NFS SERVER WITH 10 GIGABIT ETHERNET NETWORKS

A Dell Technical White Paper

DEPLOYING 10GigE NETWORK FOR HIGH PERFORMANCE CLUSTERS

By Li Ou
Massive Scale-Out Systems
delltechcenter.com

Contents

Introduction

Cluster computing has become one of the most popular platforms for high-performance computing today. Like traditional parallel computing systems, the I/O subsystems of clusters may form a bottleneck that affects overall system performance. An efficient way to alleviate the I/O bottleneck is to deploy a high performance file system that utilizes the aggregate bandwidth and capability of existing I/O resources in the cluster. Doing so provides high performance and scalable storage service for cluster computing platforms.

While many parallel file systems gradually gain popularity on cluster computing platforms, NFS is still the most popular shared file system used in high performance clusters. Although NFS is not optimized for parallel I/O access, it provides decent performance for most applications. NFS is tightly coupled with Linux®/Unix® operating systems, therefore it is more cost effective than other parallel file systems, and is easy to configure, deploy, and maintain. For HPC customers who do not have heavily I/O intensive applications or are price constrained in architecting their clusters, NFS provides a robust solution.

On a typical HPC platform, high performance network infrastructure is dedicated for inter-process computational communications (IPC). In such systems, NFS is normally built on a GigE fabric, isolated from the sensitive IPC traffic. For a single client or small number of clients, the GigE network bandwidth is enough to support NFS traffic. However, as the number of clients increases, the aggregate bandwidth requirement between an NFS server and the multiple NFS clients can quickly exceed the capability of a GigE fabric. The bottleneck is between an NFS server, and the GigE switching fabric, because all client requests are aggregated there. For cost reasons, most customers do not invest in a high performance network for NFS. Therefore, the NFS potential of aggregate performance is seriously limited by the GigE fabric.

10 Gigabit Ethernet (10 GigE) network devices provide a simple but cost effective way to boost NFS performance in an HPC system. The combination of 10GigE host cards for NFS servers and 10GigE aggregation modules for GigE switches eliminates the network bottleneck between NFS servers and clients. The hardware cost is considerably lower than investing in a new network fabric because the on-board GigE cards of the compute nodes are reused. Another obvious benefit of this solution is that NFS architecture and software is kept unchanged.

This document covers deploying a 10GigE network with NFS on Dell™ HPC clusters, HW/SW configuration best practices, and the installation process. Benchmark testing using this architecture in a 32-node testing cluster shows that 10GigE network devices greatly improve aggregated sequential I/O performance of NFS, while moderately increasing the bandwidth for random I/O.

10GigE Network Standard

The 10 GigE standard provides a data rate of 10 Gbit/s, up to ten times as fast as Gigabit Ethernet (GigE). The 10 Gigabit Ethernet standard encompasses a number of different physical layer, including fiber and copper. There are several sub-standards for both fiber and copper layers.

The most popular **fiber** standards are 10GBASE-SR and 10GBASE-LR. 10GBASE-SR uses 850 nm lasers and multi-mode fiber to support short distances network, with a range from 26 meters to 82 meters. 10GBASE-SR can be extended up to 300 meters with a new OM3 fiber. 10GBASE-LR is a Long Range Optical technology with 1310 nm lasers and single-mode fiber. It could be extended up to 25 kilometers with no data loss.

The two most popular **copper** standards are 10GBASE-CX4 and 10GBASE-T. 10GBASE-CX4 has the lowest cost per port of all 10GigE interconnects. It is designed to transmit signal over four copper lanes in each direction similar to the technology used in InfiniBand™. The maximum distance of 10GBASE-CX4 is 15 meters. 10GBASE-T uses unshielded or shielded twisted pair cables over distances up to 100 meters. The advantage of 10GBASE-T is to allow a gradual upgrade from 1000BASE-T and autonegotiation to select which speed to use. However, the downsides of this flexibility are higher latency and more power consumption than other 10GigE physical layers.

Benefits of Enabling 10GigE Network for NFS Servers

Deploying NFS servers with 10GigE network on Dell HPC clusters provides a cost effective solution for customers who do not have heavily I/O intensive applications. The key benefits of this solution include the following:

- **High-performance:** The combination of 10GigE host cards for NFS servers and 10GigE aggregation modules for GigE switches is designed to eliminate the network bottleneck between NFS servers and clients. The aggregated bandwidth requirement for a NFS server and the multiple NFS clients is satisfied.

- **Cost Effective:** The hardware cost is considerably lower than investing in a new network fabric because the on-board GigE cards of the compute nodes are used. Customers may only need to purchase a few new hardware devices including the NFS server host 10GigE cards and 10GigE aggregator modules of Ethernet switches. Moreover, there are no additional software costs because the NFS architecture and software are unchanged.

- **Deployment and Manageability:** Using 10GigE interconnects makes no changes to NFS software stack, so minimum updating is required. There are no configuration changes on the compute nodes, which maintain the manageability of NFS architecture. The solution described in this paper builds on Dell's standard high performance clustering architecture that already provides exceptional flexibility and management.

- **Reliability and Robustness:** After decades of wide deployment and bug-fixing, NFS has become a very reliable and robust network file system used in high performance clusters. Enabling a 10GigE network does not require any changes to the native NFS software stack.

Architecture

Dell offers all necessary components to deploy a NFS file system with a 10GigE network, including high performance Dell PowerEdge™ servers, 10GigE host cards, Ethernet™ switches with 10GigE modules, and PowerVault™ storage devices. Figure 1 illustrates the reference hardware architecture for such a solution.

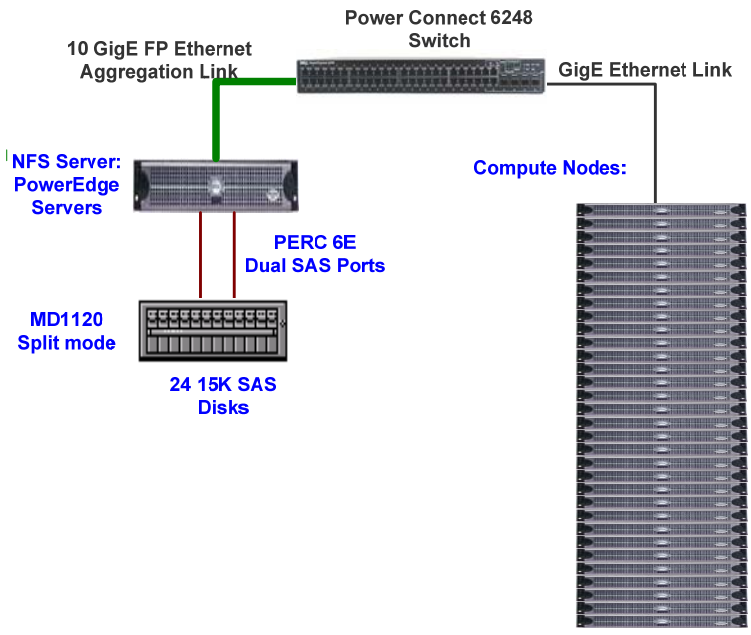

Figure 1. Hardware components of 10GigE NFS system

The PowerConnect™ 6248 switch is a 48-port Gigabit Ethernet Layer 3 switch, with 10GigE uplinks, stacking, and IPv6 (Internet Protocol Version 6) capabilities. It is able to support up to four 10 Gigabit Ethernet uplinks directly to core equipment such as 10GigE routers, Enterprise Backbones and Data Centers. In addition, the product family supports stacking up to 12 switches at 48 Gbps, low 10GigE latency, advanced security including 802.1x port authentication, and advanced QoS (Quality of Service) that make it ideal for applications such as Layer 3 routing, High Performance Cluster Computing (HPCC), and iSCSI storage.

Compute nodes of a cluster are connected to PowerConnect 6248 switches with GigE links. 10GigE uplinks from PowerConnect 6248 to 10GigE host cards of the NFS servers help eliminate the network bottleneck between NFS servers and clients. Dell offers the Intel® 10 Gigabit XF SR Server Adapter to enable 10GigE links for NFS servers. The Intel® 10 Gigabit XF SR Server Adapter is a PCIe x8 v1.1, nanometer optical fiber adapter that has driver support for Microsoft® Windows® and Linux operating systems. It is compliant with IEEE 802.3ae, IEEE 802.3ac, IEEE 802.3X, and IEEE 802.3z. It supports a Jumbo Frame (up to 9,000 bytes) for use with larger IO packets.

Dell offers a wide range of high performance PowerEdge servers, from 1U to 4U, to enable NFS services in a HPC platform. At the backbone, PowerVault MD1120 storage systems are used to support NFS servers with high performance direct-attached block level services. The MD1120 is a Serial Attached SCSI (SAS) connected external hard drive enclosure. It can be connected to a SAS/RAID controller for use as a Just a Bunch of Disks (JBOD) enclosure, and supports up to 24x 2.5" SAS hot-pluggable hard disk drives, either 15K RPM SAS drives available in 36 GB[1] and 73 GB[1] capacity, or 10K RPM SAS drives available in 73 GB[1] and 146 GB[1] capacity. The maximum capacity is 3.5 TB[1] using 24x 146GB[1] SAS hard drives. The enclosure supports dual Enclosure Management Modules (EMM), and each has one x4 3-Gb SAS Host Connectivity. With dual EMMs, MD1120 has two distinct modes: unified and split mode. In unified mode, a host directly connects to all 24 disk drives per enclosure and supports daisy chaining of up to three enclosures. In split mode, the enclosure provides direct connectivity for hosts to drives 0 to 11, and separate connectivity to drives 12 to 23. In the Figure 1, a NFS server with a Dell PERC6 RAID controller connects to a MD1120 in the split mode, fully utilizing dual SAS connections to maximize I/O performance.

The standard NFS software stack is unchanged, and no configuration modifications are required on the compute nodes. The Linux driver of the Intel 10 Gigabit XF SR Server Adapter needs to be installed on the NFS servers.

Best Practices for Deploying NFS Servers with 10GigE Network on Dell HPC Clusters

Platform Open Cluster Stack (Platform OCS) Dell Edition is a modular software stack that enables the consistent deployment and management of scale-out Linux clusters. For more information, see **http://www.platform.com/Products/platform-open-cluster-stack5**. One advantage of such a stack is seamless integration of various components. The version used in this study was Open Cluster Stack is OCS 5.1, which is based on Redhat® Enterprise Linux 5.1.

Using OCS 5.1 to automate deployment of NFS servers and clients is strongly recommended. Once the deployment process is completed, one needs to manually configure the 10GigE network and optimize I/O performance

Deploying NFS Servers With Platform OCS 5.1

With OCS 5.1, creating a dedicated node group for NFS servers is recommended to simplify the deployment. The following steps create the node group and deploy NFS servers.

1. At the frontend of the OCS5.1 cluster, run the command `ngedit`.
2. At the **Node Group Editor** screen, use the arrow keys to select `compute-rhel-5-x86_64`, then press <Tab> to select `Copy` and press <Enter>.
3. A new node group “compute-rhel-5-x86_64 Copy 1” is created. Select this node group, and then edit.

4. Navigate to the **General Info** screen, change the **Node Group Name** to `NFS-rhel-5-x86_64`, and change the **Node Name Format** to `NFS-#RR-#NN`.
5. Navigate to the **Components** screen and disable the two components **lava** and **platform ofed**. The reason for disabling `lava` is that jobs are not allowed to schedule to NFS servers. NFS servers have 10GigE cards, so OFED is not necessary. `Platform ofed` contains some components not compatible with the kernel patches applied later.
6. Navigate to the **Summary of Changes** screen and apply **Accept**.
7. Exit **ngedit**.
8. Run the command `addhost`, select the newly created node group `NFS-rhel-5-x86_64`, and PXE an NFS server for deployment. After installation is complete, the name of the first NFS server will be `NFS-00-00`.

Tuning and kernel parameters

The NFS server implementation of Redhat Enterprise Linux 5.1 has two bugs: “drop in nfs-server rewrite performance compared to rhel 4.6+ ” (bug 436004: **https://bugzilla.redhat.com/show_bug.cgi?id=436004**), and “drop in cfq read performance compared to rhel 4.6+ ” (bug 448130: **https://bugzilla.redhat.com/show_bug.cgi?id=448130**). To correct those two bugs, Dell recommends applying kernel updates and modifying system parameters on NFS servers. Specifically, Red Hat Enterprise Linux 5.2 errata kernel (kernel 2.6.18-92.1.13.el5) addresses the issue of low performance with NFS rewrite. Download the kernel update rpms **(kernel-2.6.18-92.1.13.el5.x86_64.rpm**, and **kernel-devel-2.6.18-92.1.13.el5.x86_64.rpm**) from Redhat Network, and install those rpms on the NFS servers running Red Hat Enterprise Linux 5.1. After rebooting the system to the new kernel, the NFS rewrite performance will be back to an optimized level. There is no kernel patch to correct the NFS read issue in Red Hat Enterprise Linux 5.1, but a workaround for modifying system parameters may be used to address this issue. The workaround is to add the following line to the /etc/rc.local boot script and then to run that script: (bug 436004: **https://bugzilla.redhat.com/show_bug.cgi?id=436004**)

```
for n in /sys/block/sd*/queue/iosched/slice_idle ; do echo 1 > $n ; done
```

Adapter Driver

The latest version of Intel 10GigE adaptor driver at the time of this article is 1.3.31.5. The latest driver is available at **support.dell.com**. The file name for the current version is **Intel_LAN_10.5_Linux_DKMS_A00.tar.gz**. After decompressing the tar file, locate the rpm file **ixgbe-1.3.31.5-sb_dkms.noarch.rpm**. This is the Dell DKMS driver for Intel 10GigE adapters. Because DKMS driver compiles module files for the current kernel at the time of installation, Dell recommends installing the Intel driver after the kernel patch is applied and the server is rebooted to the latest kernel.

Configuring the NFS Server Daemon

The default value of the number of NFS server daemon is 8. It is defined with a key `RPCNFSDCOUNT` in the NFS configuration file **/etc/sysconfig/nfs**. Benchmark tests show that NFS systems experience a serious performance drop with large numbers of NFS clients at this default value. Therefore, a larger number is desired for a normal application. A value of 128 is a good start for a middle size cluster consisting of less than 128 nodes.

Configuring Clients to Mount NFS Services

Using OCS 5.1 to deploy NFS clients is strongly recommended. After clients are ready, the following steps mount NFS services on the clients.

1. At the frontend of the OCS5.1 cluster, add the following line to the file **/etc/cfm/compute-rhel-5-x86_64/etc/fstab.append**.

```
10GigE-IP-Address:/NFS-server-directory      local-mount-point     nfs
defaults,nfsvers=3,tcp,intr,noatime   0   0
```

`10GigE-IP-Address` is the IP address assigned to the 10GigE interface of the NFS server. NFS`-server-directory` is the directory name exported from the NFS server. `local-mount-point` is the directory used to mount NFS services on clients.

2. At the frontend of the OCS5.1 cluster, run the following command to create a directory used to mount NFS on all clients.

   ```
   pdsh –a “mkdir local-mount-point”
   ```

   `local-mount-point` is the same directory name added in **/etc/cfm/compute-rhel-5-x86_64/etc/fstab.append** on the frontend.

3. At the frontend of the OCS5.1 cluster, run `cfmsync -f` to trigger the commands on clients to modify **/etc/fstab**.
4. Restart all clients to mount the NFS service, or at the frontend of the OCS5.1 cluster, run the following command to mount NFS on all clients.

   ```
   pdsh –a “mount local-mount-point”
   ```

Deploying an NFS System with 10GigE

The following steps are recommended to deploy NFS system with 10GigE network.

1. Deploy NFS servers and clients with Platform OCS 5.1.
2. Apply kernel patch rpms (**kernel-2.6.18-92.1.13.el5.x86_64.rpm**, and **kernel-devel-2.6.18-92.1.13.el5.x86_64.rpm**) on NFS server.
3. Add “`for n in /sys/block/sd*/queue/iosched/slice_idle ; do echo 1 > $n ; done`” into **/etc/rc.local** on NFS server.
4. Modify `RPCNFSDCOUNT` in **/etc/sysconfig/nfs** to `128` on the NFS server.
5. Reboot the NFS server.
6. Install Intel 10GigE adapter driver **ixgbe-1.3.31.5-sb_dkms.noarch.rpm** on the NFS server.
7. Configure the IP address of the 10GigE interface on the NFS server.
8. Export a NFS directory through the 10GigE interface by modifying **/etc/exportfs** on the NFS server.
9. Restart nfs service with the command `service nfs restart` on the NFS server.
10. Configure clients to mount the NFS service though the IP address of 10GigE interface of the NFS server.

Performance Studies

To evaluate the effectiveness of this solution, a comprehensive performance study was conducted with the reference architecture shown in Figure 1. The benchmarking results for a 32-node test cluster show that 10GigE network devices greatly improved aggregated sequential I/O performances of NFS, while moderately increasing the bandwidth of random I/Os.

Experimental Setup

The test cluster deployed for this study had 32 compute nodes, and one NFS server. The NFS server was supported by an MD1120 box, and connected to a 10GigE module on the Ethernet switch. The cluster was installed and managed using Platform OCS 5.1.

Hardware Configuration

- NFS server: PowerEdge 2950
  - Memory: 16 GB[1]
  - Processor: 2 socket, quad core Intel 5460, 3.16 GHz
  - Intel XR FP 10GigE Ethernet adapter
  - PERC 6E card, RAID 5 configuration, Dual SAS link to storage
- Storage: MD1120
  - 24 Seagate 34 GB[1] 15K SAS drives
  - Split mode
- Compute nodes: 32 x PowerEdge SC1435
  - Memory: 4 GB[1] each
  - Processors: 2 socket, AMD Opteron 2218, 2.59GHz
  - On board Gigabit Ethernet interface
- Switch: Dell PowerConnect 6248 with 10GigE modules

Software Configuration

- OS: Redhat Enterprise Linux 5.1 (kernel version 2.6.18-54.el15) for NFS server and clients
- Intel 10GigE card driver: 1.3.31.5 for NFS server
- Kernel patch 2.6.18-92.1.13.el15 for NFS server
- Modify parameter of **/sys/block/sdb/queue/iosched/slice_idle** from `8` to `1` on NFS server

Benchmarks

To evaluate performance of the NFS system with 10GigE network, two popular benchmarks were used in this study: NetPerf, and Iozone.

Netperf was used to study the network bandwidth of the 10GigE network with Intel server adapters and Dell Power Connect 6248 switches. The result of netperf set a bandwidth boundary for network file systems and helped locate the bottlenecks of NFS. For more information on Netperf, go to **http://www.netperf.org/netperf/**

Iozone was used to determine the performance boundary of the local file system on NFS servers, the bandwidth boundary of the NFS protocol on the 10GigE network, and the aggregated bandwidth of multiple clients with both sequential and random access. Iozone has two working modes: single client and clustering. Single client mode was used to test the performance boundary of the local file system on NFS servers. Clustering mode was chosen to test aggregated bandwidth of multiple clients. For more information on Iozone, go to **http://www.iozone.org/**

Performance Analysis

The following section details the benchmarking results for the test architecture.

Network Bandwidth

With an Intel 10GigE adapter installed on one of the compute nodes (Figure 2), Netperf was used to test the network bandwidth of 10GigE links between the NFS server and a single client. Figure 3 compares the TCP bandwidth of various MTU sizes, showing that the 10GigE network provides a sustained bandwidth of 1200MB/s with an MTU size of 9000 bytes. The overall CPU utilization of netperf is 12.5% for the eight cores in the PowerEdge 2950 server.

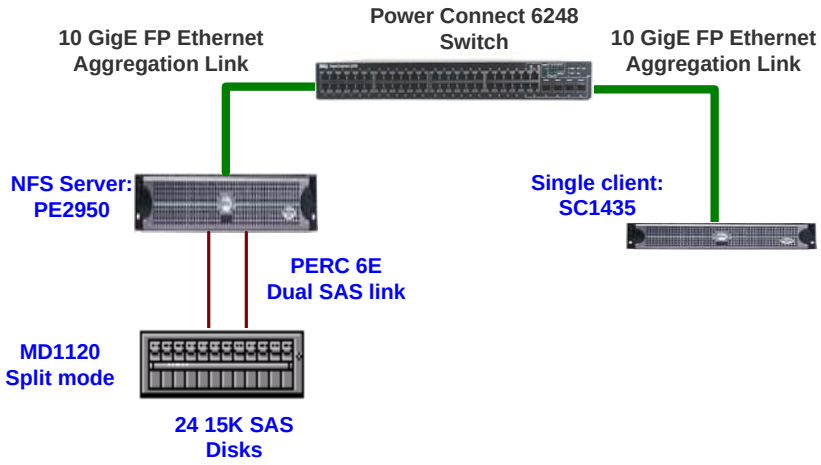

Figure 2. Hardware configuration to test NFS protocol with a single 10GigE client

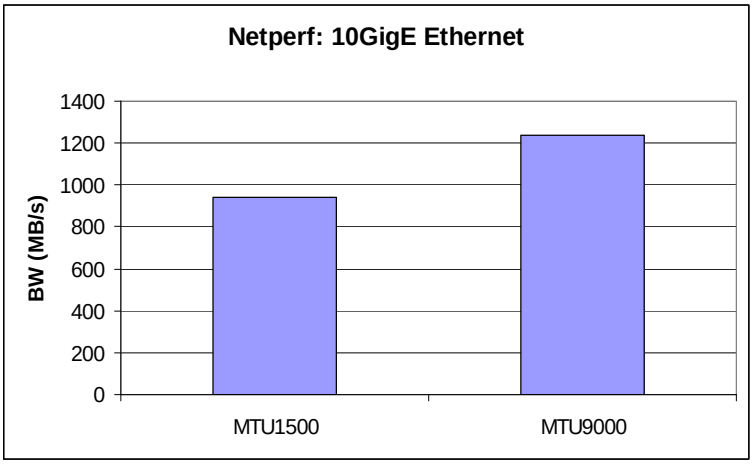

Figure 3. Network bandwidth of 10GigE network with 1500 and 9000 byte MTUs

Local File System on the NFS Server

Iozone was run on the NFS server to test the sequential bandwidth of the file system located on the attached MD1120 storage. Iozone reads/writes a 32GB file. The results could provide a performance boundary for the aggregated bandwidth of the NFS system. Figure 4 shows that the write bandwidth remained at 600MB/s on all request sizes and read ahead sizes, but the read ahead size of a hard disk could seriously impact read performance. The read ahead parameter is shown in terms of sector sizes where each sector is 0.5 KB. With the default read ahead size of 256 sectors, the read performance is relatively low, down to 300MB/s. After changing the read ahead parameter to 8192 sectors, the read performance increases up to 950MB/s. Figure 5 shows how the read ahead parameter impacted read performance, and confirmed that the maximum performance is reached with the read ahead size of 8192 sectors.

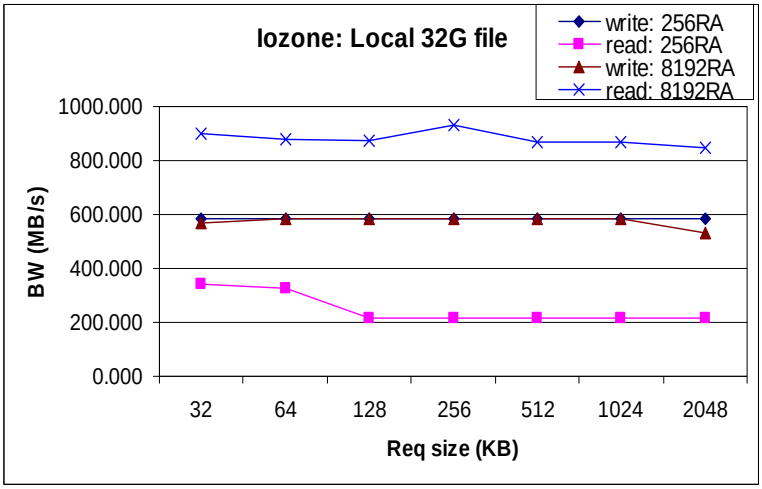

Figure 4. Comparison of read/write bandwidth of local file system with 256 sector (128 KB) and 8192 sector (4096 KB) read ahead parameters

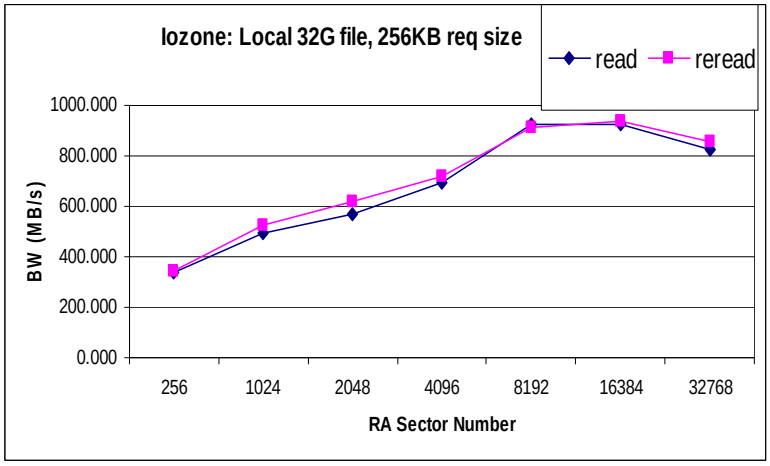

Figure 5. Comparison of read bandwidth of local file system with various Read Ahead sizes and a fixed 256KB request size

NFS Protocol on 10GigE Network with a Single Client

Iozone was used to test the sequential I/O performance of NFS protocol with a single 10GigE client (Figure 2). The first test used a 4GB file, which is much smaller than the memory size of the NFS server. The results of this test in Figure 6 gave a bandwidth boundary of NFS protocol with a single 10GigE client because all read/write data is cached by the NFS server. The gap of the NFS bandwidth on a single 10GigE link with the network bandwidth shown in Figure 3 clearly indicates that the NFS protocol could not fully utilize the physical bandwidth of 10GigE network with a single client. Figure 7 outlines the NFS performance of large file sequential accesses with a single 10GigE client. In the second test, a 32GB file size was used to minimize the cache effect of NFS servers. The write performance is very close to the bandwidth boundary shown in Figure 6, but read performance is considerably lower. Those read performances are much closer to the performance of a small read ahead size (256 sectors) in local file system shown in Figure 4 and Figure 5. The similarity indicates that NFS is using a small read ahead cache.

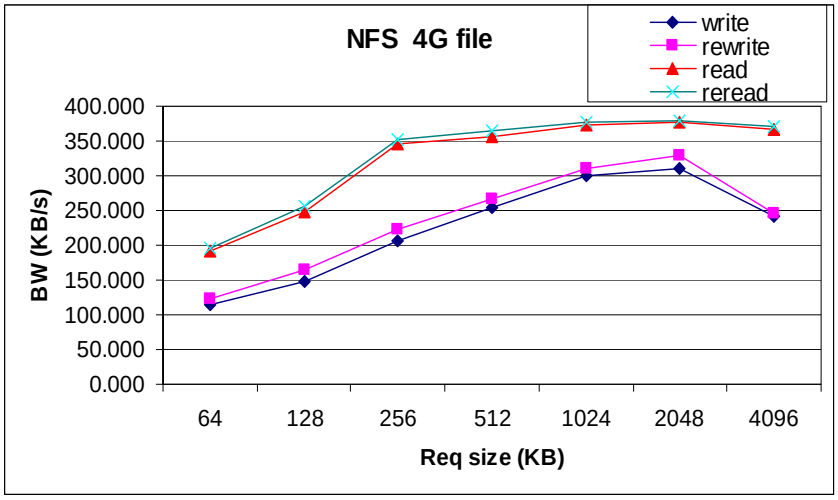

Figure 6. NFS bandwidth of a 4GB file size with a single 10GigE client

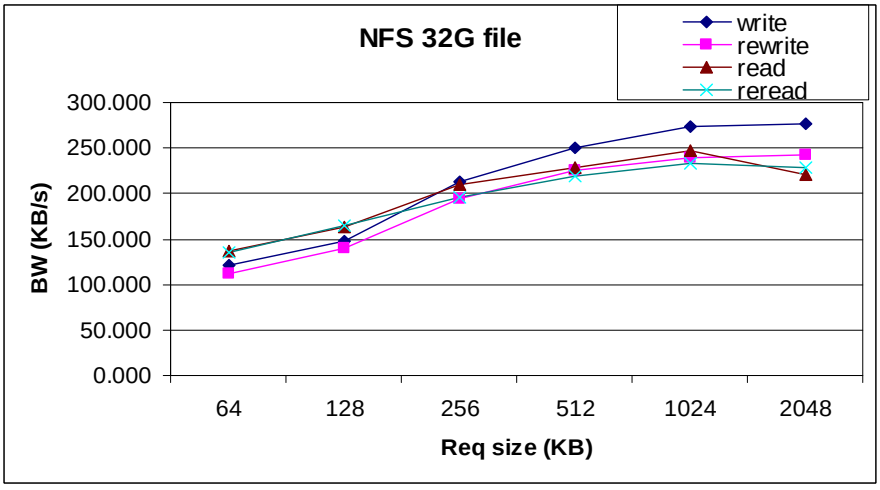

Figure 7. NFS sequential access bandwidth of a 32GB file size with a single 10GigE client

NFS Protocol on 10GigE Network with Multiple GigE Clients

Clustering mode of Iozone was enabled to test aggregated bandwidth of multiple NFS clients. As shown in Figure 1, the NFS server uses 10GigE link to connect to the Ethernet fabric, and all compute nodes are connected to the 6248 switch with their on-board GigE interfaces. An 8GB file size, half of the memory size of the NFS server, was used to find out an aggregated bandwidth boundary of NFS protocol. A small (64KB) and large (512 KB) request size were tested, as shown in Figure 8 and Figure 9, respectively. Read performance is higher than write performance for the small request sizes. However, write requests have higher bandwidth with large request size. The best write performance of 800MB/s with 512KB request size and the best read performance of 800MB/s with 64KB request size are much closer to the physical network boundary of Figure 3 than the performance of a single 10GigE client in Figure 6. This data indicates that the aggregation of multiple clients efficiently utilizes the potential of 10GigE links.

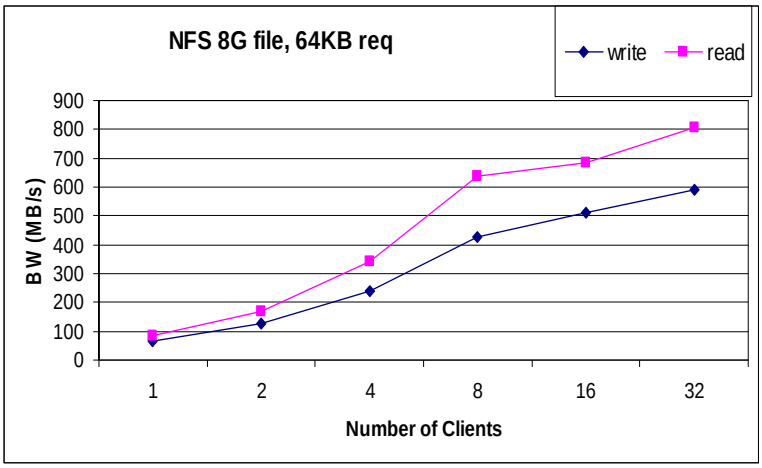

Figure 8. NFS aggregated bandwidth of a 8GB file size and 64KB request size with one 10GigE server and multiple GigE clients .

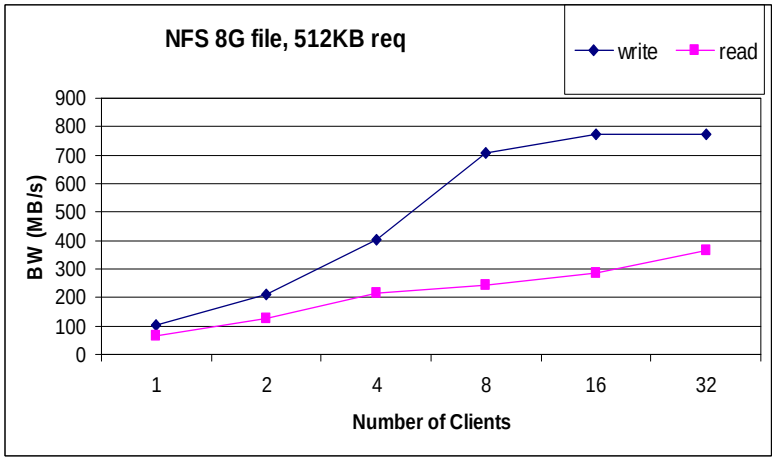

Figure 9. NFS aggregated bandwidth of a 8GB file size and 512KB request size with one 10GigE server and multiple GigE clients .

Both sequential and random requests were tested with a 32GB file size with the Iozone clustering mode, illustrated from Figure 10 to Figure 13. Sequential write performance peaks at 550MB/s with eight clients and then drop slightly with more clients. Sequential read performance peaks at 450MB/s with 32 clients. The bandwidth of sequential write almost matches the local file system boundary of Figure 4. The bandwidth of sequential read lies between the best and the worst read performances of the local file system with the various read ahead cache sizes of Figure 5. With sequential access, a large request size has better write performance than a small request size, while a small request size has better read performances than a large request size. Random write performance peaks at 350MB/s with eight clients and then drops slightly with additional clients. Random read performance peaks at 200MB/s with 32 clients. In random access, seek time of hard drives dominates overall latencies. Large request size reduces the number of arm seeks and thus has better performance for both read and writes requests than small request size.

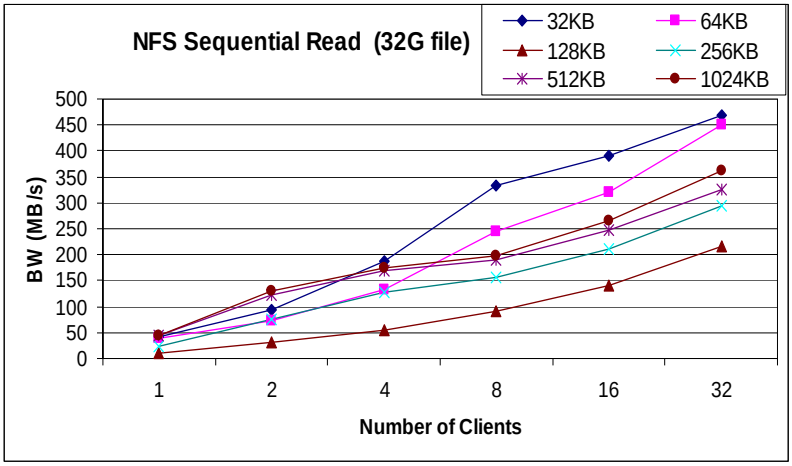

Figure 10. NFS aggregated bandwidth of sequential read of 32GB file with request sizes from 32KB to 1024KB

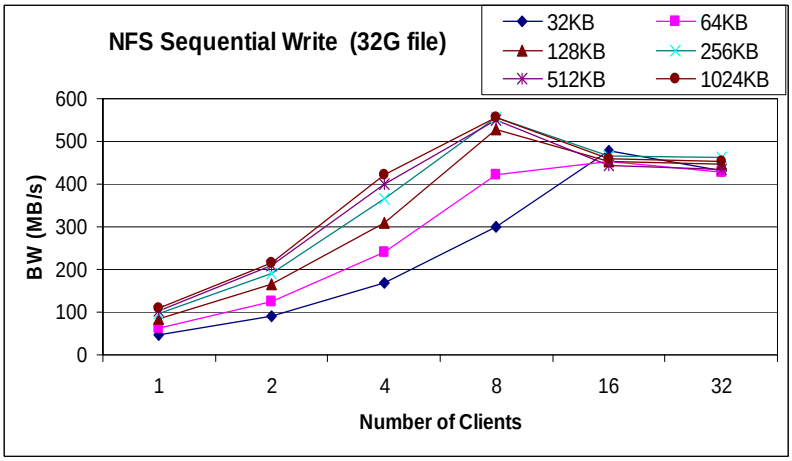

Figure 11. NFS aggregated bandwidth of sequential write of 32GB file with request sizes from 32KB to 1024KB

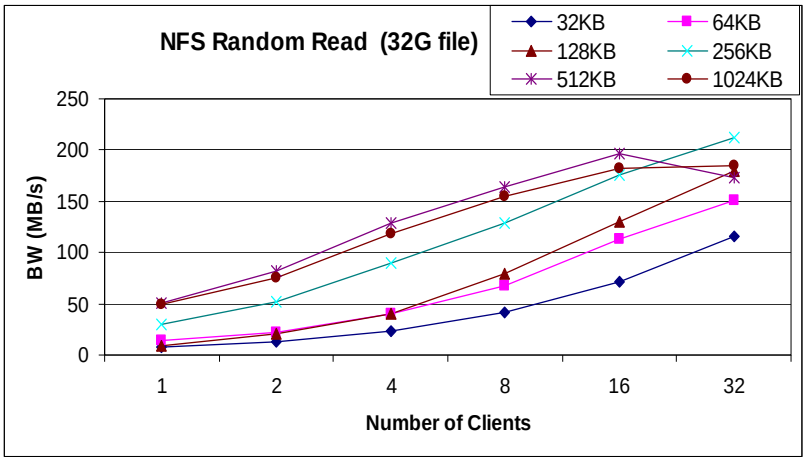

Figure 12. NFS aggregated bandwidth of random read of 32GB file with request sizes from 32KB to 1024KB

Figure 13. NFS aggregated bandwidth of random write of 32GB file with request sizes from 32KB to 1024KB

Figure 14 and Figure 15 compare aggregated NFS bandwidth between an NFS server with 10GigE link and an NFS server with GigE link. The configurations of the 32 clients are kept unchanged from previous tests where all clients are connected to the network switch with on-board GigE interfaces. The solution introduced in this paper of deploying 10GigE links for NFS servers has a considerable performance advantage over an NFS server built on a traditional GigE network, especially for sequential access.

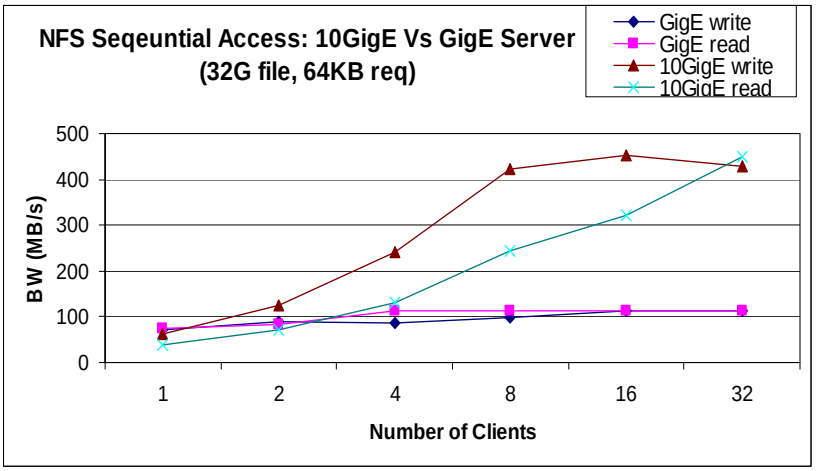

Figure 14. Comparisons of NFS aggregated bandwidth of sequential accesses between a 10GigE and a GigE NFS server

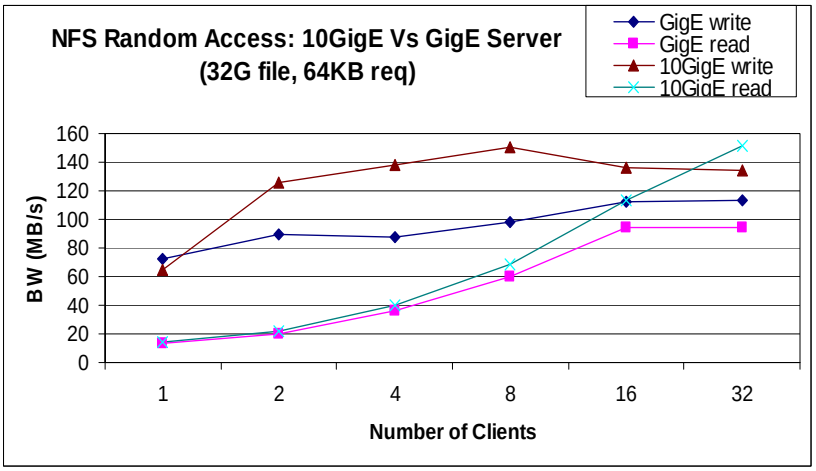

Figure 15. Comparisons of NFS aggregated bandwidth of random accesses between a 10GigE and a GigE NFS server

Conclusions

While many parallel file systems gradually gain popularity on cluster computing platforms, NFS is still the most popular shared file system used in high performance clusters. On a typical HPC platform, NFS is normally built on a GigE fabric, isolated from the sensitive IPC traffic. However, as the number of clients increases, the aggregate bandwidth requirement between an NFS server and the multiple NFS clients can quickly exceed the capability of a GigE fabric.

This article describes a solution of deploying a 10GigE network with NFS on Dell HPC clusters, and provides the HW/SW configuration best practices, and the installation process. As demonstrated by the testing results gathered from the performance study, the solution has the capability to deliver up to 450 MB/s aggregate bandwidth for

sequential requests with 32 clients. Sequential write performance peaks at eight clients and just drop slightly with additional clients. Sequential read performance scales out easily to 32 clients. In a small to medium size cluster consisting of less than one hundred nodes, typical applications whose major access patterns are sequential I/Os can benefit from this solution. The NFS protocol or storage links could be saturated by large number of clients, but the aggregated bandwidth of the NFS over 10GigE links is still much better than the bandwidth over a traditional GigE network. For those applications whose major I/Os are random requests, overall latencies are dominated by the seek time and the rotational delay of hard drives, thus 10GigE links can only moderately increase the performance.

Deploying NFS servers with 10GigE network on Dell HPC clusters provides a cost effective solution for customers to help boost NFS performance. The combination of 10GigE host cards for NFS servers and 10GigE aggregation modules for GigE switches eliminates the network bottleneck between NFS servers and clients. The hardware cost is considerably lower than investing in a new network fabric because the on-board GigE cards of the compute nodes are used. Using 10GigE interconnects makes no changes to NFS software stack, so minimum updating is required. This solution builds on Dell's standard high performance clustering architecture that already provides exceptional flexibility and management.

[1]GB means 1 billion bytes and TB equals 1 trillion bytes; actual capacity varies with preloaded material and operating environment and will be less.