided in *Appendix A: NSS-HA* .

Assuming that the cluster service is running on the active server, Table 1 - NSS-HA Mechanisms to Handle Failure

lists types of failure and the behavior of the NSS-HA cluster when the failure occurs.

Table 1 - NSS-HA Mechanisms to Handle Failure

FAILURE TYPE	MECHANISM TO HANDLE FAILURE
Single local disk failure on a server	Operating system installed on a two-disk RAID 1 device with one hot spare. Single disk failure is unlikely to bring down server.
Single server failure	Monitored by the cluster service. Service fails over to passive server.
Power supply or power bus failure	Dual power supplies in each server. Each power supply connected to a separate power bus. Server will continue functioning with a single power supply.
Fence device failure	iDRAC used as primary fence device. Switched PDUs used as secondary fence devices.
SAS cable/port failure	Dual port SAS card with two SAS cables to storage. A single SAS port/cable failure will not impact data availability.
Dual SAS cable/port failure	Monitored by the cluster service. If all data paths to the storage are lost, service fails over to the passive server.
InfiniBand /10GbE link failure	Monitored by the cluster service. Service fails over to passive server.

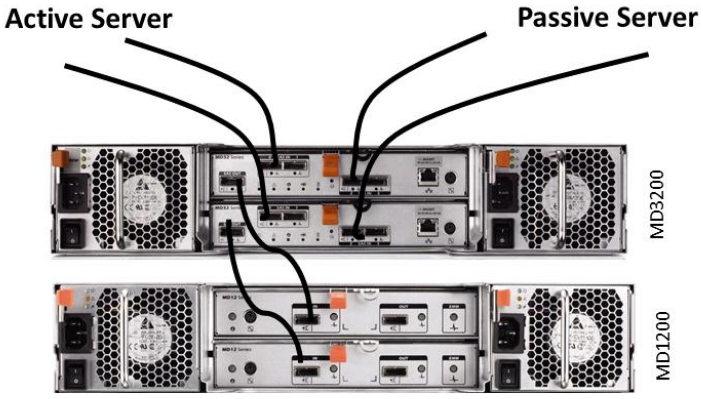
FAILURE TYPE	MECHANISM TO HANDLE FAILURE
Private switch failure	Cluster service continues on the active server. If there is an additional component failure, service is stopped and system administrator intervention required.
Heartbeat network interface failure	Monitored by the cluster service. Service fails over to passive server.
RAID controller failure on MD3200 storage array	Dual controllers in MD3200. The second controller handles all data requests. Performance may be degraded but functionality is not impacted.

3.5. NSS-HA Solution Configurations

The NSS-HA solution is available in two configurations – Medium and Large. The Medium configuration provides 40TB of usable storage, while the Large configuration scales to 80TB of usable storage.

Both configurations consist of a pair of PowerEdge R710 servers connected to a single PowerVault MD3200 storage array. The Medium and Large configurations differ only in the capacity of storage. The Medium configuration uses one PowerVault MD1200 expansion array to extend the MD3200 while the Large configuration uses three MD1200 arrays. Figure 3 shows the Large configuration; Figure 5 shows the Medium configuration. *Appendix A: NSS-HA* contains detailed instructions on setting up a Medium or a Large configuration.

Figure 5 - NSS-HA Medium Storage Configuration



3.6. Upgrade from Medium Configuration to Large Configuration

The NSS-HA architecture described in this white paper and the design of the PowerVault storage arrays makes it easy to upgrade a Medium NSS-HA configuration to a Large NSS-HA configuration. The upgrade process involves addition of two PowerVault MD1200 storage arrays to the NSS-HA Medium cluster. The new storage arrays must be configured to create virtual disks. The cluster services, switches and configuration remain the same.

There are two methods to upgrade the solution:

1) Add capacity, same performance.
   In this method user data is preserved during the upgrade. The final configuration provides 80TB of capacity, but the performance is similar to a Medium configuration.

2) Add capacity, improved performance
   In this second method all user data must be backed up. The upgrade will wipe out the existing Medium configuration and create a Large configuration.

Performance of the two configurations is discussed in the section on *Performance Benchmark Results*. *Appendix B: Medium to Large Configuration Upgrade* contains detailed instructions on the two upgrade methods.

4. Evaluation

The architecture proposed in this white paper was evaluated in the Dell HPC lab. This section describes the test methodology and the test bed used for verification. It also contains details on the functionality tests. Performance results are presented in subsequent sections.

4.1. Methodology

The verification focused on three key areas – functionality, data correctness and performance.

Different component failures were simulated to test the failover of the cluster service. Apart from hardware redundancies like dual power supplies, dual SAS links to the storage, dual PDUs described in previous sections, the HA cluster is designed to automatically detect and tolerate several types of component failures.

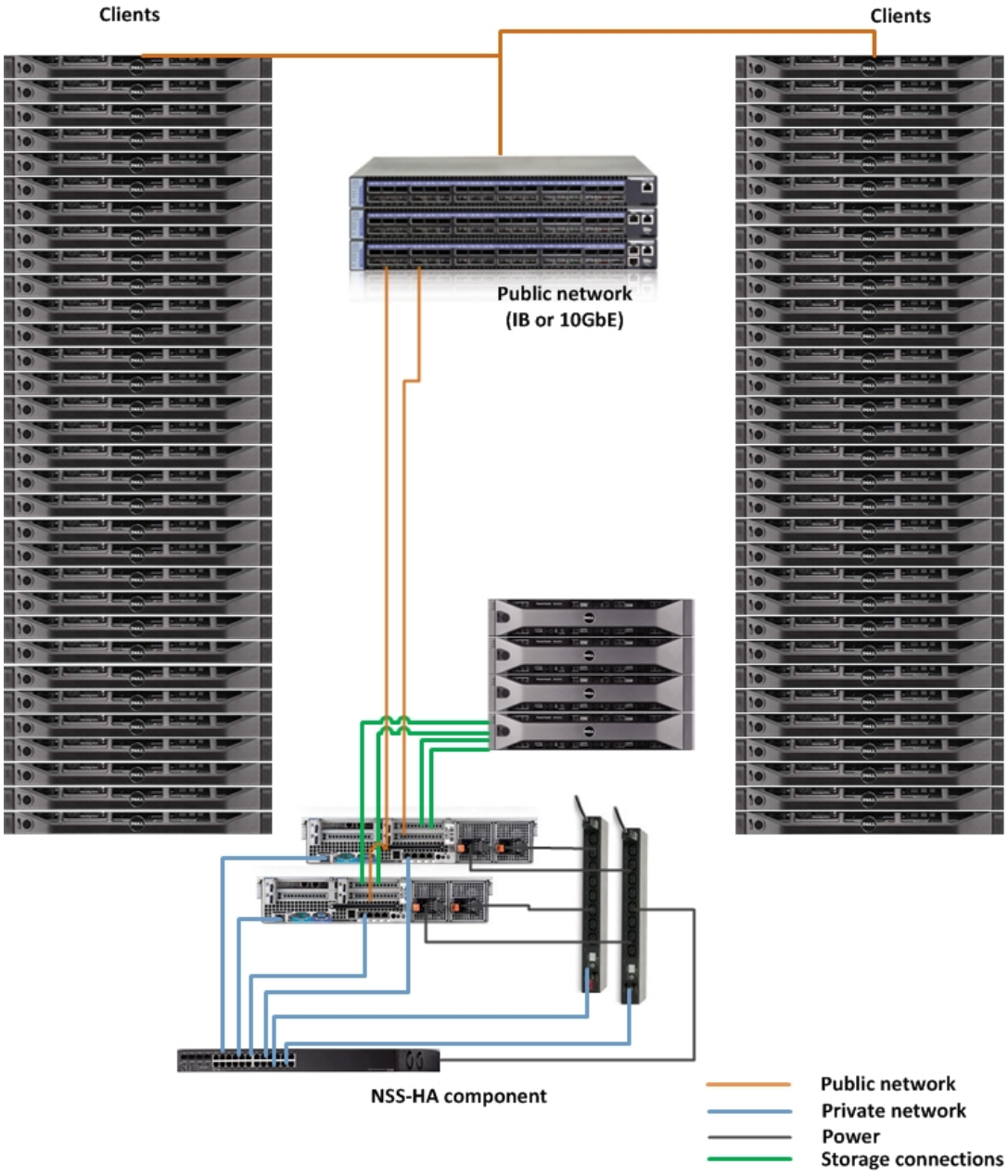
There are likely to be active data connections with clients reading and writing from the storage during a failure scenario. Ensuring data correctness during a failover operation is a critical requirement.

Performance testing of the solution was performed under a no failure case to understand the maximum capability of the solution.

4.2. Test Bed (updated May 2011)

The test bed used to evaluate the functionality and performance of the NSS-HA solution is described here. Figure 6 shows the test bed used in this study.

Figure 6 – Test Bed Configuration



Two PowerEdge R710 servers were used as the NFS gateway servers. Both servers were connected to PowerVault MD3200 storage extended with PowerVault MD1200 arrays. The Medium configuration used one MD1200 array while the Large configuration used three MD1200s. A PowerConnect 5424 Gigabit Ethernet switch was used as the private HA cluster network between the servers. APC switch power PDUs were used for the HA cluster.

The HPC compute cluster consisted of 64 PowerEdge R410 servers deployed using Platform Cluster Manager – Dell Edition version 2.0.1(6).

Table 2, Table 3, Table 4, and Table 5 give details of the configuration. In this test bed, flow control was disabled on the PowerConnect 8024 switch and two PowerConnect 6248 switches listed in Table 5.

Table 2 – NSS-HA Hardware Configuration Details

SERVER CONFIGURATION	
NFS Gateway Server Model	Two PowerEdge R710
Processor	Dual Intel Xeon E5630 @ 2.53GHz
Memory	12 * 4GB 1333MHz RDIMMs
Local disks and RAID controller	PERC H700 with five 146GB 10K SAS hard drives
Optional InfiniBand HCA (slot 4)	Mellanox ConnectX-2 QDR PCI-E card
Optional 10 Gigabit Ethernet card (slot 4)	Intel X520 DA 10Gb Dual Port SFP+ Advanced
External storage controller (slot 3)	6Gbps SAS HBA
Systems Management	iDRAC Enterprise
Power Supply	Dual PSUs
STORAGE CONFIGURATION	
Storage Enclosure	One PowerVault MD3200, one or three MD1200 expansion arrays High Performance Tier feature enabled on the PowerVault MD3200
RAID controllers	Duplex RAID controllers in the MD3200
Hard Disk Drives	Twelve 2TB 7200 rpm NL SAS drivers per array
OTHER COMPONENTS	
Private Gigabit Ethernet switch	PowerConnect 5424
Power Distribution Unit	Two APC switched Rack PDUs, model AP7921

Table 3 - NSS-HA Software Configuration Details

SOFTWARE	
Operating System	Red Hat Enterprise Linux (RHEL) 5.5
Kernel version	2.6.18-194.el5 x86_64
Cluster Suite	Red Hat Cluster Suite from RHEL 5.5
File system	Red Hat Scalable File System (XFS) 2.10.2-7
Systems Management	Dell OpenManage Server Administrator 6.4.0
Storage Management	Dell Modular Disk Storage Manager 1.2.0.12

Table 4 - NSS-HA Firmware and Driver Configuration Details

FIRMWARE AND DRIVERS	
PowerEdge R710 BIOS	2.2.10
PowerEdge R710 iDRAC	1.54
InfiniBand firmware	2.8.00
InfiniBand driver	Mellanox OFED 1.5.1
10 Gigabit Ethernet driver	ixgbe 2.0.44-k2
PERC H700 firmware	12.10.0-0025
PERC H700 driver	megaraid_sas 00.00.04.17-RH1
6Gbps SAS firmware	07.01.33.00
6Gbps SAS driver	mpt2sas 01.101.06.00

Table 5 - NSS-HA Client Configuration Details

CLIENTS / HPC COMPUTE CLUSTER	
Clients	64 PowerEdge R410 compute nodes
InfiniBand	Mellanox ConnectX-2 QDR HCAs Mellanox OFED 1.5.1
InfiniBand fabric	All clients connected to a single large port count InfiniBand switch. Both R710 NSS-HA servers also connected to the InfiniBand switch.
Ethernet	Onboard 1 GbE Broadcom 5716 network adapter. bnx2 driver v2.0.8e
Ethernet fabric	Two sets of 32 compute nodes connected to two PowerConnect 6248 Gigabit Ethernet switch. Both PowerConnect 6248 switches uplinked to a 10GbE PowerConnect 8024 switch. Both R710 NSS-HA servers connected directly to the PowerConnect 8024 switch. Flow control was disabled on the PowerConnect 8024 switch and two PowerConnect 6248 switches.

4.3. Design Choices that Impact Functionality and Performance

The HA cluster configuration includes several design choices that impact functionality and performance. These choices are described in this section. Details on configuring an NSS-HA solution are provided in *Appendix A: NSS-HA* .

Storage array configuration

1) Storage arrays are configured with twelve 2TB NearLine SAS disks. The 3.5” 2TB disks provide large capacity at a cost-effective price point.

2) Virtual disks are created using RAID 6, with 10 data disks and 2 parity disks. This RAID configuration provides a good balance between capacity and reliability to tolerate multiple disk failures. (7)

3) Virtual disks are created with a segment size of 512k to maximize performance (7). This value should be set based on the expected application I/O profile for the cluster.

4) Cache block size is set to 32k to maximize performance. This value should be set based on the expected application I/O profile for the cluster.

5) The RAID controllers read and write caches are enabled to maximize performance by allowing the controller to prefetch data into the caches based on data access patterns. (8)

6) Write cache mirroring is enabled between the two PowerVault MD3200 RAID controllers to protect data if there is a controller failure. Cache mirroring between the controllers ensures that the second controller can complete the writes to disk. (7) This design choice does have a performance penalty. With cache mirroring enabled, writes are 20-25% slower from server direct to storage when compared to the case where cache mirroring is disabled. Cache mirroring has little to no impact on read performance.

NFS Server configuration

7) The XFS file system is mounted with the **wsync** option. By default XFS uses asynchronous journaling to prevent the journaling subsystem from slowing down operations. This means that each operation is signaled as complete before the transaction recording the change has been written to disk. If the HA service on the NFS server fails over before an operation is completed to disk it can lead to an inconsistent view between the NFS client and the actual data on the storage. To avoid this problem, the XFS file system needs to run transactions synchronously and this is enforced by the wsync option. This reduces XFS metadata performance due to the increase in journal I/O, but if the server crashes and the HA service fails over to the passive server, the passive server will recover the transactions correctly.

8) The XFS file system is exported using the NFS **sync** option. This ensures that the client waits for each write operation to complete at the NFS server, thus, data integrity can be guaranteed as much as possible in the event of a server failure. The **sync** option does have a performance penalty but is an absolute requirement to ensure data integrity.

9) Increased number of concurrent NFS threads (from a default of 8 to 256) on the servers to maximize performance.(3)

10) Change in default OS scheduler from **cfq** to **deadline** to maximize performance. (3)

11) Set large MTU (MTU=8192) for 10 Gigabit Ethernet networks to maximize performance. (3)

4.4. Functionality Tests

To test the failover of the HA cluster service, several component failures were simulated.

1) Server failure
2) Heartbeat link failure
3) Public link failure

4) Private switch failure
5) Fence device failure
6) One SAS link failure
7) Multiple SAS link failures

This section describes the NSS-HA response to failures. Details on how to configure the solution to handle these failure scenarios are provided in *Appendix A: NSS-HA* .

Server response to a failure

Server response was recorded in how the HA cluster responds to a failure event. Time to recover from a failure was used as a performance metric. Time was measured from the point when the fault was injected in the server running the HA service (active) till the service was migrated and running on the other server (passive).

1) Server failure - simulated by introducing a kernel panic.
   When the active server fails, the heartbeat between the two servers is interrupted. The passive server waits for a defined period of time and then attempts to fence the active server. The default timeout period before a server is declared as dead is 10 seconds. This parameter is tunable. Once fencing is successful, the passive server takes ownership of the cluster service.

Figure 7 - Failover Procedure in Case of a Server Failure

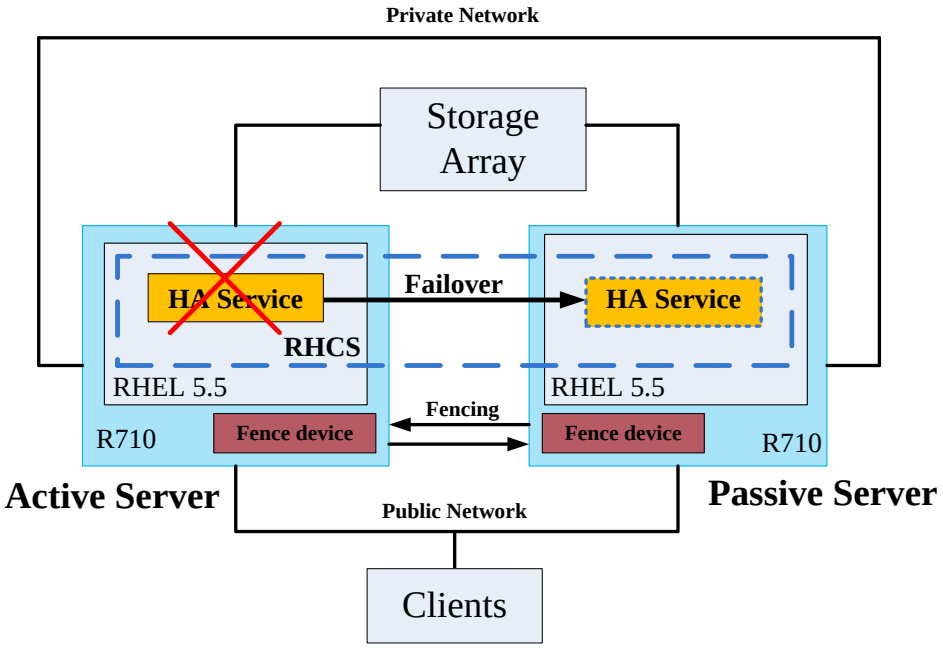


Figure 7 shows the failover procedure in this case. After the occurrence of a failure in the active server, the RHCS agent running on the passive server detects the missing heartbeat. (The process of detection may take a few seconds according to the set timeout value.) Once the failure on the active server is detected, the passive server fences and reboots the active server via a fence device before attempting to take ownership of the cluster service. This is to ensure data integrity. At this point the HA service is migrated or failed over to the passive

server by the Red Hat Service (resource group) Manager, **rgmanager**. Clients cannot access the data until the failover process is complete.

When the active server boots up, it rejoins the cluster and the HA service remains running on the passive server.

2) Heartbeat link failure - simulated by disconnecting the private network link on the active server.
When the heartbeat link is removed from the active server, both servers detect the missing heartbeat and attempt to fence each other. The active server is unable to fence the passive since the missing link prevents it from communicating over the private network. The passive server successfully fences the active server and takes ownership of the HA service.

When the active server boots up, it attempts to start the cluster and fence the passive node but fencing is still unsuccessful since the heartbeat link is still down. The active server believes that the passive server is offline. Since fencing was unsuccessful, the HA service is not started on the active server and the passive server continues to provide the file system to the clients.

When the heartbeat link is reconnected on the active server, the passive server shuts down the cluster daemons on the active server since the active server attempted to join the cluster without a clean restart. At this point there are no cluster daemons running on the active server and it is not a part of the cluster.

After the active server is manually power cycled, it rejoins the cluster. The passive server continues to own the cluster service and provide the file system to the clients.

3) Public link failure - simulated by disconnecting the InfiniBand or 10 Gigabit Ethernet link on the active server.
The HA service is configured to monitor this link. When the public network link is disconnected on the active server, the cluster service stops on the active server and is relocated to the passive server. The time to detect the failed link takes about 30 seconds for the InfiniBand case and 20 seconds for the 10 Gigabit Ethernet case. Note that until the public link is repaired on the active server it will not be able to own and start the cluster service.

After the public link on the active server is repaired, the cluster service continues to run on the passive server with no interruption in service to the clients.

4) Private switch failure - simulated by powering off the private network switch.
When the private switch fails, both servers detect the missing heartbeat from the other server and attempt to fence each other. Fencing is unsuccessful since the network is unavailable and the HA service continues to run on the active server.

When the switch is functional again, both servers kill each other's cluster daemons since the server attempted to rejoin the cluster without a clean restart. At the point the HA service is still functional and continues to run on the active server. This is not a good state since the cluster management daemons are dead. Restarting the cluster daemons does not succeed. The HA service can be stopped using debug tools to stop client access to the file system.

Once both servers are manually power cycled, they rejoin the cluster and one server takes ownership of the HA service to provide the file system to the clients.

5) Fence device failure - simulated by disconnecting the iDRAC cable from the server.
If the iDRAC on a server fails, the server will be fenced via the network PDUs which are defined as secondary fence devices in the cluster configuration files.
At the time of fencing, one server will attempt to fence the other server via the iDRAC. When that fails, it will attempt to fence via the network PDUs. The server will rotate through these fence devices until fencing is successful.

6) One SAS link failure - simulated by disconnecting one SAS link between the PowerEdge R710 server and the PowerVault MD3200 storage.
In the case where only one SAS link fails, the cluster service is not interrupted. Since there are multiple paths from the server to the storage, a single SAS link failure does not break the data path from the clients to the storage and thus does not trigger a cluster service failover.

Note that for all cases (1) through (6) it was observed that the **HA service failover takes in the range of one to two minutes.** Thus in a healthy cluster, any failure event should be noted by the Red Hat cluster management daemon and acted upon within minutes.

7) Multiple SAS link failures - simulated by disconnecting all SAS links between one PowerEdge R710 server and the PowerVault MD3200 storage.
When all SAS links on the active server fail, the multipath daemon on the active server retries the path to the storage based on the parameters configured in the `multipath.conf` file. This is set to 150 seconds by default. After this process times out, the HA service will attempt to failover to the passive server.

If the cluster service is unable to cleanly stop the LVM and the file system because of the broken path, a watchdog script will reboot the active server after five minutes. At this point the passive server will fence the active server, restart the HA service and provide the data path to the clients. This failover can therefore take anywhere in **the range of three to eight minutes.**

When the active server reboots, it will rejoin the cluster. It will, however, not be a candidate to run the HA service until at least one SAS link is established between the server and the storage.

Impact to clients

Clients mount the NFS file system exported by the server using the HA service IP. This IP is associated with either an InfiniBand or a 10 Gigabit Ethernet network interface on the server. To measure any impact on the client, the dd utility and the `iozone` benchmark were used to read and write large files between the client and the file system. Component failures were introduced on the server while the client was actively reading and writing data from the file system.

In all scenarios it was observed that the client processes complete the read and write operations successfully. As expected, the client processes take longer to complete if the process is actively accessing data during a failover event. In the InfiniBand case, the client process takes 15-35%

longer to complete. In the 10 Gigabit Ethernet case, the client process takes 5-10% longer to complete. The actual additional time taken by the client process is of the order of minutes – one to three minutes. During the failover period when the data share is temporarily unavailable, the client process was observed to be in an uninterruptible sleep state.

Depending on the characteristics of the client process it can be expected to abort or sleep while the NFS share is temporarily unavailable during the failover process. Any data that has already been written to the file system will be available. The cluster configuration includes several design choices to protect data during a failover scenario. These and other choices were explained in the section on Design Choices that Impact Functionality and Performance.

For read and write operations during the failover case, data correctness was successfully verified using the `checkstream` utility.

Details on the tools used are provided in *Appendix C: Benchmarks and Test Tools*.

5. Performance Benchmark Results (updated May 2011)

This section presents the results of performance benchmarking on the NSS-HA Solution. Performance tests were run to evaluate the following common I/O patterns.

- Large sequential reads and writes
- Small random reads and writes
- Metadata operations

These tests were performed for the 10 Gigabit Ethernet as well as the IP-over-InfiniBand (IPoIB) case. The `iozone` and `mdtest` benchmarks were used for this study. Details of the benchmarks are provided in *Appendix C: Benchmarks and Test Tools*.

Iozone was used for the sequential tests as well as the random tests. The I/O access patterns are N-to-N, i.e., each thread reads and writes to its own file. Iozone was executed in clustered mode and one thread was launched on each compute node. For the sequential tests, the performance metric used was throughput in terms of MB/s. For random tests, I/O operations per second (IOPS) was the metric.

The large sequential read and large sequential write tests were conducted using a request size of 1024KB. The total amount of data written was 128GB to ensure that the NFS server cache is saturated. The small random tests were performed with 4 KB record sizes since the size corresponds to typical random I/O workloads. All clients read and write a 2GB file for these tests.

The metadata tests were performed with the `mdtest` utility and include file creates, stats, and removals. While these benchmarks do not cover every I/O pattern, they help characterize the I/O performance of the NSS-HA solution.

Each set of tests was run on a range of clients to test the scalability of the solution. The number of simultaneous clients involved in each test was varied from one to 64 clients.

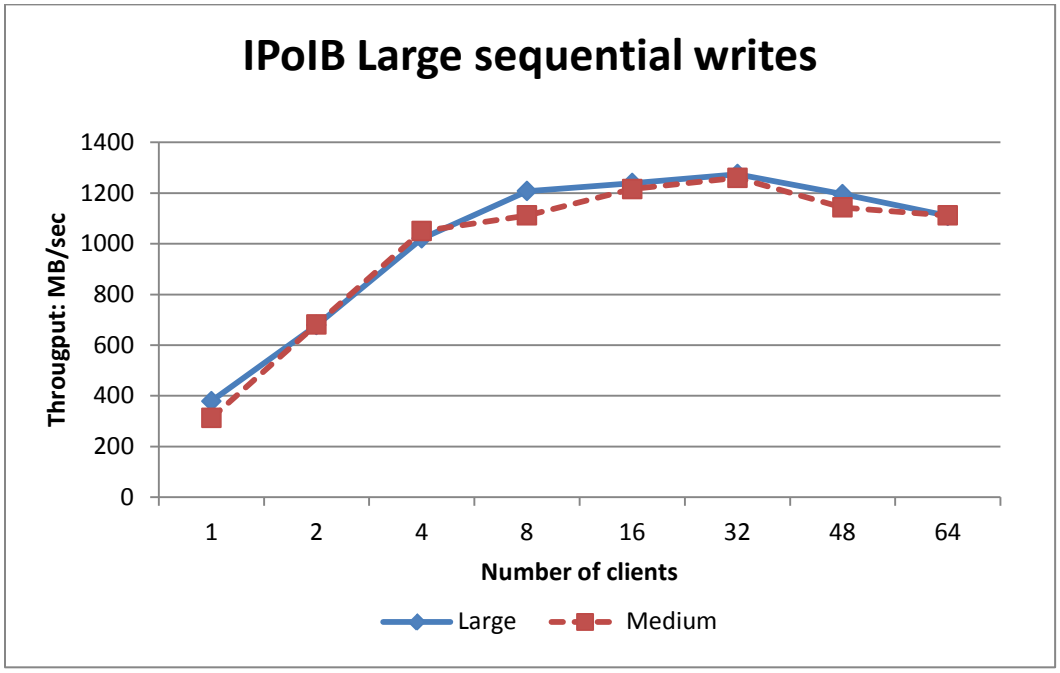
Tests were performed on the two NSS-HA configurations – Medium and Large. The Medium configuration provided 40TB of usable space across two storage arrays with each storage controller managing one virtual disk. The Large configuration provides 80TB of usable space across four storage arrays, with each storage controller owning two virtual disks.

As mentioned before, all performance benchmarking was done in a failure-free situation to understand the maximum capability of the solution.

5.1. InfiniBand Sequential Reads and Writes

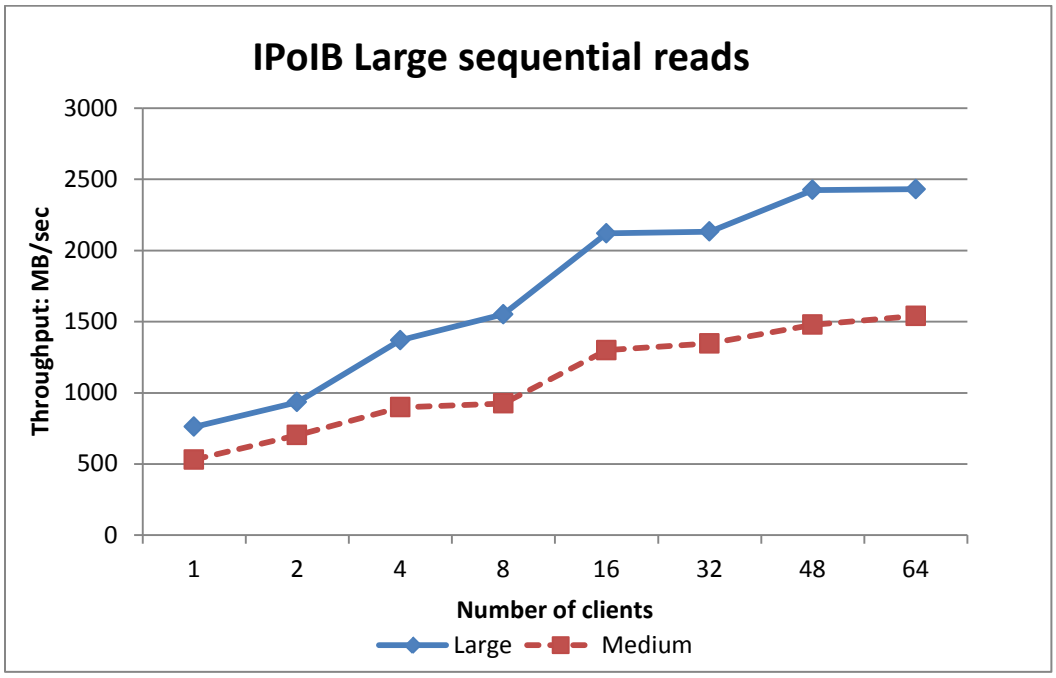
The results of the IPoIB sequential write tests are shown in 8. The figure shows the aggregate throughput that can be achieved when a number of clients are simultaneously writing to the storage over the InfiniBand fabric. Results are presented for the NSS-HA Medium and Large configurations. The results show that the peak performance of the solution is about 1260 MB/s. Both the Medium and the Large configuration provide similar bandwidth. In the Medium configuration, the storage controller cache is used by two virtual disks, while in Large configuration the cache is shared across four virtual disks. The disadvantage of the smaller cache per virtual disks balances out the advantage of the additional disk spindles in the Large configuration, resulting in similar performance across the two configurations. Additionally, the NFS sync export option and cache mirroring between the RAID controllers on the MD3200 limit the write performance of the solution. The data shows that in both configurations, there is a drop in performance after a certain number of clients. This is due to the fact that the storage arrays are populated with NearLine SAS drives and NFS concurrent write traffic causes the disks to become seek bound causing the decline in performance.

Figure 8 - IPoIB Large Sequential Write Performance



IPoIB sequential read results are shown in Figure 9. The figure shows the aggregate throughput that can be achieved when a number of clients are simultaneously reading from the storage over the InfiniBand fabric. The results show that the peak read performance of the NSS-HA Large solution is about 2430 MB/s. The Medium configuration peaks at 1510 MB/second. The Large configuration has double the number of disks than the Medium configuration and the additional disk spindles help the read performance. The solution scales well and the saturation point is not observed with even 64 clients.

Figure 9 - IPoIB Large Sequential Read Performance



5.2. 10 Gigabit Ethernet Sequential Reads and Writes (updated May 2011)

For the 10 Gigabit Ethernet tests, flow control was disabled on the PowerConnect 8024 switch and two PowerConnect 6248 switches. 10GbE sequential write results are shown in Figure 10. The figure shows the aggregate throughput that can be achieved when a number of clients are simultaneously writing to the storage over the 10 Gigabit Ethernet fabric.

The results show that the peak write performance of the NSS-HA solution is close to 1100 MB/s. As all the compute nodes in the test bed are finally connected to the NFS server via a single 10 Gigabit connection, the theoretical network bandwidth is limited to 1.25 GB/sec which is much less than the I/O bandwidth of the shared storage array. Since the bottleneck in this solution is the network and not the disk, both the Medium and the Large configuration have similar results.

10GbE sequential read results are shown in Figure 11. The figure shows the aggregate throughput that can be achieved when a number of clients are simultaneously reading from the storage over the 10 Gigabit Ethernet fabric. The results show that the peak read performance of the NSS-HA solution is about 1170 MB/s. As the network is the bottleneck, both the Medium and the Large configuration also have similar read results.

Figure 10 - 10GbE Large Sequential Write Performance

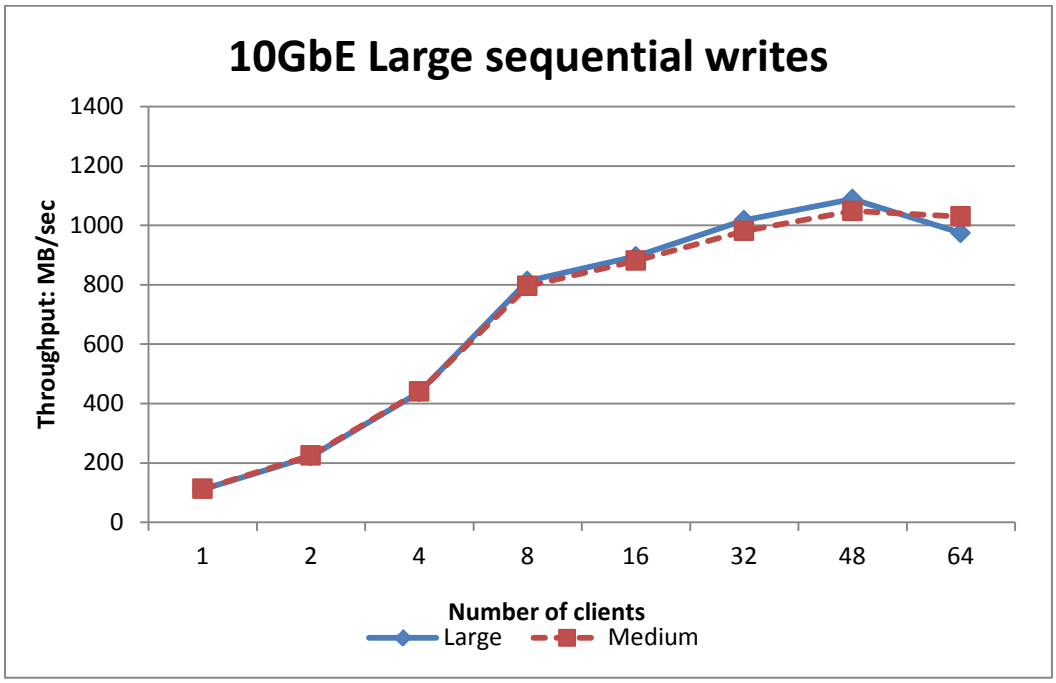
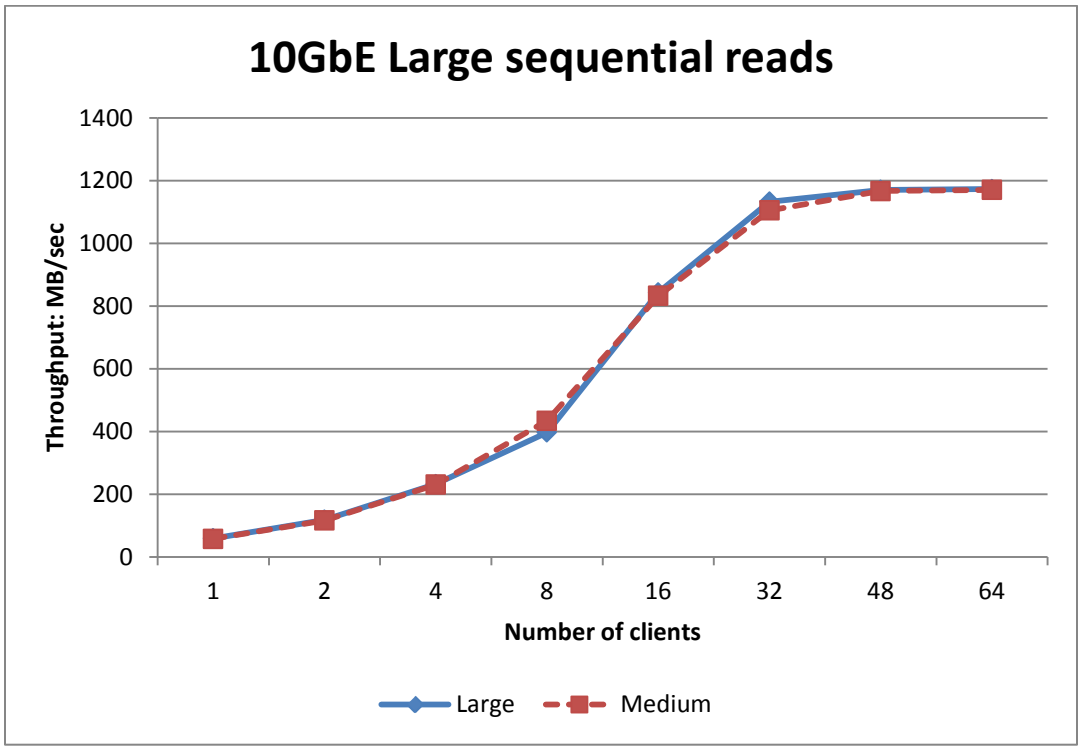


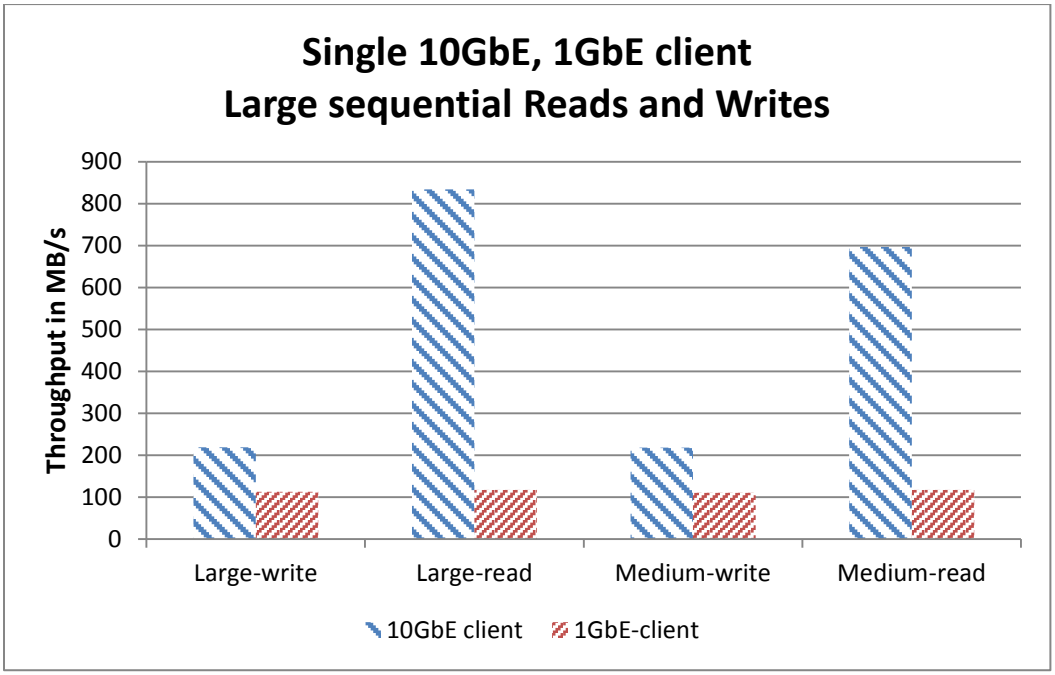
Figure 11 - 10GbE Large Sequential Read Performance



A single NFS client with a 10 Gigabit Ethernet connection to the NFS server was tested as well. In this case the NFS client had only a 10 Gigabit Ethernet card (and no InfiniBand card). Both the NFS client and the NSS server were directly connected to the PowerConnect 8024 10 Gigabit Ethernet

switch. Note that the network switch setting might vary for single client performance tests as opposed to multiple concurrent clients accessing the storage. Hence the network switch options should be tuned accordingly. These results are shown in Figure 12. The large sequential read throughput was 864 MB/sec for the Large configuration and 700 MB/sec for the Medium configuration from this 10GbE client. Large sequential write throughput was 219 MB/sec for both the Large and Medium configurations.

Figure 12 - Single 10GbE Client - Large Sequential Reads and Writes



5.3. Random Reads and Writes

The results for the random read and write tests were very close for InfiniBand and 10 Gigabit Ethernet, with the average difference being less than 3%. Only the InfiniBand results are presented and discussed in this section.

Similar IPoIB and 10GbE performance indicates that the bottleneck in this case is the disks and not the network. This is to be expected as the performance of random reads and writes are dominated by the disk seek latency and cache plays less of a role for random operations.

Figure 13 - IPoIB Random Write Performance

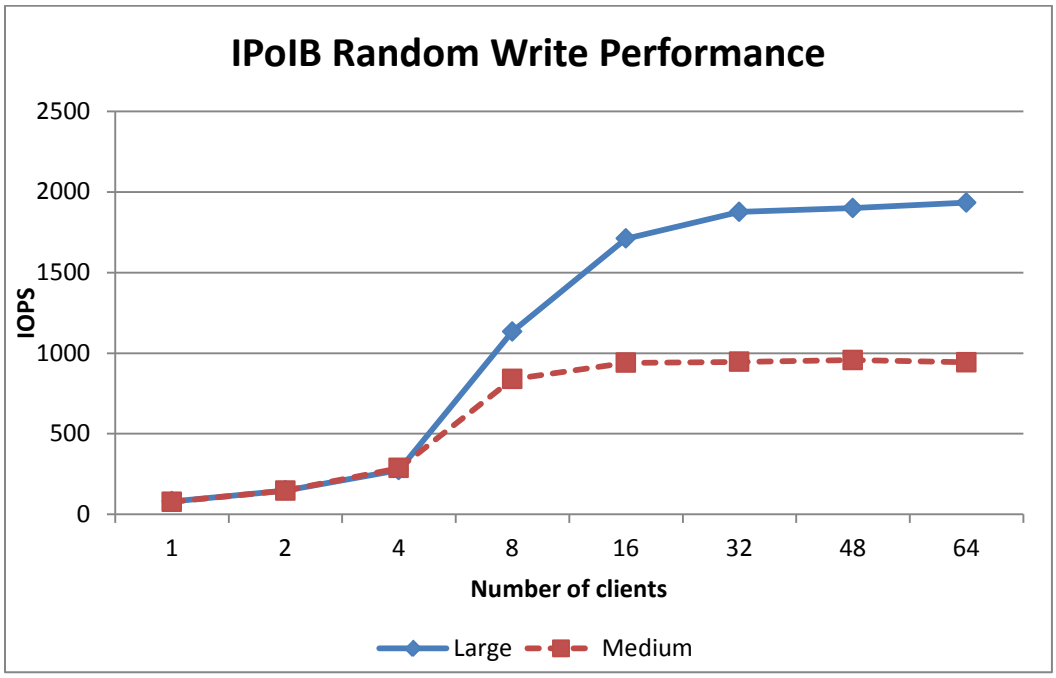
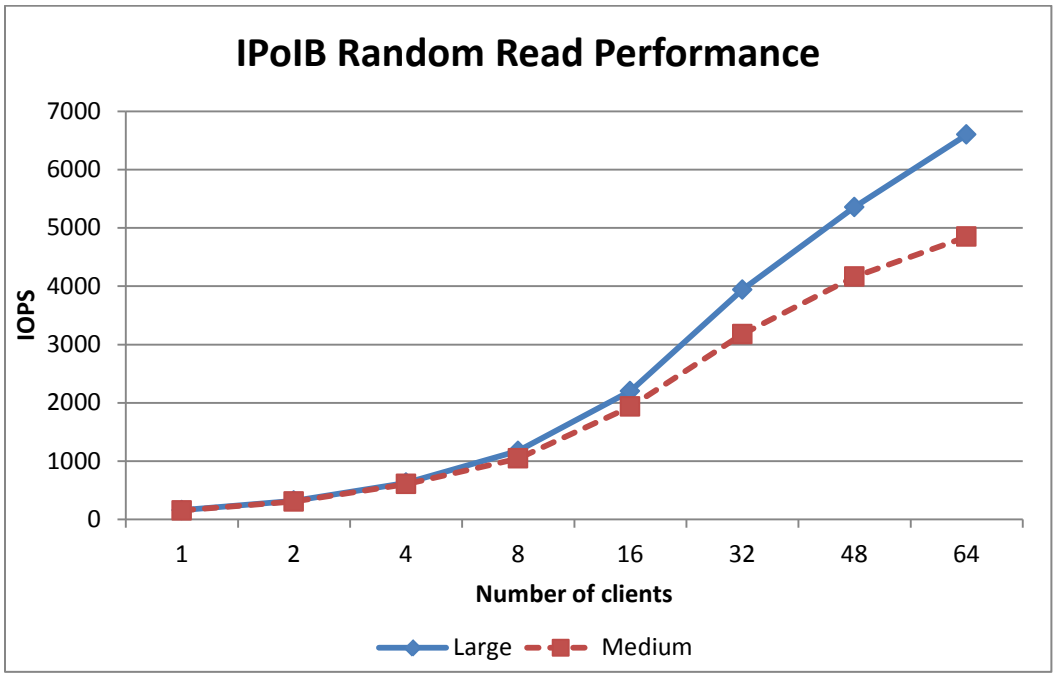


Figure 13 shows the IPoIB random write performance. The figure shows the aggregate I/O operations per second when a number of clients are simultaneously writing to the storage over the InfiniBand fabric. The results show that the peak random write IOPS of the NSS-HA Large configuration is about 1930 IOPS. The peak of the NSS-HA Medium is about 950 IOPS. The additional disk spindles in the NSS-HA Large configuration help the performance scale better. Both configurations maintain the peak IOPS across a range of clients. The performance of NSS-HA random writes are limited by the design choices of NFS sync export option, cache mirroring on the MD3200 controllers and the well-known performance limitation of small writes for RAID 6. The random read performance shown in Figure 14 is better than the write performance for this reason. The read results demonstrate that both the NSS-HA Large and Medium configurations scale well with the number of clients. 64 clients are not enough to note the saturation.

Figure 14 - IPoIB Random Read Performance



5.4. Metadata tests

The results for the `mdtest` (file create, stat, remove) were very close for InfiniBand and 10 Gigabit Ethernet, with the average difference being less than 5%. Only the InfiniBand results are presented and discussed in this section.

Similar IPoIB and 10GbE performance indicates that the bottleneck in this case is the disks and not the network. Similar to the random test results, this is to be expected as these tests generate a large amount of small random I/O requests.

Figure 15, Figure 16 and Figure 17 show the results of file create, stat, and remove, respectively. The results show that all three metadata operations are highly scalable in terms of the number of NFS clients.

Figure 15 - IPoIB File Create Performance

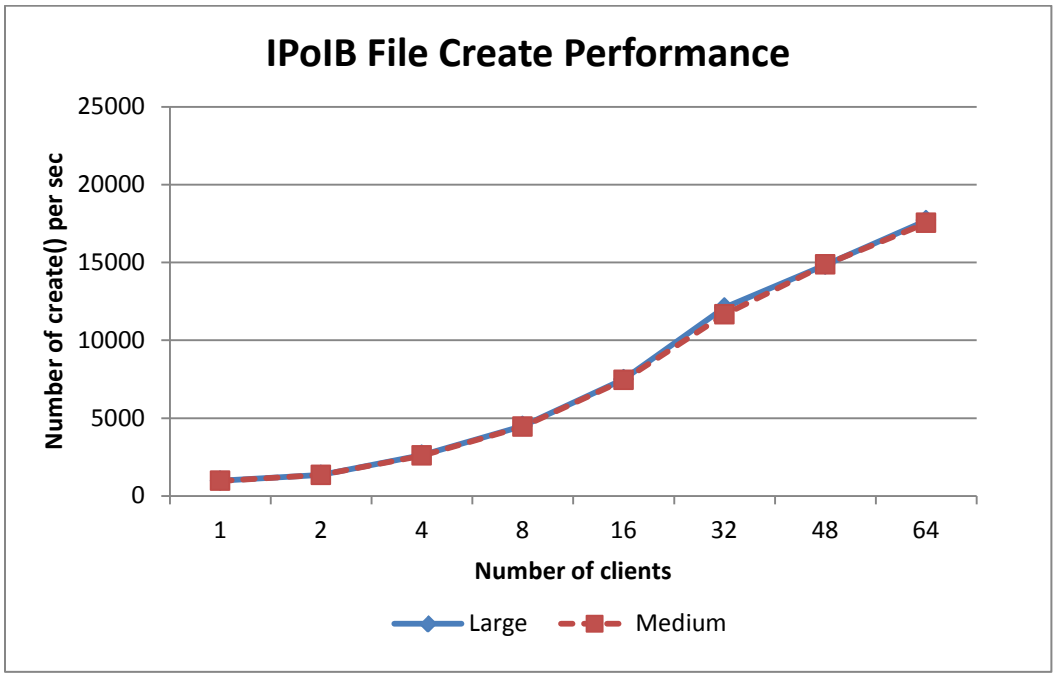


Figure 16 - IPoIB File Stat Performance

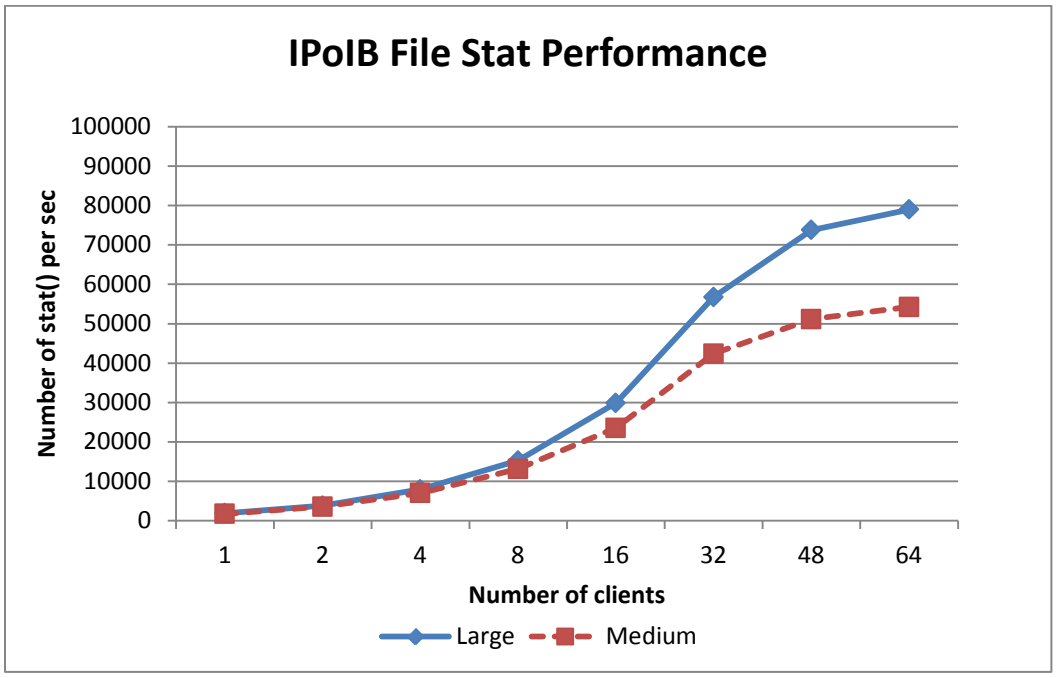
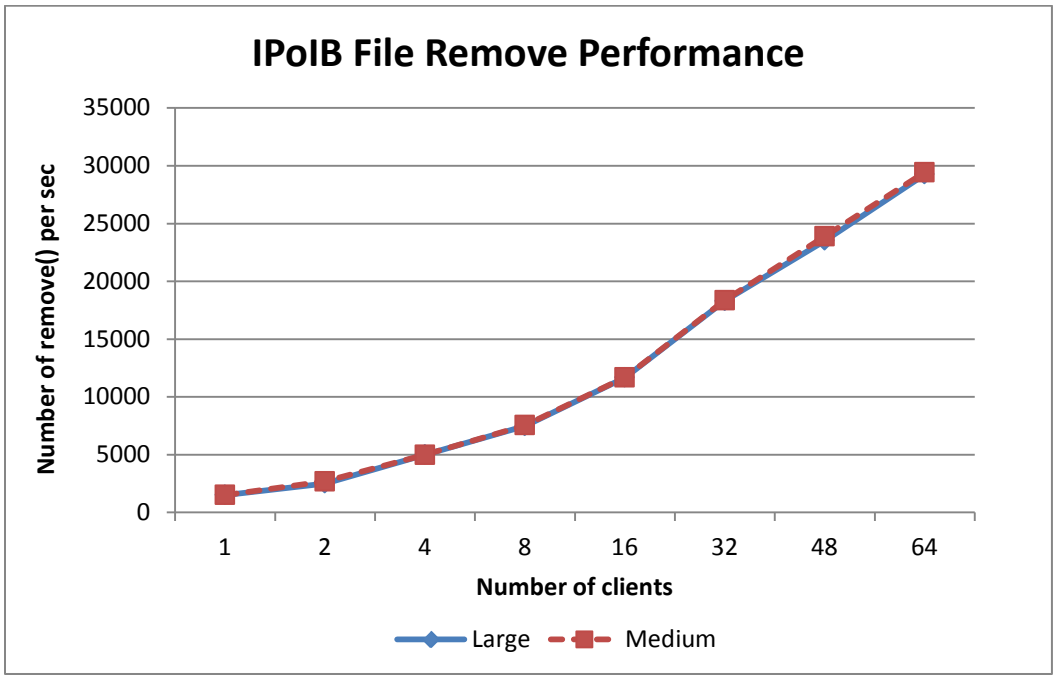


Figure 17 - IPoIB File Remove Performance



6. Comparison of the NSS Solution Offerings

This document describes a method to build on the existing non-HA NFS Storage Solution offered by Dell to provide High Availability features. Addition of the HA feature changes a few solution components. Table 6 and

Table 7 summarize the differences and similarities between the HA and non-HA NSS configurations.

Table 6 - Comparison of Hardware Components

HARDWARE COMPONENT	NSS without HA	NSS with HA
Available Solution Configurations	Small 20TB, Medium 40TB and Large 80TB	Medium 40TB and Large 80TB
Storage	One, two or four MD1200s	One MD3200; One or three MD1200s
Server	Single PowerEdge R710	Two PowerEdge R710s
HBA for storage connectivity	One or two PERC H800 cards	One SAS 6Gbps card
Systems Management	iDRAC enterprise optional	iDRAC enterprise mandatory
Server Power Supply Units (PSU)	Two power supply optional	Two power supplies mandatory
PDUs	Switched rack PDU optional	Two switched rack PDUs mandatory
Private network	No private network	Gigabit Ethernet switch for private

HARDWARE COMPONENT	NSS without HA	NSS with HA
		network
Storage controller for local disks	PERC H700, 5 local disks	
Public network to clients	InfiniBand or 10 Gigabit Ethernet	

Table 7 - Comparison of Software Configuration

SOFTWARE COMPONENT	NSS without HA	NSS with HA
Operating System	Red Hat Enterprise Linux 5.5 x86_64	
HA Cluster Software	None required	Red Hat Cluster Suite for RHEL 5.5
Systems Management	Dell OpenManage Server Administrator	
Storage Management	None required	Dell Modular Disk Storage Manager
InfiniBand driver	Mellanox OFED 1.5.1	
10 GbE driver	Native to RHEL 5.5	
File system	Red Hat XFS	
LVM create parameters	lvcreate -i <number_of_arrays> -I 1024 -l 100%FREE VGName	
XFS create parameters	mkfs.xfs -l size=128m <path_to_LV>	
XFS mount options	“noatime, allocsize=1g, nobarrier, inode64, logbsize=262144”	“noatime, allocsize=1g, nobarrier, inode64, logbsize=262144, wsync”
NFS export options	“async” if performance is the key criteria	“sync” option must be used for data corrected. “async” will not guarantee data correctness.

7. Conclusion

This solution guide enhances Dell’s HPC storage solutions with an NFS offering that includes a High Availability feature. The Dell NFS Storage Solution is available with deployment services and full hardware and software support from Dell. This document provides complete information on architecting, deploying and tuning such a solution. The guidelines include hardware and software information along with detailed configuration steps and best practices make it easy to deploy and manage. The performance tuning notes and results describe the capability of the architecture.

8. References

1) Clusters have become the most popular architecture for HPC today. http://www.top500.org/overtime/list/36/archtype

2) Dell | Terascala HPC Storage Solution (DT-HSS)
http://content.dell.com/us/en/enterprise/d/business~solutions~hpcc~en/Documents~Dell-terascala-dt-hss2.pdf.aspx

3) Dell NFS Storage Solution for HPC (NSS)
http://i.dell.com/sites/content/business/solutions/hpcc/en/Documents/Dell-NSS-NFS-Storage-solution-final.pdf

4) Red Hat Enterprise Linux 5 Cluster Suite Overview
http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5/pdf/Cluster_Suite_Overview/Red_Hat_Enterprise_Linux-5-Cluster_Suite_Overview-en-US.pdf

5) Deploying a Highly Available Web Server on Red Hat Enterprise Linux 5
http://www.redhat.com/f/pdf/rhel/Deploying_HA_Web_Server_RHEL.pdf

6) Platform Cluster Manager
http://www.platform.com/cluster-computing/cluster-management

7) Optimizing DELL™ PowerVault™ MD1200 Storage Arrays for High Performance Computing (HPC) Deployments
http://i.dell.com/sites/content/business/solutions/power/en/Documents/Md-1200-for-hpc.pdf

8) Array Tuning Best Practices
http://www.dell.com/downloads/global/products/pvaul/en/powervault-md3200i-performance-tuning-white-paper.pdf

Appendix A: NSS-HA Recipe (updated May 2011)

Sections

A.1. Pre-install preparation

Figure 2 shows an NSS HA cluster. The equipment list for NSS-HA includes:

- 2 PowerEdge R710 servers
- 1 PowerConnect 5424 Gigabit Ethernet switch
- 2 Switched APC AP7921 power PDUs
- 1 PowerVault MD3200
- 1 or 3 PowerVault MD1200s

Gigabit Ethernet IP addresses for the private network between the servers and the PDUs.
All IPs should be on the same subnet. Noted here are the IP addresses that were used in the test configuration.

- Active server NIC1 IP – 15.15.10.1/24
- Passive server NIC1 IP – 15.15.10.2/24
- Active server iDRAC IP – 15.15.10.201/24
- Passive server iDRAC IP – 15.15.10.202/24
- PDU 1 IP – 15.15.10.101/24
- PDU 2 IP – 15.15.10.105/24

Public network IP addresses for the network that is shared with the NFS clients.
These IPs should be on the same subnet as the clients. Noted here are the IP addresses that were used in the test configuration.

- Active server 10GbE/IPoIB IP address – 10.10.10.201/16
- Passive server 10GbE/IPoIB IP address – 10.10.10.202/16
- Floating resource IP that the clients will use to mount NFS – 10.10.10.200

A.2. Server side hardware set-up

1) Prepare two PowerEdge R710 servers (called "active" and "passive"). Configure each server as follows.
   - One PERC H700 and 5 local disks each of 146 GB.
   - Configure 2 disks in RAID 1 with 1 disk designated as the hot spare. This will be used for the operating system. Configure 2 disks in RAID 0, this will be used as swap.
   - 10 Gigabit Ethernet card OR InfiniBand card in slot 4, a PCI-E x8 slot.
   - SAS 6 Gbps card in slot 3, a PCI-E x8 slot. This is the card that will connect to the MD3200 storage.
   - iDRAC enterprise
   - Dual power supplies

   Reference for the PERC H700 storage controller is provided below. The PowerEdge R710 server might ship from the Dell factory with 2 disks already configured in RAID 1. Insert the additional 3 disks into the server and use this document to add a disk as a hot spare for the RAID 1 setup and to configure the remaining two disks in RAID 0.
   http://support.dell.com/support/edocs/storage/Storlink/H700H800/en/UG/PDF/H700H800.pdf

2) Set up a Gigabit Ethernet switch for the private network. Power the gigabit switch from one of the Power PDUs. You will need at least 6 Ethernet ports on the switch. One port each for:
   - Ethernet cable from power PDU1
   - Ethernet cable from power PDU2
   - iDRAC enterprise from active server
   - iDRAC enterprise from passive server
   - NIC1 from active server
   - NIC1 from passive server

3) Set up two switched power PDUs with at least 3 power ports each.
   - PDU 1, port 1 for the Gigabit switch for the private network.
   - PDU 1, port 2 for power supply 1 on active
   - PDU 1, port 3 for power supply 1 on passive
   - PDU 2, port 2 for power supply 2 on active
   - PDU 2, port 3 for power supply 2 on passive

4) Configure the IP of the power PDUs to be on the private network. Connect the Ethernet port of each PDU to the Gigabit Ethernet private switch.

5) For each R710, cable the onboard NIC1 and iDRAC enterprise port to the Gigabit Ethernet private switch.

6) For each R710, plug in the two power supplies on the server to each of the two switched power PDUs.

7) For each R710, set the IP address of the iDRAC Enterprise on the private network.

8) Connect the IB or 10GbE network to the "public" network. This is the network the NFS clients are connected to.

A.3. Initial software configuration on each PowerEdge R710

1) Install the RHEL5.5 x86_64 operating system on the RAID1 virtual disk.
   - Make sure MD storage is not attached to the servers during the OS install.

2) After the OS is installed, setup swap on the RAID0 device.
   - `multipath –ll` will show the device associated with the PERC H700, such as `/dev/mapper/mpath1`
   - `# mkswap -L SWAP /dev/mapper/mpath1`
   - `# swapon –p 10 /dev/mapper/mpath1`
   - Edit `/etc/fstab` and add an entry for this swap device.
     "LABEL=SWAP swap swap pri=10 0 0"
     Make sure the entry is listed before the default swap space the OS install created, if any.
   - Test that swap can be enabled automatically when the server boots
     ```
     # swapoff –a
     # swapon -s
     # swapon –a
     # swapon –s
     ```

3) Copy the RHEL 5.5 DVD to the server and create yum repositories. On each server:
   - ```
     # cp –r /media/RHEL5.5 /root/
     # rpm -ivh /root/RHEL5.5/Server/createrepo-0.4.11-3.el5.noarch.rpm
     # cd /root/RHEL5.5/Server; createrepo .
     # cd /root/RHEL5.5/VT; createrepo .
     # cd /root/RHEL5.5/Cluster; createrepo .
     # cd /root/RHEL5.5/ClusterStorage; createrepo .
     ```

   - Create files similar to the example below for each repository.
     ```
     /etc/yum.repos.d/Server.repo
     /etc/yum.repos.d/Cluster.repo
     /etc/yum.repos.d/ClusterStorage.repo
     /etc/yum.repos.d/VT.repo
     ```

     Example `.repo` file:
     ```
     [root@active]# cat /etc/yum.repos.d/Server.repo
     [Server]
     name=Server
     baseurl=file:///root/RHEL5.5/Server
     enabled=1
     gpgcheck=0
     ```

   - Check the repositories.
     ```
     [root@passive]# yum repolist
     Loaded plugins: rhnplugin, security
     This system is not registered with RHN.
     RHN support will be disabled.
     repo id         repo name          status
     VT              VT                 enabled:    71
     Cluster         Cluster            enabled:    32
     Clusterstorage ClusterStorage      enabled:    39
     ```
     ```

```
        Server          Server              enabled: 3,116
        repolist: 3,258
```

4) Obtain the XFS packages from the Red Hat Network (http://rhn/redhat.com) and install them.
   xfsdump-2.2.48-3.el5.x86_64.rpm
   xfsprogs-devel-2.10.2-7.el5.x86_64.rpm
   xfsprogs-2.10.2-7.el5.x86_64.rpm

5) Install Dell OpenManage Server Administrator (OM-SrvAdmin-Dell-Web-LX-6.4.0-1266.RHEL5.x86_64_A00.21.tar.gz)

   If the setup fails citing missing dependencies, install the missing rpms from `<dir_where_OM_extracted>/linux/RPMS/supportRPMS/opensource-components/RHEL5/x86_64`

   libcmpiCppImpl0-2.0.0Dell-1.3.el5.x86_64.rpm
   sblim-sfcc-2.2.1-1.4.1.el5.x86_64.rpm
   sblim-sfcb-1.3.7-1.4.2.el5.x86_64.rpm
   openwsman-server-2.2.3.9-1.5.3.el5.x86_64.rpm

6) Install the MD3200 management software (MGFM4A04_md3200_1_2_0_12.iso).
   http://support.dell.com/support/downloads/driverslist.aspx?c=us&l=en&s=gen&ServiceTag=&SystemID=PWV_MD3200&os=RHEL5&osl=EN

7) Install Mellanox OFED 1.5.1 if using InfiniBand (MLNX_OFED_LINUX-1.5.1-rhel5.5.iso).

8) Enable IP ports on both the cluster servers. The list of ports to be enabled is in the Red Hat Cluster Administration Guide.
   http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5/pdf/Cluster_Administration/Red_Hat_Enterprise_Linux-5-Cluster_Administration-en-US.pdf
   Section 2.3
   Launch the firewall configuration tool (`system-config-securitylevel`), click on “Other ports” and add the port numbers for TCP and UDP.

   Alternately, turn off the firewall. Ensure that your public and private interfaces are on a secure network and be aware of the security implications of turning off the firewall before implementing this alternative:
   `# service iptables stop; chkconfig iptables off`

9) chkconfig ipmi on

10) chkconfig nfs on

11) chkconfig multipathd on

12) Update `/etc/hosts` on both servers to contain entry for both servers
```
[root@passive ~]# cat /etc/hosts
# Do not remove the following line, or various programs
```

```
# that require network functionality will fail.
127.0.0.1             localhost.localdomain localhost
15.15.10.1            active.hpc.com  active
15.15.10.2            passive.hpc.com passive
```

13) Set up password-less ssh between the active and passive servers.

```
active> ssh-keygen -t rsa

active> ssh-copy-id –i ~/.ssh/id_rsa.pub passive

passive> ssh-keygen -t rsa

passive> ssh-copy-id –i ~/.ssh/id_rsa.pub active
```

14) Configure the IPoIB ib0 address or 10GbE address for the public network.

15) To work around an issue that can impact cluster service failover when all SAS links from the server to the storage fail, two rpms need to be updated on both the active and the passive server.

    These rpms can be obtained from Red Hat Network.
    device-mapper-multipath-0.4.7-42.el5_6.2.x86_64.rpm
    kpartx-0.4.7-42.el5_6.2.x86_64.rpm

    Update both the active and passive server
    `# rpm –Uvh kpartx-0.4.7-42.el5_6.2.x86_64.rpm device-mapper-multipath-0.4.7-42.el5_6.2.x86_64.rpm`
    References:
    http://rhn.redhat.com/errata/RHBA-2011-0379.html
    https://bugzilla.redhat.com/show_bug.cgi?id=677821
    https://bugzilla.redhat.com/show_bug.cgi?id=683447

A.4. Performance tuning on the server

1) If the clients access the NFS server via 10GbE, configure the MTU on the 10GbE device to be 8192 for both the active and the passive server. Note that the switches need to be configured to support large MTU as well.

    On the server, if the value of MTU is not specified in `/etc/sysconfig/network-scripts/ifcfg-ethX`,

    `echo “MTU=8192” >> /etc/sysconfig/network-scripts/ifcfg-ethX,`

    Else change the old value to 8192 in `/etc/sysconfig/network-scripts/ifcfg-ethX.`

    where `ifcfg-ethX` is the 10GbE network interface.

    Restart the networking services.
    `service network restart`

2) On both the active and the passive server, change the number of NFS threads from a default of 8 to 256.

Make a backup of `/etc/sysconfig/nfs` and change the number of threads

```
# cp /etc/sysconfig/nfs{,.orig}
# sed -i 's/#RPCNFSDCOUNT=8/RPCNFSDCOUNT=256/' /etc/sysconfig/nfs
```

Restart the NFS service.

```
service nfs restart
```

Reference - http://i.dell.com/sites/content/business/solutions/whitepapers/en/Documents/hpc-pv-md1200-nfs.pdf

3) On both the active and the passive server, change the OS I/O scheduler to "deadline".

   To the end of the kernel line in `/etc/grub.conf`, add `elevator=deadline`

   Reboot the server for this to take effect.

A.5. Storage hardware set-up

1) Cable the MD3200 to the SAS 6 Gbps card on servers as shown in Figure 3.
   - Do not cable storage to R710 before OS install.

   Reference: http://support.dell.com/support/edocs/systems/md3200/en/DG/PDF/DG.pdf, Section "Dual Controller Configurations"

2) Cable MD1200 to MD3200
   Reference: http://support.dell.com/support/edocs/systems/md3200/en/index.htm

A.6. Storage Configuration

1) Launch the MDSM management GUI on one R710. Discover the attached storage array via in-band management and add the storage array to the management GUI.

2) Create a host group and add the active and passive servers to the group.

3) Create LUNS on storage using MDSM GUI
   - For each array, create a disk group and a virtual disk that contains all 12 disks in a 10+2 RAID6 with a segment size of 512k. For the NSS-HA Medium configuration, this is a total of two LUNS. For NSS-HA Large, this is a total of four LUNS.
   - Enable read cache, write cache, write cache mirroring and dynamic cache read prefetch for each virtual disk.
   - Enable the high performance tier on the MD3200.
   - Set the cache block size to 32k.

For instruction on how to configure the storage array, use reference:
http://support.dell.com/support/edocs/systems/md3200/en/OM/PDF/MD3200.pdf

4) Map all the virtual disks to this host group that contains the active and passive servers.

Reference: http://support.dell.com/support/edocs/systems/md3200/en/index.htm

5) On each R710, run the command `rescan_dm_devs`

6) On each R710, `cat /proc/partitions` and `multipath –ll` should show all the LUNs on the storage.
Reference http://support.dell.com/support/edocs/systems/md3200/en/OM/HTML/config_n.htm

A.7. NSS HA Cluster setup

In this recipe the term "cluster" refers to the active-passive NSS-HA Red Hat cluster.

1) On both R710s install the cluster software packages.
```
# yum install -y ricci rgmanager cman openais
# service ricci start; chkconfig ricci on
```

2) On one R710 (the 'active' server), install the cluster management GUI
```
# yum install luci
# luci_admin init
# service luci restart
```

3) Set up HA LVM. HA LVM ensures that only 1 server will access the LVM at a time.

On the active R710:

- edit `/etc/lvm/lvm.conf` and edit `locking_type` to be `1`
- Create an LVM using the mpath device names from `multipath –ll` output

FOR NSS-HA Medium

```
# pvcreate /dev/mapper/mpath2 /dev/mapper/mpath3;
# vgcreate DATA_VG /dev/mapper/mpath2 /dev/mapper/mpath3
# lvcreate -i 2 -I 1024 -l 100%FREE DATA_VG  -n DATA_LV
```

FOR NSS-HA Large

```
# pvcreate /dev/mapper/mpath2 /dev/mapper/mpath3 /dev/mapper/mpath4
/dev/mapper/mpath5
# vgcreate DATA_VG /dev/mapper/mpath2 /dev/mapper/mpath3
/dev/mapper/mpath4 /dev/mapper/mpath5
# lvcreate -i 4 -I 1024 -l 100%FREE DATA_VG  -n DATA_LV
```

- `lvdisplay` should show DATA_LV

- Edit `/etc/lvm/lvm.conf` and change the volume list to
`volume_list = [ "VolGroup00" , "@active" ]`
where `VolGroup00` is the volume group that contains the "`/`" file system for the OS. "`active`" is the name of the server as defined in the `cluster.conf` file that will be used for the cluster configuration.

- Remake the initrd
```
# mkinitrd -f /boot/initrd-2.6.18-194.el5.img 2.6.18-194.el5
```

Set up HA LVM on the other (passive) server:

- `lvdisplay` should show DATA_LV

- Edit `/etc/lvm/lvm.conf` and edit `locking_type` to be 1
- Edit `/etc/lvm/lvm.conf` and change the volume list to
  `volume_list = ["VolGroup00" , "@passive" ]`
  where `VolGroup00` is the volume group that contains the "`/`" file system for the OS. "`passive`" is the name of the server as defined in the `cluster.conf` file that will be used for the cluster configuration..

- Remake the initrd
  `# mkinitrd -f /boot/initrd-2.6.18-194.el5.img 2.6.18-194.el5`

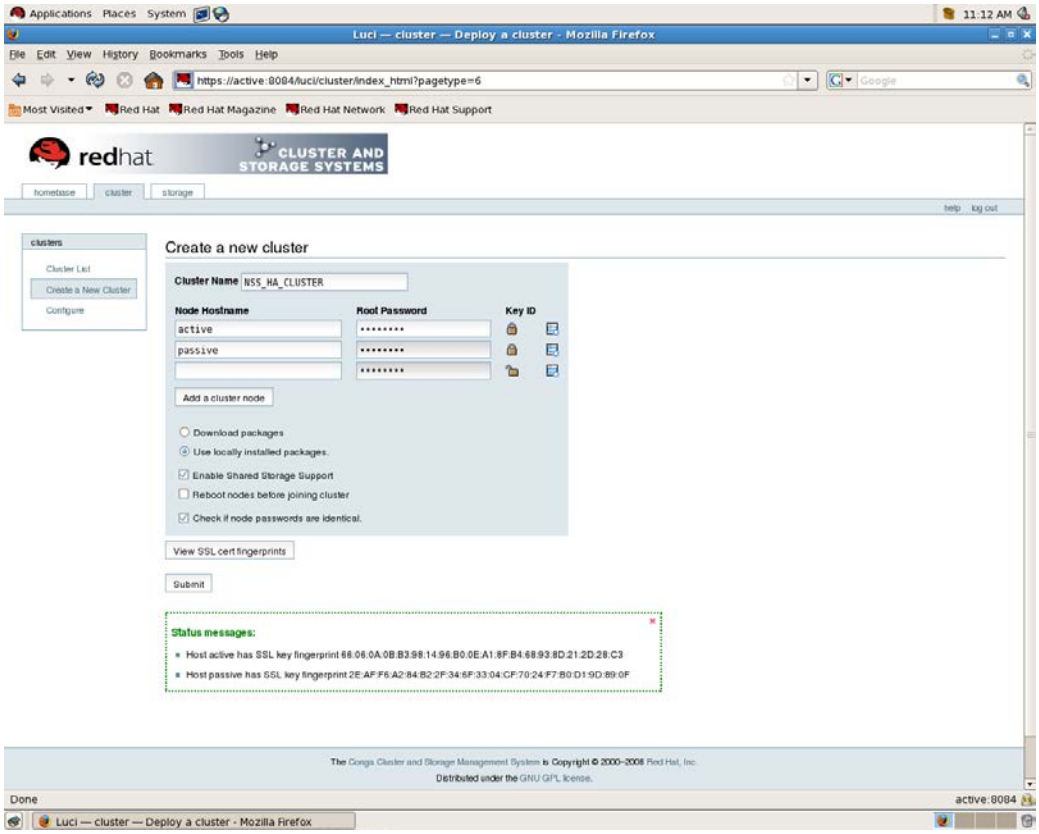
Reference - http://kbase.redhat.com/faq/docs/DOC-3068

4) On the active server, create the XFS file system
   `# mkfs.xfs -l size=128m /dev/DATA_VG/DATA_LV`

5) On both servers, create a mount point for the XFS file system. This is the directory where the XFS file system will be mounted and that will be exported over NFS to the clients.
   `# mkdir /mnt/xfs_data`

6) Configure cluster. Refer to the screen shot in Figure 18
   On the 'active' R710:
   - Launch luci web GUI (https://active:8084).
   - Navigate to the 'cluster' tab and click on "Create a New Cluster"
   - Enter the cluster name, for example 'NSS_HA_CLUSTER'
   - Enter the node hostnames 'active' and 'passive'. Enter the passwords.
   - Select "Use locally installed packages"
   - Check "Enable Shared Storage Support"
   - Check "Check if node passwords are identical" if the node passwords are identical
   - Click on "View SSL cert fingerprints"
   - Click on "Submit"

Figure 18 - Create a New Cluster



7) On both servers, check that the cman service is running.
   `# service cman status`

8) On both servers, check the cluster status:
   `# clustat`

   Cluster status should show both servers online. If the cluster is up and running, go on to the next step.

```
[root@active ~]# clustat
Cluster Status for NSS_HA_CLUSTER @ Wed Mar  2 17:36:36 2011
Member Status: Quorate

 Member Name                     ID   Status
 ------ ----                     ---- ------
 active                             1 Online, Local
 passive                            2 Online
```

9) Re-set any HA LVM configuration that might have changed as a result of cluster packages installed by luci.
   - On both servers, edit `/etc/lvm/lvm.conf` and set `locking_type` to be 1
   - Remake the initrd
     `# mkinitrd -f /boot/initrd-2.6.18-194.el5.img 2.6.18-194.el5`

10) For InfiniBand clusters, copy the **ibstat-script.sh** file provided in Section A.11 to `/root/ibstat-script.sh` on both servers. This script checks the InfiniBand link status using the `ibstat` command. It is included as a resource of the cluster service to ensure that InfiniBand link is monitored. (For 10 Gigabit Ethernet clusters, RHCS monitors the 10GbE link and no additional scripts are needed).

11) Copy the **sas_path_check_script.sh** file provided in Section A.11 to **`/root/sas_path_check_script.sh`** on both servers. This script checks the status of a device on the shared storage. If the device is not accessible within a set period of time, the server will reboot prompting a failover of the cluster service to the other server. The reboot will trigger if the cluster service is unable to stop gracefully because of the failed path.

    The device to check (`/dev/mapper/mpath2`) and the timeout period (300 second) are tunable. Check that the device in the script `/dev/mapper/mpath2` points to a LUN on the shared MD3200 storage. This can be checked by looking at the output of the `multipath –ll` command.

12) Check that the public interface is up on both servers. This is the 10GbE link or the InfiniBand link.

13) **ONLY** on the active server, modify the `/etc/cluster/cluster.conf`
    - Using the example cluster.conf file provided in Section A.11, update the `/etc/cluster/cluster.conf` file on the active server. Add in the sections for fence devices, resources, services, etc.

    - Be sure to edit the `/etc/cluster/cluster.conf` file to match the unique hardware and software configuration.
      This is a critical step. It defines the resources and the cluster service for the NSS-HA cluster. An incorrect or incomplete cluster.conf file will yield a non-working cluster.
      Pay attention to the following environment specific parameters. The instructions below include xml snippets to help identify the relevant portions of the xml file.

        a) Cluster name
        Make this change in **two** places in the xml file.

        ```
        <cluster alias="NSS_HA_CLUSTER" config_version="2"
        name="NSS_HA_CLUSTER">
        ```

        b) correct hostnames for both R710s (active, passive)
        Make this change in **two** places in the xml file for **each** host.

        ```
        <clusternode name="active" nodeid="1" votes="1">

        <clusternode name="passive" nodeid="2" votes="1">

        <failoverdomainnode name="active" priority="1"/

        <failoverdomainnode name="passive" priority="2"/>
        ```

        c) DRAC IP address for both servers

Make this change in **three** places in the xml file for **each** host.

```
<fence>
   <method name="1">
         <device name="15.15.10.201"/>


<fencedevice agent="fence_drac5" cmd_prompt="admin1-&gt;"
ipaddr="15.15.10.201" login="root" name="15.15.10.201"
passwd="calvin" secure="1"/>

<fence>
   <method name="1">
         <device name="15.15.10.202"/>

<fencedevice agent="fence_drac5" cmd_prompt="admin1-&gt;"
ipaddr="15.15.10.202" login="root" name="15.15.10.202"
passwd="calvin" secure="1"/>
```

d) DRAC user name and password for both servers
Make this change in **one** place in the xml file for **each** host.

```
 <fencedevice agent="fence_drac5" cmd_prompt="admin1-&gt;"
 ipaddr="15.15.10.201" login="root" name="15.15.10.201"
 passwd="calvin" secure="1"/>


<fencedevice agent="fence_drac5" cmd_prompt="admin1-&gt;"
ipaddr="15.15.10.202" login="root" name="15.15.10.202"
passwd="calvin" secure="1"/>
```

e) PDU IP addresses for both PDUs
Make this change in **six** places in the xml for **each** PDU

```
<method name="2">
   <device name="15.15.10.101" option="off" port="2"/>
   <device name="15.15.10.105" option="off" port="2"/>
   <device name="15.15.10.101" option="on" port="2"/>
   <device name="15.15.10.105" option="on" port="2"/>
</method>

<method name="2">
   <device name="15.15.10.101" option="off" port="3"/>
   <device name="15.15.10.105" option="off" port="3"/>
   <device name="15.15.10.101" option="on" port="3"/>
   <device name="15.15.10.105" option="on" port="3"/>
</method>

<fencedevice agent="fence_apc" ipaddr="15.15.10.101"
login="apc" name="15.15.10.101" passwd="apc"/>

<fencedevice agent="fence_apc" ipaddr="15.15.10.105"
login="apc" name="15.15.10.105" passwd="apc"/>
```

f) PDU login user name and password for both PDUs

Make this change in **one** place in the xml for **each** PDU

```
<fencedevice agent="fence_apc" ipaddr="15.15.10.101"
login="apc" name="15.15.10.101" passwd="apc"/>

<fencedevice agent="fence_apc" ipaddr="15.15.10.105"
login="apc" name="15.15.10.105" passwd="apc"/>
```

g) PDU ports for the all four power supplies
Make this change in **two** places in the xml for each of the **four** PDU ports

```
<method name="2">
<device name="15.15.10.101" option="off" port="2"/>
   <device name="15.15.10.105" option="off" port="2"/>
   <device name="15.15.10.101" option="on" port="2"/>
   <device name="15.15.10.105" option="on" port="2"/>
</method>

<method name="2">
   <device name="15.15.10.101" option="off" port="3"/>
   <device name="15.15.10.105" option="off" port="3"/>
   <device name="15.15.10.101" option="on" port="3"/>
   <device name="15.15.10.105" option="on" port="3"/>
</method>
```

h) VG and LV names
Make this change in **two** places in the xml for the VG and LV

```
<lvm lv_name="DATA_LV" name="HA_LVM" self_fence="1"
vg_name="DATA_VG"/>

<fs device="/dev/DATA_VG/DATA_LV" force_fsck="0"
force_unmount="1" fstype="xfs" mountpoint="/mnt/xfs_data"
name="XFS"
options="noatime,allocsize=1g,nobarrier,inode64,logbsize=26214
4,wsync" self_fence="0"/>
```

i) XFS mount point
Make this change in **one** place in the xml.

```
<fs device="/dev/DATA_VG/DATA_LV" force_fsck="0"
force_unmount="1" fstype="xfs" mountpoint="/mnt/xfs_data"
name="XFS"
options="noatime,allocsize=1g,nobarrier,inode64,logbsize=26214
4,wsync" self_fence="0"/>
```

j) Service IP for the public network (floating IP that the clients will use to mount NFS)
Make this change in **two** places in the xml.

```
<ip address="10.10.10.200" …

<ip ref="10.10.10.200"/>
```

k) NFS export options
Make this change in **one** place in the xml. "sync" is a required option for NSS-HA.

```
<nfsclient allow_recover="1" name="NFS_client"
options="fsid=55,rw,sync,no_root_squash" target="*"/>
```

l) For InfiniBand clusters, location of the ibstat_script file. This location must be the same on both the servers.
Make this change in **one** place in the xml.

```
<script file="/root/ibstat_script.sh" name="ibstat_script"/>
```

m) The location of the sas_path_check_script file. This location must be the same on both the servers.
Make this change in **one** place in the xml.

```
<script file="/root/sas_path_check_script.sh" name="
sas_path_check_script "/>
```

- Increment the config_version by one. This is on line 2 of `/etc/cluster/cluster.conf`.

- This next command will synchronize the cluster configuration file between the active and the passive server***. It will also start the cluster service that includes activating the HA LVM, mounting the XFS file system, exporting it via NFS and creating the floating resource IP.***
On the active server, run the command
`# ccs_tool update /etc/cluster/cluster.conf`

14) Check that the cluster service is running.
- Cluster status should show both servers online. If the cluster is up and running, go on to the next step.

```
[root@active ~]# clustat
Cluster Status for NSS_HA_CLUSTER @ Thu Mar  3 10:41:48 2011
Member Status: Quorate

   Member Name                              ID   Status
 ------ ----                             ---- ------
active                                      1 Online, rgmanager
passive                                     2 Online, Local,
rgmanager
Service Name                  Owner (Last)                  State
------- ----                  ----- ------                   -----
service:HA_service            active                        started
```

- On the server that is running the service, check that the XFS file system is mounted.

```
[root@active ~]# mount | grep xfs
/dev/mapper/DATA_VG-DATA_VG_LV on /mnt/xfs_data type xfs
(rw,noatime,allocsize=1g,nobarrier,inode64,logbsize=262144,wsync)
```

- On the server that is running the service, check that the resource IP is assigned. The interface to the public network should have two IP addresses - the statically assigned address and the floating service IP address.

```
[root@active ~]# ip addr show ib0
9: ib0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 65520 qdisc pfifo_fast
qlen 256
    link/infiniband
80:00:00:48:fe:80:00:00:00:00:00:00:00:02:c9:03:00:07:7f:a7 brd
00:ff:ff:ff:ff:12:40:1b:ff:ff:00:00:00:00:00:00:ff:ff:ff:ff
    inet 10.10.10.201/24 brd 10.10.10.255 scope global ib0
    inet 10.10.10.200/24 scope global secondary ib0
    inet6 fe80::202:c903:7:7fa7/64 scope link
       valid_lft forever preferred_lft forever
```

15) Update the Selinux policy. This is needed to allow a cluster server to fence the other cluster member and take ownership of the cluster service.
Use the Selinux policy Type Enforcement file (.te) provided in Section A.11 to build a policy module. Install the policy module on both servers.

```
# checkmodule -M -m  NSSHApolicy.te -o NSSHApolicy.mod
# semodule_package -o NSSHApolicy.pp -m NSSHApolicy.mod
# semodule -i NSSHApolicy.pp
```

Alternately, an SELinux policy can be generated from logs of denied operations. Check `/var/log/audit/audit.log` for denied operations. If there are none relating to the cluster, test fencing as described in the "Quick test of HA set-up" section and then follow the steps below.

```
# grep avc /var/log/audit/audit.log | audit2allow -M NSSHApolicy
```

Install the module on bot servers

```
# semodule -i NSSHApolicy.pp
```

Reference https://bugzilla.redhat.com/show_bug.cgi?id=588902

16) On both servers, turn off CLVM.

```
# chkconfig clvmd off; service clvmd stop
```

17) On both servers turn off GFS.

```
# chkconfig gfs off; service gfs stop;
# chkconfig gfs2 off; service gfs2 stop
```

18) Launch the luci web GUI (https://active:8084) to see the latest cluster configuration. An example is shown in Figure 19.

NOTE: There are several known issues that have been worked around in the example `cluster.conf` file. If any changes are saved via the web GUI, please note these fixes and make sure they are included.

a) If changes are saved using the luci web GUI, edit the `cluster.conf` file manually on one server
   - Change file system type `fstype` back to "`xfs`"

```
<fs device="/dev/DATA_VG/DATA_LV" force_fsck="0" force_unmount="1"
fstype="xfs" mountpoint="/mnt/xfs_data" name="XFS"
```

```
options="noatime,allocsize=1g,nobarrier,inode64,logbsize=262144,wsyn
c" self_fence="0"/>
```

Reference https://bugzilla.redhat.com/show_bug.cgi?id=636554

- Check that the fence device for the iDRAC has the cmd_prompt parameter filed in and the secure parameter set to 1 for both iDRAC devices.

```
<fencedevice agent="fence_drac5" cmd_prompt="admin1-&gt;"
ipaddr="15.15.10.201" login="root" name="15.15.10.201"
passwd="calvin" secure="1"/>
```

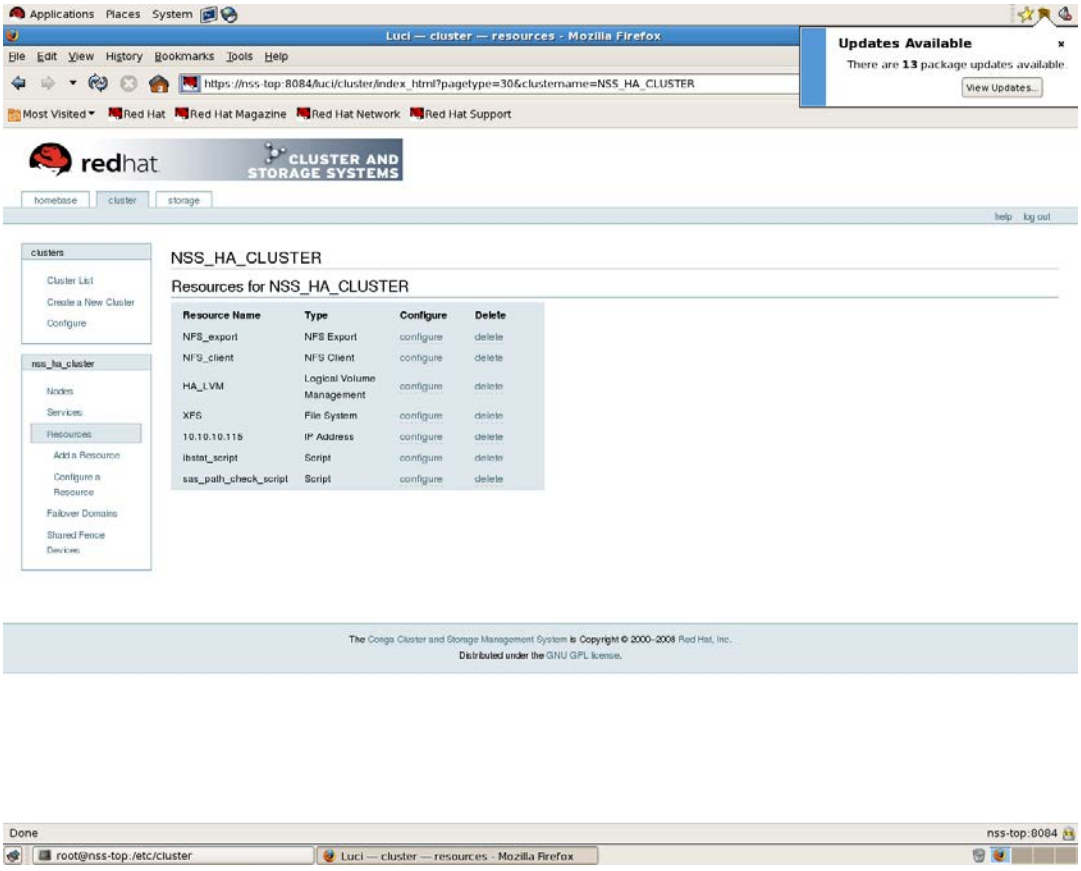
References https://bugzilla.redhat.com/show_bug.cgi?id=577913
https://bugzilla.redhat.com/show_bug.cgi?id=496749
http://linux.dell.com/wiki/index.php/Products/HA/DellRedHatHALinuxCluster/Cluster#Configure_iDRAC6_Fencing

b) Now increment the version number by one and propagate changes to all nodes.

```
ccs_tool update /etc/cluster/cluster.conf
```

Alternately, an updated version of the luci rpm can be used that incorporates these fixes.
luci rpm version : luci-0.12.2-24.el5_6.1.x86_64.rpm
Reference: http://rhn.redhat.com/errata/RHBA-2011-0033.html

Figure 19 - Cluster Resources



A.8. Quick test of HA set-up

1) Test fencing:
   - First disable the cluster service:
     `[root@active ~]# clusvcadm –d HA_service`
   - From the active server run the command `fence_node passive`. This should power cycle the passive server via DRAC. Check `/var/log/messages` on active.
   - From passive run the command `fence_node active`. This should power cycle the active server via DRAC. Check `/var/log/messages` on passive
   - Disconnect the DRAC cable on passive. From "active" run the command `fence_node passive`. This should power cycle the passive server via the switched PDU.
   - Disconnect the DRAC cable on active. From passive run the command `fence_node active`. This should power cycle the "active" server via the switched PDU.

2) Simple HA failover test:
   - Start the service on the "active" server
     `[root@active ~]# clusvcadm –e HA_service –m active`
   - Power down the active server by holding down the power button/pulling out both power cords.
   - Service should migrate to passive. Watch `/var/log/messages` and check `clustat` status on passive.

A.9. Useful commands and references

1) `clustat`

2) `cman_tool status`

3) If the `/etc/cluster/cluster.conf` file is edited manually, make the changes only on one server and increment the version number field. Use the following command to propagate the changes to both servers. `ccs_tool update /etc/cluster/cluster.conf`

4) `service cman status`

5) `service rgmanager status`

6) `clusvcadm` to manually stop, start, disable and relocate the cluster service.

7) `group_tool`

8) `rg_test` is a debugging tool. Use it with care. Ensure the cluster service is stopped and disabled before you use this tool.
   - `clusvcadm –s HA_service`
   - `clusvcadm –d HA_service`
   - `rg_test test /etc/cluster/cluster.conf start service HA_service`
     - `mount` will show the file system mounted.
     - `ip addr list` will show the floating resource IP configured.

- Clients will be able to access the file system, so use `rg_test` with care and ONLY for debugging. `clustat` and `clusvcadm` should manage the cluster and cluster service during normal operation.

  - `rg_test test /etc/cluster/cluster.conf status service HA_service`
  - `rg_test test /etc/cluster/cluster.conf stop service HA_service`
  - `clusvcadm –e HA_service`

9) To turn or reboot the active or passive server, it must first leave the cluster gracefully. Else the other server will assume that it has died and fence it. To leave the cluster gracefully, run the following sequence of commands.
   - `fence_tool leave`
   - `service rgmanager stop`
   - `cman_tool leave remove`
   - `service cman stop`

10) A very good FAQ on Red Hat Cluster Suite is at http://sources.redhat.com/cluster/wiki/FAQ

A.10. Performance tuning on clients (updated May 2011)

1) If the clients access the NFS server via 10GbE, configure the MTU on the 10GbE device to be 8192 for all the clients. Note that the switches need to be configured to support large MTU as well.

   On the client, if the value of MTU is not specified in `/etc/sysconfig/network-scripts/ifcfg-ethX`,

   ```
   echo “MTU=8192” >> /etc/sysconfig/network-scripts/ifcfg-ethX,
   ```

   Otherwise change the old value to 8192 in `/etc/sysconfig/network-scripts/ifcfg-ethX.`

   where `ifcfg-ethX` if the 10GbE network interface.

   Restart the networking services. `service network restart`

2) If the clients access the NFS server via 10GbE, the Ethernet switches on the fabric should have flow control disabled. The following instructions are for the PowerConnect 6248 and the PowerConnect 8024 and can be used as a reference.

   Start a console session to the switch and type the following commands in sequence:

   ```
   console>enable
   console#configure
   console(config)#no flowcontrol
   ```

   In order to verify the setting of the option, type the following command:

   ```
   console#show interface status
   ```

   A message of “**`Flow Control:Disabled`**” will be printed out.

3) For each client, Add the following to `/etc/sysctl.conf`

   ```
   # increasing the default TCP receive memory size
   net.ipv4.tcp_rmem = 4096 2621440 16777216
   ```

Activate the changes with `sysctl -p.`

A.11. Example scripts and configuration files

1) `/etc/cluster/cluster.conf` file for InfiniBand clusters
2) `/etc/cluster/cluster.conf` file for 10 Gigabit Ethernet clusters
3) `/root/ibstat_script.sh` file for InfiniBand clusters
4) `/root/sas_path_check_script.sh` file
5) `NSSHApolicy.te` SELinux policy Type Enforcement file

Appendix B: Medium to Large Configuration Upgrade

The Medium configuration consists of one PowerVault MD3200 and one PowerVault MD1200 and provides capacity of about 40TB. The Large configuration consists of one PowerVault MD3200 and three PowerVault MD1200s with a capacity of about 80TB.

To upgrade from a Medium configuration to a Large configuration, the two additional PowerVault MD1200s must be identical in configuration to the original set-up. Cable the two additional PowerVault MD1200s to the storage.

Similar to the storage configuration done for the initial Medium configuration, create a RAID 6, 10+2 virtual disk with segment size 512k on each new PowerVault MD1200. Add this virtual disk into the host group so that both active and passive servers can access the two new LUNS.

There are two paths to expand the XFS file system from a Medium configuration to a Large configuration.

Option 1: Extend the Medium configuration by an additional 40TB of capacity keeping the current file system intact. In this case:

- The first 40TB of data is stored on the original Medium configuration of one PowerVault MD3200 and one PowerVault MD1200. Data is striped across the PowerVault MD3200 and the first PowerVault MD1200.
- The second 40TB of data is striped across the two additional PowerVault MD1200s.
- The total capacity of the system is 80TB, similar to a Large configuration.
- The performance of the storage is similar to a Medium configuration.
- The file system and user data stay intact during the upgrade. However, it is recommended that the user data be backed up before the upgrade.

Option 2: The second option is to create a Large configuration on the storage. In this case:

- The user data must be backed up externally. This is a requirement. Data will not be preserved.
- The volume group, logical volume and file system is re-created using all four PowerVault MD storage enclosures. All 80TB of data is stripped across all four PowerVault MD arrays.
- The total capacity of the system is 80TB.
- The performance of the system is like a Large configuration.
- User data must be restored after the upgrade.

Implementing Option 1- Extending the Medium configuration

These steps must be executed on the server currently running the cluster service.

1) Backup user data.

2) Create new physical volumes on the two new LUNS.

```
pvcreate /dev/mapper/mpath4 /dev/mapper/mpath5
```

3) Extend the existing volume group to include these new physical volumes.

```
vgextend DATA_VG /dev/mapper/mpath4 /dev/mapper/mpath
```

4) Extend the existing logical volume to include these new physical volumes.

```
lvextend /dev/DATA_VG/DATA_LV /dev/mapper/mpath4 /dev/mapper/mpath5 -I
1024 –i 2
```

5) Extend the file system to include the additional space in the logicial volume.
```
xfs_growfs -d /mnt/xfs_data
```

6) Check the new file system size. `df –h` should show the total size of the file system to be close to 80TB, double what it was earlier.

7) Check that the user data is intact. Restore if needed.

Implementing Option 2 - Creating the Large configuration.

1) Back up user data. This is a requirement. Data will not be preserved.

2) Stop and disable the cluster service. Stop the cluster.
   On the server running the cluster service:
```
clusvcadm –s HA_service
clusvcadm –d HA_service
fence_tool leave
service rgmanager stop
cman_tool leave remove
service cman stop
```
   On the other server:
```
fence_tool leave
service rgmanager stop
cman_tool leave remove
service cman stop
```

3) On both servers, edit `/etc/lvm/lvm.conf` to remove HA_LVM. Comment out the `volume_list` line.

4) On both servers, remake the initrd
```
mkinitrd -f /boot/initrd-2.6.18-194.el5.img 2.6.18-194.el5
```

5) On one server, remove the old logical volume
```
lvchange –a y /dev/DATA_VG/DATA_LV
lvremove /dev/DATA_VG/DATA_LV
```

6) On one server, remove the old volume group.
```
vgremove DATA_VG
```

The steps that follow are similar to the NSS HA configuration recipe.

7) On the active server, create new physical volumes on the two new LUNS.
```
pvcreate /dev/mapper/mpath4 /dev/mapper/mpath5
```

8) On the active server, create a volume group across all four LUNS. Use the same VG name as before; else the `/etc/cluster/cluster.conf` file will need to be edited.
```
vgcreate DATA_VG /dev/mapper/mpath2 /dev/mapper/mpath3
/dev/mapper/mpath4 /dev/mapper/mpath5
```

9) On the active server, create a logical volume using this volume group.
```
lvcreate -i 4 -I 1024 -l 100%FREE DATA_VG  -n DATA_LV
```

10) Set up HA LVM
    - On the active server, edit `/etc/lvm/lvm.conf` and change the volume list to
      `volume_list = [ "VolGroup00" , "@active" ]`
      where `VolGroup00` is the volume group that contains the "`/`" file system for the OS. "`active`" is the name of the server as defined in the `cluster.conf` file.

    - On the passive server, edit `/etc/lvm/lvm.conf` and change the volume list to
      `volume_list = [ "VolGroup00" , "@passive" ]`
      where `VolGroup00` is the volume group that contains the "`/`" file system for the OS. "`passive`" is the name of the server as defined in the `cluster.conf` file.

    - On both servers, remake the initrd
      `mkinitrd -f /boot/initrd-2.6.18-194.el5.img 2.6.18-194.el5`

11) On the active server, create the file system
```
mkfs.xfs -l size=128m /dev/DATA_VG/DATA_LV
```

12) Restart the cluster. This will activate the HA LVM, mount the XFS file system, export it via NFS and create the floating resource IP.

    Simultaneously on both servers:
```
service cman start
service rgmanager start
```

13) Check the new file system size. `df –h` should show the file system mounted and the total size of the file system to be close to 80TB, double what it was earlier.

14) Restore user data.

Appendix C: Benchmarks and Test Tools

The iozone benchmark was used to measure sequential read and write throughput (MB/sec) as well as random read and write I/O operations per second (IOPS).

The mdtest benchmark was used to test metadata operation performance.

The checkstream utility was used to test for data correctness under failure and failover cases.

The Linux dd utility was used for initial failover testing and to measure data throughput as well as time to complete for file copy operations.

C.1. IOzone

IOzone can be downloaded from http://www.iozone.org/. Version 3.353 was used for these tests and installed on both the NFS servers and all the compute nodes.

The IOzone tests were run from 1-64 nodes in clustered mode. All tests were N-to-N, i.e. N clients would read or write N independent files.

Between tests, the following procedure was followed to minimize cache effects:

- Unmount NFS share on clients.
- Stop the cluster service on the server. This umounts the XFS file system on the server.
- Start the cluster service on the server.
- Mount NFS Share on clients.

The following table describes the IOZone command line arguments.

IOzone ARGUMENT	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
-O	Give results in ops/sec.

For the sequential tests, file size was varied along with the number of clients such that the total amount of data written was 128G (number of clients * file size per client = 128G).

IOzone Sequential Writes

```
# /usr/sbin/iozone -i 0 -c –e –w –r 1024k –s 2g –t 64 -+n -+m ./clientlist
```

IOzone Sequential Reads

```
# /usr/sbin/iozone -i 1 -c -e -w -r 1024k -s 2g -t 64 -+n -+m ./clientlist
```

For the random tests, each client read or wrote a 2G file. The record size used for the random tests was 4k to simulate small random data accesses.

IOzone IOPs Random Access (Reads and Writes)

```
# /usr/sbin/iozone -i 2 -w -r 4k -I -O -w -+n -s 2G -t 1 -+m ./clientlist
```

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.

C.2. mdtest

`mdtest` can be downloaded from http://sourceforge.net/projects/mdtest/. Version 1.8.3 was used in these tests. It was compiled and installed on a NFS share that was accessible by compute nodes. `mdtest` is launched with `mpirun`. For these tests, OpenMPI from Mellanox OFED 1.5.1 was used.

The following table describes the `mdtest` command line arguments.

mpirun ARGUMENT	DESCRIPTION
-np	Number of Processes
--nolocal	Instructs mpirun not to run locally
--hostfile	Tells mpirun where the hostfile is
mdtest ARGUMENT	**DESCRIPTION**
-d	The 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
-u	unique working directory for each task
-C	Create files and directories
-R	Randomly stat files
-T	Only stat files and directories
-r	Remove files and directories left over from run

As with the IOzone random access patterns, the following procedure was followed to minimize cache effects during the metadata testing:

- Unmount NFS share on clients.
- Stop the cluster service on the server. This umounts the XFS file system on the server.
- Start the cluster service on the server.
- 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
```

C.3. Checkstream

The `checkstream` utility is available at http://sourceforge.net/projects/checkstream/. Version 1.0 was installed and compiled on the NFS servers and used for these tests.

First, a large file was created using the `genstream` utility. This file was copied to and from the NFS share by a client using `dd` to mimic write and read operations. Failures were simulated during the file copy process and the NFS service was failed over from one server to another. The resultant output files were checked using the `checkstream` utility to test for data correctness and ensure that there was no data corruption.

Given below is sample output of a successful test with no data corruption.

```
checkstream[genstream.file.100G]: ------------------------------------------------
checkstream[genstream.file.100G]: valid data for 107374182400 bytes at offset 0
checkstream[genstream.file.100G]: ------------------------------------------------
checkstream[genstream.file.100G]: end of file summary
checkstream[genstream.file.100G]: [valid data] 1 valid extents in 261.205032 seconds
(0.00382841 err/sec)
checkstream[genstream.file.100G]: [valid data] 107374182400/107374182400 bytes (100
GiB/100 GiB)
checkstream[genstream.file.100G]: read 26214400 blocks 107374182400 bytes in
261.205032 seconds (401438 KiB/sec), no errors
```

For comparison, here is an example of a failing test with data corruption in the copied file. For example, if the file system is exported via the NFS async operation and there is an HA service failover during a write operation, data corruption is likely to occur.

```
checkstream[compute-00-10]: ------------------------------------------------
checkstream[compute-00-10]: valid data for 51087769600 bytes at offset 45548994560
checkstream[compute-00-10]:
checkstream[compute-00-10]: ------------------------------------------------
checkstream[compute-00-10]: end of file summary
checkstream[compute-00-10]: [valid data] 1488 valid extents in 273.860652 seconds
(5.43342 err/sec)
checkstream[compute-00-10]: [valid data] 93898678272/96636764160 bytes (87 GiB/90 GiB)
checkstream[compute-00-10]: [zero data] 1487 errors in 273.860652 seconds (5.42977
err/sec)
checkstream[compute-00-10]: [zero data] 2738085888/96636764160 bytes (2 GiB/90 GiB)
checkstream[compute-00-10]: read 23592960 blocks 96636764160 bytes in 273.860652
seconds (344598 KiB/sec)
checkstream[compute-00-10]: ------------------------------------------------
checkstream[compute-00-10]: encountered 1487 errors, failing
```

C.4. dd

dd is a Linux utility provided by the coreutils rpm distributed with RHEL 5.5. It is used to copy a file. The file system was mounted at /mnt/xfs on the client.

To write data to the storage, the following command line was used.

```
# dd if=/dev/zero of=/mnt/xfs/file bs=1M count=90000
```

To read data from the storage, the following command line was used.

```
# dd if=/mnt/xfs /file of=/dev/null bs=1M
```