Dell HPC NFS Storage Solution – High Availability (NSS7.0-HA) Configuration

Dell HPC Engineering
May 2016

Xin Chen

A Dell Technical White Paper

Revisions

Date	Description
May 2016	Initial release

Contents

Executive summary

This Dell technical white paper describes the Dell NFS Storage Solution—High Availability configuration (NSS7.0-HA). The paper compares all available NSS-HA offerings so far, and provides performance results for a configuration with a storage system providing 480 TB of raw storage capacity.

The NSS-HA solution described here is designed to enhance the availability of storage services to the HPC cluster by using a pair of Dell PowerEdge servers and PowerVault storage arrays along with Red Hat HA software stack. The goal of the solution is to improve storage service availability and maintain data integrity in the event of possible failures or faults and to optimize performance in a failure-free situation.

1 Introduction

This white paper provides information on the latest Dell NFS Storage Solution - High Availability configurations with Dell PowerEdge 13th generation servers. The solution uses Dell PowerEdge servers and PowerVault storage arrays along with Red Hat High Availability software stack to provide an easy to manage, reliable, and cost effective storage solution for HPC clusters. It leverages the Dell PowerEdge 13th generation servers (R730s) and Red Hat Enterprise Linux 7.2 operating system (RHEL 7.2 OS) to deliver more powerful storage solutions than the earlier NSS-HA solutions. This version of the solution is NSS7.0-HA.

The design principle for this release remains the same as previous Dell NSS-HA solutions. The major updates between the current and the previous version of NSS-HA solution (NSS6.0-HA) are:

- The NFS servers are changed from Dell PowerEdge R630s to R730s
- The OS is updated from RHEL 7.0 to RHEL 7.2
- Intel Omni-Path Network Fabric is supported in NSS7.0-HA.

For more information about the NSS-HA solution family, read this technical white paper with the previous NSS-HA white papers and blogs[1] [2] [3] [4] [5].

The sections hereafter describe the technical details, evaluation method, and the expected performance of the solution.

2 Overview of NSS-HA solutions

Along with the current version, four versions of NSS-HA solutions have been released since 2011. This section provides a brief description of the NSS-HA solution, and lists the current available Dell NSS-HA offerings.

2.1 A brief introduction to NSS-HA solutions

The design of the NSS-HA solution for each version is similar. In general, the core of the solution is a high availability (HA) cluster[6], which provides a highly reliable and available storage service to HPC compute clusters by using a high performance network connection such as Intel Omni-Path (OPA), InfiniBand (IB), or 10 Gigabit Ethernet (10GbE).

The HA cluster consists of a pair of Dell PowerEdge servers and a network switch. The two PowerEdge servers have shared access to disk-based Dell PowerVault storage in a variety of capacities, and both are directly connected to the HPC cluster by using OPA, IB or 10GbE. The two servers are equipped with two fence devices: iDRAC8 Enterprise, and an APC Power Distribution Unit (PDU). If failures such as storage disconnection, network disconnection, and system stopping from functioning, etc., occur on one server, the HA cluster will failover the storage service to the healthy server with the assistance of the two fence devices; and also ensure that the failed server does not return to life without the administrator's knowledge or control.

The disk-based storage array is formatted as a Red Hat Scalable file system (XFS) and exported to the HPC cluster by using the NFS service of the HA cluster. Note that large capacity file systems (greater than 100 TB) have been supported since the 2nd version of NSS-HA solution [2].

Figure 1 shows the general infrastructure of the NSS-HA solution. For more information about NSS-HA solution, refer to the previous NSS-HA white papers [1] [2] [3].

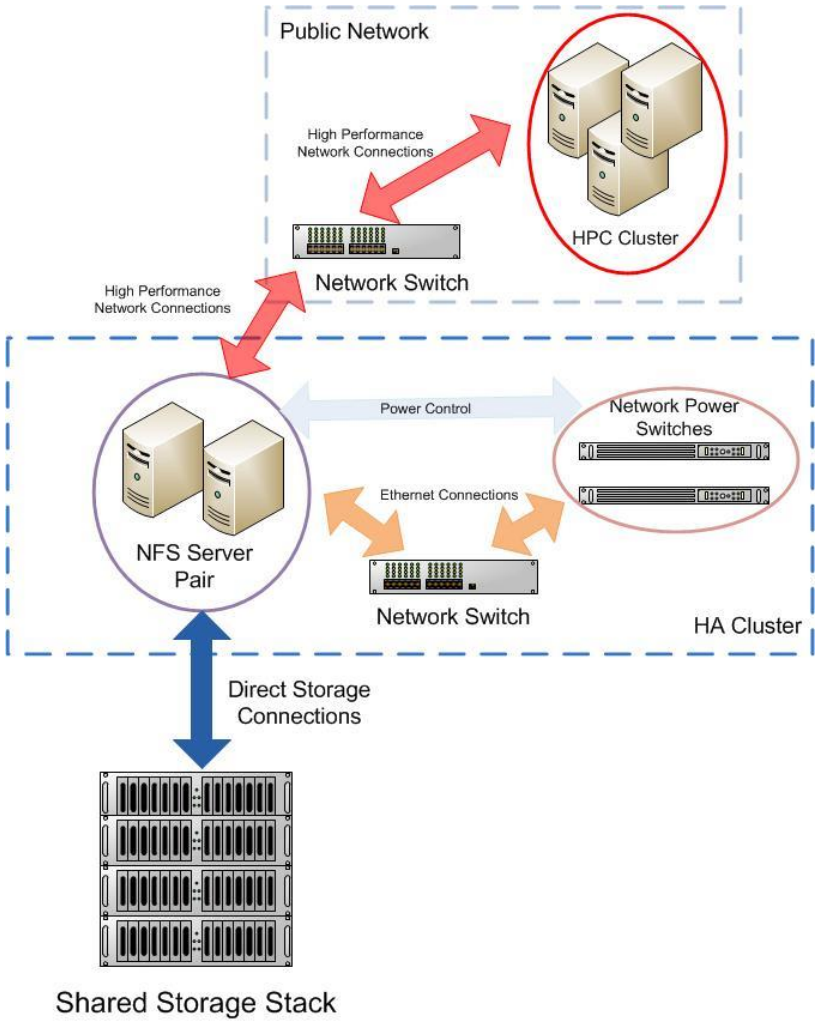

Figure 1 The infrastructure of the NSS-HA solution

Note: The iDRAC 8 Enterprise is not shown in the figure, and it is installed on each NFS server for Dell NSS-HA solutions. The term Network Power Switches refers to APC PDU (Power Distribution Unit) in Dell NSS-HA solutions.

2.2 NSS-HA offerings from Dell

Table 1 lists the available Dell NSS-HA solutions with standard configurations.

Table 1. NSS-HA Solutions[1], [2], [3], [4], [5]

	NSS6.0-HA Release (November 2014) "PowerEdge 13th generation server based solution"	NSS7.0-HA Release (May 2016) "PowerEdge 13th generation server based solution"
Storage Capacity	240 TB and 480 TB of raw storage capacity.	
Network Connectivity	FDR InfiniBand or 10 GbE Connectivity.	Intel Omni-Path, FDR InfiniBand, or 10GbE Connectivity.
Direct Storage Connection	12 Gbps SAS connections	
NFS servers	Dell PowerEdge R630 (1U) servers. CPU: Dual Intel Xeon E5-2697v3 @ 2.60 GHz, 14 cores per processor. Memory: 16 × 8 GiB 2133 MT/s RDIMMs.	Dell PowerEdge R730 (2U) servers. CPU: Dual Intel Xeon E5-2660v4 @ 2.0 GHz, 14 cores per processor. Memory: 8 × 16GiB 2400 MT/s RDIMMs.
Software	Red Hat Enterprise Linux 7.0 Red Hat Scalable File system (XFS) v3.2.0-0.10	Red Hat Enterprise Linux 7.2 Red Hat Scalable File system (XFS) v3.2.2-2
Storage Devices	Dell PowerVault MD3460 and Dell PowerVault MD3060e. 4 TB 7.2K NL SAS hard drives.	
Local Switch and Power Distribution Units (PDUs)	Dell Networking S3048-ON. Two APC-switched PDUs to manage high availability. For information about supported models of APC PDUs, see Fence device and Agent Information for Red Hat Enterprise Linux.	
Support and Services	Three years of Dell ProSupport for IT and mission-critical 4HR 7x24 onsite pack. Dell deployment services are available to speed up installation, optimize performance, and integrate NSS-HA solution with customer's HPC Cluster environment.	

Note:

- The table only lists NSS-HA versions currently available on the market. The NSS-HA versions before NSS6.0-HA are no longer available on the market.
- Contact your Dell Sales Representative to discuss which solution would be suited for your environment. You can order any of the preconfigured solutions or a customized solution designed to fulfill your requirements. On the basis of customization selected, some of the best practices discussed in this white paper may not apply.

3 Dell PowerVault MD3460 and MD3060e storage arrays

Both PowerVault MD3460[7] and MD3060e[8] storage array are 4U, 60-drive dense enclosures. The major difference between them is that the MD3460 has dual active-active RAID controllers and is used as an RBOD (Redundant Array of Inexpensive Disks Bunch of Disks), while, the MD3060e is an EBOD (expansion bunch of disks), which is usually used to extend the capacity of the PowerVault MD3460.

NSS7.0-HA shares the same storage configurations as NSS6.0-HA[5]. Table 2 summarizes the supported storage configurations of the NSS7.0-HA.

Table 2. Storage configurations in NSS7.0-HA

Storage configurations	240 TB: One PowerVault MD3460 480 TB: One PowerVault MD3460 + One PowerVault MD3060e
Disks in storage array	4 TB NL SAS.
Virtual disk configuration	• RAID6 8+2, a virtual disk is created across all five drawers in a storage array, two drives per drawer. • Segment size 512 KiB • Write cache mirroring enabled • Read cache enabled • Dynamic cache read prefetch enabled
Storage enclosure cabling	Chain cabling scheme
Logical volume configuration	Stripe element size: 512 KiB Number of stripes: 6 Number of virtual disks per logical volume: • 6 virtual disks for 240 TB configuration • 12 virtual disks for 480 TB configuration

4 Evaluation

The architecture proposed in this technical white paper was evaluated in the Dell HPC lab. This section describes the test methodology and the test bed used for verification. Also provides information about the functionality tests. Performance tests and results are described in Section 5 later in this white paper.

4.1 Method

The NFS Storage Solution described in this solution guide was tested for HA functionality and performance. A 480 TB NSS7.0-HA configuration was used to test the HA functionality of the solution. Different types of failures were introduced and the fault-tolerance and robustness of the solution was verified. Section 4.3 describes these HA functionality tests and their results. HA functionality testing was similar to the work done in the previous versions of the solution [(1), (2),(3),(4),(5)].

4.2 Test Bed

The test bed used to evaluate the NSS7.0-HA functionality and performance is shown in Figure 2. The following configuration was used.

- A 32-node HPC compute cluster (also known as "the clients") was used to provide I/O network traffic for the test bed.
- A pair of Dell PowerEdge R730 servers were configured as an active-passive HA pair and function as a NFS server for the HPC compute cluster.
- Both NFS servers were connected to a shared Dell PowerVault MD3460 storage enclosure extended with one Dell PowerVault MD3060e storage enclosure (Figure 2 shows a 480 TB solution with the two PowerVault MD storage arrays) at the back-end. The user data is stored on an XFS file system created on this storage. The XFS file system was exported to the clients by using NFS.
- The NFS servers were connected to the clients by using the public network. This network was Intel OPA.
- For the HA functionality of the NFS servers, a private 1 Gigabit Ethernet network was configured to monitor server health and heartbeat, and to provide a route for the fencing operations by using a Dell Networking 3048-ON Gigabit Ethernet switch.
- Power to the NFS servers was provided by two APC switched PDUs on two separate power buses.

Complete configuration details are provided in Table 3, Table 4, and Table 5.

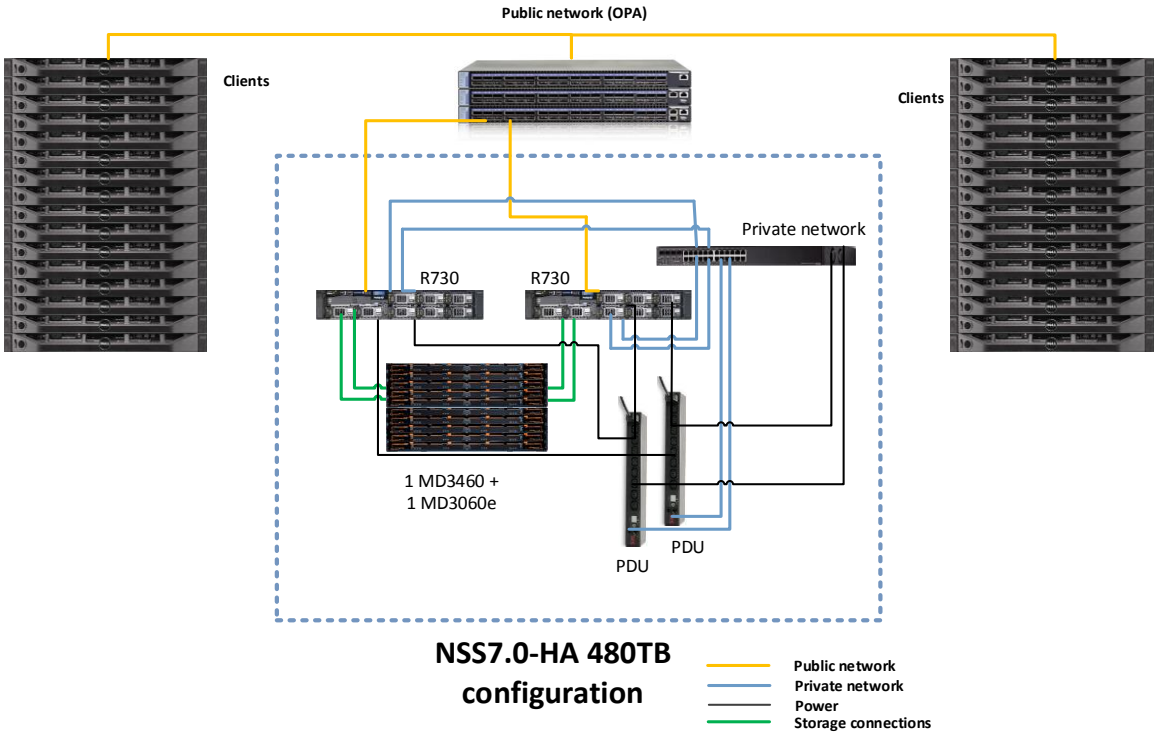

Figure 2 NSS7.0-HA test bed

Table 3. NSS7.0-HA hardware configuration

Server configuration	
NFS server model	Two Dell PowerEdge R730s
Processor	Dual Intel Xeon E5-2660 v4 @ 2.0 GHz, 14 cores per processor (The test bed used E5-2660 v3 @ 2.60 GHz.)
Memory	8 x 16 GiB 2400 MT/s RDIMMs. (The test bed used 16 x 8GiB 2133 MT/s RDIMMS.)
Local disks and RAID controller	PERC H730 with five 300GB 15K SAS hard drives. Two drives are configured in RAID1 for the OS, two drives are configured in RAID0 for swap space, and the fifth drive is a hot-spare for RAID1 disk group.
Optional OPA HCA (slot 4)	Intel Omni-Path Host Fabric Interface (HFI)
1 GbE Ethernet Card (daughter card slot)	Broadcom 5720 QP 1 Gigabit Ethernet network daughter card

SAS HBAs (slot 1 and slot 6)	Two 12 Gbps SAS HBAs
Systems Management	iDRAC8 Enterprise version
Power Supply	Dual Power Supply Units (PSUs)
Storage configuration	
Storage Enclosure	One Dell PowerVault MD3460 array and one MD3060e array for the 480TB solution
RAID controllers	Duplex RAID controllers in the Dell MD3460
Hard Disk Drives	60 numbers of 4 TB, 7.2K NL SAS drives per array
Other components	
Private Gigabit Ethernet switch	Dell Networking S3048-ON
Power Distribution Unit	Two APC switched Rack PDUs, model AP7921

Table 4. NSS7.0-HA software versions

Software	
Operating system	Red Hat Enterprise Linux (RHEL) 7.2 x86_64
Kernel version	3.10.0-327.el7.x86_64
Cluster Suite	Red Hat Cluster Suite from RHEL 7.2
File system	Red Hat Scalable File System (XFS) 3.2.2-2
Systems management tool	Dell OpenManage Server Administrator 8.3.0
Storage management package	Consolidated Resource DVD (RDVD): 6.3.0.10

Table 5. NSS7.0-HA client cluster configuration

	Client / HPC Compute Cluster
Clients	32 PowerEdge R630s Each compute node has: • CPU: Dual Intel Xeon E5-2697 v4 @ 2.3GHz, 18 cores per processor • Memory: 16 x 8GiB 2400 MT/s • Red Hat Enterprise Linux 7.1, kernel 3.10.0-229.el7.x86_64
HCA card	Intel OPA HFI
Switch	Intel Omnipath Fabric Edge Switch

4.3 HA functionality

The HA functionality of the solution was tested by simulating several component failures. The design of the tests and the test results are similar to previous versions of the solution because the general architecture of the solution has not changed in this release. This section reviews the failures and fault-tolerant mechanisms in NSS-HA solutions, and then presents the HA functionality tests with regards to different potential failures and faults.

4.3.1.1 Potential failures and fault tolerant mechanisms in NSS-HA

There are many different types of failures and faults that can impact the functionality of NSS-HA. Table 6 lists the potential failures that are tolerated in NSS-HA solutions.

Note: The analysis below assumes that the HA cluster service is running on the active server; the passive server is the other component of the cluster.

Table 6. NSS-HA mechanisms to handle failures

Failure type	Mechanism to handle failure
Single local disk failure on a server	Operating system installed on a two-disk RAID1 device with one hot-spare. Single disk failure is unlikely to make server non-functioning.
Single server failure	Monitored by the cluster service. Service fails over to passive server.

Failure type	Mechanism to handle failure
Power supply or power bus failure	Dual PSUs in each server. Each PSU is connected to a separate power bus. Server continues functioning with a single PSU.
Fence device failure	iDRAC8 Enterprise used as primary fence device. Switched PDUs used as secondary fence devices.
SAS cable/port failure	Two SAS cards in each NFS server. Each card has a SAS cable to each controller in the shared storage. A single SAS card or cable failure will not impact data availability.
Dual SAS cable/card failure	Monitored by the cluster service. If all data paths to the shared storage are lost, service fails over to the passive server.
OPA / InfiniBand / 10GbE link failure	Monitored by the cluster service. Service fails over to passive server.
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 Dell PowerVault MD3460 storage array	Dual controllers in the Dell PowerVault MD3460. The second controller handles all data requests. Performance may be degraded, but functionality is not impacted.

4.3.1.2 HA tests for NSS-HA

Functionality was verified for an NFSv4-based solution. The following failures were simulated on the cluster with the consideration of the failures and faults listed Table 6.

- Server failure
- Heartbeat link failure
- Public link failure
- Private switch failure
- Fence device failure
- Single SAS link failure
- Multiple SAS link failures

The NSS-HA behaviors in response to these failures are:

- Server failure: Simulated by introducing a Kernel panic. When the active server stops functioning, 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. After fencing is successful, the passive server takes

ownership of the cluster service. Clients cannot access the data until the failover process is completed.

- 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 server because 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.
- Public link failure: Simulated by disconnecting the Intel OPA, 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.
- Private switch failure: Simulated by turning 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 because the network is unavailable and the HA service continues to run on the active server.
- Fence device failure: Simulated by disconnecting the iDRAC8 Enterprise cable from a server.
  If the iDRAC on a server stops responding, the server is fenced by using the network PDUs, which are defined as secondary fence devices during the configuration.

  For the above cases, it was observed that the HA service failover takes in the range of 30–60 seconds. In a healthy cluster, any failure event must be noted by the Red Hat cluster management daemon and acted upon within minutes. Note that this is the failover time on the NFS servers; the impact to the clients could be longer.

- Single SAS link failure: Simulated by disconnecting one SAS link between the Dell PowerEdge R630 server and the Dell PowerVault MD3460 storage.
  In case where only one SAS link fails, the cluster service is not interrupted. Because 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 does not trigger a cluster service failover.
- Multiple SAS link failures: Simulated by disconnecting all SAS links between one Dell PowerEdge R630 server and the Dell PowerVault MD3460 storage.
  When all SAS links on the active server fail, the HA service will attempt to failover to the passive server. At this point, the passive server fences the active server, restarts the HA service, and provides a data path again to the clients. This failover can usually take less than two minutes.

4.3.1.3 Impact to clients

Clients mount the NFS file system exported by the server by using the HA service IP. This IP is associated with either OPA, IPoIB or a 10 Gigabit Ethernet network interface on the NFS 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 clients and the file system. Component failures were introduced into the server while the clients were actively reading and writing data from/to the file system.

In all tests, the client processes completed the read and write operations successfully. As expected, the client processes take longer to complete if the process was actively accessing data during a failover event.

During the failover period, when the data share is temporarily unavailable, the client processes were in an uninterruptible sleep state.

Depending on the characteristics of the client processes, the processes can be expected to either stop abruptly 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 after the failover is completed.

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

The information about `IOzone`, `checkstream`, and `dd` is available in Appendix A of this technical white paper.

5 NSS7.0-HA I/O Performance

This section presents the results of the I/O performance tests for the current NSS-HA solution. All performance tests were conducted in a failure-free scenario to measure the performance of the solution. The tests focused on two types of I/O patterns: large sequential read- and write operations, and small random read- and write operations.

The 480 TB configuration was benchmarked with Intel OPA cluster network connectivity. The 32-node compute cluster described in section 4.2 is used to generate workload for the benchmarking tests. Each test was run over a range of clients to test the scalability of the solution.

The `IOzone` tool was used in this study, which was used for the sequential and random tests. For sequential tests, a request size of 1024 KiB was used. The total size of data transferred was 512 GiB to ensure that the NFS server cache was saturated. Random tests used a 4 KiB request size and each client read and wrote a 4 GiB file. Metadata tests were performed using the `mdtest` benchmark and included file create, stat, and remove operations. All of the test results were based on NFSv4. For a list of commands used in the tests, see Appendix A later in this technical white paper.

5.1 OPA sequential writes and reads

In the sequential write and read tests, the I/O access patterns are N-to-N. That is, each client reads and writes to its own file. Iozone was run in clustered mode and one thread was launched on each compute node. As the total transferred data size was kept constant at 512 GiB, the file size per client varied accordingly for each test case. For example, 512 GiB file was read or written in one-client test case, 256 GiB file was read or written per client node in two-client test case.

Figure 3 shows the sequential write and read performance. The figure shows the aggregate throughput that can be achieved when a number of clients are simultaneously writing or reading from the storage over the Intel OPA network fabric.

During the benchmark tests, we observed that

- The max read performance was up to 6.57 GB/sec
- The max write performance was up to 1.84 GB/sec.

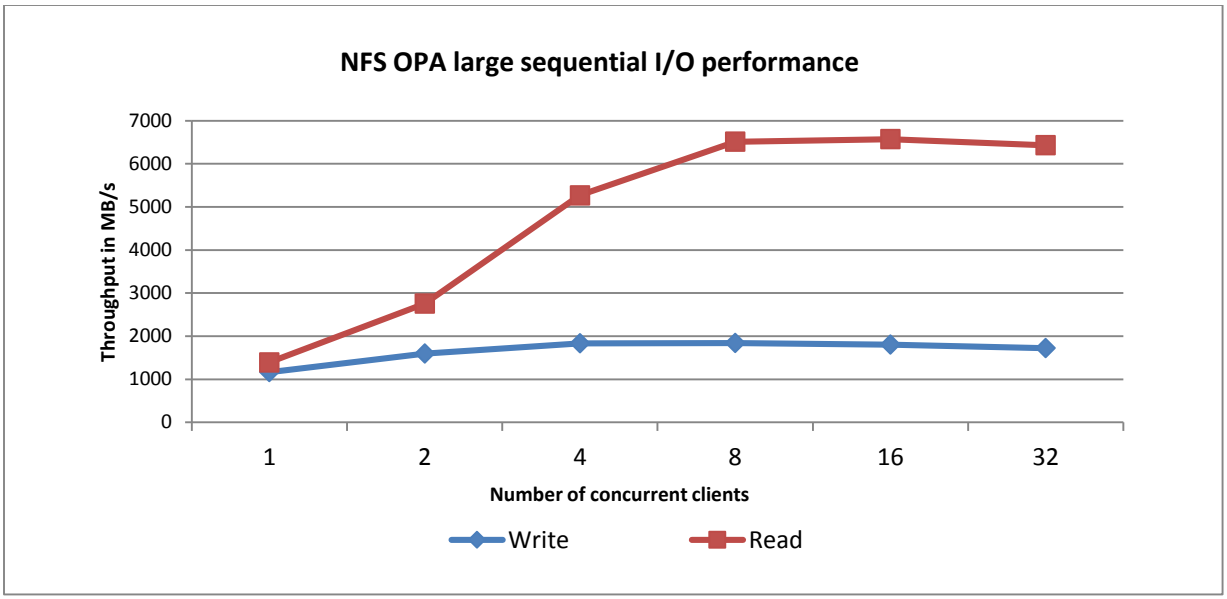

Figure 3 OPA large sequential write and read performance

5.2 OPA random write and read operations

Figure 4 shows the random write and read performance. The figure shows the aggregate I/O operations per second when a number of clients are simultaneously writing or reading to or from the storage over the Intel OPA fabric.

During the benchmark tests, we observed that

- The max random write performance was up to 5.6K IOPS
- The max random read performance was up to 18.7K IOPS.

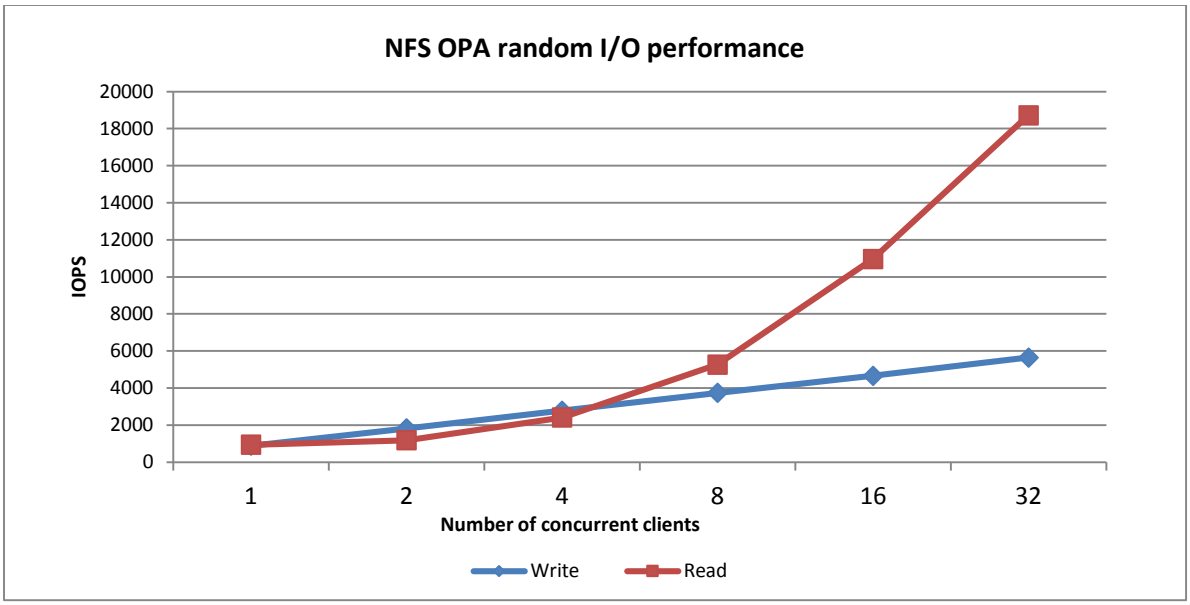

Figure 4 OPA random write and read performance

5.3 OPA metadata operations

Figure 5, Figure 6, and Figure 7 show the results of file create, stat, and remove operations respectively. Because the test bed has 32 compute nodes, in the graphs here, each client executed one thread for client counts up to 32. For client counts of 64, 128, 256, and 512, each client executed 2, 4, 8, or 16 simultaneous operations.

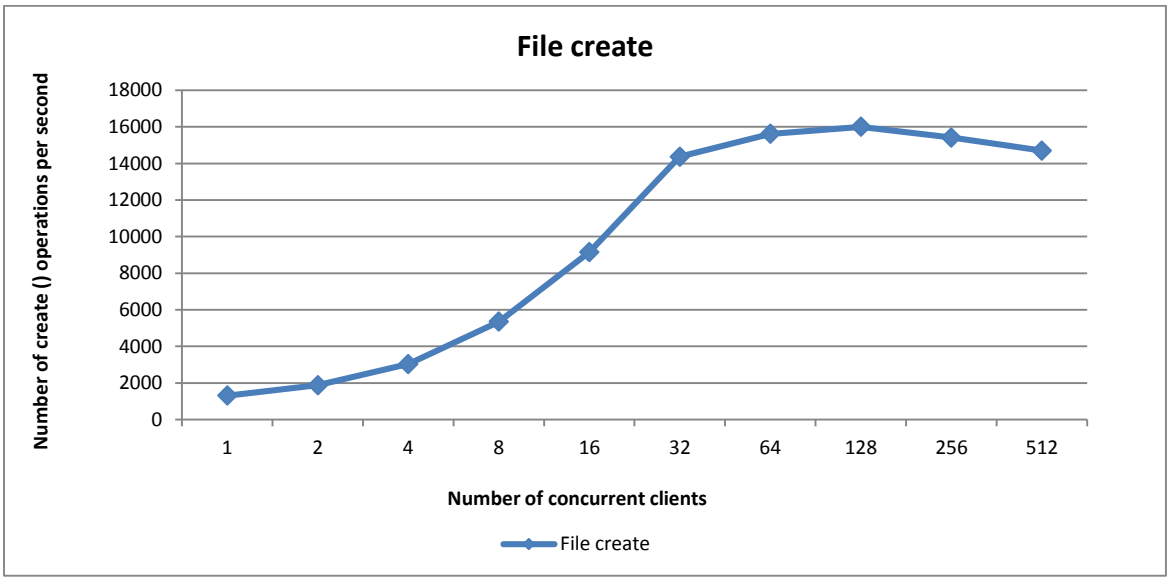

Figure 5 OPA file create performance

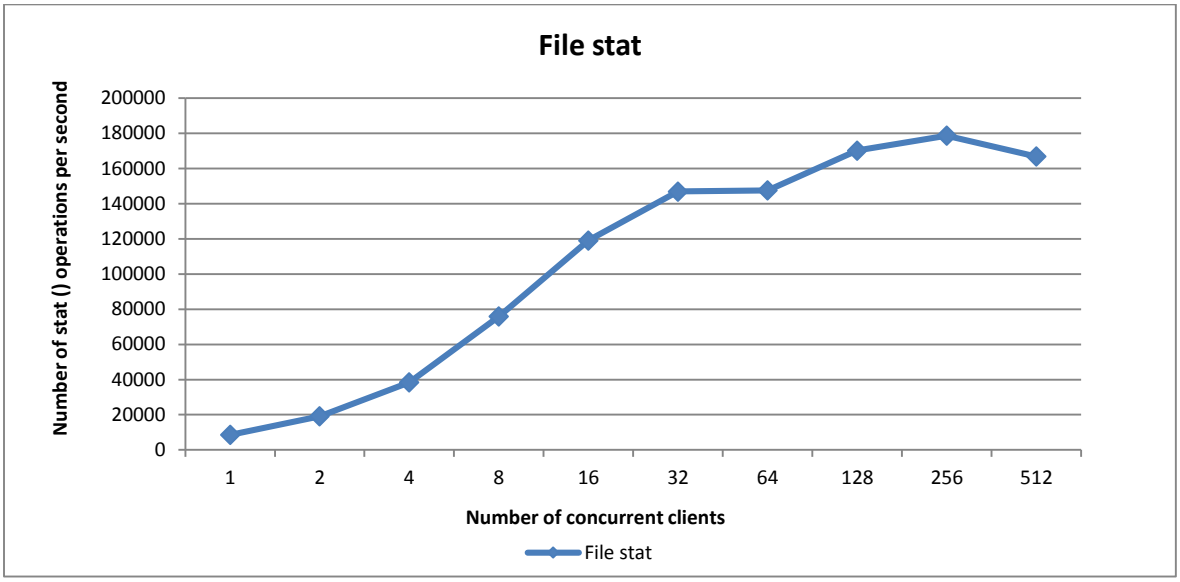

Figure 6 OPA file stat performance

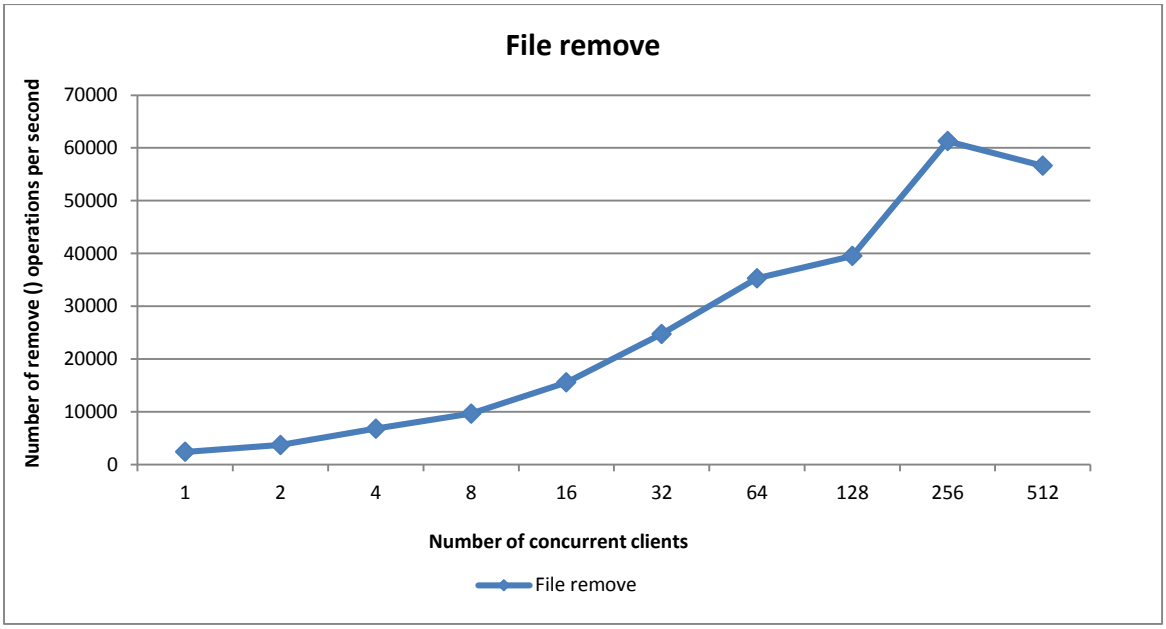

Figure 7 OPA file remove performance

6 Summary

This Dell technical white provides information about the latest Dell HPC NSS-HA Solution, including the solution configuration, HA functionality evaluation, and perfomrnace evaluation of the solution. With this version, the Dell NSS7.0-HA solution supports the Intel OPA network connection and delivers good sequential and random I/O performance. The Dell NSS-HA solution is available with deployment services and full hardware and software support from Dell.

7 References

1. Dell HPC NFS Storage Solution High Availability Configurations, Version 1.1
   http://i.dell.com/sites/content/business/solutions/whitepapers/en/Documents/dell-hpc-nssha-sg.pdf
2. Dell HPC NFS Storage Solution — High availability with large capacities, Version 2.1
   http://i.dell.com/sites/content/business/solutions/engineering-docs/en/Documents/hpc-nfs-storage-solution.pdf
3. Dell HPC NFS Storage Solution - High Availability Solution NSS5-HA configurations,
   http://en.community.dell.com/techcenter/high-performance-computing/b/hpc_storage_and_file_systems/archive/2013/10/28/dell-hpc-nfs-storage-solution-high-availability-solution-nss5-ha-configurations.aspx
4. Dell HPC NFS Storage Solution High Availability (NSS5.5-HA) Configuration with Dell PowerVault MD3460 and MD3060e Storage Arrays, Version 1.0
   http://i.dell.com/sites/doccontent/shared-content/solutions/en/Documents/Nss5-5-ha-v1-0_final.pdf
5. Dell HPC NFS Storage Solution High Availability (NSS6.0-HA) Configuration with Dell PowerEdge 13th Generation Servers, Version 1.0
   http://i.dell.com/sites/doccontent/business/solutions/whitepapers/en/Documents/dell-nfs-hpc-storage-solution-nss6-0-ha.pdf
6. Red Hat Enterprise Linux 7 High Availability Add-On Reference
   https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/High_Availability_Add-On_Reference/index.html
7. Manuals & documentation for your PowerVault MD3460
   http://www.dell.com/support/home/us/en/19/product-support/product/powervault-md3460/manuals
8. Manuals & documentation for PowerVault MD3060e
   http://www.dell.com/support/home/us/en/19/product-support/product/powervault-md3060e/manuals

Appendix A: Benchmarks and test tools

- The `IOzone` benchmark tool was used to measure sequential read- and write throughput (MBps) and random read- and write I/O operations per second (IOPS).
- 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, to measure data throughput, and the time to complete file copy operations.

A.1. IOzone

You can download `IOzone` from http://www.iozone.org/. Version 3.420 was used for these tests and installed on both the NFS servers and all the compute nodes.

The `IOzone` tests were run from 1–32 nodes in clustered mode. All tests were N-to-N. Meaning, 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 unmounts the XFS file system on the server.
- Start the cluster service on the server.
- Mount NFS Share on clients.

Table 7 describes the `IOzone` command line arguments.

Table 7. Appendix A – 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` when in clustered mode
-w	Does not unlink (delete) temporary file

IOzone Argument	Description
-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 512 GiB (number of clients * file size per client = 512GiB).

`IOzone` Sequential Writes

```
# /usr/sbin/iozone -i 0 -c –e –w –r 1024k –s 16g –t 32 -+n -+m ./clientlist
```

`IOzone` Sequential Reads

```
# /usr/sbin/iozone -i 1 -c -e -w -r 1024k -s 16g -t 32 -+n -+m ./clientlist
```

For the random tests, each client read or wrote a 4GiB file. The record size used for the random tests was 4KiB 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 4g -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.

A.2. mdtest

You can download `mdtest` from http://sourceforge.net/projects/mdtest/. Version 1.9.3 was used in these tests. It was compiled and installed on an NFS share that was accessible by compute nodes. `mdtest` is used with `mpirun`. For these tests, OpenMPI version 1.10.0 was used. The following table describes the `mdtest` command-line arguments.

Table 8. Appendix A – 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 located
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:

1. Unmount NFS share on clients.
2. Stop the cluster service on the server. This unmounts the XFS file system on the server.
3. Start the cluster service on the server.
4. Mount NFS Share on clients.

- Metadata file and directory creation test:

```
# 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 test:

```
# 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 test:

```
# 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
```

A.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 each client by using `dd` to simulate 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 by using the `checkstream` utility to test for data correctness and ensure that there was no data corruption.

A sample output of a successful test with no data corruption is given here:

```
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 by usinng 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
```

A.4. The dd Linux utility

`dd` is a Linux utility provided by the coreutils rpm distributed with RHEL 7.2. It was used to copy a file. The NFS file system was mounted at `/mnt/xfs` on the clients.

- 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
```