Dell™ | Terascala HPC Storage Solution

A Dell Technical White Paper

Li Ou, Scott Collier
Dell
Massively Scale-Out Systems Team

Rick Friedman
Terascala

May 2010

Contents

Figures

Introduction

Cluster computing has become the most popular platform for high performance computing in use today. In all clusters, there are three key challenges that need to be addressed for clusters to run efficiently: the compute nodes, the networking, and the storage I/O systems.

For compute nodes, the challenges are to ensure that scalable applications are available, that the servers deployed as compute nodes can be easily installed and maintained, and that the nodes themselves deliver high price/performance to the make the whole system cost effective. Dell is a leader in solving these challenges.

Once an efficient processing solution is available, networking is the next key element to a successful cluster. As the processing nodes become more efficient, inter-node communication performance becomes critical. A cluster interconnect needs to deliver high bandwidth, low latency communication while being easily managed on a large scale and being cost effective as a percentage of the compute engine cost. Dell's InfiniBand and 10Gb Ethernet offerings address these challenges today.

With a powerful compute platform and high speed inter-node communication available, the final challenge to creating an efficient, balanced compute cluster is the storage and I/O system. With multiple compute nodes accessing data simultaneously, today's single access storage solutions cannot deliver data fast enough to keep the cluster running at peak efficiency. One alternative has been to use parallel file system approach which delivers high throughput, parallel access and scalable capacity in one system. However, historically, such solutions have been complex to deploy and maintain and too expensive.

The Dell | Terascala HPC Storage Solution (DT-HSS) is a unique new storage solution providing high throughput Lustre-based storage as an appliance. The DT-HSS is a Base Object Solution which consists of Metadata Servers (MDS), Object Storage Servers (OSS) and pre-configured storage array(s). With performance of up to 1.45 GB/sec per Base Object Solution, the HSS delivers the performance necessary to get maximum utilization from your high performance computing infrastructure. The redundant metadata solution ensures high availability of all metadata through an active – passive MDS architecture and use of RAID controllers on the storage arrays. The OSS solution consists of two servers and 2 storage arrays that are also configured in a highly available solution which eliminates any single point of failure and ensures access to data. Designed to scale from 30TB installations up to 180TB, the DT-HSS is delivered as a fully configured, ready to go storage solution and is available with full hardware and software support from Dell. Leveraging the Dell PowerVault™ MD3000 storage array, the Dell | Terascala HPC Storage Solution delivers a great combination of performance, reliability and cost-effectiveness.

This paper describes the Dell | Terascala HPC Storage Solution which delivers all the benefits of a parallel file system based storage solution in a simple to use, cost effective appliance.

Lustre Overview

Lustre is an open source, high performance parallel file system for applications needing very high throughput, scalability, and capacity. It is used in some of the largest supercomputing sites in the world, delivering 100's of GB/sec of throughput and supporting multiple PetaBytes of data in production environments for the last 10 years.

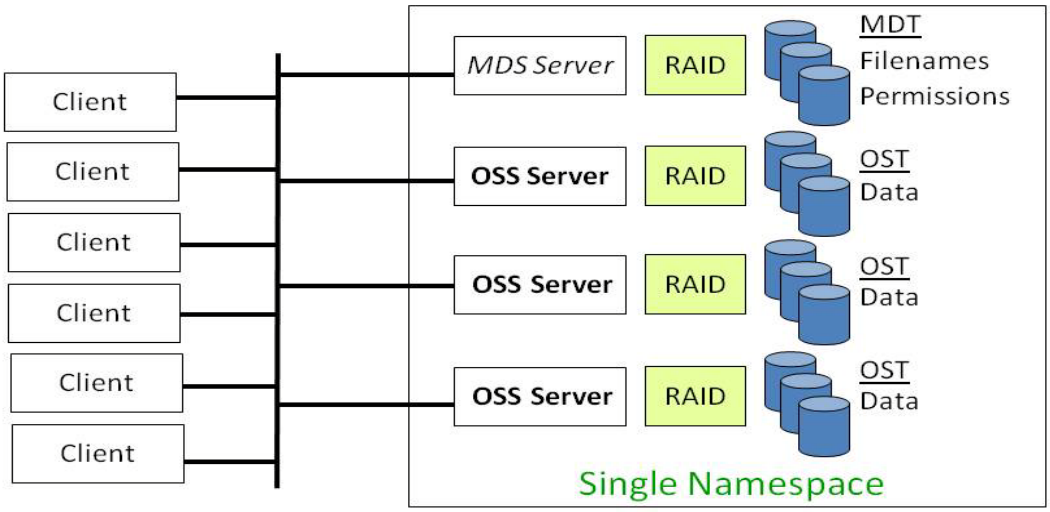
A parallel file system like Lustre delivers its performance and scalability by distributing (or striping) data across multiple access points, allowing multiple compute engines to access data simultaneously. A

Lustre installation consists of three main key systems: the metadata system, the object system, and the compute clients.

The metadata system consists of the Meta Data Target (MDT) and the Meta Data Server (MDS). The MDT stores all the metadata for the file system including file names, permissions, timestamps and where the data objects are storage within the object system. The MDS is the dedicated server that manages the MDT. There is only one active MDS running at any time.

The object system consists of the Object Storage Target (OST) and the Object Storage Server (OSS). The Object Storage Target - provides the storage for the file object data. The Object Storage Server is the server which manages one or more OSTs. There are typically multiple active OSS's at any time. Lustre is able to deliver increased throughput with the addition of OSSs since each additional OSS provides additional networking and processing throughput and capacity. See Figure 1 for more information.

Figure 1. Lustre Overview



Installed on the compute nodes is the Lustre client software that allows access to data stored within the Lustre file system. To the clients, the file system appears as a single namespace, single entity making application access to data simple.

To summarize the functionality of the different elements of the Lustre parallel file system:

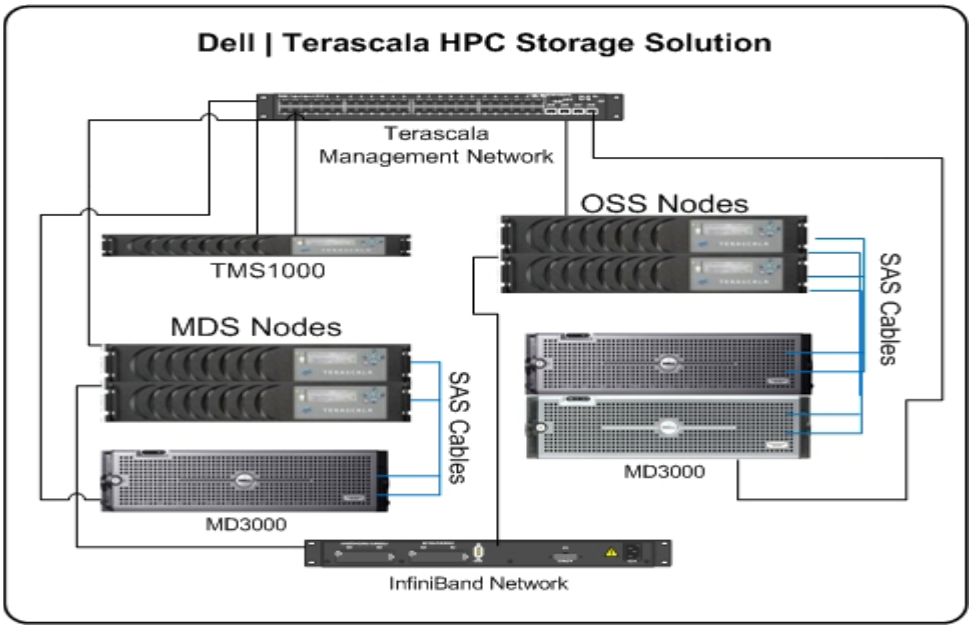
- Meta Data Target (MDT)
  - Keeps track of the location of "chunks" of data
- Object Storage Target (OST)
  - Stores the "chunks" (The chunks are really just blocks on a disk)

- Lustre Client
  - Accesses the MDS to find out where a file is (i.e., on what OSTs)
  - Accesses the OSSs to read and write the data

Typically Lustre deployments and configurations are considered complex and time consuming. Generally available open source Lustre is typically installed and administered via a command line interface which may hinder a systems administrator who is not familiar with Lustre, and therefore won't reap the benefits of such a powerful file system. The Dell | Terascala High Performance Storage Solution (HSS) removes these complexities and minimizes both Lustre deployment time and configuration so the file system can be tested and production ready as soon as possible.

Dell | Terascala HPC Storage Solution Description

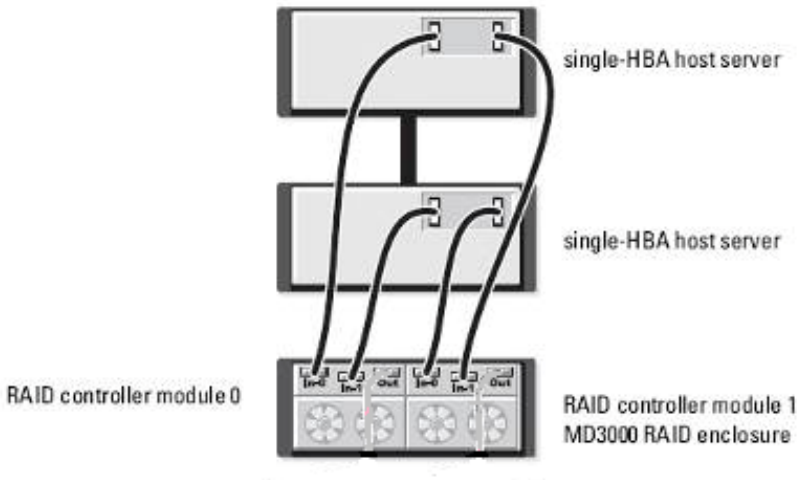
Figure 2. Sample HSS 30 Configuration



A sample configuration of a 30TB Dell |Terascala HPC Storage Solution (HSS) is shown in Figure 2. The key hardware systems include the MDS (Metadata Servers), the OSS (Object Storage Server), and the TMS1000 Management appliance. The MDS and OSS nodes are connected to the compute nodes via InfiniBand. This allows the file system traffic to traverse a high-speed, low-latency network thus improving performance.

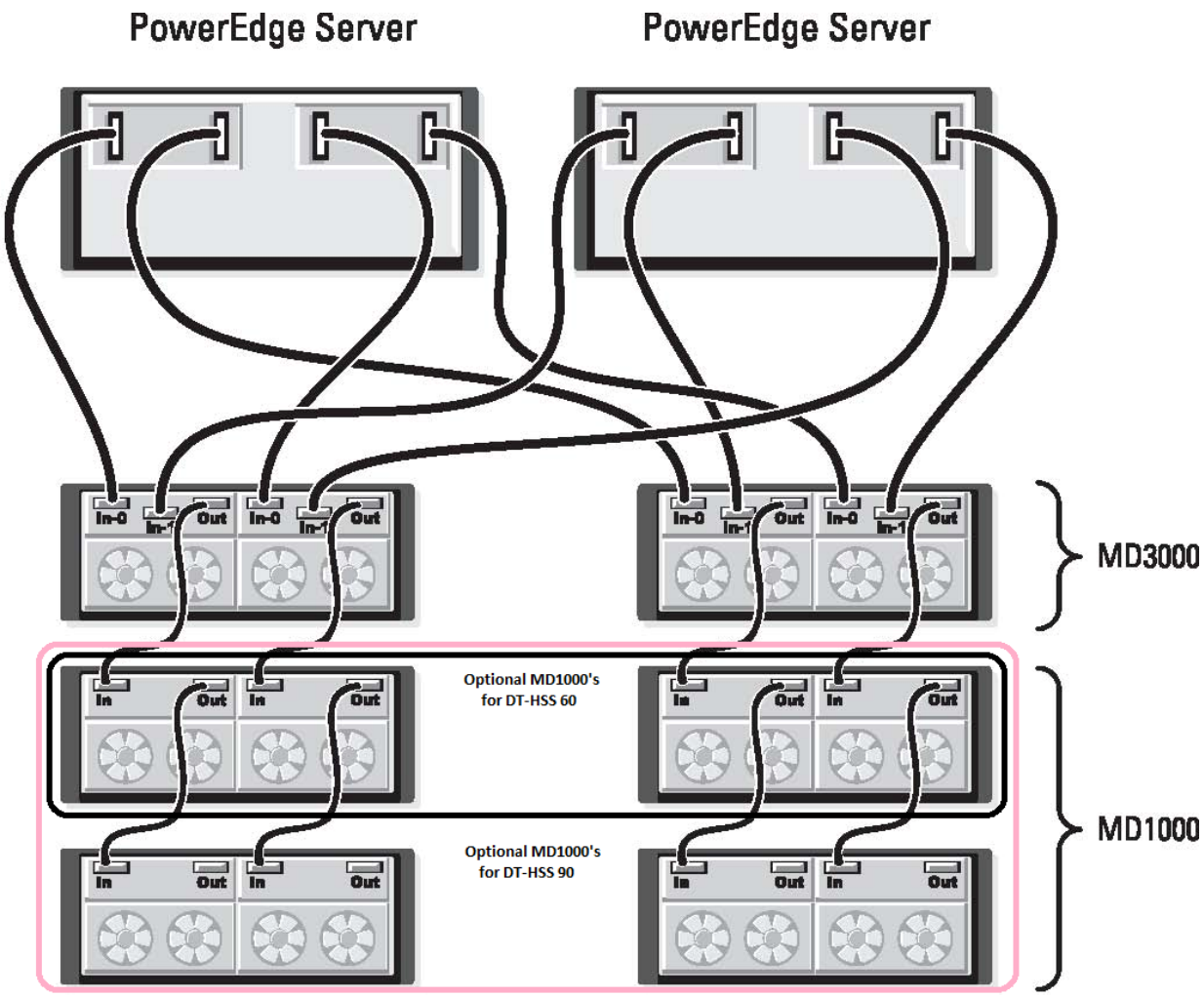
The MDS portion consists of two Terascala storage servers connected in an active-passive configuration to a MD3000 storage array as shown in Figure 3. The active-passive configuration provides high availability and reliability to the metadata itself. It uses advanced monitoring and management to enable rapid, complete failover in case of MDS failure while insuring no spurious writes. The MDS can deliver up to 15,000 file creates/sec and store over 7 TB of metadata information in a RAID 10 volume on the MD3000 array.

Figure 3. Example MDS Cable Configuration



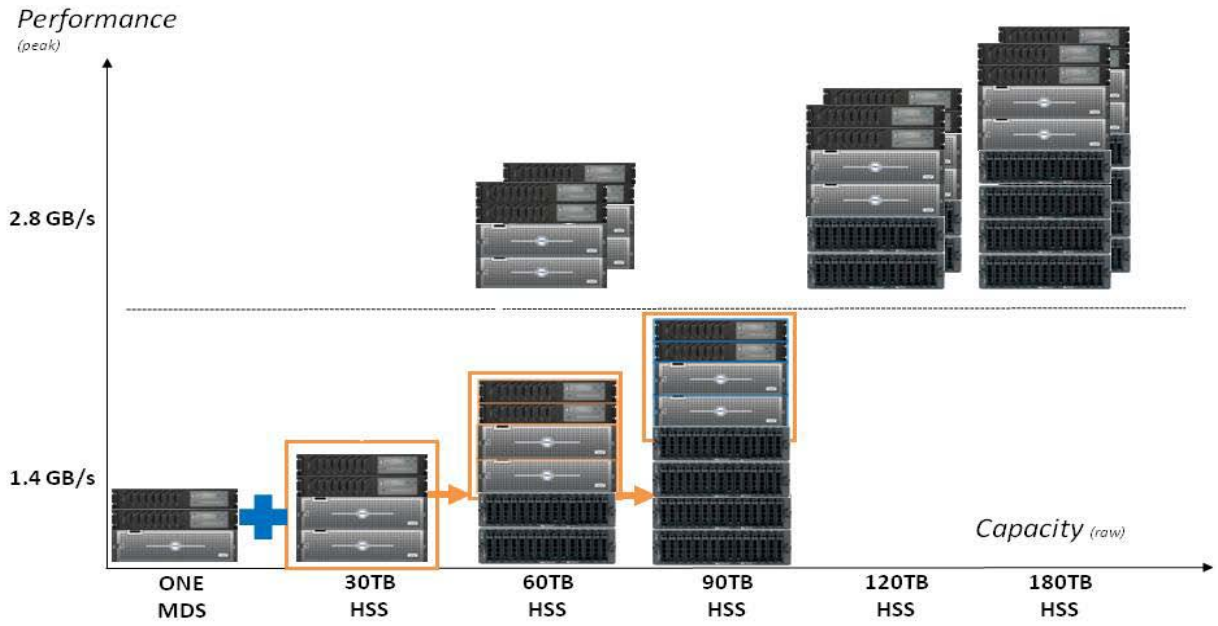
The OSS portion consists of two Terascala storage servers and two PowerVault MD3000 storage arrays cross connected in an active-active configuration as shown in Figure 4. The active-active configuration allows both servers to see data from both storage arrays. With this configuration, all of the object data can be accessed through redundant paths. The storage capacity can be expanded by adding PowerVault MD1000 storage arrays, attached to the existing MD3000 arrays.

Figure 4. Example OSS Cable Configuration



The HSS is designed to be delivered in a set number of combinations and expansion units. The first is the HSS 30 which provides 30 drives and ~1.45GB/sec of throughput. The HSS 30 is a pre-configured, ready to deploy solution that includes the full redundant metadata solution (two Terascala storage servers and one MD3000) and one Base Object (a pair of OSS servers cross connected to 2 MD3000 storage arrays). This configuration is also available in two other preconfigured sizes, a HSS 60 which provides 60 drives that expand the storage array with 2 MD1000's to the Object Store, and a HSS 90 which provides 90 drives that expand the storage array with four MD1000s, see Figure 5. The HSS can also scale performance by adding additional Base Objects to the initial configuration. For example, adding one more Base Objects will increase the performance to ~ 3GB/s of throughput. The HSS can easily scale to the needs of many users and does not increase the administrative overhead.

Figure 5. HSS Expansion Options



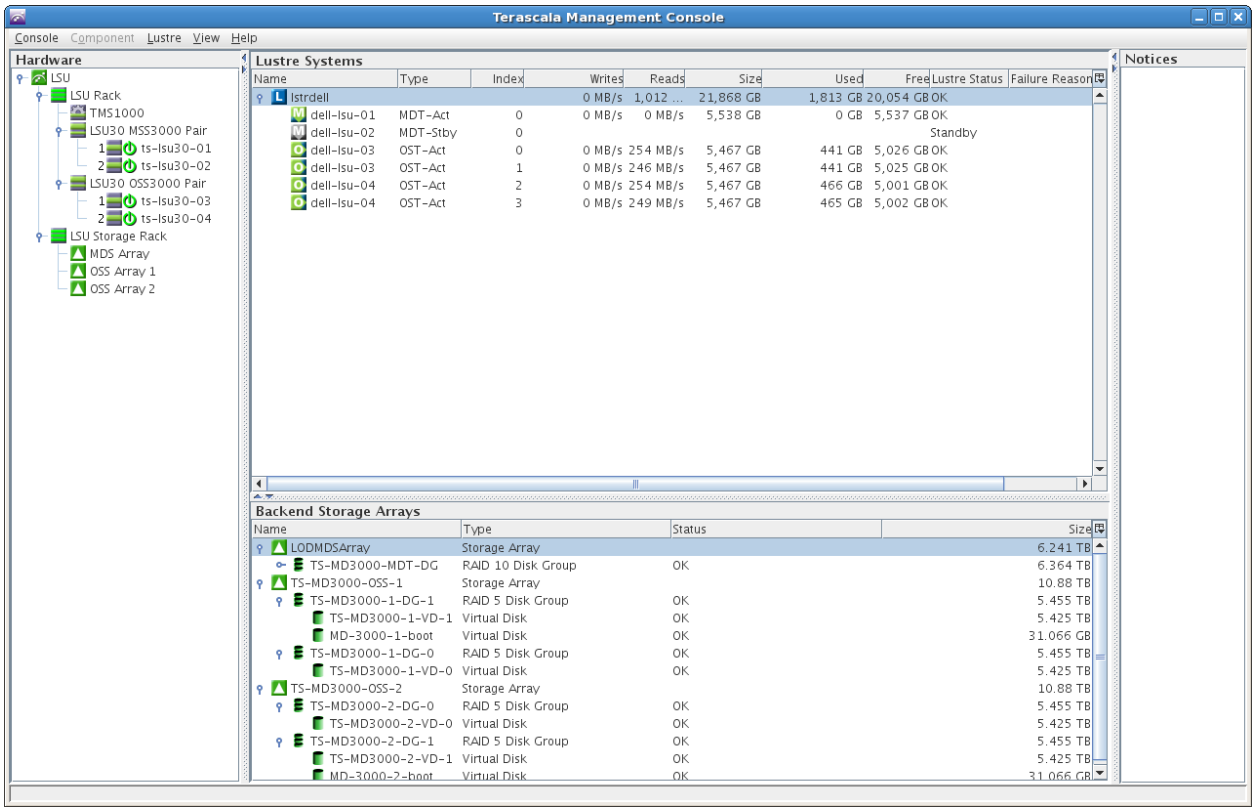
In addition to scaling performance, the HSS can also scale capacity. By adding additional Base Objects, the capacity of the HSS can be doubled. For example, on an existing HSS 30, add one additional HSS 30 to double both the throughput and the capacity. The same goes for a HSS 60 and HSS 90. The HSS can easily scale in both capacity and throughput to meet the needs of the cluster user base and applications.

The HSS has three networks. The primary data network is the InfiniBand fabric on which the Lustre file system traffic traverses. This network is also, typically, the primary data network used by the compute nodes. The HSS is currently configured with DDR IB HCA's. This allows integration into an existing DDR or QDR InfiniBand fabrics, utilizing QDR to DDR crossover cables as required. The second network is an Ethernet network that the Terascala Management Console uses to collect data from the HSS and presents that data via the GUI. The third network is also an Ethernet network that provides a heartbeat between the MDS nodes and the OSS nodes used for failover traffic.

The complete Dell | Terascala HPC Storage solution is managed through the TMS1000 Management appliance. This appliance maintains management paths to all elements of the solution for both capturing system status and file system information and configuration. Users interact with the TMS1000 through the Terascala Management console.

The Dell | Terascala HPC Storage Solution provides a plug and play Lustre file system appliance by providing pre-configured storage arrays and integrating Lustre administration into the Terascala Management Console. Both systems administrators and cluster users alike can benefit from such a solution because they can focus on the work at hand, and not on file system and storage administration.

Figure 6. Terascala Management Console Summary



The Terascala Management Console (TMC) takes the complexity out of administering a Lustre-based file system by providing a centralized graphical user interface for management purposes. The TMC can be used as a tool to standardize the following actions: mount and unmount the file system, initiate failover of the file system from one node to another, and monitor the performance of file system and the status of its components. See Figure 6 for the main interface of the TMC.

The TMC is a Java-based application that can be run from any computer and that remotely manages the complete solution (assuming all security requirements are met). It provides a complete view of both the hardware and file system, while allowing complete management of the solution.

Figure 6 shows the initial window view of a particular system. In the left pane of the window are all the key elements of the system. Each element can be selected to get additional information. In the center pane is a view of the system from a Lustre perspective, showing the status of MDS and the various OSS nodes. In the right pane is a message window that highlights any conditions or status changes. The bottom pane displays a view of the Dell PowerVault storage arrays.

Using the TMC, many tasks that required complex CLI instructions, can now be completed easily with a few mouse clicks. The following figures show how to shutdown a file system (see Figure 7), initiate a failover (see Figure 8) and monitor the MD3000 array (see Figure 9).

Figure 7. Unmounting the File System in TMC

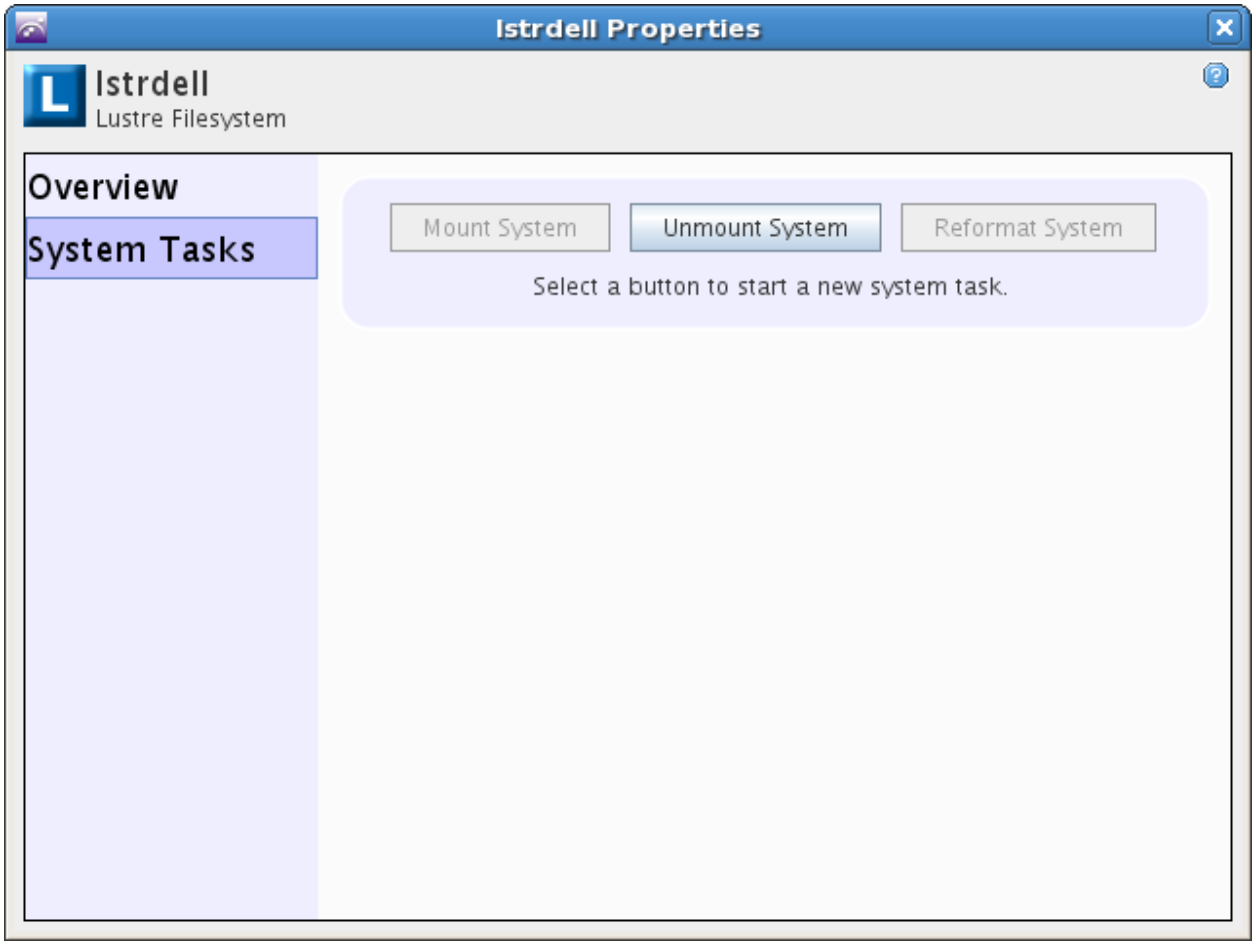


Figure 8. Initiate a Failover in TMC

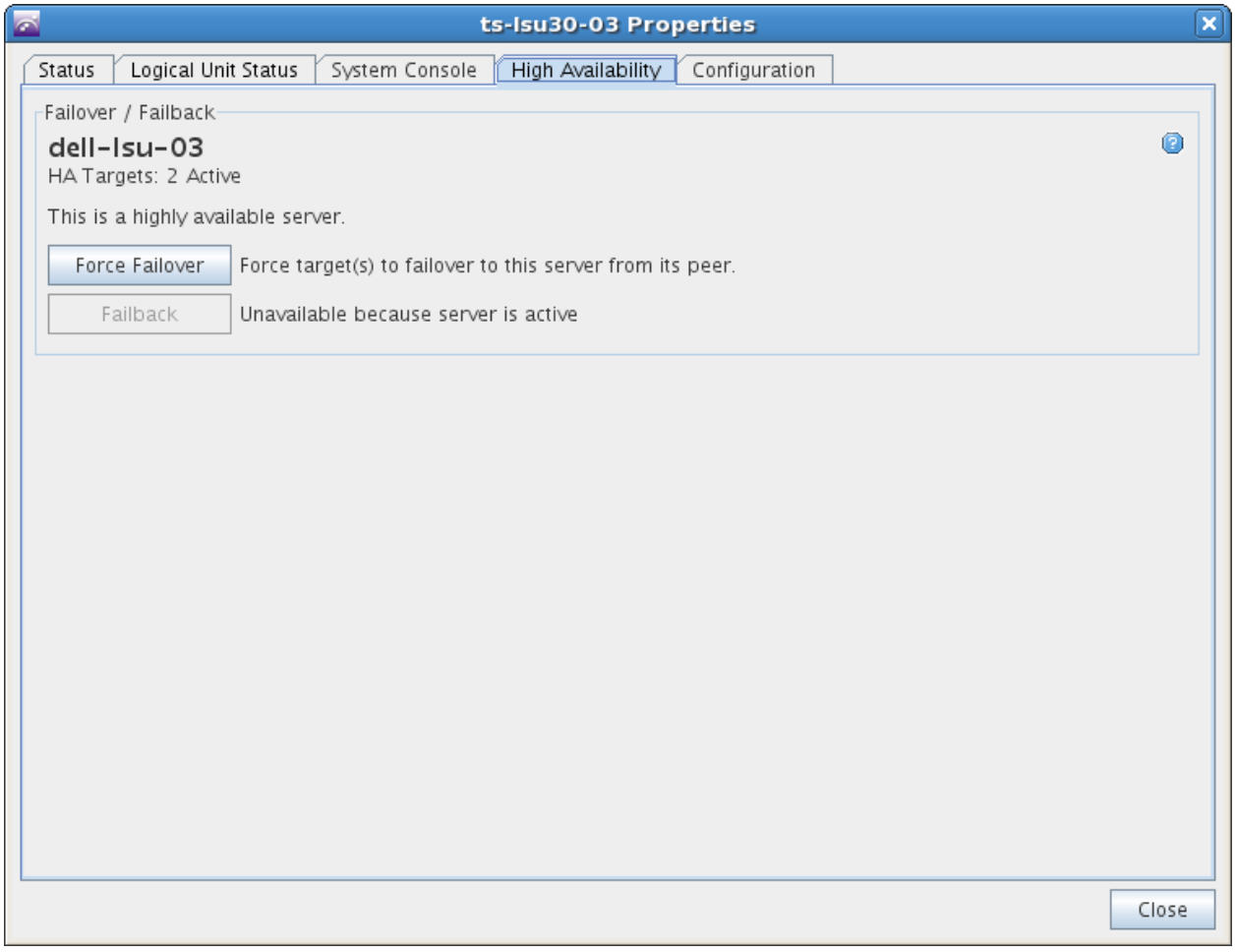
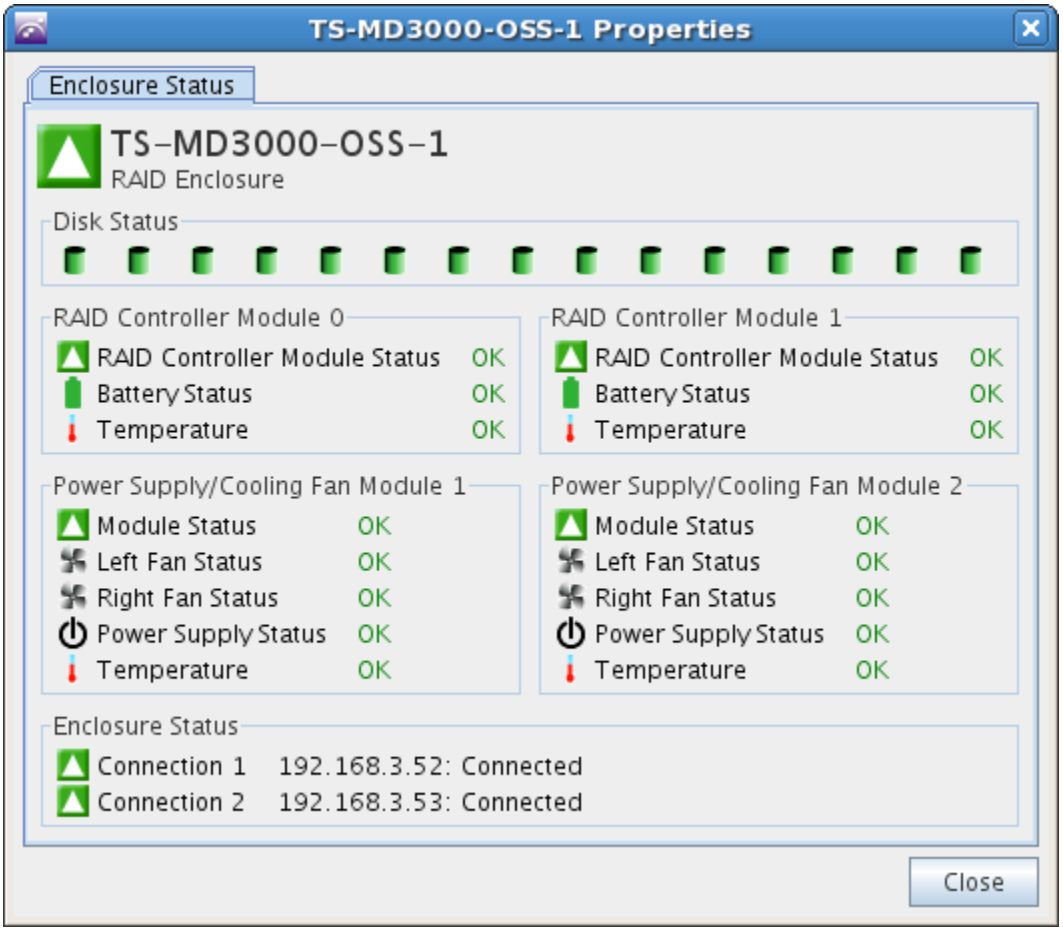


Figure 9. Monitoring the MD3000 in TMC



Dell PowerVault MD3000 / 1000 Overview

The Dell | Terascala HPC Storage Solution leverages the Dell PowerVault MD3000 storage array as the primary data repository. This storage array has proven to be a reliable, high performance solution across multiple application spaces. Within the HSS, the MD3000 is fully configured with 15 nearline SAS drives per shelf to provide the maximum number of spindles to deliver maximum performance.

The MD3000 is a modular disk storage unit housing up to 15 3.5-inch disk drives in a single 3U rack enclosure. The MD3000 direct-attached storage array is expandable by adding up to two additional expansion MD1000 enclosures for a total of 45 drives of capacity. The MD3000 direct-attached array is loaded with data-protection features to keep your applications up and running. The storage array is architected to avoid single points of failure.

A key feature of the MD3000 is two active/active RAID controllers that provide the logic to govern everything that occurs within the MD3000 disk storage array, by performing necessary calculations,

controlling I/O performance operations, handling communication with management applications and storing firmware. The MD3000 direct-attached storage array is designed to incorporate dual active/active RAID controllers with mirrored cache for high-availability. If one controller should fail, the remaining controller handles the full processing load until its mate is brought back online.

Additionally, the MD3000 direct-attached storage array supports removing and replacing redundant components without disruption to operation. In the rare event of component failure, disk drives, storage controllers, power supplies and cooling fan modules are all hot-pluggable, i.e., those components can be removed and replaced for easy repair while the system remains up and running. The MD3000 disk storage array enclosure contains two integrated power supply and fan modules; if one is removed, the other can handle the demand, virtually eliminating downtime to the storage array.

With this level of performance, reliability and expandability, the MD3000 provides a great base for the Dell | Terascala HPC Storage Solution

Integrating Dell | Terascala HPC Storage Solution into a High Performance Cluster

Dell, Platform Computing, and Terascala have created a Terascala Lustre Kit that allows seamless integration and deployment of Lustre clients onto the compute nodes in the cluster. To ensure consistency of the software and configuration on the cluster, use the Terascala Lustre Kit to deploy and manage the Lustre software. As new patches are released, revision control can be performed by using the kit deployment method as well.

The following integration steps were performed using Platform Cluster Manager (PCM) from Platform Computing as the cluster management software. PCM provides the cluster middleware solution that eases deployment and administration of the cluster. Platform Computing and Dell have partnered to test and validate PCM on Dell solutions to ensure software and hardware compatibility. Some of the key features of PCM are the inclusion of common HPC tools (compilers, MPI stacks, etc), web-based management, bare metal cluster provisioning, simplified node management and job submission / management tools.

Terascala Lustre Kits

The Platform Cluster Manager software uses the following terminology to describe the software provisioning concepts:

1. Installer Node – Runs DNS, DHCP, HTTP, TFTP, etc., services
2. Components – RPMS
3. Kits – Collection of “Components”
4. Repositories – Collection of “Kits”
5. Node Groups – Allows association of software to a particular set of compute nodes.

The following is an example of how to integrate a Dell | Terascala HPC Storage Solution into an existing PCM cluster.

1. Access a list of existing repositories on the head node:

```
# repoman -l
Repo name:      rhel5.3-5-x86_64
Repository:     /depot/repos/1000
Installers:     172.20.0.1;192.168.254.100
Ostype:         rhel-5-x86_64
Kits:           PCM-GUI 1.2 noarch, base 5.2 noarch,
                cacti 0.8.7 noarch, dell-vendor 5.3 noarch,
                ganglia 3.0 noarch, java-jre 1.5.0 noarch,
                lava 1.0 noarch, nagios 2.12 noarch,
                ntop 3.3 noarch, platform-hpc 1.0 noarch,
                platform_mpi 3.13.10 x86_64, platform_ofed RH53 noarch,
                rhel5.3 5 x86_64, test1 1 x86_64
```

The key here is the "Repo name:" Line.

2. Add the Terascala kit to the cluster.

```
# kitops -a -m terascala-kit.iso -k terascala-kit
```

3. Add the kit to the repository.

```
# repoman -r "rhel5.3-5-x86_64" -a --kit terascala-kit
```

4. Refresh the repository.

```
# repoman -r "rhel5.3-5-x86_64" –u
```

5. Confirm the kit has been added.

```
# kitops –l
```

6. Associate the kit to the compute node group on which the Lustre client should be installed:
    a. Launch **ngedit** at the console.
        i. `# ngedit`
    b. On the **Node Group Editor** screen, select the compute node group to add the Lustre client.
    c. Select the **Edit** button on the bottom.
    d. Accept the defaults until you reach the **Components** screen.
    e. Use the down arrow and select the **Terascala Lustre** kit.
    f. Expand and select the **Terascala Lustre** kit component.
    g. Accept the defaults and on the **Summary of Changes** screen, accept the changes and push the packages out to the compute nodes.

On the frontend node, there will now be a /root/terascala directory that contains the IOzone benchmark scripts. There will also be a /home/apps-lustre directory that contains the Lustre client configuration parameters. This directory contains a file that gets used by the Lustre file system startup script to optimize the clients for Lustre operations.

The Terascala kit configures IPoIB so the clients can access the Lustre file system over the InfiniBand network. Also, the Terascala kit installs the patchless Lustre clients. The Lustre client is version 1.8.2 for the Terascala 1.0 kit.

Next, verify that the Lustre file system is mounted and accessible from all the clients. Use *pdsh* for verification.

```
# pdsh -w compute-00-[00-63] mount | grep lustre | dshbak -c
```

Performance Studies

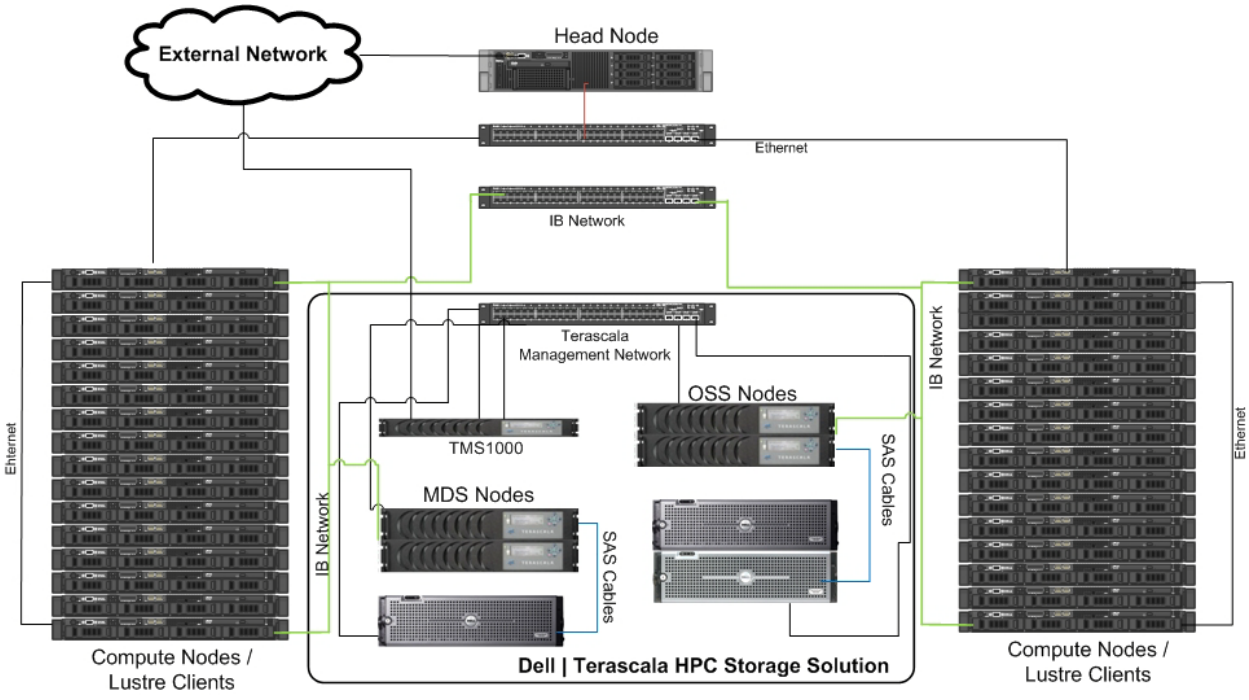
These performance studies characterize what types of applications will benefit from using a Dell | Terascala HPC Storage Solution as a storage appliance. Using a Dell compute test bed, a number of performance studies were performed. The goal was to stress the Dell | Terascala HPC Storage Solution with different types of workloads to find out what the maximum performance would be in addition to how well the solution can sustain that performance.

This study tested different aspects of the solution, specifically: Throughput, I/O Operations Per Second (IOPs) and Metadata Operations of the system. Throughput is the amount of data that can be carried from one point to another point within a particular time frame. IOPs results can be influenced by multiple factors including average seek time of a disk, latency and rotational speed of the disks. IOzone and IOR benchmarks were used to measure throughput and IOPS. Metadata is the final aspect of the Dell | Terascala HPC Storage Solution tested. Metadata is data about data. The benchmark *mdtest* was used to examine the speed with which the Dell | Terascala High Performance Storage Solution creates files, deletes files, obtains the status of files and how quickly files are updated. The types of tests performed on the HSS were sequential read/writes and, random read/writes.

There are two types of file access methods used in these benchmarks. The first file access method is N-to-N, where N-to-N means every thread of the benchmark writes to one file on the storage system. IOzone and IOR can both use N-to-N for a file access method. N-to-1 means that every thread writes to the same file. IOR can use MPI-IO, HDF5 or POSIX to run the N-to-1 file access tests. N-to-1 testing determines how much overhead is involved when multiple clients (threads) are writing to the same file. The overhead encountered comes threads having to deal with single file locking and the serialization of writes. See *Appendix A* for examples of the commands used to run these benchmarks.

Figure 10 shows a diagram of the cluster configuration used for this study.

Figure 10. DT-HSS Cluster Diagram



The storage is an HSS-30:

- Two MDS connected to a single MD3000 that has 15 1TB, 7,200 RPM Near Line SAS drives
- Two OSS connected in a redundant configuration to two MD3000s
- Each OSS MD3000 has 2 OSTs (virtual disks) for a total of 4 OSTs

There are 64 PowerEdge™ R410 servers being used as compute nodes, running Platform PCM 1.2a, which includes Red Hat Enterprise Linux® 5.3 and Platform OFED kit (v.1.4.2). The compute nodes are also running a patch-less Lustre client version 1.8.2. The compute nodes are connected to QDR InfiniBand to a DDR InfiniBand switch via QDR to DDR InfiniBand crossover cables that the MDS and OSS servers are connected to. For a more detailed description of the components of this cluster, please refer to Table 1.

Table 1. Cluster Setup

Compute Nodes (PowerEdge R410)

Processor	Two Intel Xeon™ E5540 2.53 GHz quad core processors
Memory	24 GB (Six 4 GB DDR3-Registered DIMMs)
OS	Red Hat Enterprise Linux 5 U3
Lustre	Lustre 1.8.2 Patch-less Client
Kernel	2.6.18-128.7.1.el5
BIOS	1.2.4

HSS

OSS Nodes	Terascala Servers
MDS Nodes	Terascala Servers
Storage Array	MD3000
Drives in Storage Array	15 1TB 7200RPM Near Line SAS

InfiniBand Network

Terascala Servers	Mellanox Infinihost III Lx DDR HCA
Compute Nodes	Mellanox ConnectX QDR InfiniBand HCA
DDR IB Switch	Qlogic Silverstorm 9024
QDR IB Switch	Qlogic 12800-040
IB Switch Connectivity	5 QDR to DDR Crossover Cables Connecting Qlogic Switches

IOzone Benchmark

IOzone is an industry standard benchmark that can generate many types of loads on a disk subsystem. IOzone is used to test the sequential read and write throughput of the Dell | Terascala HPC Storage Solution. IOzone can be run in one of two ways: in standalone mode where it launches locally and it can spawn X number of threads and report the throughput, or launched in "cluster" mode. In cluster mode, IOzone spawns a master process on the head node and launches IOzone threads on the compute nodes that each write to an individual file. When the IOzone tests are finished, the IOzone master process collects the results and reports the aggregate bandwidth. The version of IOzone used was 3.283 and is available at www.iozone.org.

For this study, the cache on the compute nodes, the OSS server, as well as the MD3000 will be saturated. A file size of 48GB was used for each IOzone thread to ensure that the cache was exhausted and the tests were getting realistic results.

Figure 11. IOzone N-to-N Sequential Write Performance

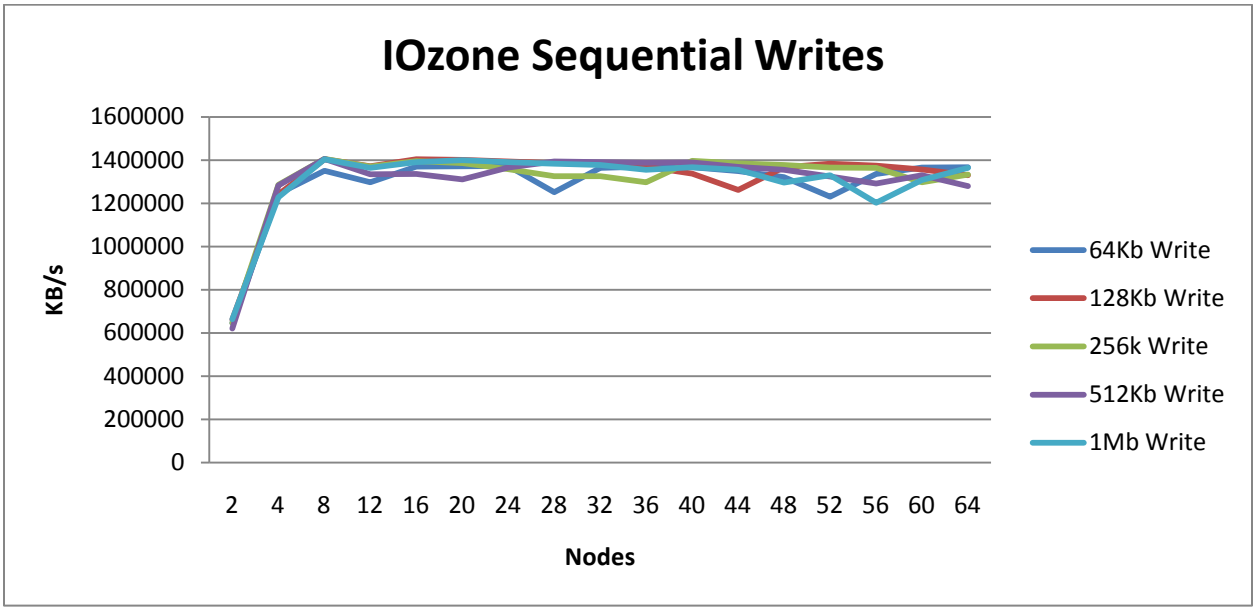


Figure 11 shows that the Dell | Terascala HPC Storage Solution can sustain sequential write bandwidth of 1400 MB/s starting at eight nodes. For this test each IOzone client was pinned to a particular OST and the benchmark was configured to ensure the load was evenly distributed among the OSTs. The quick saturation of the array can be attributed to the Lustre file system traffic going over the InfiniBand network. Different block sizes did not cause significant variance in the test results because Lustre takes the small I/O requests and collects them into larger I/O requests and then does the write. This is a pattern that Lustre prefers and provides better performance. This test was run with an HSS 30. When additional Base Objects are added, the performance will increase significantly.

Figure 12. IOzone N-to-N Sequential Read Performance

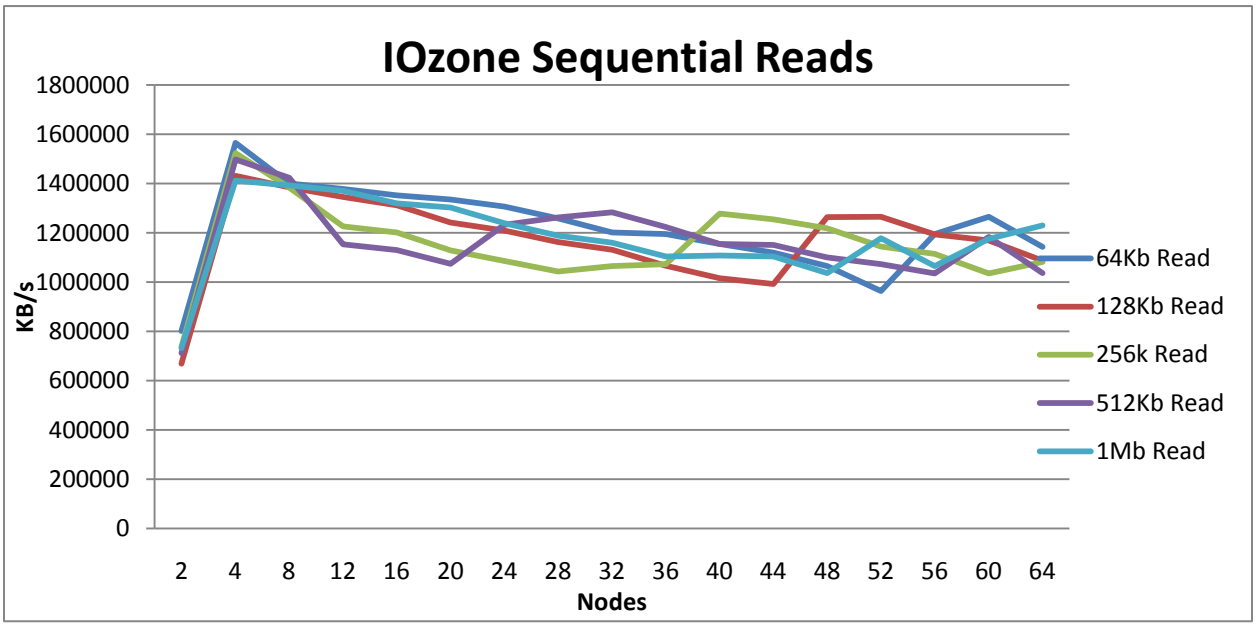


Figure 12 shows that sequential reads can saturate the link at approximately 1500 MB/s using four clients. Also in this case, the saturation point is the same for all block sizes. The drop in performance can be attributed to the fact that the more sequential client IO is added, the more random the data appears. The block sizes also do cause a variation in read performance. The reason for this variation is that the large data set used in the test ensures Lustre is not reading from cache, it is reading directly from disk and typically, as the block sizes get larger, so does the variance.

Figure 13. IOzone IOPs – Sequential Writes

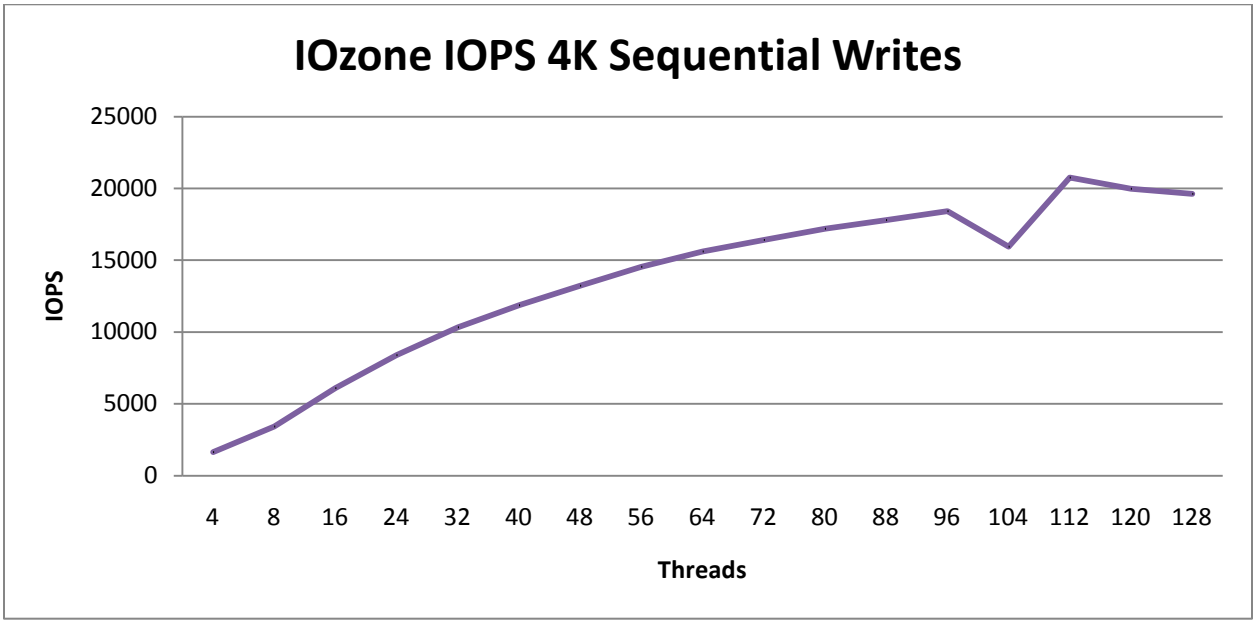


Figure 13 shows how many IOPs can be obtained by issuing sequential writes. It was not possible to saturate the Dell | Terascala HPC Storage Solution by using one IOzone thread per compute node. The

numbers of threads were doubled in order to find the saturation point at around 112 IOzone threads. For this test a 4K block size was used for the sequential writes. The default Lustre block size is 4K.

IOR N-to1 Testing

IOR is an industry standard tool that is used to benchmark parallel file systems using POSIX, MPIIO or HDF5 interfaces. In this study, IOR was run using the POSIX API to test the raw throughput of the file system and avoid the overhead of the HDF5 or MIP-IO which involve using collectives and file views. IOR is launched with *mpirun* and creates a thread on each compute node where each thread can either write to a separate file (N-to-N) or all the threads can write to a single file (N-to-1). For this study, N-to-1 was used to determine the type of overhead encountered based on the file locking and serialization involved in coordinating multiple threads writing to a single file.

The IOR benchmark used in this study was version 2.10.2 and is available at http://sourceforge.net/projects/ior-sio/. The MPI stack used for this study was openmpi version 1.2.8 and was provided as part of the Platform Compute Manager OFED kit.

Figure 14. IOR – Parallel IO Using Sequential Reads and Writes

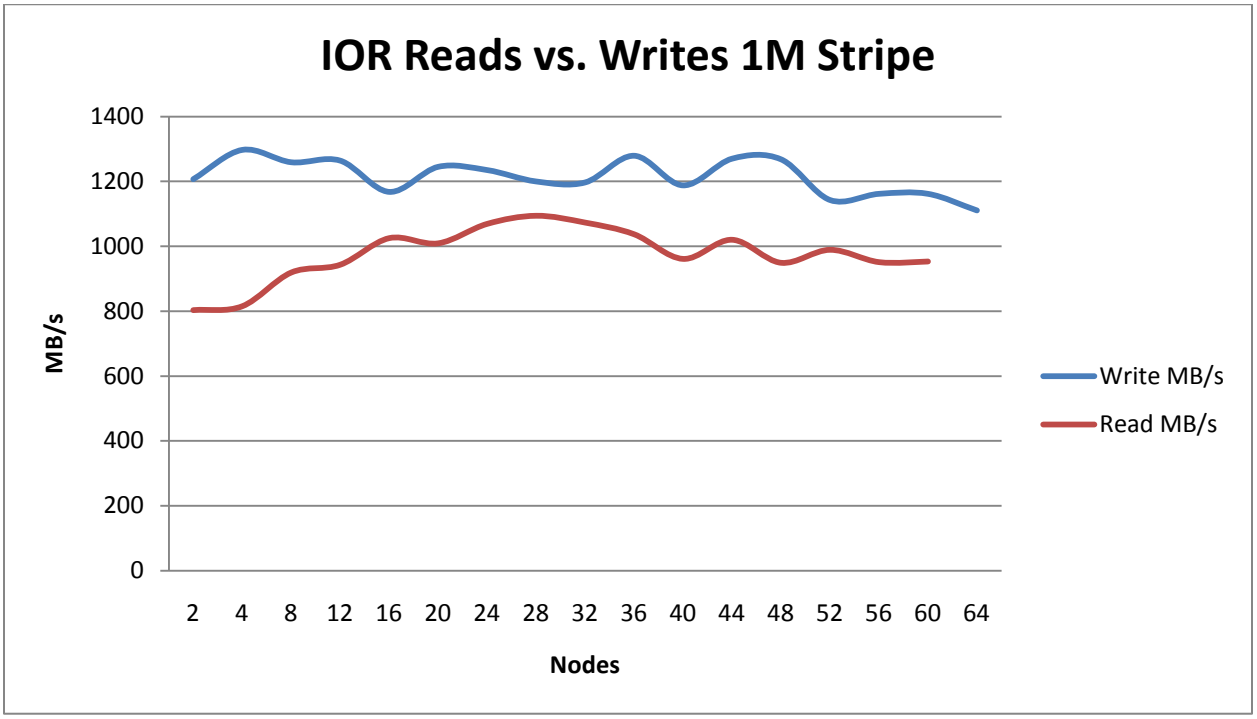


Figure 14 shows that the link can be saturated at around 1300 MB/s with sequential N-to-1 writes and the link is able to sustain that throughput until about 52 nodes. The write throughput is slightly slower than the N-to-N IOzone tests. When IOR performs N-to-1 tests many clients are writing to the same file, causing some overhead because locking on the single file is being performed. The figure illustrates that Lustre is performing better with reads rather than writes. Lustre is optimized for better performance with writes. The Lustre driver is able to profile write requests, and when appropriate it will collect those smaller requests into larger requests and then issue a single write which reduces locks and improves performance.

Metadata Testing

Metadata means data about data. *Mdtest* is an MPI-coordinated benchmark that performs create, stat and delete operations on files and directories and then provides timing results. This version of *mdtest* was modified by one of our test engineers to include *utime*.

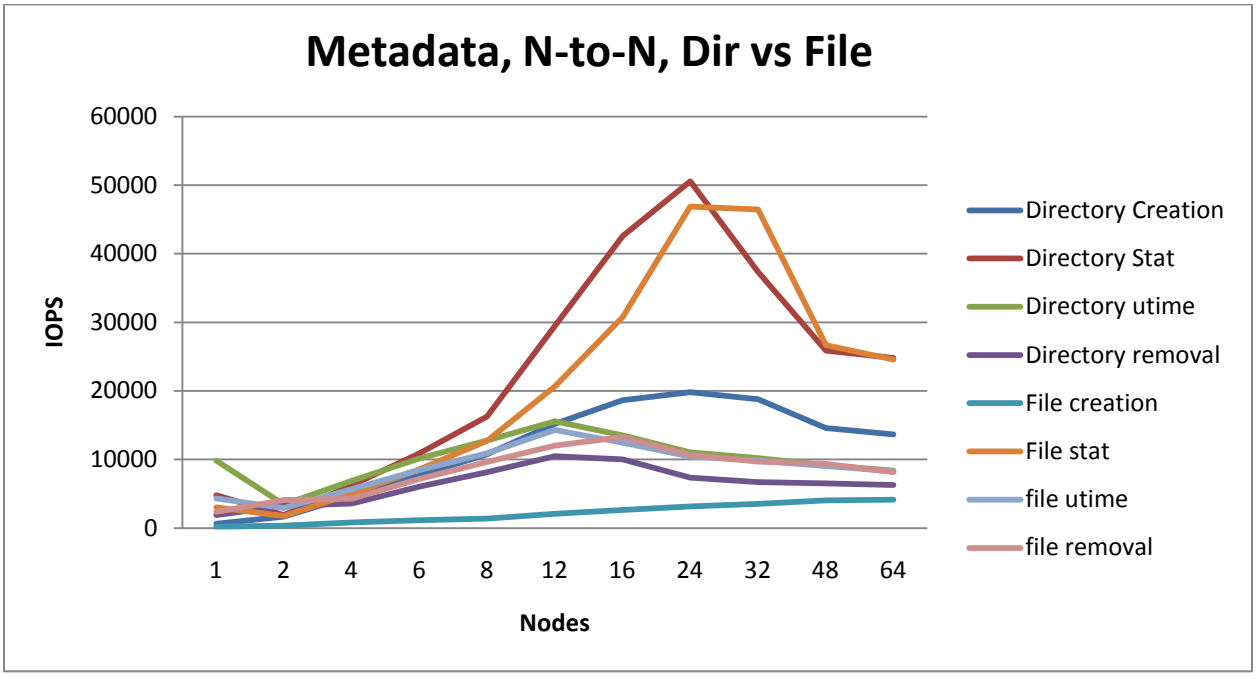
The *mdtest* capabilities includes:

- Create – time to create a file or directory both empty and directories containing data
- Delete – time to delete a file or directory
- Access
    - Stat – time to get the status of a file. This represents a metadata read operation.
    - Utime – time required to update the file. This represents a metadata update operation.

The mdtest benchmark used for this study was version 1.7.4 and is available at http://sourceforge.net/projects/mdtest/. The proper way to launch *mdtest* is with *mpirun* which creates a thread on each compute node. Each thread can perform operations in one of two ways -- to a separate file or directory (N-to-N) or to a single file (N-to-1). For this study, both types of tests were performed and the results compared. The objective was to determine the locking and serialization encountered in coordinating multiple threads writing to a single file as opposed to writing to individual files.

This study presents results based on these two scenarios. The first examines how many IOPS can be expected when all the operations are handled within the MDS cache. The second examines how many IOPS can be expected when the MDS cache is exhausted. This demonstrates the importance of sufficient MDS memory for metadata intensive applications.

Figure 15. Metadata N-to-N, Directory Vs. File



The metadata results shown in Figure 15 compare file and directory create/stat/utime/removal while the operations are handled in cache on the MDS. The results show that directory creation is much faster than file creation. Directory creation only requires an operation on the metadata server whereas a file creation requires two operations: one on the metadata server and one on the object store server. In this metadata test, the metadata operations do not scale well past 30 nodes. This test used an HSS 30. The file creation speed is influenced by the number of OSSs. It is expected that with additional base object solutions that the file creations will speed up and scale accordingly.

Figure 16. Metadata file N-to-N Vs. N-to-1 in Cache

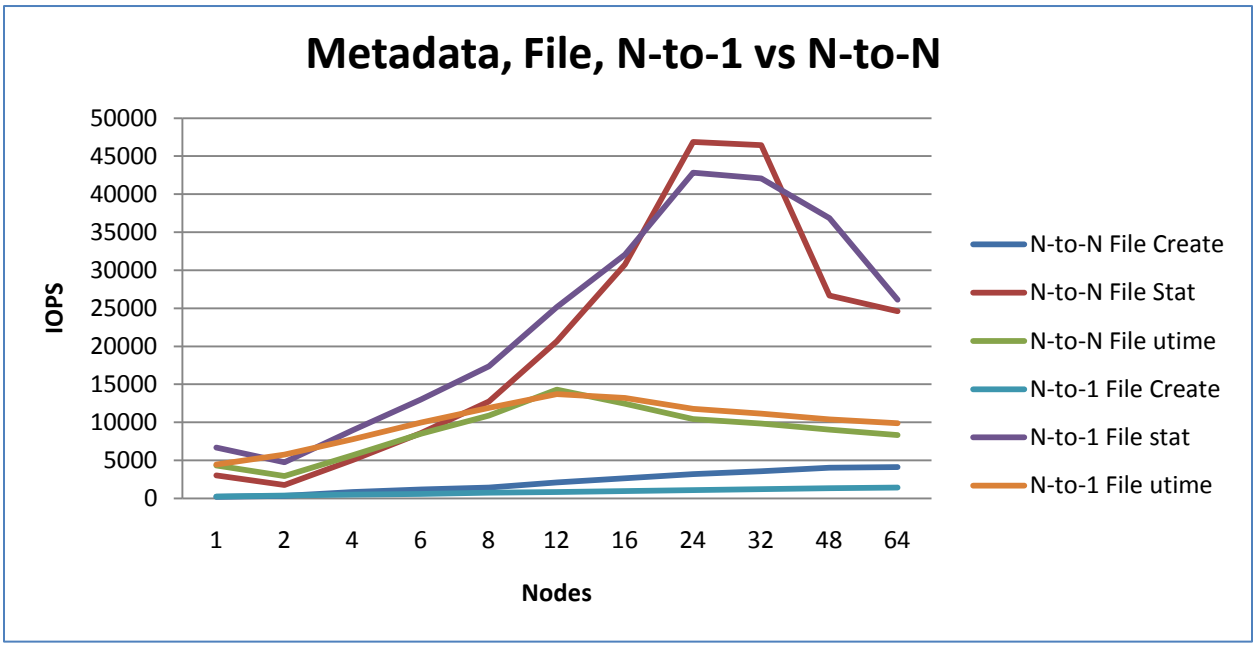


Figure 16 demonstrates that N-to-N sequential file creates are much faster than N-to-1. This is because N-to-1 create requires locks and serializations on both the MDT and OSTs. Utime shows that N-to-N and N-to-1 are similar, indicating that the locks and serializations on the MDT are not the bottleneck. Also, here we don't compare N-to-1 delete performance because the rank 0 node is always responsible for removing the shared file for all the nodes.

In some cases the workload may require the creation of several million files. Lustre consumes about 2k of memory per file to handle the locks and kernel structures, so the DT-HSS can serve approximately 4 million files out of cache. Once that amount of files is exceeded, it is expected that the performance will degrade.

Conclusion

Some applications need a high performance and high capacity file system to be as productive as possible. Also, the file system solution needs to be easy to deploy and integrate as well as easy to administer. The solution to these problems is the Dell | Terascala HPC Storage Solution (HSS).

By providing 30TB to 180TB of storage and 1.45-2.90GB/s throughput, the HSS meets these capacity and performance requirements.

The HSS is a plug and play appliance that is managed via the TMC. This console provides a centralized administration interface that can be used to perform all the necessary actions on the Lustre file system. By using the TMC, there is no need to manually issue Lustre commands or become a Lustre file system expert – it's all taken care of upfront.

The performance studies in the paper clearly show that the HSS is able to provide very high throughput and IOPS for both N-to-N and N-to-1 file access types. Industry standard tools like IOzone, IOR and

mdtest provide a way to capture performance results from the HSS and also characterize what types of applications will run well with this type of storage backend.

The HSS provides a scalable, easy to administer storage array that removes the complexities for deploying and administering Lustre. By providing a fully supported and pre-configured hardware and software Lustre file system appliance, the HSS is quick to deploy and easy to maintain. From the data presented in this paper, it is clear that the HSS solution delivers excellent performance helping users to maximize the performance of their applications and the utilization of their servers.

References

http://www.dell.com/us/en/enterprise/storage/pvaul_md3000/pd.aspx?refid=pvaul_md3000&cs=555&s=biz
http://www.dell.com/us/en/business/storage/pvaul_md1000/pd.aspx?refid=pvaul_md1000&cs=04&s=bsd
Dell HPC Home Page – http://www.dell.com/hpc
Dell HPC Wiki – http://www.HPCatDell.com
Terascala Home Page – http://www.terascala.com
Platform Computing Home Page – http://www.platform.com
Lustre Home Page – http://www.lustre.org

Appendix A – Benchmark Command Reference

This section describes the commands used to benchmark the Dell | Terascala HPC Storage Solution.

IOzone –

IOzone Sequential Writes –

```
/usr/sbin/iozone -i 0 -+n -c –e –r 1m –s 48g -Rb /tmp/iozone_$$.wks -l
$NUMTHREAD -u $NUMTHREAD -+m clientlist
```

IOzone Sequential Reads -

```
/usr/sbin/iozone -i 1 -+n -c –e –r 1m –s 48g -Rb /tmp/iozone_$$.wks -l
$NUMTHREAD -u $NUMTHREAD -+m clientlist
```

IOzone IOPs Sequential Writes –

```
/usr/sbin/iozone -i 0 -r 4K -I -O -w -+n -s 32G -l $NUMTHREAD -u $NUMTHREAD -
+m clientlist
```

Description of command line arguments:

IOzone Command Line Arguments	Description
-i 0	Write test
-i 1	Read test
-+n	No retest
-c	Includes close in the timing calculations
-e	Includes flush in the timing calculations
-r	Records size
-s	File size
-Rb	Excel report with filename
-l	Minimum number of processes
-u	Maximum number of processes
+m	Location of clients to run IOzone on when in clustered mode
-I	Use O_Direct
-w	Does not unlink (delete) temporary file
-+n	No retests selected

By using *-c* and *-e* in the test, IOzone provides a more realistic view of what a typical application is doing.

The O_Direct command line parameter allows us to bypass the cache on the compute node on which we are running the IOzone thread.

IOR -

IOR Reads Vs. Writes –

```
mpirun -np 64 --hostfile "$HOSTFILES" "$IOR" -a POSIX -i 3 -d 32 -k -r -E -o
"$IORFILE" -s 1 -b 48G -t 1m >> "$LOG"
```

Description of command line arguments:

IOR Command Line Arguments	Description
-np	Number of processes
--hostfile	File with names of compute nodes
"$IOR"	Location of IOR executable
-a	Defines which API to use
-i	Number of repetitions of test
-d	Delay between repetitions in seconds
-k	Keeps the file on program exit
-r	Reads the existing file
-E	Does not remove test file before write access
-o	Name of test file
-s	Number of segments
-b	Contiguous bytes to write per test
-t	Size of transfer in bytes

Metadata

Files N-to-1 access:

```
mpirun  -np $i --hostfile hosts mdtest -d $dir -n 10 -i 200 -y -w 1 -N $tpn -
u -t | tee /home/scott/mdtest_Testing/mdtest_results/md-n-d_out_$i.txt
```

Empty files N-to-1 Access:

```
mpirun -np $i --hostfile hosts mdtest -d $dir -n $n -i 200 -y -N $tpn -t -u |
tee /home/scott/mdtest_Testing/mdtest_results/mdtest_out_md-n-e-noac-$i-
$n.li_test.txt
```

Files N-to-1 access:

```
mpirun -np $i --hostfile ./hosts -nolocal ./mdtest -d $dir -n 10 -i 200 -y -w
1 -S -N $tpn -u -t | tee md-1-d_out_noac_$i.txt
```

Empty files N-to-1 Access:

```
mpirun -np $i --hostfile ./hosts mdtest -d $dir -n 10 -i 200 -y -S -N $tpn -u
-t | tee /home/scott/mdtest_Testing/mdtest_results/mdtest_out-1-e-
noac_out_$i.txt
```

Description of command line arguments:

mdtest Command Line Arguments	Description
-d	Directory where the tests will run
-n	Every process will create/get status/remove
-i	Number of iterations each test runs
-y	Ensures the file gets synced after each operation
-N	Sets stride to tasks per node
-u	Creates a unique working directory for each task
-t	Time the unique working directory overhead
-y	Syncs the file write after completion
-S	Shared file access