High Performance, Open Source, Dell Lustre Storage System

Dell/Cambridge HPC Solution Centre

Wojciech Turek, Paul Calleja

July 2010

Abstract

The following paper was produced by the Dell™ Cambridge HPC Solution Centre and is based on operational experience gained from using mass storage technologies within a production University HPC environment. The paper provides a detailed description of how to build a commodity "Dell Lustre™ Storage Brick" and provides comprehensive performance characteristics obtained from the Lustre storage brick when integrated into a gigabit Ethernet storage network. The paper also discusses the operational characteristics and system administration best practices derived from over two years of production usage. The performance data shows good I/O throughput via the Lustre filesystem on top of the Dell storage brick, yielding 80% of the bare metal MD3000 storage array performance. Each Dell Lustre Storage brick is able to provide 400MB/s read/write I/O bandwidth through the filesystem layer, this performance scaling linearly with each additional storage brick. Over gigabit Ethernet each client is able to achieve an I/O bandwidth of 100MB/s which scales linearly with successive clients until the back-end bandwidth is saturated. We have scaled such a system to hundreds of terabytes with over 2 GB/s total back-end storage I/O performance and 600 clients.

The Dell Lustre storage brick shows good overall performance and scalability characteristics with a high data security and availability record when in production. From usage experience over several years, it can be said to be a good fit for departmental and workgroup HPC installations. A large 270 TB (6 brick) configuration within the Cambridge production environment has demonstrated very good operational characteristics with an unscheduled downtime of less than 0.5% over 2 years of 24/7 service.

Whitepaper Structure

1.0 Introduction

The continued growth of clustered computing and the large computational power of even modest departmental and workgroup HPC systems has resulted in a storage architecture challenge in which traditional NFS, NAS, SAN and DAS based storage solutions have not kept pace with the performance growth of a large segment of the HPC install base. This has led to many mid-range HPC users struggling to meet the I/O demands of their applications. It is widely recognised within the research computing community that cost effective, high performance, scalable and resilient centralised storage is the greatest challenge facing modern day HPC facilities.

Over recent years the HPC industry has seen a rise in use of parallel filesystems such as Lustre™, GPFS™ (General Parallel File System™) and pNFS as an attempt to overcome the HPC storage architecture problem. This paper will describe in detail how to build and configure a Lustre Parallel filesystem storage brick using Dell™ MD3000 / MD1000 commodity storage arrays. The paper details a Lustre storage array build for use in an Ethernet or Infiniband I/O network. Comprehensive performance data will be presented from the storage array described when used in a gigabit Ethernet I/O network. The paper will also comment on operational characteristics and system administration best practices derived from over two years production usage of a large-scale mass storage solution of the same architecture. The main focus of this paper is how to build Lustre on a commodity storage array, with the objective of allowing the reader to replicate the system described. A follow-up Lustre paper will take this work further and focus on detailed performance characteristics when using the latest Dell MD3200 /1200 in conjunction with a QDR Infiniband network. These two papers together will then provide detailed build instructions and comprehensive performance data on leading edge up-to-date commodity storage hardware.

This paper has been produced from the Dell/Cambridge HPC Solution Centre, which is part of the wider University of Cambridge HPC Service. The Service runs one of the largest HPC facilities in the UK with a large user base and a high I/O workload. The Solution Centre acts as an HPC requirements capture vehicle, solution development lab and production test environment for Dell whereby the operational experience of the Cambridge HPC service and the commodity computing knowledge of Dell are combined to produce innovative, robust and fully tested HPC solutions. These solutions are then to be made openly available to the HPC community via white papers and technical bulletins.
The Cambridge HPC Service has been successfully using the Lustre Parallel file system within a large scale 300 TB central cluster filesystem for over three years. The Dell Lustre storage brick as described here has evolved to fill the operational requirements of a busy working HPC centre and has been used in production in Cambridge for several years. Its development was critical in order to meet pressing operational requirements caused by a growing demand for storage volume and the inability of our NFS file systems to keep up with the I/O workload.

The Lustre filesystem has several key features that make it a good fit within the departmental and workgroup HPC centre as well as in the largest national level HPC centres. In fact Lustre is now used in over 50% of the top 50 supercomputers in the world. Key Lustre features responsible for Lustre's high level of usage within the HPC segment include:

- Lustre is a true parallel filesystem allowing multiple nodes in a cluster to read and write from the same file at the same time. This can greatly increase the file system I/O for applications which support parallel I/O.
- Lustre serves files horizontally across any number of storage servers with data striped across standard low cost commodity storage arrays, providing the aggregate performance of all storage servers and storage arrays. In this way Lustre can provide a large and scalable back-end storage performance from low cost hardware.
- The Lustre filesystem I/O can be delivered to cluster client nodes over a wide range of network technologies, gigabit, 10 gigabit or Infiniband. This enables high I/O performance to a single client, which can be advantageous for legacy HPC applications that do not take advantage of parallel I/O.
- The Lustre filesystem can be used to agglomerate many separate disk storage arrays into one single filesystem with a single global name space which can grow dynamically simply by adding more storage elements. This can be very useful when having to manage a large HPC central storage facility.
- Lustre leverages Linux and is itself an open-source software effort. This, combined with its usage of commodity storage array hardware, drastically reduces the cost of Lustre filesystem installations. It also means there is an active open-source community for future development and mailing lists for troubleshooting problems. Also, as there is a range of companies offering professional support of Lustre storage solutions there is no vendor lock-in with Lustre.
- Lustre has many HA and resiliency features which can be used to configure fault-tolerant and highly available storage solutions essential for use in large-scale HPC environments.

2.0 Lustre Filesystem Design

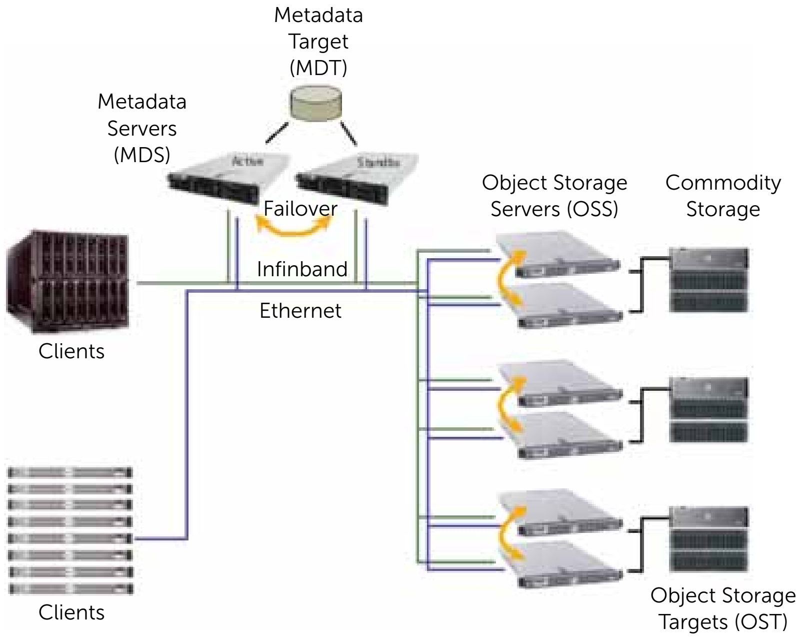
Lustre provides a storage architecture for clusters which allows significant freedom in hardware implementation. At the user level the Lustre filesystem provides a POSIX-compliant UNIX filesystem interface. The main components of Lustre are the Metadata Server (MDS), Object Storage Server (OSS) and Lustre client. The Lustre filesystem uses an object-based storage model and provides several abstractions designed to improve both performance and scalability. At the filesystem level, Lustre treats files as objects which are located through the MDS. Metadata Servers support all filesystem name space operations, such as file lookups, file creation, and file and directory attribute manipulation. This information is physically stored on the metadata target device (MDT). Lustre allows only one MDT per filesystem. File data is stored in objects on the object storage targets (OST) which are managed by OSSs. The MDS directs actual file I/O requests from a Lustre client to the appropriate OST, which manages the objects that are physically located on the underlying storage block devices. Once the MDS identifies the storage location of a file, all subsequent file I/O is performed between the client and the OSSs.

The Lustre clients are typically HPC cluster compute nodes which run Lustre client software and communicate with Lustre servers over Ethernet or Infiniband. The Lustre client software consists of an interface between the Linux virtual filesystem and the Lustre servers. Each server target has a client counterpart: Metadata Client (MDC), Object Storage Client (OSC), and a Management Client (MGC). OSCs are grouped into a single Logical Object Volume (LOV) which is the basis for transparent access to the filesystem. Clients mounting the Lustre filesystem see a single, coherent, synchronised namespace at all times. Different clients can write to different parts of the same file at the same time, while other clients read from the file.

This design divides filesystem operation into two distinct parts: filesystem metadata operations on the MDS and file data operations on the OSSs. This approach not only improves filesystem performance but also other important operational aspects such as availability and recovery times.

As shown in Figure 1, the Lustre filesystem is very scalable and can support a variety of hardware platforms and interconnects.

Figure 1. High Availability Lustre filesystem high-level architecture



3.0 Dell MD3000 – Ethernet Lustre Storage System Architecture Overview

There are three main design goals of the Dell™ Lustre™ Storage System, namely; maximised HPC I/O performance, simplified implementation and configuration and a competitive price point. These goals are accomplished by using a modular approach with standardised commodity hardware components and open source software. This modular approach allows the HPC data centre to start with a number of storage modules that meet their workload requirements and easily grow capacity and throughput as needed by adding more OSS modules. The following design can utilise either Ethernet or Infiniband fast interconnects, making it able to meet most data centre deployment requirements. In this paper we describe performance for an Ethernet-based solution and in a follow-up paper we will look at detailed performance analysis using an Infiniband mediated I/O network.

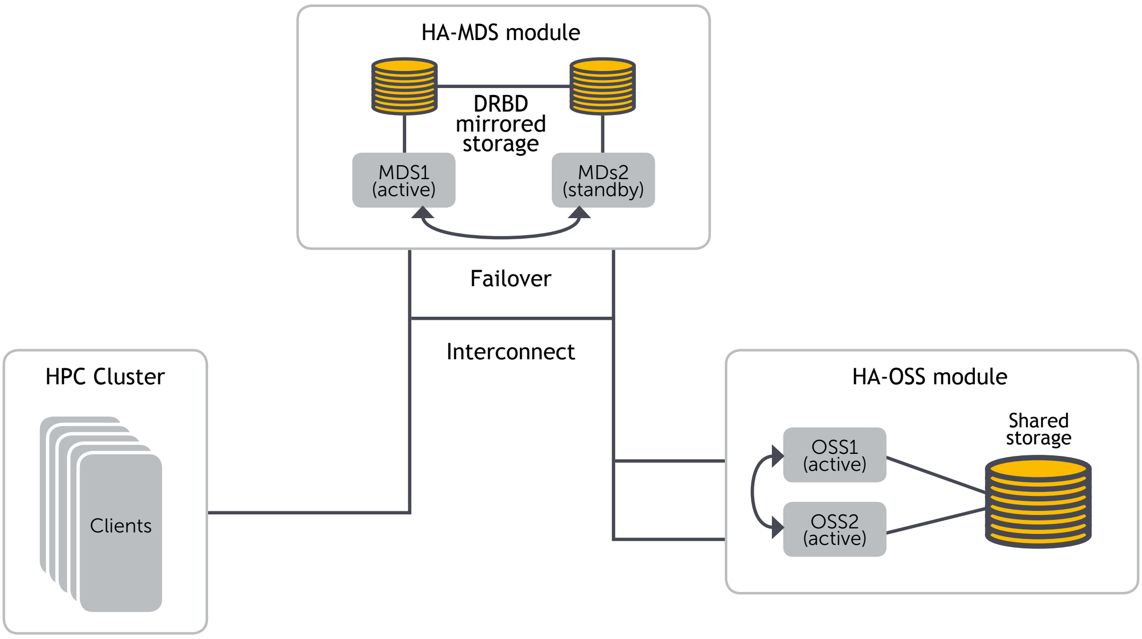
The following modules are used in the Dell Lustre Storage System design:

HA-MDS/MGS – High Availability Metadata Server and Management Server module.

HA-OSS – High Availability Object Storage Server module.

See Figure 2. for the logical relationship between clients and the storage modules.

Figure 2. Overview of Dell Lustre Storage System design



4.0 Hardware Components

The Dell™ PowerEdge™ and Dell™ PowerVault™ product lines are the foundation of the Dell Lustre Storage System. Specifically, PowerEdge 2950 and 1950 servers and PowerVault MD3000 and MD1000 disk arrays are used as the building blocks of the Dell Lustre Storage System modules.

4.1 HA-MDS Module

The HA-MDS module is comprised of two Dell PowerEdge 2950 servers, each equipped with high-performance Intel processors, 16GB of RAM, six fast, low-latency, enterprise class SAS hard drives and an internal PERC 5/i RAID controller (Figure 3). The internal drives are divided into two RAID arrays. The first two 73GB SAS drives are configured as RAID1 and act as the boot device and the other four 300GB SAS drives are configured as RAID10 and act as the MDS storage. In order to provide High Availability, the MDS block devices are mirrored by means of the network-based RAID1 software known as DRBD (Figure 7). An MDS storage configuration with a typical inode size of 4KB per file can support a filesystem containing over 150 million files. This HA-MDS module can support both Ethernet and Infiniband interconnect networks.

It is commonplace to use a variant design whereby the MDS storage is an external SAS raid array as opposed to using the two internal SAS disks and DRBD. When looking at Ethernet network latency compared to SAS disk latency, one can calculate the additional overhead from DRBD for metadata performance as about 10%. This is acceptable when one considers the added simplicity and cost saving. This overhead can be reduced to 1% if RDMA-based network technologies are used for the DRBD network. We have used both external storage array and DRBD networked MDS storage in our production Lustre system in Cambridge and found the practical performance characteristics to be identical.

Figures 3. and 4. illustrate the HA-MDS as described.

Figure 3. Dell Lustre HA-MDS module configuration

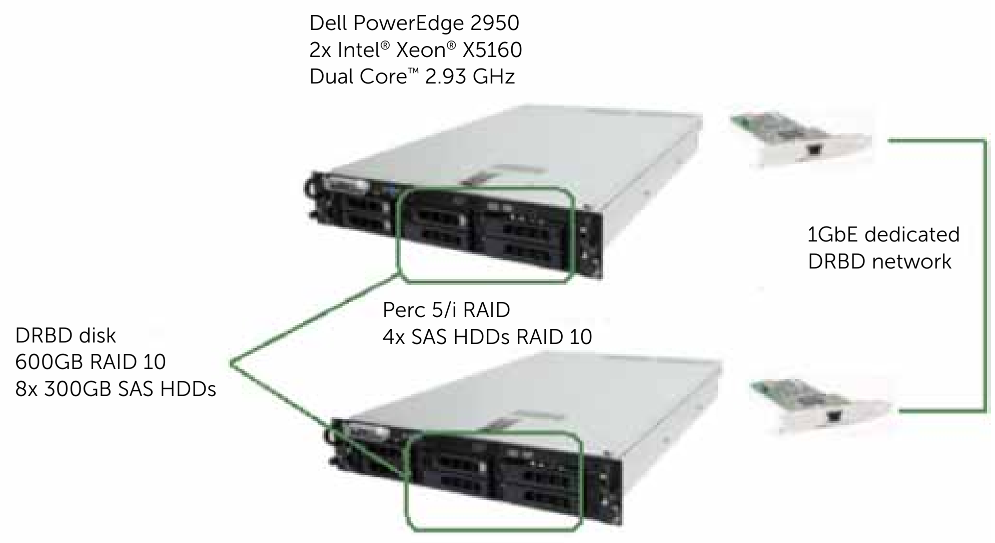
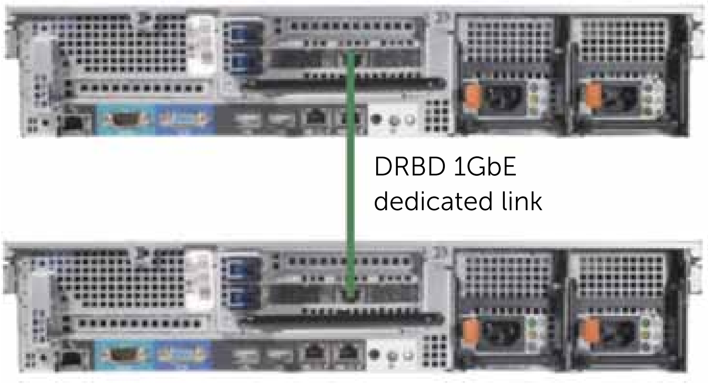


Figure 4. Dell Lustre HA-MDS module rear view



4.2 HA-OSS Module

The HA-OSS module consists of two Dell™ PowerEdge™ 1950 servers attached to a single PowerVault™ MD3000 disk array extended with two PowerVault MD1000 disk enclosures (Figure 5 and Figure 6). The PowerEdge servers are each equipped with high-performance Intel® multicore processors, 8GB of RAM, 2 internal hard drives, an internal PERC 5 RAID controller and an SAS HBA card. All disk enclosures are populated with fast 1TB SATA disks. The Dell PowerVault MD3000 direct attached external RAID storage array is designed to increase cluster I/O performance and functionality. Equipped with data-protection features to help keep the storage system online and available, the storage array is designed to avoid single points of failure by employing redundant RAID controllers and cache coherency provided by direct mirroring. The Dell PowerVault disk array provides performance and reliability at the commodity price point required when constructing very large HPC central data storage facilities.

Figure 5. Dell Lustre HA-OSS module configuration

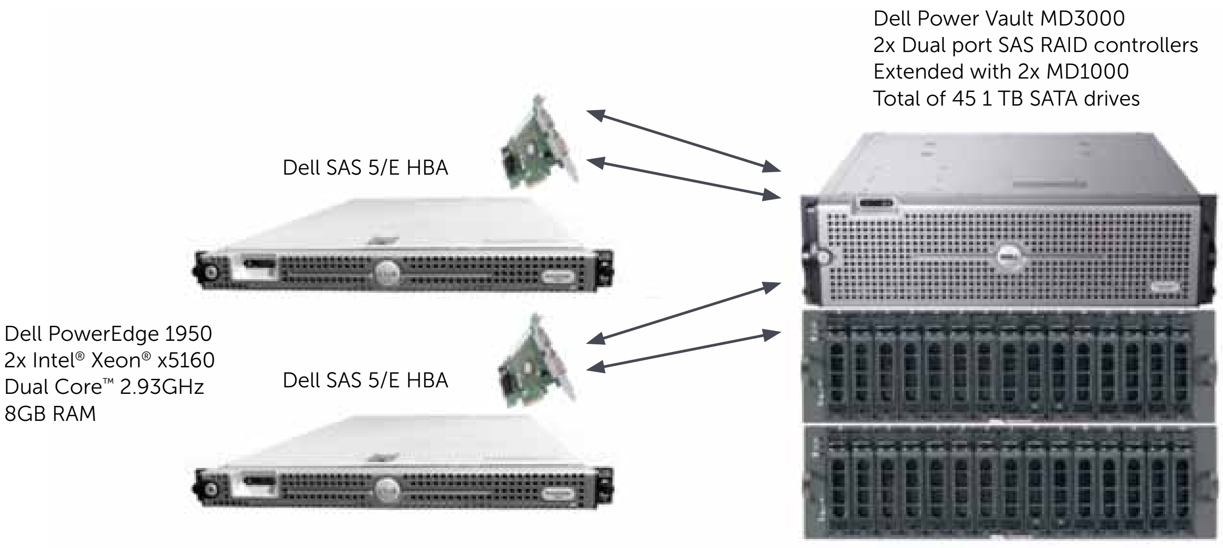
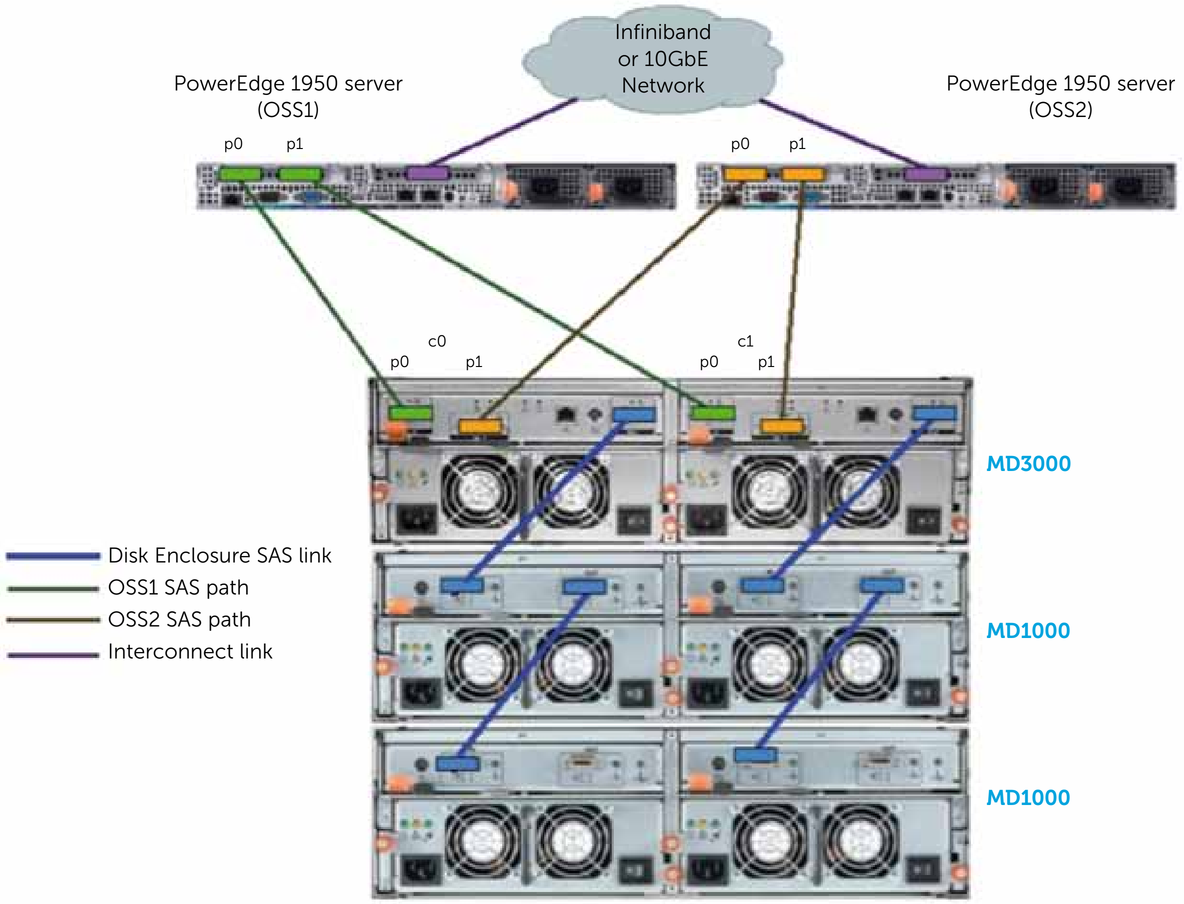


Figure 6. Dell Lustre HA-OSS module configuration - rear view



The HA-OSS module is configured to provide high availability of data. The Dell PowerEdge Object Storage Servers connect to the Dell PowerVault MD3000 via dual port SAS HBA cards. To ensure high availability failover, each OSS connects to both external controllers, providing two redundant links to the PowerVault MD3000 disk array per OSS. This design allows for either OSS to drive all the volumes on the Dell PowerVault disk arrays. During normal operation, each OSS exclusively owns a subset of volumes. In the event of failure, the remaining server in the failover pair will take over the volumes from the failed server and subsequently service requests to all the volumes.

MD3000 storage array key features:

- Mirrored data cache of up to 512 MB on each RAID controller
- In-band and out-of-band management using Dell Modular Storage Manager
- Redundant, hot-swappable components and virtual disk failover
- Transparent failover for clustered operating systems to maximise data protection
- RAID-level migration, capacity expansion, consistency checks, and Self-Monitoring, Analysis and Reporting Technology (SMART).

The storage array configuration described here of one MD3000 extended by two MD1000’s was designed to provide a balanced solution in terms of performance, disk capacity and cost. We used 1 TB disks here but 2TB disks could also be used to increase storage capacity further and reduce the cost per TB of the total solution. A second configuration is possible that maintains the optimum Lustre RAID configuration but increases the I/O performance per storage unit volume. The configuration in this case is a storage array of one MD3000 extended by one MD1000. Both the 3 array unit and the 2 array unit have the same I/O performance of 400 MB/s, thus a 6 array solution with the 3 array module would have a total performance of 800 MB/s compared to a 6 array solution of the 2 array module of 1200MB/s. The 2 array solution has a higher per TB because of an increased number of RAID controllers and servers.

5.0 Software Components Overview

The Dell Lustre™ Storage System software stack installed on the Metadata Servers and Object Storage Servers includes the following software components:

- Linux operation system
- Lustre filesystem
- HA-Linux Heartbeat failover
- DRBD network mirroring tool
- Dell configuration and monitoring tools

The Dell Lustre Storage system uses the Scientific Linux 5.4 Linux operating system distribution (SL5.4). This is a rebuild of the Red Hat® Enterprise Linux® distribution (RHEL5.4) This Linux distribution is an Enterprise-level operating system which provides the industry's most comprehensive suite of Linux solutions. RHEL5.4 delivers a high performance platform with most recent Linux features and highest levels of reliability. The Dell Lustre Storage system uses Lustre version 1.8 which is the latest stable version of Lustre, incorporating several key features compared to older versions of Lustre.

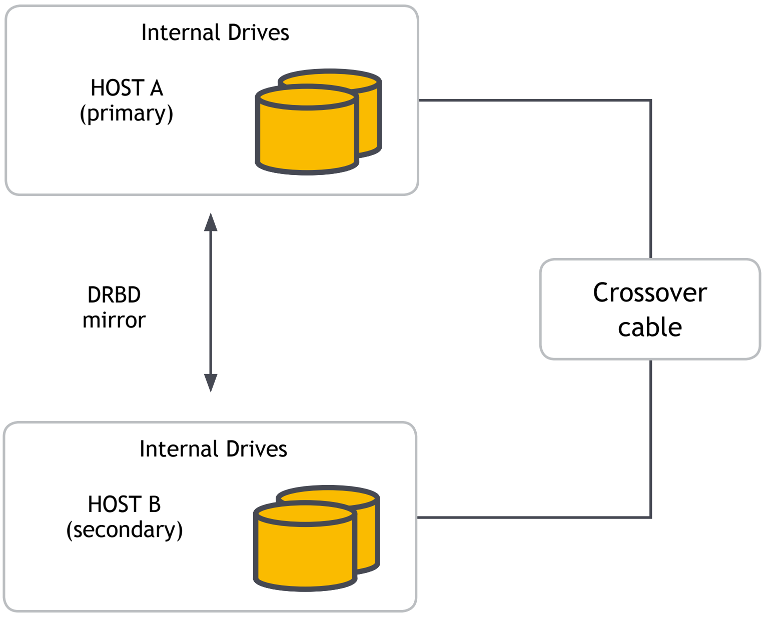
Key features of Lustre 1.8:

- **Adaptive timeouts** - offers simplified management for small and large clusters, automatically adjusts RPC timeouts as network conditions and server loads change, and reduces server recovery time.
- **OSS read cache** - provides read-only caching of data on the OSS side, allows OSTs to cache read data more frequently, improves repetitive reads to match network speeds instead of disk speeds.
- **OST pools** - allows the system administrator to create a named group of OSTs for file striping purposes, easier disk policy implementation and ease of disk management.
- **Version-based recovery (VBR)** - improves the speed of client recovery operations and allows Lustre to recover even if multiple clients fail at the same time as the server. VBR permits clients not to be evicted if some miss recovery and unrecovered clients may try to recover after the server recovery window closes.

The Dell Lustre storage system offers a failover solution based on HA-Linux Heartbeat software. HA-Linux Heartbeat software is responsible for detecting failure of the primary Lustre servers and controlling the resource transition to secondary nodes. The Dell Lustre Storage High Availability Modules use Heartbeat version 1 configuration which provides reliable two-node failover and simplified configuration. To ensure full data protection by forcing the server to disconnect from the shared storage, Heartbeat software is able to completely power-off the failed server. The process of automatic power control and management is called STONITH (Shoot The Other Node In The Head).

Heartbeat-stonith is configured to control and manage power via the servers IPMI interface.
The Dell Lustre HA-MDS module uses DRBD® to provide a shared storage for Lustre metadata targets (Figure 7). DRBD® is a distributed storage system for the Linux platform. DRBD® refers to both the software (kernel module) and also a specific logical block device managed by the software.
DRBD® creates a logical block device which mirrors a real block device via the network.
DRBD® intercepts the data, writes it to the local disk of a primary host and sends it to the other (secondary) host's local disk. DRBD® is configured to work with the Heartbeat software. In fact Heartbeat is responsible for control and management of the DRBD® logical block device. If the primary node fails, Heartbeat switches the secondary device into primary state and restarts the application there. If the failed node comes up again, it becomes a secondary node and synchronises its content to the primary. Synchronisation is performed in the background and does not affect the operation of the primary device.

Figure 7. DRBD® high-level architecture

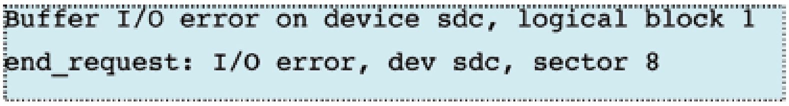


6.0 System Software Configuration

This section describes the steps to install required software packages and configuration of the Dell Lustre™ storage system.

6.1 Dell Storage Configuration Prerequisites

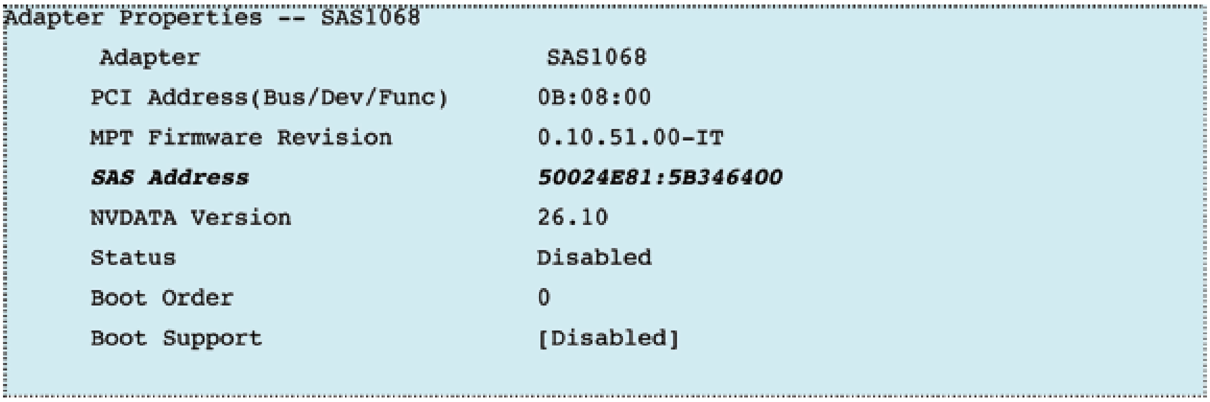
The Dell™ PowerVault™ MD3000 physically connects to each OSS node via a dual port SAS 5/E HBA (see Figure 6). Until all software is properly installed, Linux operating system errors such as the following may occur in the system log while booting:



```
Buffer I/O error on device sdc, logical block 1
end_request: I/O error, dev sdc, sector 8
```

It is safe to ignore such errors until installation of all software has been completed. It is recommended to disconnect the MD3000 from the server prior to Linux® OS and RDAC driver installation to avoid the lengthy boot time caused by above messages flooding the system logs.

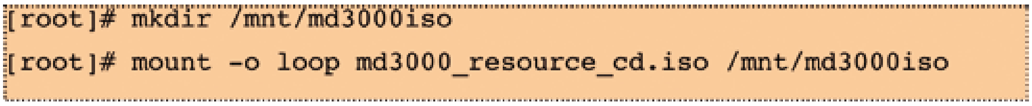
The SAS address of the SAS 5/E HBA is required for the MD3000 host to access manual configuration data. This information can be found using the SAS 5/E configuration utility. The configuration utility can be invoked during OSS server initialisation by pressing the <Ctrl><C> key combination. Once the configuration utility is loaded, the adapter properties can be viewed by selecting an adapter from the available menu. The adapter properties window is displayed, from which the SAS address can be found.



```
Adapter Properties -- SAS1068
       Adapter                         SAS1068
      PCI Address(Bus/Dev/Func)       0B:08:00
      MPT Firmware Revision           0.10.51.00-IT
      SAS Address                     50024E81:5B346400
      NVDATA Version                  26.10
      Status                          Disabled
      Boot Order                      0
      Boot Support                    [Disabled]
```

All necessary Linux host software is available on the Dell PowerVault MD3000 resource CD image. The latest resource CD image can be downloaded from the Dell Support website at support.dell.com.

The downloaded image can be mounted as a loop device. It is recommended to place the mount point on, or copy the contents of the MD3000 resource CD to a shared NFS location in order to make it available to all OSS servers.



```
[root]# mkdir /mnt/md3000iso
[root]# mount -o loop md3000_resource_cd.iso /mnt/md3000iso
```

The MD3000 installation process includes the following software modules:

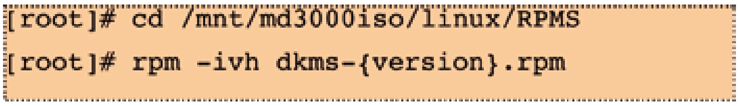
DKMS - some software modules are provided in the form of a Dynamic Kernel Module Support (DKMS) package. It is necessary to install the DKMS rpm prior to installation of other software modules.

RDAC Multi Path Proxy Driver – since each OSS server has multiple paths to the same Virtual Disk (VD) via SAS cables it is necessary to install a special Multi Path Proxy (MPP) driver which consolidates all paths to a VD into a single pseudo block device. If path failure occurs the MPP driver automatically switches paths, and the OSS server continues to access data through the same pseudo block device but via the alternate path.

MPT Driver – provides latest driver for the SAS 5/E HBA.

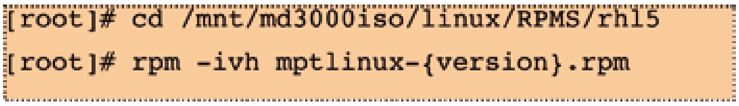
Dell Configuration and Management – the MD3000 resource CD provides the Modular Disk Storage Manager and Command Line Interface. This software tool provides a means of accessing the MD3000 storage for configuration and monitoring purposes.

DKMS installation:



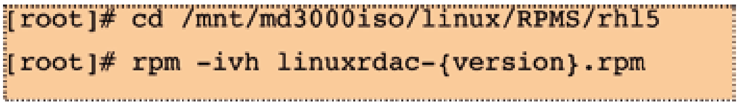
```
[root]# cd /mnt/md3000iso/linux/RPMS
[root]# rpm -ivh dkms-{version}.rpm
```

SAS 5/E HBA driver installation:



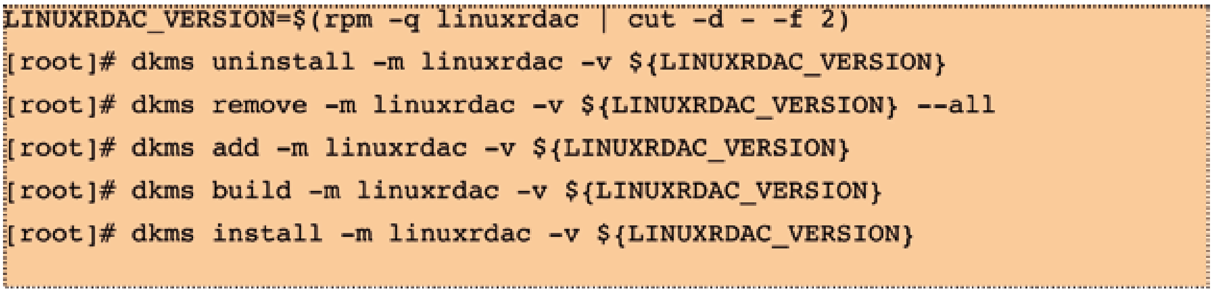
```
[root]# cd /mnt/md3000iso/linux/RPMS/rhl5
[root]# rpm -ivh mptlinux-{version}.rpm
```

RDAC Multi-Path Proxy Driver Installation



```
[root]# cd /mnt/md3000iso/linux/RPMS/rhl5
[root]# rpm -ivh linuxrdac-{version}.rpm
```

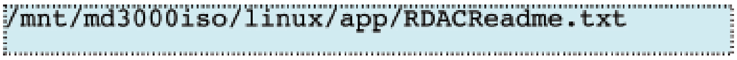
The Dell MPP driver has wide range of configuration parameters that can be changed for the current session or permanently by updating the initial ramdisk. The mppUtil tool can be used to update some configuration parameters with new values for the current session. The configuration parameters are stored in the /etc/mpp.conf file. The system administrator can edit this file and change the values of the parameters to match the required configuration. In order to make these changes persistent through a reboot, a new initial ram disk should be built using the following steps:



```
LINUXRDAC_VERSION=$(rpm -q linuxrdac | cut -d - -f 2)
[root]# dkms uninstall -m linuxrdac -v ${LINUXRDAC_VERSION}
[root]# dkms remove -m linuxrdac -v ${LINUXRDAC_VERSION} --all
[root]# dkms add -m linuxrdac -v ${LINUXRDAC_VERSION}
[root]# dkms build -m linuxrdac -v ${LINUXRDAC_VERSION}
[root]# dkms install -m linuxrdac -v ${LINUXRDAC_VERSION}
```

If the installation process completes without error, a reboot of the server is necessary to ensure that the new initial ramdisk with the updated parameters is loaded.

Additional installation instructions can be found in the README file located on the Dell™ PowerVault™ MD3000 resource CD:

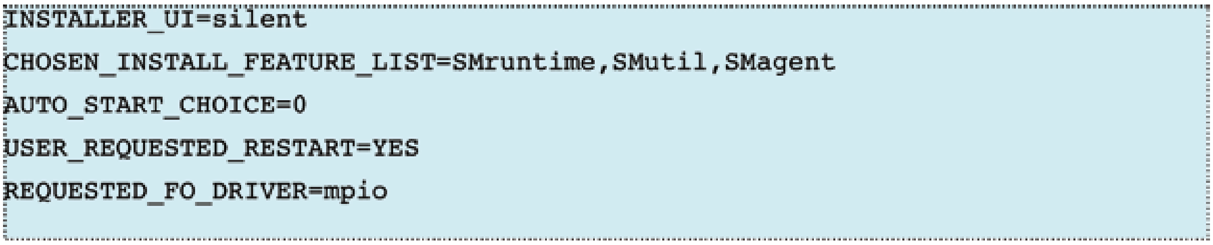


```
/mnt/md3000iso/linux/app/RDACReadme.txt
```

When installing the PowerVault Modular Disk Storage Manager and Host Utilities the System administrator has the option of using an interactive installer which requires the X Windows System or of using a silent mode which doesn't require X Windows. The second method is a good choice for the cluster environment where software has to be installed on many nodes, allowing automation of the installation process to all available servers.

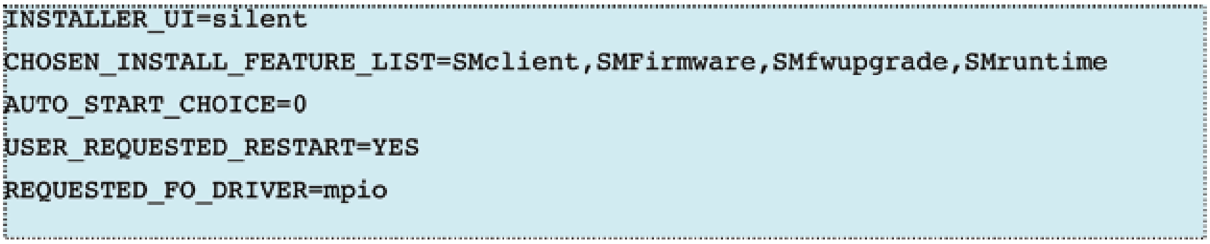
To install the Modular Disk Storage Manager (MDSM) using silent mode, create a config file called mdsm installer.option. The contents of the file depend on the role of the server in the Dell Lustre Storage System. Typically OSS storage servers require only agent software and utility packages to be installed. Management software and firmware upgrade software may be installed on the server that is not a part of the HA-OSS module. Usually these software packages are installed on a headnode or dedicated management node.

OSS servers file contents:



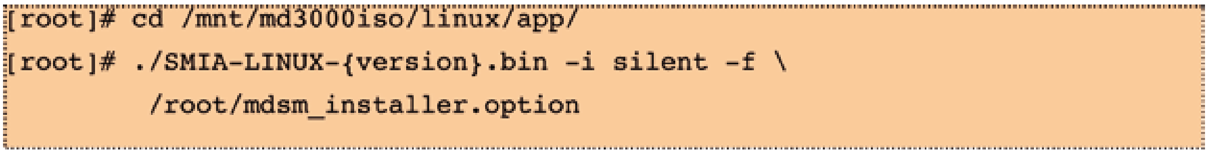
```
INSTALLER_UI=silent
CHOSEN_INSTALL_FEATURE_LIST=SMruntime,SMutil,SMagent
AUTO_START_CHOICE=0
USER_REQUESTED_RESTART=YES
REQUESTED_FO_DRIVER=mpio
```

Management servers file contents:



```
INSTALLER_UI=silent
CHOSEN_INSTALL_FEATURE_LIST=SMclient,SMFirmware,SMfwupgrade,SMruntime
AUTO_START_CHOICE=0
USER_REQUESTED_RESTART=YES
REQUESTED_FO_DRIVER=mpio
```

Once the appropriate mdsm_installer.option file is created, software can be installed with the following commands:

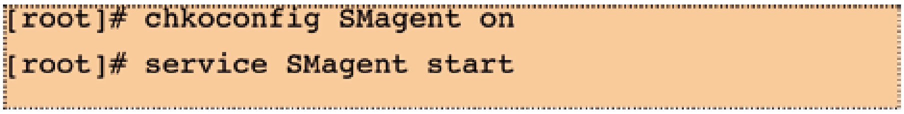


```
[root]# cd /mnt/md3000iso/linux/app/
[root]# ./SMIA-LINUX-{version}.bin -i silent -f \
         /root/mdsm_installer.option
```

The above process can be streamlined to use a single shell script which will run on the server to install all necessary software - an example script is shown in appendix A.

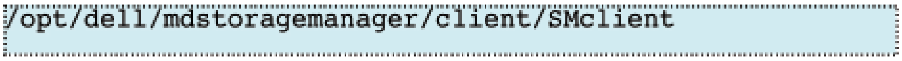
6.2 Dell PowerVault MD3000 Disk Array Initial Configuration

Before attempting MD3000 array configuration, care should be taken to ensure that all storage arrays are correctly connected (check Figure 6 for details) and the necessary software described in the previous section has been successfully installed. On the OSS servers the SMagent service should be started prior to attempting any configuration. The SMagent service can be configured in the Linux OS to start automatically at boot time:



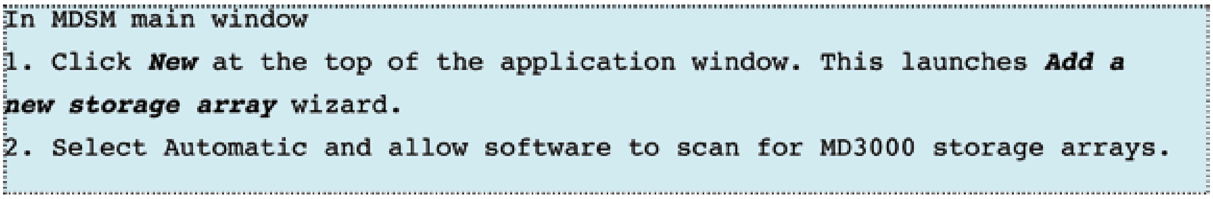
```
[root]# chkoconfig SMagent on
[root]# service SMagent start
```

To start the configuration process run the MDSM application that has been installed on the management node:



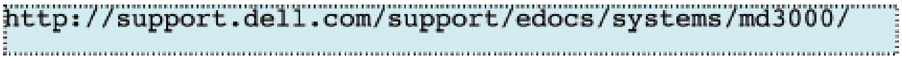
```
/opt/dell/mdstoragemanager/client/SMclient
```

Perform the initial setup tasks as follows:



```
In MDSM main window
1. Click New at the top of the application window. This launches Add a
new storage array wizard.
2. Select Automatic and allow software to scan for MD3000 storage arrays.
```

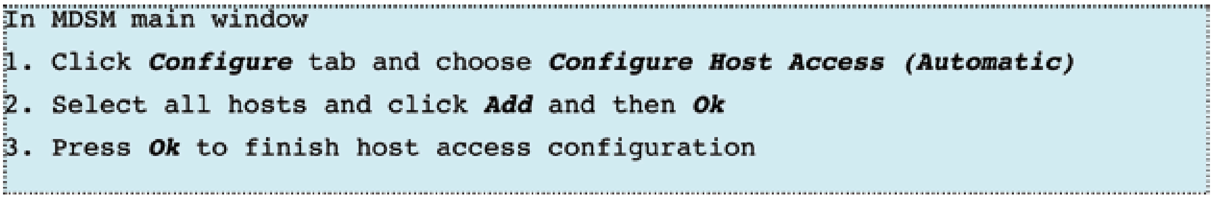
Once the scan is complete, all connected MD3000 storage arrays should be visible. The Initial Setup Tasks window will pop up and offer a range of tasks that should be performed on a newly installed MD3000 disk array. It is recommended to start with upgrading the storage array component firmware. Detailed instructions can be found in the User's Guide on the Dell PowerVault MD3000 documentation site:



```
http://support.dell.com/support/edocs/systems/md3000/
```

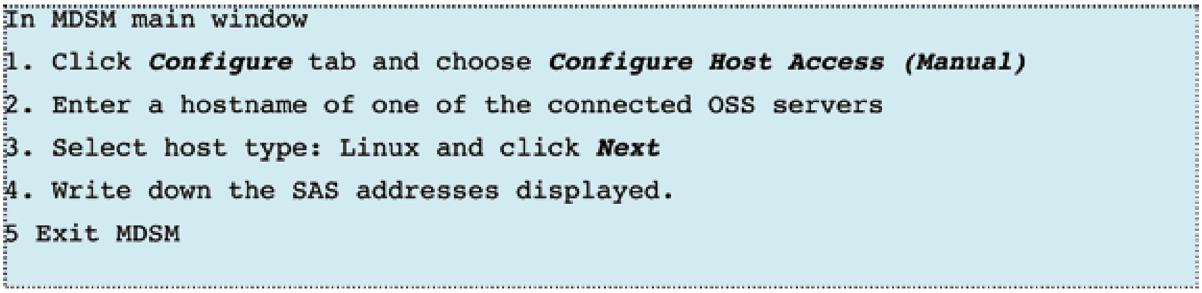
The next step of the configuration process is enabling the OSS Linux server host access to the MD3000 storage array. Host access can be configured either automatically or manually. In the automatic method all hosts running the SMagent service should be automatically detected by the MDSM software. If MDSM cannot detect hosts automatically then the alternative manual method can be used to complete the task.

Follow these steps to configure host access automatically:



```
In MDSM main window
1. Click Configure tab and choose Configure Host Access (Automatic)
2. Select all hosts and click Add and then Ok
3. Press Ok to finish host access configuration
```

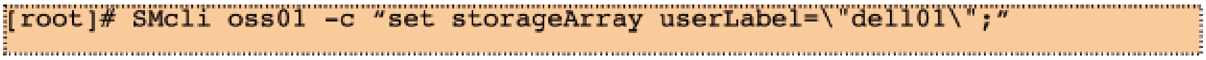
Follow these steps to obtain the host SAS adapter port addresses needed for manual host access configuration:



```
In MDSM main window
1. Click Configure tab and choose Configure Host Access (Manual)
2. Enter a hostname of one of the connected OSS servers
3. Select host type: Linux and click Next
4. Write down the SAS addresses displayed.
5 Exit MDSM
```

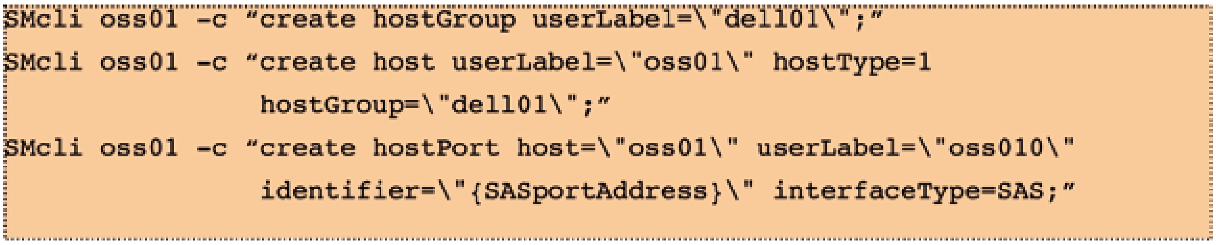
Each port on the SAS card has a unique SAS address. Port addresses on the same adapter differ only in the last two digits. Having collected all the required details, the rest of the configuration can be completed using the Storage Manager Command Line Interface SMcli. The command line tool can be used to configure storage array parameters using a batch mode as follows.

Configure storage array name:



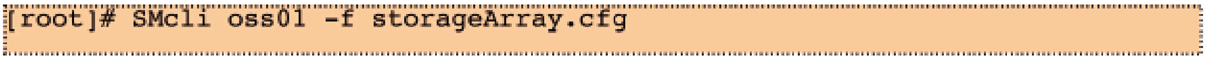
```
[root]# SMcli oss01 -c “set storageArray userLabel=\"dell01\";”
```

Example configuration of storage array host access and host group:



```
SMcli oss01 -c “create hostGroup userLabel=\"dell01\";”
SMcli oss01 -c “create host userLabel=\"oss01\" hostType=1
                hostGroup=\"dell01\";”
SMcli oss01 -c “create hostPort host=\"oss01\" userLabel=\"oss010\"
                identifier=\"{SASportAddress}\" interfaceType=SAS;”
```

The most efficient way of configuration is via configuration script:



```
[root]# SMcli oss01 -f storageArray.cfg
```

A complete script for configuring a Dell MD3000 HA-OSS module can be found in Appendix B.

6.3 Dell PowerVault MD3000 Lustre Configuration Best Practices

One of the biggest factors affecting overall Dell™ PowerVault™ MD3000 storage array performance is the correct choice and configuration of the RAID disk groups and virtual disks, especially when using RAID5 or RAID6 disk groups. Correctly configured RAID groups minimise the number of expensive read and write operations that are needed to complete a single I/O operation.

Figure 8. shows the disk configuration of the Dell Lustre HA-OSS module. There are 4 RAID6 disk groups configured. Each RAID6 disk group consists of 8 data disks and 2 parity disks. The segment size of the disk group is 512KB and the stripe size is 4096KB. Such a large segment size minimises the number of disks that need to be accessed to satisfy a single I/O operation. The Lustre typical I/O size is 1MB, therefore each I/O operation requires access to two disks in a RAID stripe. Disks in each disk group are spanned across all three disk arrays. This leads to faster disk access per I/O operation. For optimal performance only one virtual disk per disk group is configured, which results in more sequential access of the RAID disk group and which may significantly improve I/O performance. Lustre manual suggests using the full device instead of a partition (sdb vs sdb1). When using the full device, Lustre writes well-aligned 1 MB chunks to disk. Partitioning the disk removes this alignment and will noticeably impact performance. Additional performance boost can be gained by using separate devices for the EXT3 (lustre patched ext3) filesystem. The grey disks on Figure 8. can be used as journal devices. Fast SAS drives are recommended for this purpose. The Lustre configuration in this document does not use separate journal devices.

Figure 8. Disk configuration of the Dell Lustre HA-OSS module

- 🟩 OST1
- 🟦 OST2
- 🟥 OST3
- 🟨 OST4

	Disk0	Disk1	Disk2	Disk3	Disk4	Disk5	Disk6	Disk7	Disk8	Disk9	Disk10	Disk11	Disk12	Disk13	Disk14
Enclosure 0	C0P0	C0P0	C0P0	C0P0	C1P0	C1P0	C1P0	C0P1	C0P1	C0P1	C1P1	C1P1	C1P1	C1P1	▨
Enclosure 1	C0P0	C0P0	C0P0	C1P0	C1P0	C1P0	C1P0	C0P1	C0P1	C0P1	C1P1	C1P1	C1P1	▨	▨
Enclosure 2	C0P0	C0P0	C0P0	C1P0	C1P0	C1P0	C0P1	C0P1	C0P1	C0P1	C1P1	C1P1	C1P1	▨	▨

The Dell MD3000 controller configuration requires some adjustment to optimise performance of the Lustre filesystem. After performing a number of tests exercising various controller parameters, the following choice of configuration parameters seems to be best suited to the Lustre filesystem where the requirement is high performance under large I/O workload:

```
Summary:
      cacheBlockSize=16
      segmentSize=512
      readCacheEnabled=false
      cacheReadPrefetch=false
      modificationPriority=lowest
```

Disabling the read cache retains all the cache memory for write operations, which has a positive effect on write speeds. Linux® OS can compensate for the absence of a controller read cache by means of an adaptive read-ahead cache. The default setting of the read-ahead algorithm in the Red Hat® Linux® is 256 sectors per read. The default value is too small for sequential I/O and increasing this value to 8192 sectors significantly improves read performance. In addition, the Lustre 1.8 OSS is equipped with a read cache feature which also has a positive impact on storage system read performance. Some extra write performance boost can be gained by disabling the write cache mirroring feature. Making this change however needs to be carefully considered because it decreases data integrity safety so it is only recommended for scratch storage systems.

6.4 Lustre™ Installation

The easiest way to install Lustre is to obtain Lustre RPM packages from the Lustre maintainer website, currently http://wiki.lustre.org/index.php/Download

In order to determine which Lustre packages should be downloaded please check with Lustre support matrix http://wiki.lustre.org/index.php/Lustre_Release_Information#Lustre_Support_Matrix

The Lustre Servers require the following RPMs to be installed:

- kernel-lustre: lustre patched kernel (server)
- kernel-ib: OFED (optional, client and server)
- lustre-modules (client and server)
- lustre: Lustre utilities package (client and server)
- lustre-ldiskfs: Lustre-patched backing file system kernel module (server)
- e2fsprogs: Utilities package for ext3 backing file system
- lustre-test: various test utilities

The Lustre Client requires these RPMS to be installed:

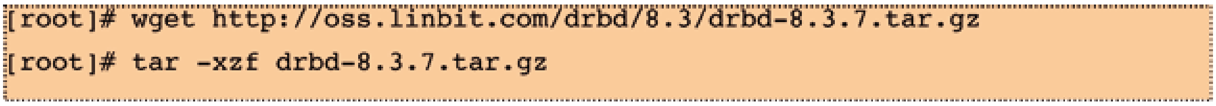
- kernel-ib: OFED (optional, client and server)
- lustre-modules (client and server)
- lustre: Lustre utilities package (client and server)
- lustre-test: various test utilities (optional)

Lustre installation is completed by installing the above RPMs and rebooting the server into the

6.5 Configure DRBD on the MDS

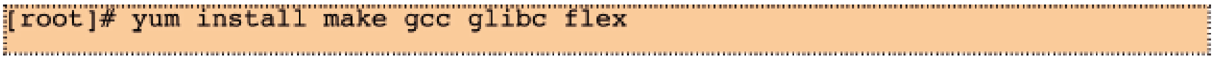
The Distributed Replicated Block Device® (DRBD) is a software-based replicated storage solution mirroring the content of block devices between servers. DRBD® mirrors data in real time - replication happens continuously, while applications modify data on the block device. Replication is completely transparent, applications which use the mirrored device are unaware of the fact that the data are being mirrored across the servers. DRBD® can be configured to work in synchronous or asynchronous mode. With synchronous mirroring an application accessing the mirrored device for writes is notified of write completion only after the modifications have been transmitted to the other computer system. This mode is the best choice for a production system and it is used for providing reliable and safe DRBD® shared storage for the Dell Lustre HA-MDS module.

The DRBD® source code can be downloaded at http://oss.linbit.com/drbd/. The latest stable version is drbd-8.3.7:



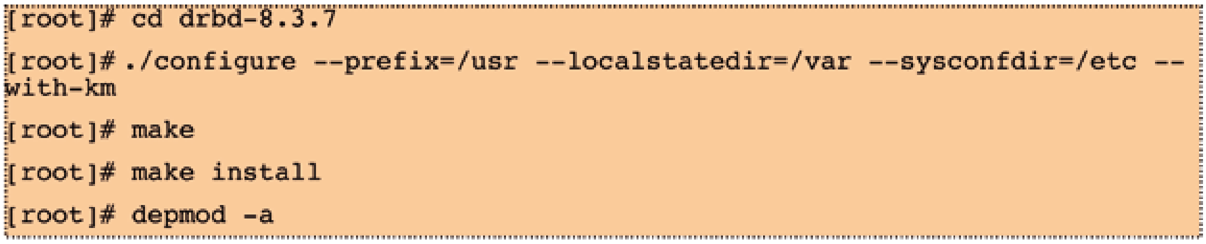
```
[root]# wget http://oss.linbit.com/drbd/8.3/drbd-8.3.7.tar.gz
[root]# tar -xzf drbd-8.3.7.tar.gz
```

Before beginning the build process the host server must have the following prerequisites installed:



```
[root]# yum install make gcc glibc flex
```

In order to build the DRBD® software these steps should be followed:



```
[root]# cd drbd-8.3.7
[root]# ./configure --prefix=/usr --localstatedir=/var --sysconfdir=/etc --
with-km
[root]# make
[root]# make install
[root]# depmod -a
```

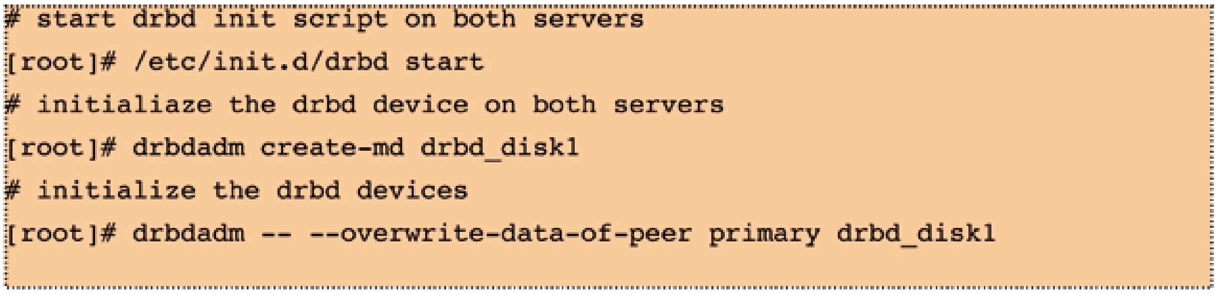
Please also visit http://www.drbd.org/users-guide/s-build-from-source.html for more detailed information about the build process.

After successful DRBD® installation a local disk block device needs to be prepared on both MDS servers. Each MDS is equipped with 6 drives of which 4 SAS 300GB drives are configured into a RAID10 virtual disk. This RAID level choice seems to be best suited to the Lustre metadata I/O patterns. The configuration is performed via PERC 5/i Integrated Bios Configuration Utility which can be accessed at boot time by pressing Ctr+R when prompted on the screen. The Virtual Disk Management (VD Mgmt) menu allows creating new Virtual Disk by pressing F2 button and choosing Create New VD option. This will display Create New VD panel where user can make the choice of which physical disks will be used to create a new VD, RAID parameters, and cache settings. The new VD which will be used by the DRBD® software uses 4 drives configured as RAID10. The stripe size can be left at the default value of 64KB, read ahead can be left as disabled to preserve cache for the write-back cache. Write back cache should be enabled only if the controller is equipped with the battery protection.

In order to work properly, DRBD devices need a network over which to keep each other in sync. It is strongly recommended that a dedicated connection is used for this purpose. The configuration described here uses a direct, back-to-back Gigabit Ethernet connection.

DRBD® configuration is controlled by a single configuration file /etc/drbd.conf. This file must be identical on both MDS servers in order for DRBD® to work properly. The Dell HA-MDS module DRBD® configuration file can be found in Appendix C. Documentation describing in detail DRBD® installation, configuration, and management can be found at http://www.drbd.org/docs/about/.

When DRBD® is started for the first time it must be initialised:



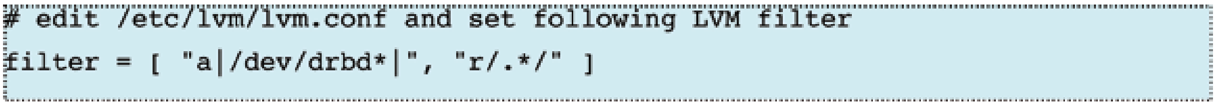
```
# start drbd init script on both servers
[root]# /etc/init.d/drbd start
# initialaze the drbd device on both servers
[root]# drbdadm create-md drbd_disk1
# initialize the drbd devices
[root]# drbdadm -- --overwrite-data-of-peer primary drbd_disk1
```

6.6 LVM Configuration

The Dell Lustre HA-MDS module uses an LVM volume as the base for the metadata target (MDT). LVM allows one to regularly backup MDT device content by using LVM snapshots. This can be done without taking the Lustre filesystem offline. Metadata are crucial for the correct functioning of the Lustre filesystem and therefore need to be protected from accidental damage caused by unexpected failures. In addition to RAID protection it is strongly recommended that regular backups of the metadata are performed. This will minimise damage caused by unexpected, irrecoverable hardware failure.

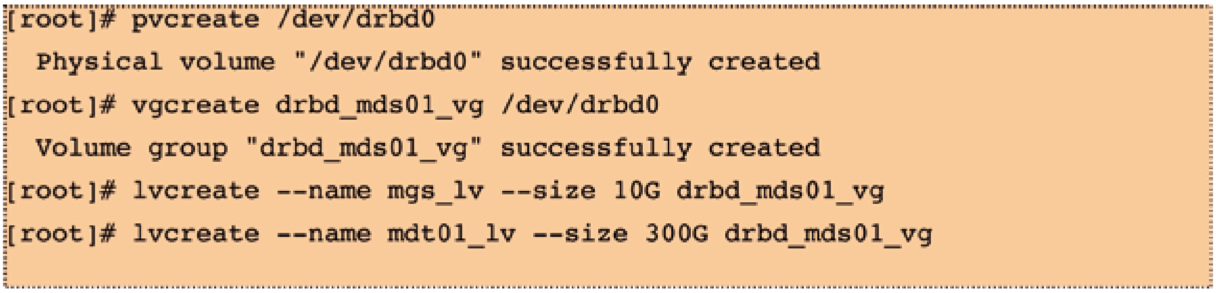
Although the MD3000 has built-in snapshot and virtual disk copy functionality it is not discussed in this document since these are premium features and require licence to activate them.

For LVM to function correctly with DRBD® an LVM filter needs to be configured to exclude any duplicates of the DRBD® device (only on MDS server).



```
# edit /etc/lvm/lvm.conf and set following LVM filter
filter = [ "a|/dev/drbd*|", "r/.*/" ]
```

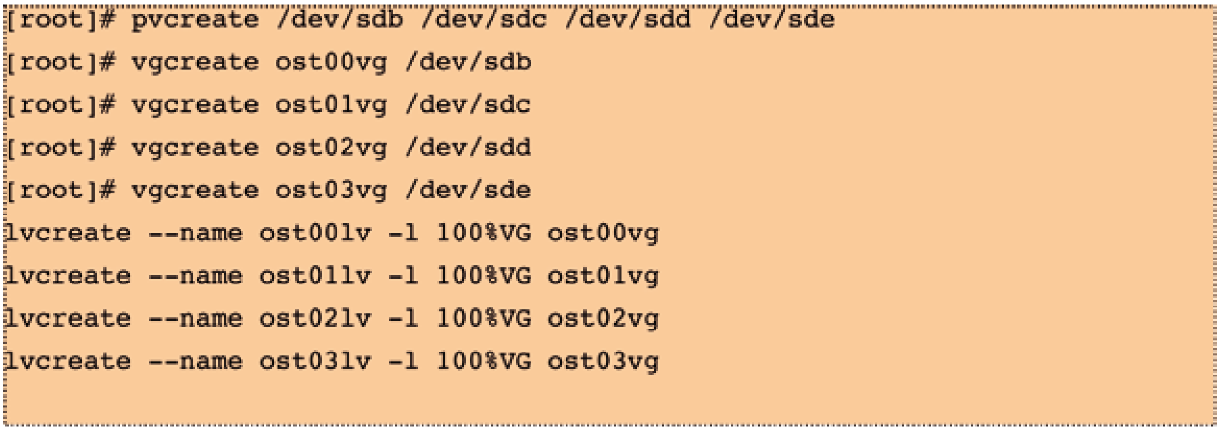
The following instructions create a 10GB MGS LVM volume and 300GB MDT LVM volume on top of the DRBD device.



```
[root]# pvcreate /dev/drbd0
  Physical volume "/dev/drbd0" successfully created
[root]# vgcreate drbd_mds01_vg /dev/drbd0
  Volume group "drbd_mds01_vg" successfully created
[root]# lvcreate --name mgs_lv --size 10G drbd_mds01_vg
[root]# lvcreate --name mdt01_lv --size 300G drbd_mds01_vg
```

For general purpose Lustre filesystems it is recommended that LVM volumes are used as a base for OST devices. LVM volumes provide additional features that can help system administrators to manage Lustre devices more efficiently. More details about the advantages of using LVM can be found in Lustre 1.8 manual (chapter 15.5).

Overhead added by LVM is negligible as long as snapshots are not used. Using snapshots can decrease performance and is not recommended on production filesystems (with the exception of short periods of time when backing up MDT devices). Lustre configuration described in this document uses LVM-based OSTs.



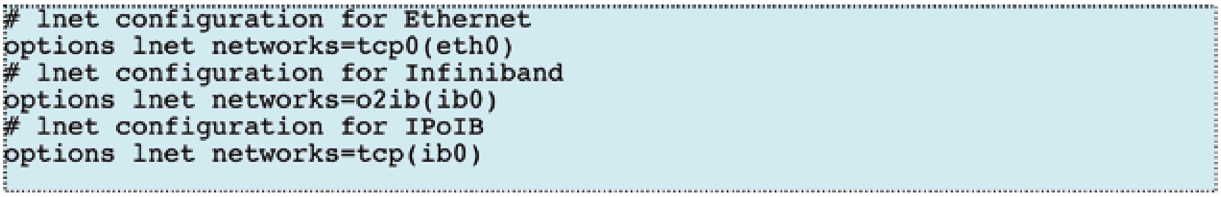
```
[root]# pvcreate /dev/sdb /dev/sdc /dev/sdd /dev/sde
[root]# vgcreate ost00vg /dev/sdb
[root]# vgcreate ost01vg /dev/sdc
[root]# vgcreate ost02vg /dev/sdd
[root]# vgcreate ost03vg /dev/sde
lvcreate --name ost00lv -l 100%VG ost00vg
lvcreate --name ost01lv -l 100%VG ost01vg
lvcreate --name ost02lv -l 100%VG ost02vg
lvcreate --name ost03lv -l 100%VG ost03vg
```

Lustre network configuration:

In order for Lustre network to operate correctly, the system administrator must verify that the cluster network (Ethernet or Infiniband) has been set up correctly and that clients and servers can communicate with each other using the standard network protocols for the particular network type used.

In a cluster with Lustre filesystem, servers and clients communicate with each other using Lustre networking API (LNET). LNET supports a variety of network transports through Lustre Network Drivers (LND). For example ksocklnd enables support for Ethernet and ko2iblnd enables support for Infiniband. End points of the Lustre network are defined by Lustre network identifiers (NID). Every node that uses Lustre filesystem must have an NID. Typically NID looks like this: address@network - where address is the name or IP address of the computer and network is one of the Lustre network type e.g. tcp0 or o2ib.

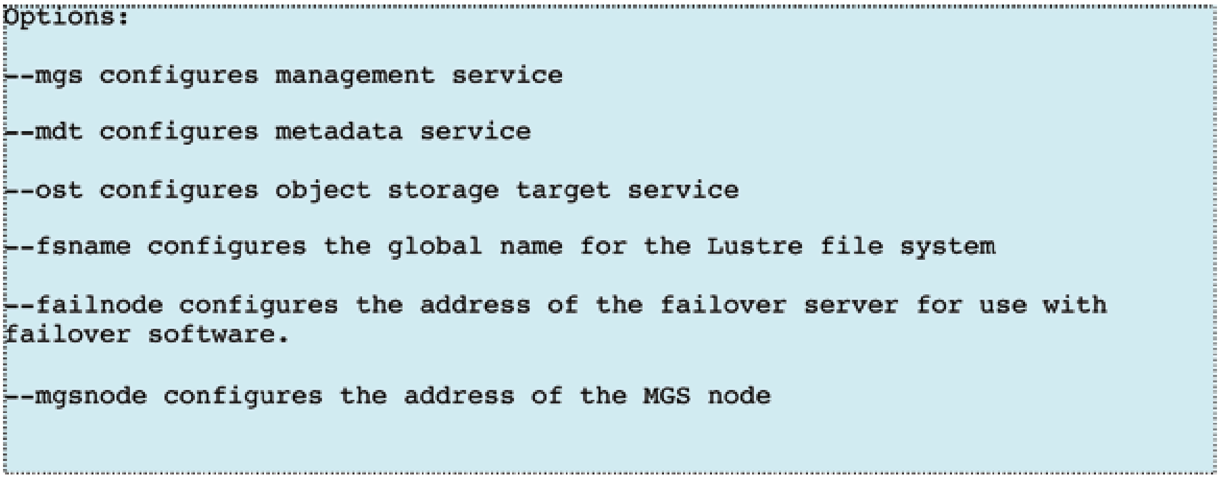
When configuring Lustre network in Red Hat® 5 the system administrator needs to add one of the following lines into `/etc/modprobe.conf file.`



```
# lnet configuration for Ethernet
options lnet networks=tcp0(eth0)
# lnet configuration for Infiniband
options lnet networks=o2ib(ib0)
# lnet configuration for IPoIB
options lnet networks=tcp(ib0)
```

Below are defined names and options used for the Dell Lustre System configuration.

Ethernet connected modules		Infiniband connected modules	
Hostname	Network Interface	Hostname	Network Interface
mds01	eth0	mds01_ib	ib0
mds02	eth0	mds02_ib	ib0
oss01	eth0	oss01_ib	ib0
oss02	eth0	oss02_ib	ib0

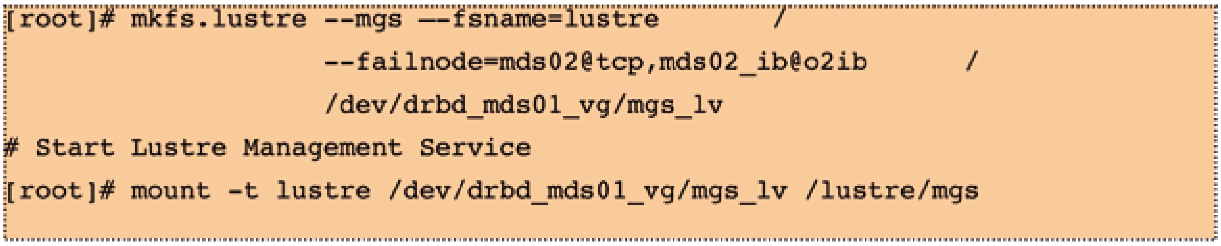


```
Options:

--mgs configures management service

--mdt configures metadata service

--ost configures object storage target service

--fsname configures the global name for the Lustre file system

--failnode configures the address of the failover server for use with
failover software.

--mgsnode configures the address of the MGS node
```

More information about Lustre networking can be found at
http://wiki.lustre.org/manual/LustreManual18_HTML/UnderstandingLustreNetworking.html

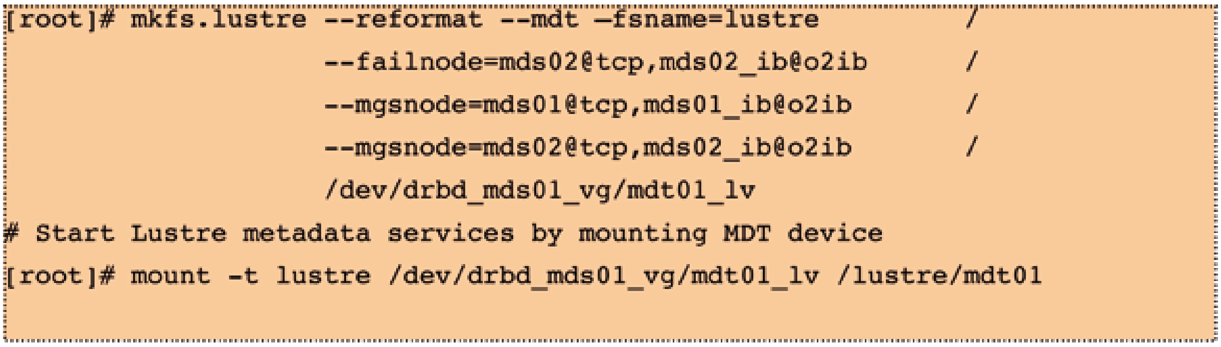
Dell Lustre HA-MDS Module Configuration

Configure and start Lustre Management Server (typically MGS and MDS run on the same host)



```
[root]# mkfs.lustre --mgs —-fsname=lustre       /
                    --failnode=mds02@tcp,mds02_ib@o2ib      /
                    /dev/drbd_mds01_vg/mgs_lv
# Start Lustre Management Service
[root]# mount -t lustre /dev/drbd_mds01_vg/mgs_lv /lustre/mgs
```

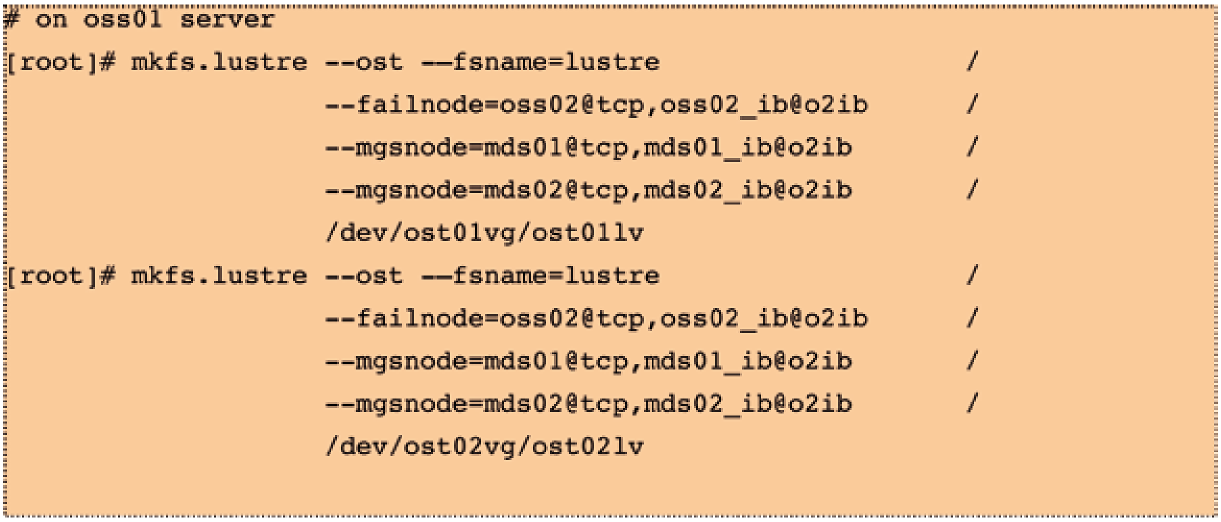
Configure and Start Lustre Matadata Server



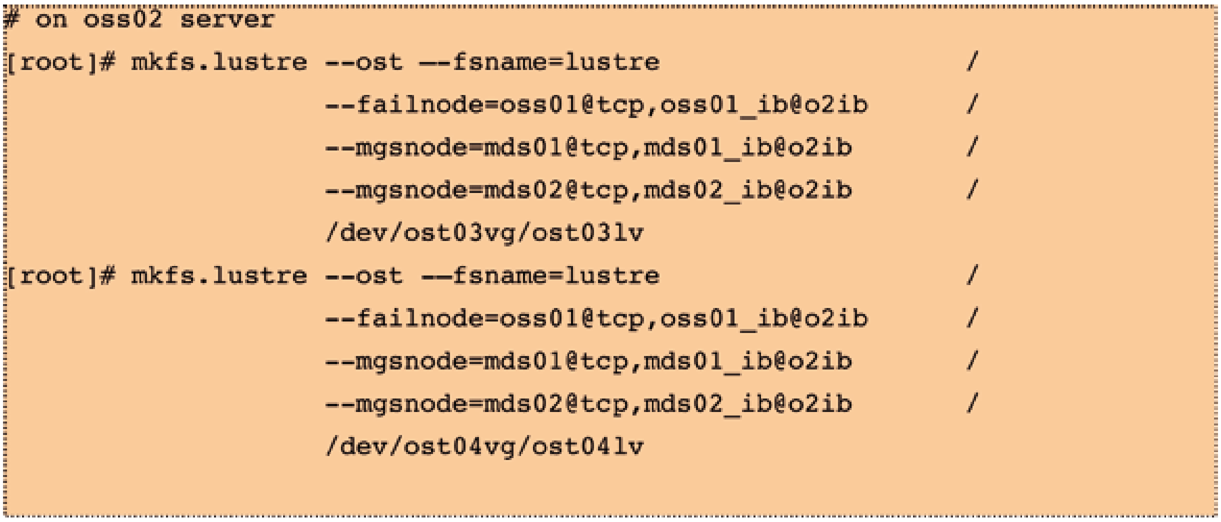
```
[root]# mkfs.lustre --reformat --mdt —fsname=lustre         /
                    --failnode=mds02@tcp,mds02_ib@o2ib      /
                    --mgsnode=mds01@tcp,mds01_ib@o2ib       /
                    --mgsnode=mds02@tcp,mds02_ib@o2ib       /
                    /dev/drbd_mds01_vg/mdt01_lv
# Start Lustre metadata services by mounting MDT device
[root]# mount -t lustre /dev/drbd_mds01_vg/mdt01_lv /lustre/mdt01
```

Dell Lustre HA-OSS Module Configuration

Configure Lustre Object Storage Servers

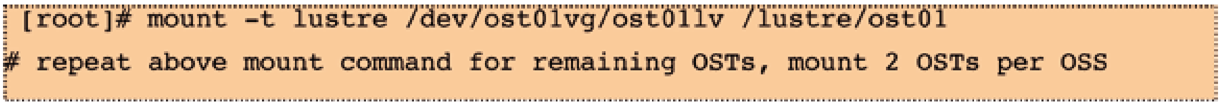


```
# on oss01 server
[root]# mkfs.lustre --ost --fsname=lustre                        /
                    --failnode=oss02@tcp,oss02_ib@o2ib           /
                    --mgsnode=mds01@tcp,mds01_ib@o2ib            /
                    --mgsnode=mds02@tcp,mds02_ib@o2ib            /
                    /dev/ost01vg/ost01lv
[root]# mkfs.lustre --ost --fsname=lustre                        /
                    --failnode=oss02@tcp,oss02_ib@o2ib           /
                    --mgsnode=mds01@tcp,mds01_ib@o2ib            /
                    --mgsnode=mds02@tcp,mds02_ib@o2ib            /
                    /dev/ost02vg/ost02lv
```



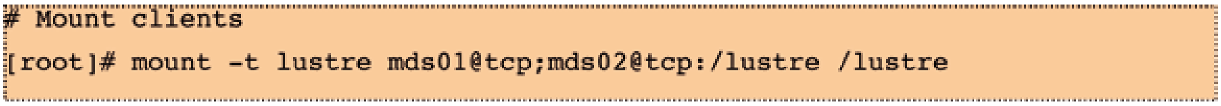
```
# on oss02 server
[root]# mkfs.lustre --ost --fsname=lustre                        /
                    --failnode=oss01@tcp,oss01_ib@o2ib           /
                    --mgsnode=mds01@tcp,mds01_ib@o2ib            /
                    --mgsnode=mds02@tcp,mds02_ib@o2ib            /
                    /dev/ost03vg/ost03lv
[root]# mkfs.lustre --ost --fsname=lustre                        /
                    --failnode=oss01@tcp,oss01_ib@o2ib           /
                    --mgsnode=mds01@tcp,mds01_ib@o2ib            /
                    --mgsnode=mds02@tcp,mds02_ib@o2ib            /
                    /dev/ost04vg/ost04lv
```

Start Lustre Object Storage services



```
 [root]# mount -t lustre /dev/ost01vg/ost01lv /lustre/ost01
# repeat above mount command for remaining OSTs, mount 2 OSTs per OSS
```

Start Lustre on the client node



```
# Mount clients
[root]# mount -t lustre mds01@tcp;mds02@tcp:/lustre /lustre
```

6.7 Lustre Administration Best Practices

Lustre™ was designed with large sequential I/O in mind and its early usage was only as scratch filesystems at large Datacentres. Nowadays, Lustre is considered to be not only one of the best cluster file systems, but it finds its place in more general purpose applications. Lustre comes with a good range of features and tools and learning how to use them effectively can help to run Lustre-based filesystems in a smooth and efficient way.

6.7.1 Small Files Lustre Performance

Although in general Lustre is designed to work with large files it is possible to tune lustre client parameters to improve small file performance. It is especially useful to perform this tuning on nodes that are interactively accessed by users, for example login nodes:

- **Disable LNET debug on client nodes:** By default Lustre records many types of debug messages. Disabling them may increase client perfomance.

```
cat /proc/sys/lnet/debug
ioctl neterror warning error emerg ha config console
# disable debug messages
sysctl -w lnet.debug=0
```

- **Increase dirty cache on client nodes:** By default Lustre will have 32Mbytes of dirty cache per OST.

```
lctl get_param osc.*.max_dirty_mb
osc.scratch2-OST0000-osc-ffff8102266f1000.max_dirty_mb=32
...
osc.scratch2-OST0003-osc-ffff8102266f1000.max_dirty_mb=32
# increase dirt cache per OST
lctl set_param osc.*.max_dirty_mb=256
```

- **Increase number of RPCs in flight:** By default Lustre have 8 RPCs in flight. Increasing this number up to 32 can improve perfomance of both normal data and metadata.

```
lctl get_param osc.*.max_rpcs_in_flight
osc.scratch2-OST0000-osc-ffff8102266f1000.max_rpcs_in_flight=8
...
osc.scratch2-OST0003-osc-ffff8102266f1000.max_rpcs_in_flight=8
# increase number of RPCs in flight
lctl set_param osc.*. max_rpcs_in_flight=32
```

- **Lustre striping:** Lustre performance degrades if small files are striped on many OSTs. It is a good practice to set stripe count for `/home directories to 0 or 1`

  ```
  lfs setstripe -c 0 /home/user
  ```

6.7.2 Managing OST Free Space

Lustre provides many tools and methods for managing free space on the OSTs. It is important to monitor and keep OSTs free space in balance to avoid OSTs getting full. Making use of OST pools enables administrators to create default file allocation policies which can improve file access and I/O performance.

- **Working with OST pools:** The OST pool feature enables administrators to group OSTs together to make object placement more flexible. A "pool" is a name associated with an arbitrary subset of OSTs. Use the *lctl* command to create/destroy a pool, add/remove OSTs in a pool, list pools and OSTs in a specific pool. The *lctl* command **MUST** be run on the MGS node. Pools can be used to group OSTs with the same technology or performance (slower or faster) or those that are preferred for certain jobs. Examples are SATA OSTs versus SAS OSTs.

  ```
  # create OST poll <poolname>
  lctl pool_new <fsname>.<poolname>

  # add even number of OSTs to newly created pool
  lctl pool_add <fsname>.<pool name> OST[0-10/2]

  # list pools in the named filesystem
  lctl pool_list <fsname>

  # list OSTs in the named pool
  lctl pool_list <fsname>.<poolname>

  # remove named OSTs from the pool
  lctl pool_remove <fsname>.<poolname> <ost_list>

  # destroy pool
  lctl pool_destroy <fsname>.<poolname>
  ```

- **Handling Full OSTs:** If an OST becomes full and an attempt is made to write more information to the file system, an error occurs. The procedures below describe how to deal with a full OST.

```
lfs df

UUID                  1K-blocks     Used          Available    Use%    Mounted on
scratch2-MDT0000_UUID 237804280     6958248       217255916    2%      /scratch2[MDT:0]
scratch2-OST0000_UUID 7687338532    7405477732    281860800    99%     /scratch2[OST:0]
scratch2-OST0001_UUID 7687338532    6499283628    797560416    84%     /scratch2[OST:1]
scratch2-OST0002_UUID 7687338532    6477962704    818872512    84%     /scratch2[OST:2]
scratch2-OST0003_UUID 7687338532    6483592540    813251428    84%     /scratch2[OST:3]
```

OST:0 is almost full and attempts to write to this OST may fail as follows:

```
writing '/scratch2/user/test' : No space left on device
```

To enable continued use of the filesystem, the full OST has to be deactivated using the *lctl* command. This is done on the MDS, since the MSD allocates space for writing.

```
[root@mds01 ~]# lctl dl | grep OST0000

   5 UP osc scratch2-OST0000-osc scratch2-mdtlov_UUID 5

[root@mds01 ~]# lctl --device 5 deactivate

# Check that OST is INactive

[root@mds01 ~]# lctl dl | grep OST0000

   5 IN osc scratch2-OST0000-osc scratch2-mdtlov_UUID 5
```

In order to free-up some space on the full OST some data can be manually migrated off that OST. This can be accomplished by copying data located on the deactivated OST to a new location. New files will be created using only active OSTs. Then original data can be deleted and copies of the data can be moved to their original location.

```
[root@client01 ~]# cp /scratch2/user/file /scratch2/user/file.tmp

[root@client01 ~]# rm scratch2/user/file

[root@client01 ~]# mv /scratch2/user/file.tmp scratch2/user/file
```

Use lfs find command to find files located on the particular OST.

```
lsf find --obd scratch2-OST0000_UUID /scratch2/user/somefiles > file.list
```

Lustre manual contains a script which helps to automate the migration task. Also Lustre BUG 22481 contains the latest version of that script.

6.7.3 Recreating Lustre Configuration Logs

If the Lustre configuration logs are in a state where the filesystem cannot be started, writeconf command can be used to erase them. After the writeconf command is run and the servers restart, the configuration logs are re-generated and saved on the MGS.

The *writeconf* command should only be used if configuration logs were corrupted and *filesystem* cannot start or a server NID has changed. Running *writeconf* command will erase pool information and conf_param settings so please make sure that those parameters are recorded in the recovery_ script that can be run after *writeconf* execution.

Before running *writeconf* Lustre filesystem should be stopped on all clients and servers.
The *writeconf* command must be run on MDT first and then on all OSTs.

```
[root@mds01 ~]# tunefs.lustre --writeconf <mdt_device>
[root@oss01 ~]# tunefs.lustre --writeconf <ost_device>

...

[root@oss10 ~]# tunefs.lustre --writeconf <ost_device>
```

Restart the file system in this order:

```
mount MGS
mount OSTs
mount MDT
Mount clients
```

6.7.4 Recovering From Errors or Corruption on a Backing ext3 File System

When an OSS, MDS, or MGS server crash occurs, it is not necessary to run e2fsck on the file system. Ext3 journaling ensures that the file system remains coherent. Also the backing file systems are never accessed directly from the client, so client crashes are not relevant. Sometimes however an event may occur that ext3 journal is unable to handle. This can be caused by hardware failure or I/O error. If ext3 detects corruption, filesystem is remounted read-only and reported in the syslog as error -30. In such a case it is required to run e2fsck on the bad device. Once e2fsck fixes the problems, the device can be put back into service. If a serious problem is suspected, it is recommended that first e2fsck is run with -n switch. It is also useful to record output of the e2fsck to a file if the information is needed later.

```
e2fsck -fn /dev/<device> #don't fix anything, just check for corruption

e2fsck -fp /dev/<device> #fix filesystem using prudent answers
```

6.7.5 Recovering From Corruption in The Lustre File System

In situations where MDS or an OST becomes corrupt, the *lfsck* distributed filesystem check can be run to determine how serious the corruption is and what sort of problems exist. This process consists of many steps and may take a very long time to complete.

- run e2fsck -f to fix any local backing filesystem errors.
- build mds database, it is quicker to write the database file to local filesystem. Depending on the number of files, this step can take several hours to complete.
  ```
  e2fsck -n -v --mdsdb /tmp/mdsdb /dev/{mdsdev}
  ```
- Make this file accessible on all OSSs and use it to generate OST database file per each OST.
  ```
  e2fsck -n -v --mdsdb /tmp/mdsdb --ostdb /tmp/{ostNdb} /dev/{ostNdev}
  ```
- Make the mdsdb file and all ostdb files available on a mounted client and run lfsck to examine the file system. Optionally, correct the defects found by lfsck
  ```
  lfsck -n -v --mdsdb /tmp/mdsdb --ostdb /tmp/{ost1db} /tmp/{ost2db} ... /lustre/mount/point
  ```

6.7.6 Lustre backups with LVM snapshots

MDT device backup

Periodic device backups are recommended - MDT backups are specially important as this device holds all the metadata information for the whole filesystem. In order to do MDT device backup, it has to be remounted as ldiskfs, which means that Lustre has to be stopped on the MDS server. However if the MDT resides on top of the LVM volume, snapshot functionality can be used to create a snapshot volume which can then be mounted as ldiskfs and used for MDT device backup, avoiding filesystem downtime.

– Create LVM snapshot:

```
# Create 50MB snapshot volume
[root@mds01 ~]# lvcreate -L50M -s -n MDTbak1 drbd_mds01_vg
```

– Mount the snapshot as ldiskfs

```
[root@mds01 ~]# mkdir -p /mnt/mdtbak
[root@mds01 ~]# mount -t ldiskfs /dev/drbd_mds01_vg/MDTbak1 /mnt/mdtbak
[root@mds01 ~]# cd /mnt/mdtbak
```

– Backup extended attributes EA and filesystem data

```
[root@mds01 ~]# getfattr -R -d -m '.*' -P . > /backup/ea.bak
[root@mds01 ~]# tar czvf /backup/mdtbak1.tgz --sparse .
[root@mds01 ~]# cd -
```

– Umount the snapshot and reclaim the used space by removing snapshot volume

```
[root@mds01 ~]# umount /mnt/mdtbak
[root@mds01 ~]# lvremove drbd_mds01_vg/MDTbak1
```

Restoring from a device backup

If the MDT device failed or needs to be replaced, the following procedure can be used to restore the device data from the backup taken earlier.

– Format new device

```
[root@mds01 ~]# mkfs.lustre {--mdt|--ost} {other options} {newdev}
```

– Mount the filesystem as ldiskfs

```
[root@mds01 ~]# mount -t ldiskfs {newdev} /mnt/mdt
```

– Restore data from the backup

```
[root@mds01 ~]# cd /mnt/mdt
[root@mds01 ~]# tar zxvpf /backup/mdtbak1.tgz --sparse
```

– Restore EAs

```
[root@mds01 ~]# setfattr --restore=/backup/ea.bak
[root@mds01 ~]# rm OBJECTS/* CATALOGS
[root@mds01 ~]# cd -
```

– Umount the ldiskfs filesystem

```
[root@mds01 ~]# umount /mnt/mdt
```

A similar procedure can be used to backup/restore OST devices.

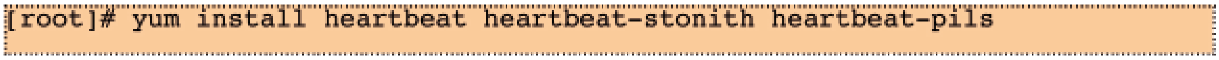
6.8 Heartbeat

Lustre does not offer a complete failover solution. It must be combined with high-availability (HA) software to enable full failover support. Linux-HA Heartbeat software provides all necessary functionality to provide a complete Lustre failover solution. HA-Linux Heartbeat is very portable and runs on many Linux platforms. It requires the following packages to be installed:

- Heartbeat: the Heartbeat subsystem for HA-Linux
- Heartbeat-stonith: provides an interface to Shoot The Other Node In The Head (STONITH)
- Heartbeat-pils: provides a general plugin and interface loading library

The easiest way to install Heartbeat is by downloading and installing RPM packages from http://mirror.centos.org/centos/5.4/extras/x86_64/RPMS/

When using CentOS Linux, HA-Linux Heartbeat can be installed via yum package manager:



```
[root]# yum install heartbeat heartbeat-stonith heartbeat-pils
```

Heartbeat configuration involves three configuration files that need to be identical on both failover nodes.

The three configuration files are:

***authkeys* -** This file contains keys for mutual node authentication, it has to have root-only readable permissions set.

***ha.cf* -** Global cluster configuration, this file is read by Heartbeat daemon on startup. It lists the communication facilities enabled between nodes, enables or disables certain features, and optionally lists the cluster nodes by host name.

***haresources* -** This file specifies the resources for the cluster and who the default owner is. The haresources file is one of the most important files to configure when using Heartbeat.

Heartbeat configuration parameters:

The Dell Lustre HA-MDS module Heartbeat parameters	
MDS node host name	mds01, mds02
MGS device	/dev/drbd_mds01_vg/mgs_lv
mount point	/lustre/mgs
MDT device	/dev/drbd_mds01_vg/mdt01_lv
mount point	/lustre/mdt01
IPMI device names	mds01_ipmi, mds02_ipmi

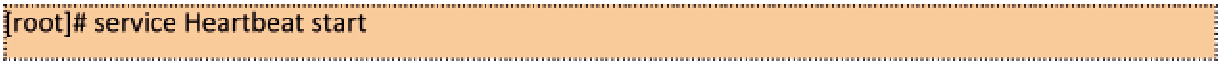
The Dell Lustre HA-OSS module Heartbeat parameters	
First OSS node parameters	
OSS node host name	oss01
OST device	/dev/ost01vg/ost01lv
mount point	/lustre/ost01
OST device	/dev/ost02vg/ost02lv
mount point	/lustre/ost02
IPMI device names	oss01_ipmi

The Dell Lustre HA-OSS module Heartbeat parameters	
Second OSS node parameters	
OSS node host name	oss02
OST device	/dev/ost03vg/ost03lv
mount point	/lustre/ost03
OST device	/dev/ost04vg/ost04lv
mount point	/lustre/ost04
IPMI device names	oss02_ipmi

The complete configuration files can be found in the appendix D.

Once the configuration is complete, the Heartbeat service can be started. It has to be started on both nodes at the same time to work properly. The role of the service is to start Heartbeat resources which mount Lustre™ filesystem and to manage the resource ownership. The Heartbeat service on the first node monitors the status of the partner node. If the partner node fails, the Heartbeat service can move resources mounted on that node to the failover node. In addition, the STONITH subsystem can power off or reboot the failed node to ensure integrity of the Lustre filesystem data. This keeps the Lustre filesystem availability as high as possible.

The following command starts Heartbeat service on the server node.

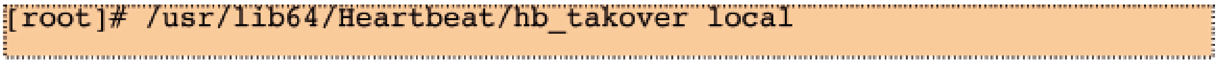


```
[root]# service Heartbeat start
```

It is recommended that the Heartbeat service is configured to start at server boot time.

The system administrator can issue a failover request manually by using `hb_takover` and `hb_standby` tools. Both tools can use one of following options: `all|foreign|local|failback`

For example



```
[root]# /usr/lib64/Heartbeat/hb_takover local
```

More detailed information about configuration and usage of HA-Linux Heartbeat with Lustre can be found at http://wiki.lustre.org/manual/LustreManual18_HTML/Failover.html

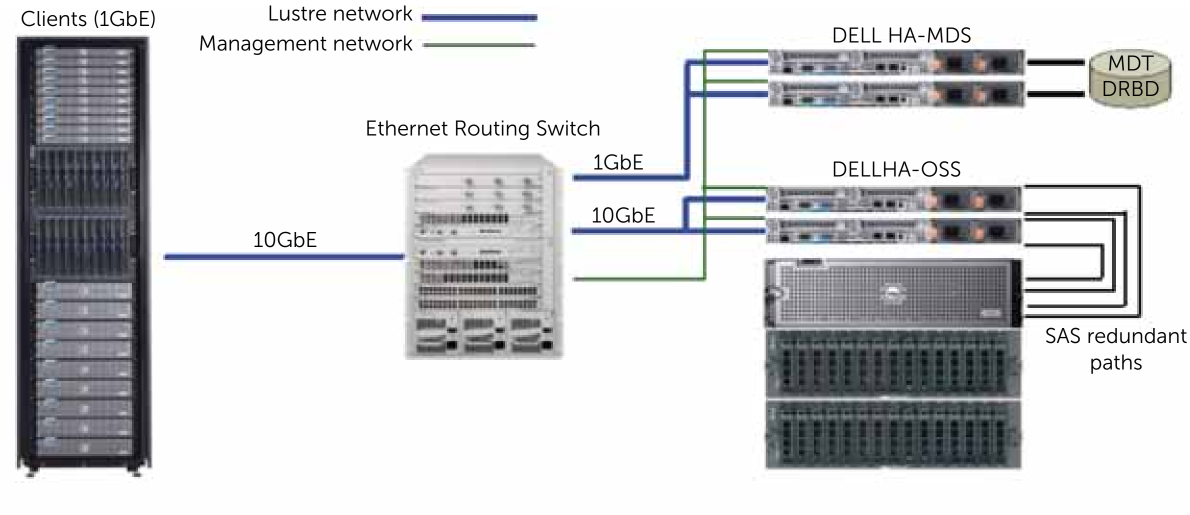
7.0 Performance Tests and Results

The purpose of the performance testing is to provide accurate Lustre filesystem throughput measurements of the Dell Lustre™ Storage System as configured in this paper. The results can then be compared with the expected theoretical performance of the tested storage system. Performance tests were conducted using one HA-MDS module and one HA-OSS module.

7.1 HA-OSS Module Raw Speed

The I/O network used for the OSS test is an Ethernet network as described in Fig 9.

Figure 9. Lustre Test Setup



The Lustre I/O kit (included in the Lustre-test package) provides a set of tools which help to validate the performance of various hardware and software layers. One of the I/O kit tools, called sgppd_survey, can test bare metal performance of the back-end storage system while bypassing as much of the kernel as possible. This tool can be used to characterise the performance of a storage device by simulating an OST serving files consisting of multiple stripes. The data collected by this survey can help to determine the performance of a Lustre OST exporting the device.

The raw bare metal performance of the Dell MD3000 would be expected to yield 600MB/s total throughput when using both array controllers simultaneously. Below are the results of the sgpdd_survey for tests using two and four LUNs.

Figure 10. sgpdd_survey with 2 LUNs - write

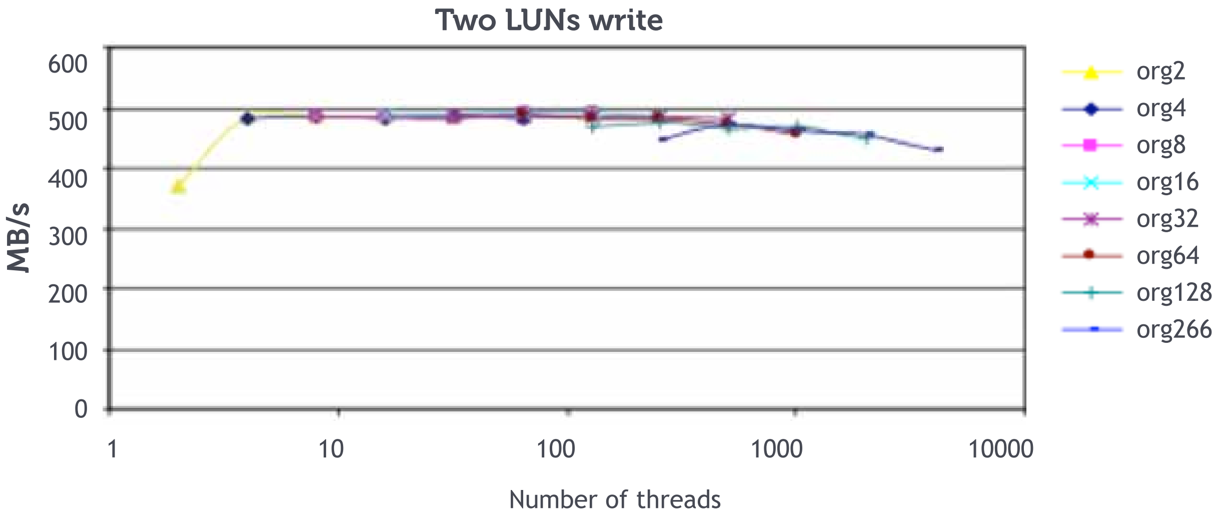


Figure 11. sgpdd_survey with 2 LUNs - read

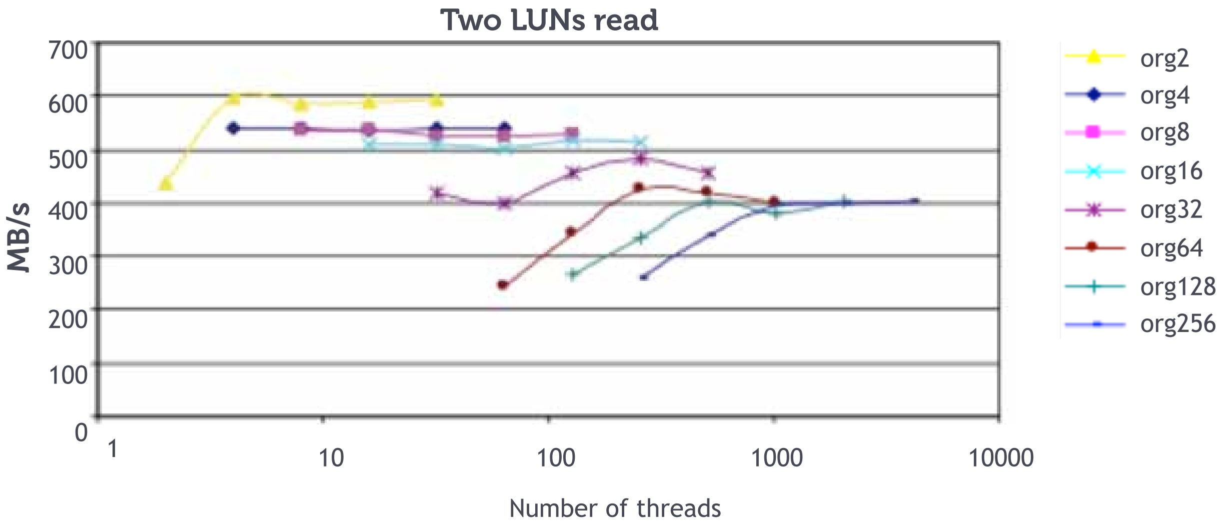


Figure 12. sgpdd_survey with 4 LUNs - read

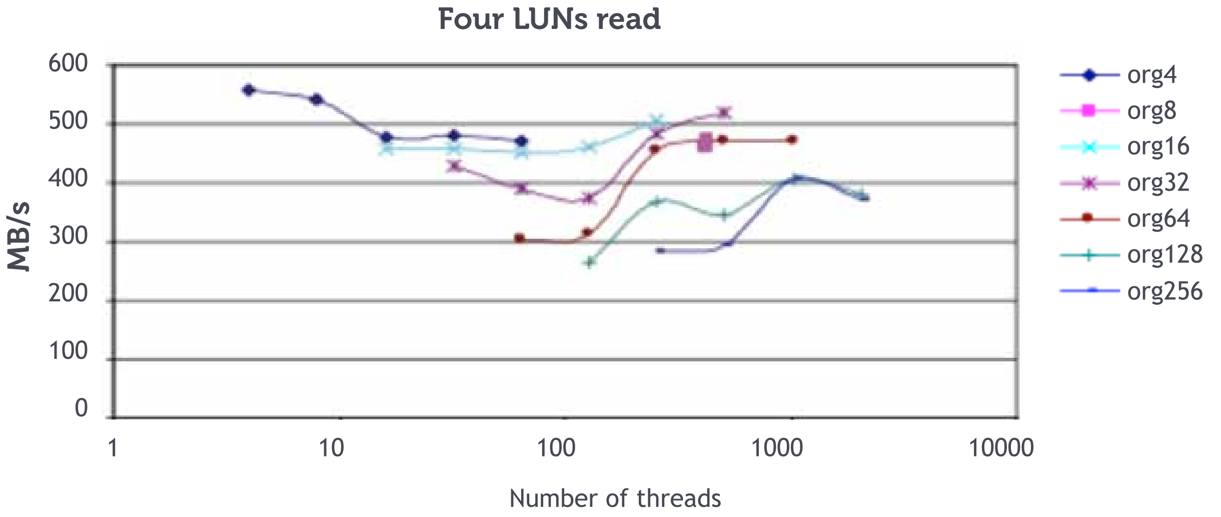
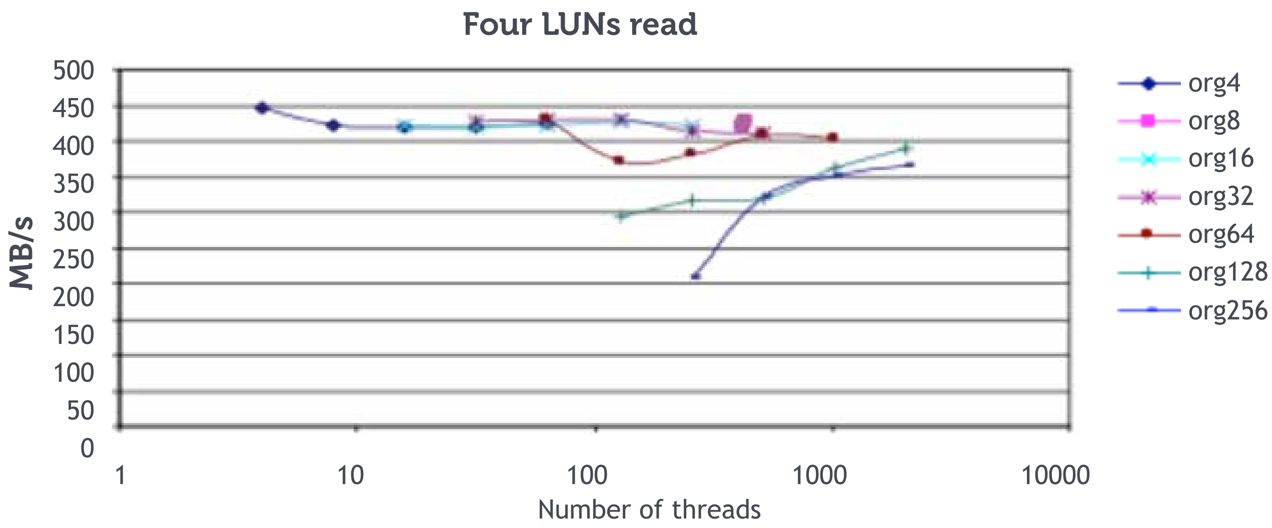


Figure 13. sgpdd_survey with 4 LUNs - write



The above results show that the PV MD3000 can provide near maximum theoretical read and write performance when tested with the sgpdd_survey tool. This demonstrates that the RAID configuration used is optimal. The tests using two LUNs show slightly better and more consistent results. This is due to the fact that both controllers are used but each controller services only one Virtual Disk. In the four LUNs test each controller services two Virtual Disks at the same time, which adds some small overhead. Summarising the above test results, it is clear that the PV MD3000 array as configured provides close to the maximum performance expected from the raw device and sets a good base line for further filesystem performance tests.

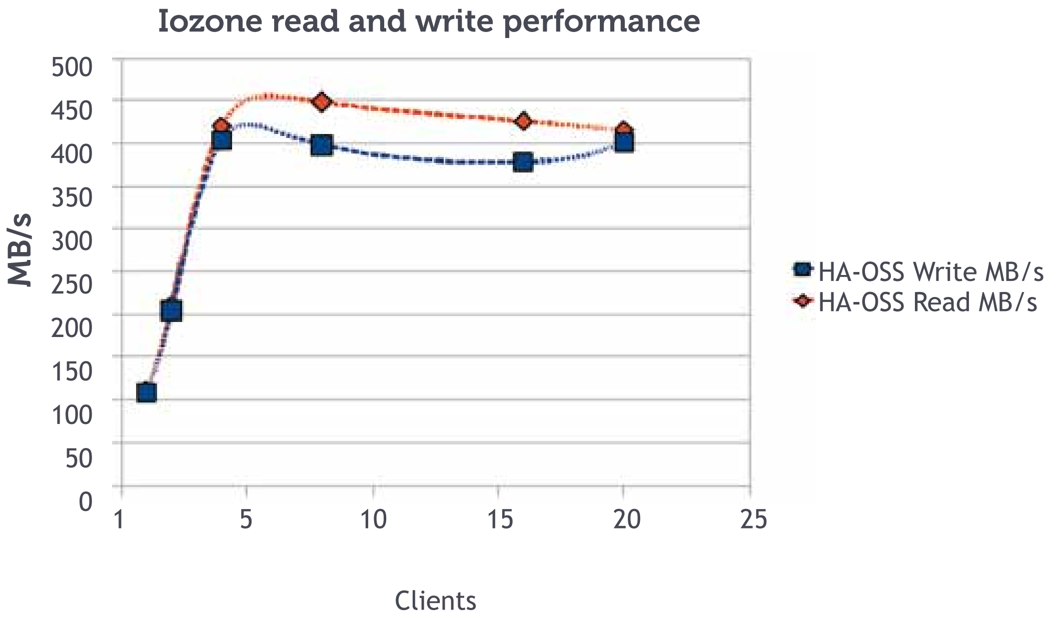
7.2 Sequential I/O Benchmark Through the Lustre Filesystem

Here we test the I/O performance obtained from the Lustre filesystem using Iozone. The Iozone benchmark tool generates and measures a variety of file operations. For instance read, write, re-read, re-write operations can be used to provide a broad analysis of filesystem performance. Iozone can work in a single client mode and cluster mode where it is provided with a list of nodes on which it will run. The following tests (Figure 14) were performed with 20 client nodes connected over Gigabit Ethernet. Each client was writing and reading a 16GB (twice the clients RAM size) file to minimise client side caching. Iozone tests were run with a 1MB block size, which is known to be best suited to getting optimum Lustre I/O performance. The following command line was used to start the Iozone test:

```
iozone -+m iozone-clients.txt -t 1 -s 16384m -r 1024k -i 0 -i 1
```

Figure 14. Dell Lustre Storage system Iozone tests

Clients	HA-OSS Write MB/s	HA-OSS Read MB/s
1	108	111
2	204	210
4	403	420
8	398	448
16	378	426
20	401	415



The above results show that Lustre filesystem scales almost linearly with the number of clients up to the point of saturation of the back-end storage performance. Adding more clients doesn't cause much performance degradation and overall performance remains at a very good level. When comparing the above results to the previous bare metal performance results, it can be observed that the loss of throughput caused by filesystem overhead is not high and the PV MD3000 disk array still provides a very high level of efficiency.

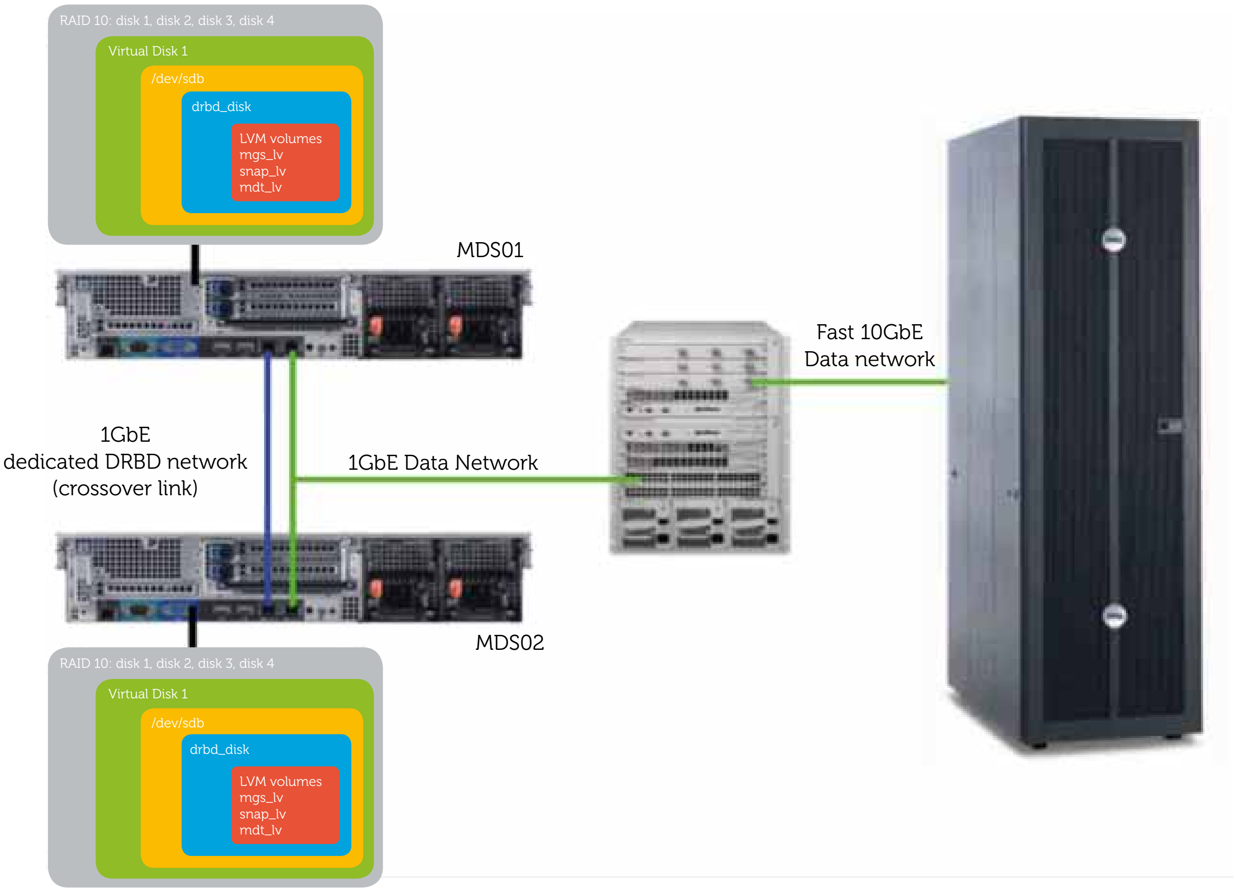
7.3 Scale-out Performance on Large Production Systems

Within the Cambridge HPC datacentre a large scale Dell™ PowerVault™ MD3000 storage system has been in production for two years. This system consists of six bricks as described in this paper providing 200TB of parallel filesystem storage and an total back-end storage of over 2 GB/s, demonstrating near linear scalability of Dell Lustre storage brick architecture.

7.4 Benchmarking Metadata Performance Of The Dell HA-MDS module

The test configuration of the Dell HA-MDS module is presented on Figure 15. The Dell HA-MDS module is connected to the cluster via 1GbE network and consists of two servers which use DRBD®, a network mirrored block device as a shared metadata target. DRBD® is configured on a dedicated 1GbE network with a latency increase of 10% as compared to writing to local disk. This can be reduced to 1% by using RDMA-based networking technology to link the two metadata servers. This makes the DRBD solution very efficient and cost effective while still providing good metadata performance for HPC workloads and enabling high availability features.

Figure 15. The Dell HA-MDS module test configuration



The metadata performance of the Dell HA-MDS module was measured using the mdtest benchmarking tool. The benchmarks measure the performance of most common metadata operations like directory and file creation and deletion and file "stat" operation. The benchmarks use MPI to coordinate and synchronise threads between the nodes, making them suitable for testing a single client performance as well as many clients, to show how performance scales with higher client numbers.

The mdtest benchmark was run using a maximum of 32 client nodes and the following parameters were used:

```
mdtest -n 10 -i 200 -y -N 10 -t -u -d $test_directory
```

-n: every process will creat/stat/remove # directories and files

-i: number of iterations the test will run

-y: sync file after writing

-N: stride # between neighbour tasks for file/dir stat (local=0)

-t: time unique working directory overhead

-u: unique working directory for each task

-d: the directory in which the tests will run

Figure 16. Dell Lustre Storage system mdtest benchmark – directory operations

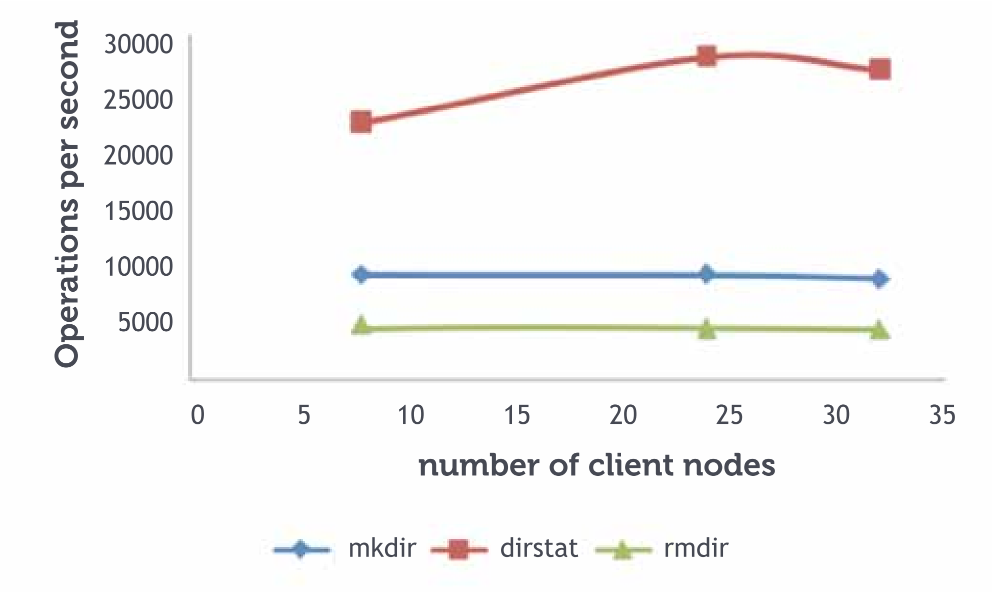
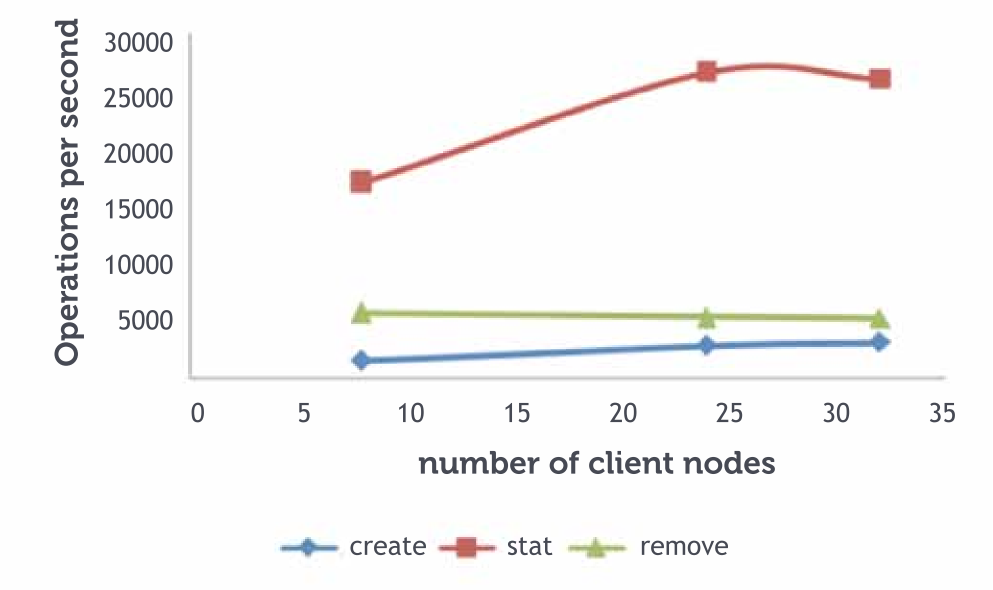


Figure 17. Dell Lustre Storage system mdtest benchmark – file operations



The above results show that Dell HA-MDS module can provide good and sustainable metadata performance. In case of “stat”, adding more clients can increase the number of operations. Directory and file “create” and “remove” operations remain roughly unchanged across the tests. Lustre filesystem has been designed to work with massive streaming parallel I/O where the typical file size is a megabyte or more. However as the above results show, Lustre not only excels in typical HPC environments but can also provide good performance for general purpose and home filesystems with mixed pattern file system access.

8.0 Summary

As commodity clusters continue to grow in performance and more applications adopt parallel I/O, traditional file systems such as NFS, NAS, SAN and DAS are failing to keep up with the HPC I/O demand. The Lustre parallel filesystem is gaining ground within the departmental and workgroup HPC space since it has proved capable of meeting the high I/O workload within HPC environments as well as presenting a range of additional operational benefits as described in the introduction.

This paper presents a detailed description of how to build a Dell™ PowerVault™ MD3000 Lustre storage brick. The storage brick as described shows good sequential I/O throughput, high redundancy and high availability with good data safety features, all of which are important within the HPC environment.

The storage brick demonstrates good Lustre filesystem performance attaining 80% of the raw bare metal performance and delivering 400MB/s back-end read/write I/O bandwidth. When the Lustre storage brick is connected to the client nodes via gigabit Ethernet each client can attain an I/O bandwidth of 100 MB/s which scales linearly with each successive client until the back-end storage bandwidth is saturated. Scale-out performance across multiple bricks is linear, achieving 80% of the theoretical peak when scaled across 6 bricks on an Ethernet I/O network.

Within the Cambridge HPC Service a large scale (6 brick) Dell Lustre storage solution has been constructed and used within the production environment for several years. This solution has demonstrated scalable performance across the storage array yielding over 2 GB/s file I/O bandwidth with high levels of availability and data security. Operational experience has demonstrated less than 0.5% un-planned downtime. We have found the Dell Lustre™ storage brick is a good match to both HPC performance and operational requirements seen within a large University HPC data centre.

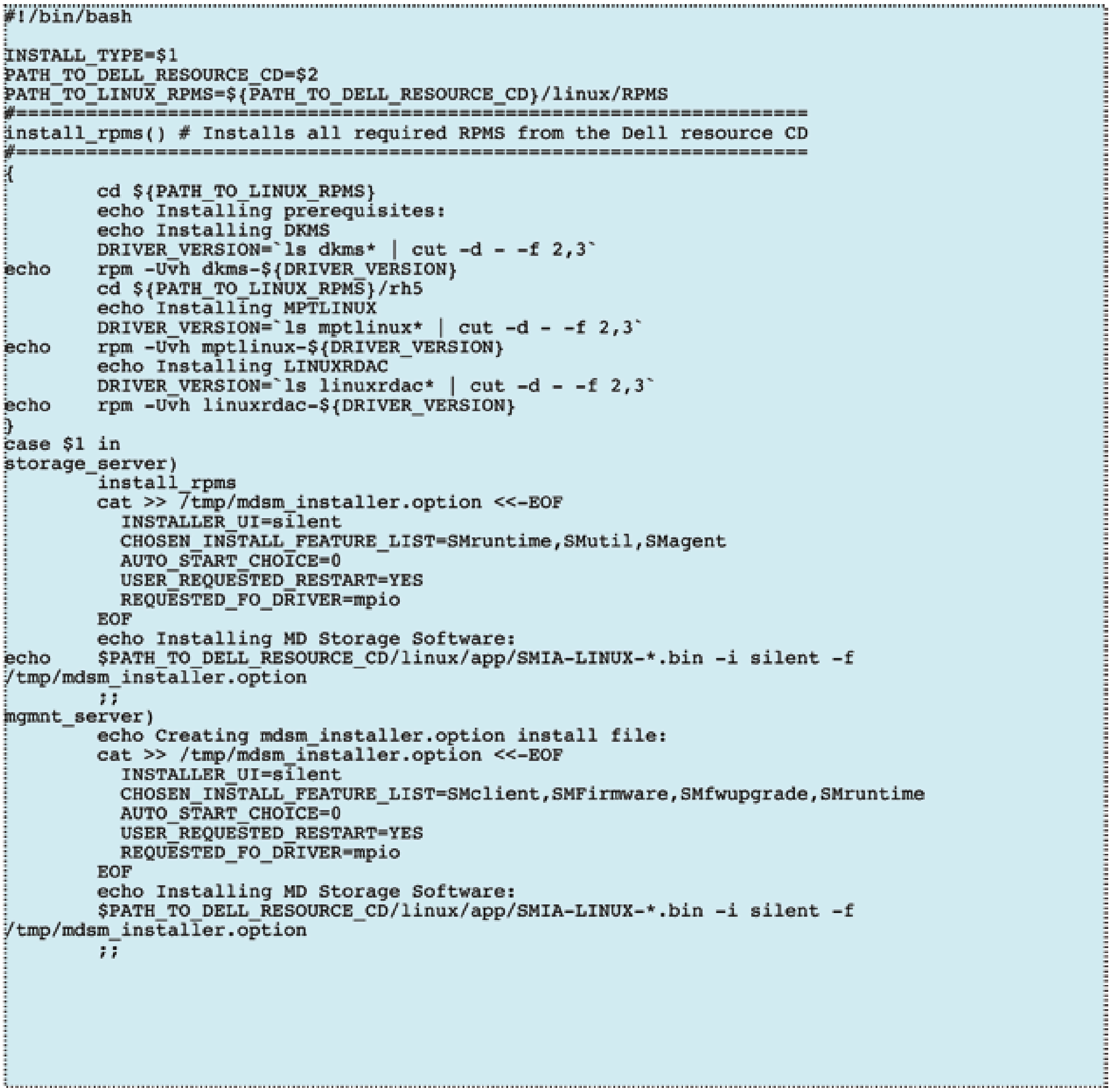
Looking forward it is likely that Lustre will continue to have strong open source development and support with a strong feature roadmap. Also the commodity storage hardware platforms will continue to improve in terms of disk density, raw performance and cost per TB. These developments combined with the use of Infiniband storage networks and solid state disks for metadata will combine to offer large sequential I/O and IOPS performance increases. This will help ensure that commodity Lustre storage arrays will continue to grow their presence within the mid range HPC segments and beyond.

A follow-up paper is being produced where we use the recently released (at the time of writing) Dell MD3200 storage arrays, with a combination of solid state disk technologies and QDR Mellanox Infiniband. This paper will concentrate on performance factors and, when referenced to the current paper, the reader will have an up-to-date view on commodity Lustre storage solutions, performance trends across hardware generations and real life scale-out / operational characteristics and system administration best practice.

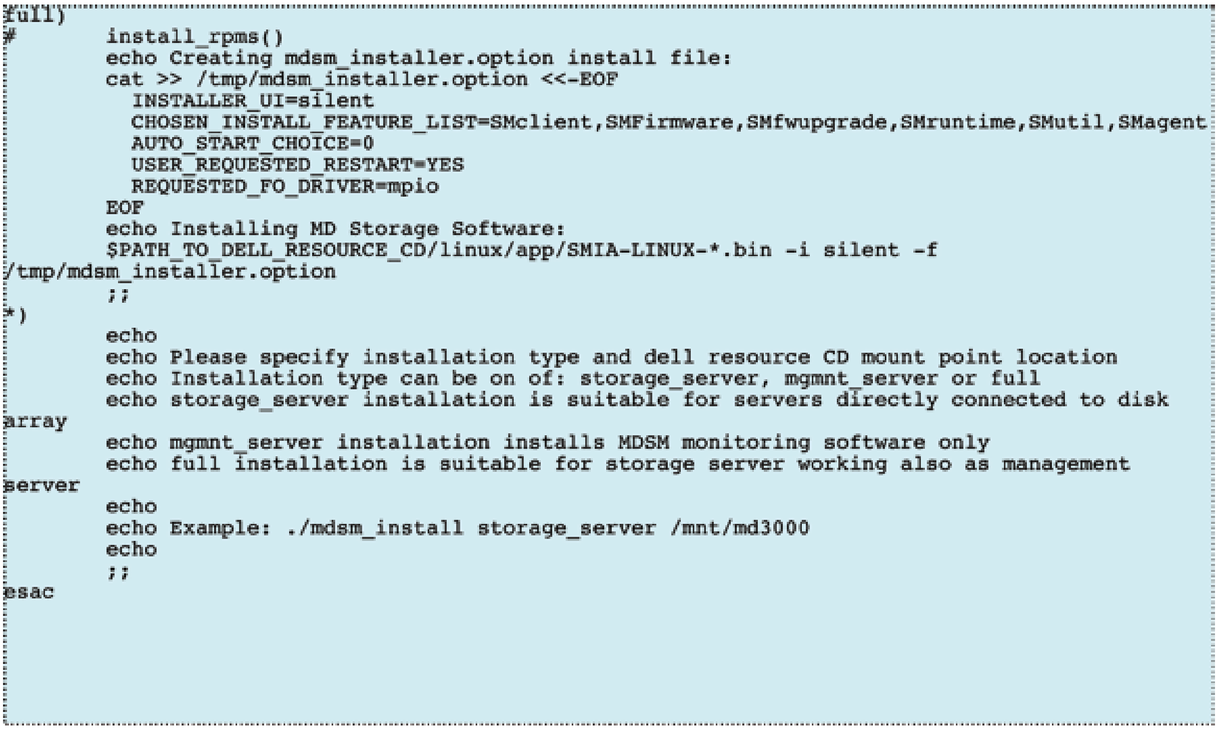
9.0 Apendix A

The following script installs software necessary to access, configure and manage the Dell™ PowerVault™ MD3000 disk arrays

`mdsm_install.sh`



```
#!/bin/bash

INSTALL_TYPE=$1
PATH_TO_DELL_RESOURCE_CD=$2
PATH_TO_LINUX_RPMS=${PATH_TO_DELL_RESOURCE_CD}/linux/RPMS
#==================================================================
install_rpms() # Installs all required RPMS from the Dell resource CD
#==================================================================
{
        cd ${PATH_TO_LINUX_RPMS}
        echo Installing prerequisites:
        echo Installing DKMS
        DRIVER_VERSION=`ls dkms* | cut -d - -f 2,3`
echo    rpm -Uvh dkms-${DRIVER_VERSION}
        cd ${PATH_TO_LINUX_RPMS}/rh5
        echo Installing MPTLINUX
        DRIVER_VERSION=`ls mptlinux* | cut -d - -f 2,3`
echo    rpm -Uvh mptlinux-${DRIVER_VERSION}
        echo Installing LINUXRDAC
        DRIVER_VERSION=`ls linuxrdac* | cut -d - -f 2,3`
echo    rpm -Uvh linuxrdac-${DRIVER_VERSION}
}
case $1 in
storage_server)
        install_rpms
        cat >> /tmp/mdsm_installer.option <<-EOF
          INSTALLER_UI=silent
          CHOSEN_INSTALL_FEATURE_LIST=SMruntime,SMutil,SMagent
          AUTO_START_CHOICE=0
          USER_REQUESTED_RESTART=YES
          REQUESTED_FO_DRIVER=mpio
        EOF
        echo Installing MD Storage Software:
echo    $PATH_TO_DELL_RESOURCE_CD/linux/app/SMIA-LINUX-*.bin -i silent -f
/tmp/mdsm_installer.option
        ;;
mgmnt_server)
        echo Creating mdsm_installer.option install file:
        cat >> /tmp/mdsm_installer.option <<-EOF
          INSTALLER_UI=silent
          CHOSEN_INSTALL_FEATURE_LIST=SMclient,SMFirmware,SMfwupgrade,SMruntime
          AUTO_START_CHOICE=0
          USER_REQUESTED_RESTART=YES
          REQUESTED_FO_DRIVER=mpio
        EOF
        echo Installing MD Storage Software:
        $PATH_TO_DELL_RESOURCE_CD/linux/app/SMIA-LINUX-*.bin -i silent -f
/tmp/mdsm_installer.option
        ;;
```

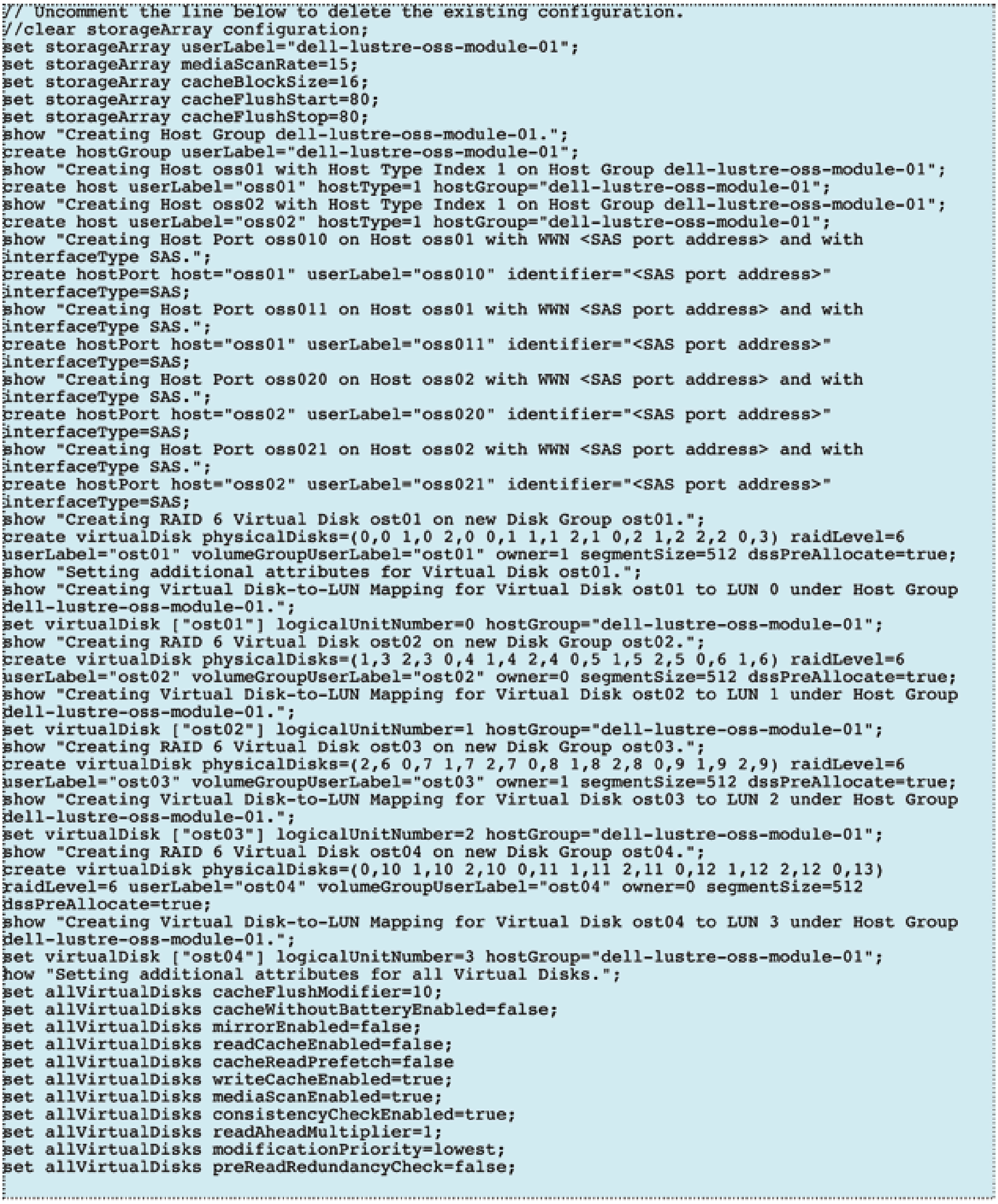


```
full)
#       install_rpms()
        echo Creating mdsm_installer.option install file:
        cat >> /tmp/mdsm_installer.option <<-EOF
          INSTALLER_UI=silent
          CHOSEN_INSTALL_FEATURE_LIST=SMclient,SMFirmware,SMfwupgrade,SMruntime,SMutil,SMagent
          AUTO_START_CHOICE=0
          USER_REQUESTED_RESTART=YES
          REQUESTED_FO_DRIVER=mpio
        EOF
        echo Installing MD Storage Software:
        $PATH_TO_DELL_RESOURCE_CD/linux/app/SMIA-LINUX-*.bin -i silent -f
/tmp/mdsm_installer.option
        ;;
*)
        echo
        echo Please specify installation type and dell resource CD mount point location
        echo Installation type can be on of: storage_server, mgmnt_server or full
        echo storage_server installation is suitable for servers directly connected to disk
array
        echo mgmnt_server installation installs MDSM monitoring software only
        echo full installation is suitable for storage server working also as management
server
        echo
        echo Example: ./mdsm_install storage_server /mnt/md3000
        echo
        ;;
esac
```

10.0 Apendix B

Dell™ PowerVault™ MD3000 configuration script for Dell Lustre™ HA-OSS module.

```
dell-ha-oss-module-01.cfg
```

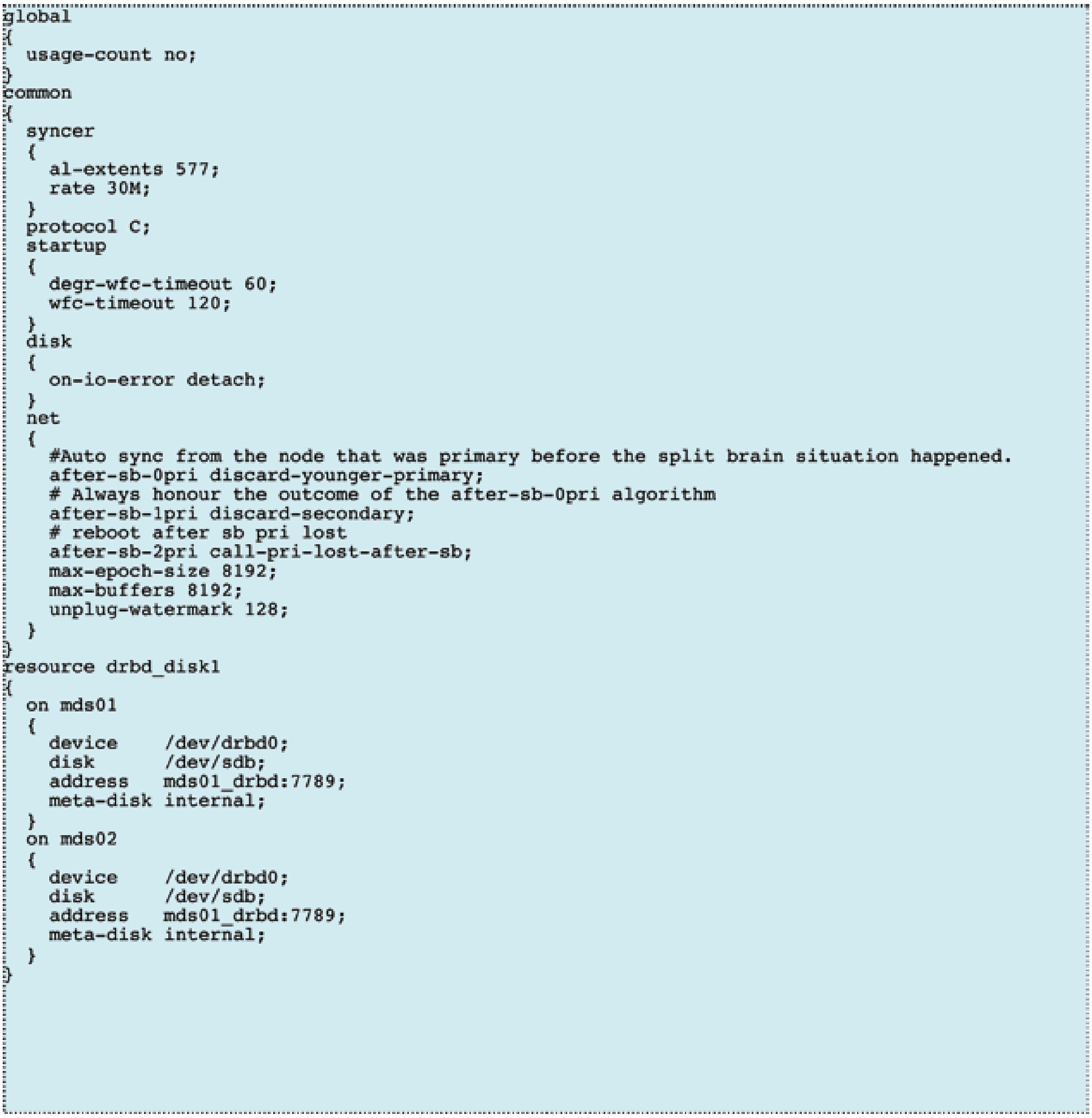


```
// Uncomment the line below to delete the existing configuration.
//clear storageArray configuration;
set storageArray userLabel="dell-lustre-oss-module-01";
set storageArray mediaScanRate=15;
set storageArray cacheBlockSize=16;
set storageArray cacheFlushStart=80;
set storageArray cacheFlushStop=80;
show "Creating Host Group dell-lustre-oss-module-01.";
create hostGroup userLabel="dell-lustre-oss-module-01";
show "Creating Host oss01 with Host Type Index 1 on Host Group dell-lustre-oss-module-01";
create host userLabel="oss01" hostType=1 hostGroup="dell-lustre-oss-module-01";
show "Creating Host oss02 with Host Type Index 1 on Host Group dell-lustre-oss-module-01";
create host userLabel="oss02" hostType=1 hostGroup="dell-lustre-oss-module-01";
show "Creating Host Port oss010 on Host oss01 with WWN <SAS port address> and with
interfaceType SAS.";
create hostPort host="oss01" userLabel="oss010" identifier="<SAS port address>"
interfaceType=SAS;
show "Creating Host Port oss011 on Host oss01 with WWN <SAS port address> and with
interfaceType SAS.";
create hostPort host="oss01" userLabel="oss011" identifier="<SAS port address>"
interfaceType=SAS;
show "Creating Host Port oss020 on Host oss02 with WWN <SAS port address> and with
interfaceType SAS.";
create hostPort host="oss02" userLabel="oss020" identifier="<SAS port address>"
interfaceType=SAS;
show "Creating Host Port oss021 on Host oss02 with WWN <SAS port address> and with
interfaceType SAS.";
create hostPort host="oss02" userLabel="oss021" identifier="<SAS port address>"
interfaceType=SAS;
show "Creating RAID 6 Virtual Disk ost01 on new Disk Group ost01.";
create virtualDisk physicalDisks=(0,0 1,0 2,0 0,1 1,1 2,1 0,2 1,2 2,2 0,3) raidLevel=6
userLabel="ost01" volumeGroupUserLabel="ost01" owner=1 segmentSize=512 dssPreAllocate=true;
show "Setting additional attributes for Virtual Disk ost01.";
show "Creating Virtual Disk-to-LUN Mapping for Virtual Disk ost01 to LUN 0 under Host Group
dell-lustre-oss-module-01.";
set virtualDisk ["ost01"] logicalUnitNumber=0 hostGroup="dell-lustre-oss-module-01";
show "Creating RAID 6 Virtual Disk ost02 on new Disk Group ost02.";
create virtualDisk physicalDisks=(1,3 2,3 0,4 1,4 2,4 0,5 1,5 2,5 0,6 1,6) raidLevel=6
userLabel="ost02" volumeGroupUserLabel="ost02" owner=0 segmentSize=512 dssPreAllocate=true;
show "Creating Virtual Disk-to-LUN Mapping for Virtual Disk ost02 to LUN 1 under Host Group
dell-lustre-oss-module-01.";
set virtualDisk ["ost02"] logicalUnitNumber=1 hostGroup="dell-lustre-oss-module-01";
show "Creating RAID 6 Virtual Disk ost03 on new Disk Group ost03.";
create virtualDisk physicalDisks=(2,6 0,7 1,7 2,7 0,8 1,8 2,8 0,9 1,9 2,9) raidLevel=6
userLabel="ost03" volumeGroupUserLabel="ost03" owner=1 segmentSize=512 dssPreAllocate=true;
show "Creating Virtual Disk-to-LUN Mapping for Virtual Disk ost03 to LUN 2 under Host Group
dell-lustre-oss-module-01.";
set virtualDisk ["ost03"] logicalUnitNumber=2 hostGroup="dell-lustre-oss-module-01";
show "Creating RAID 6 Virtual Disk ost04 on new Disk Group ost04.";
create virtualDisk physicalDisks=(0,10 1,10 2,10 0,11 1,11 2,11 0,12 1,12 2,12 0,13)
raidLevel=6 userLabel="ost04" volumeGroupUserLabel="ost04" owner=0 segmentSize=512
dssPreAllocate=true;
show "Creating Virtual Disk-to-LUN Mapping for Virtual Disk ost04 to LUN 3 under Host Group
dell-lustre-oss-module-01.";
set virtualDisk ["ost04"] logicalUnitNumber=3 hostGroup="dell-lustre-oss-module-01";
how "Setting additional attributes for all Virtual Disks.";
set allVirtualDisks cacheFlushModifier=10;
set allVirtualDisks cacheWithoutBatteryEnabled=false;
set allVirtualDisks mirrorEnabled=false;
set allVirtualDisks readCacheEnabled=false;
set allVirtualDisks cacheReadPrefetch=false
set allVirtualDisks writeCacheEnabled=true;
set allVirtualDisks mediaScanEnabled=true;
set allVirtualDisks consistencyCheckEnabled=true;
set allVirtualDisks readAheadMultiplier=1;
set allVirtualDisks modificationPriority=lowest;
set allVirtualDisks preReadRedundancyCheck=false;
```

11.0 Apendix C

DRBD configuration file for Dell Lustre HA-MDS module.

/etc/drbd.conf

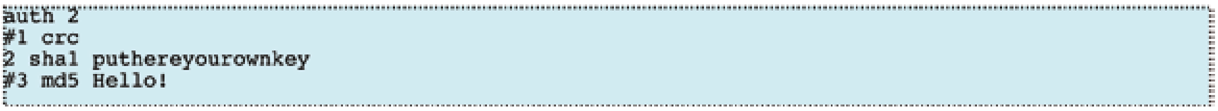


```
global
{
  usage-count no;
}
common
{
  syncer
  {
    al-extents 577;
    rate 30M;
  }
  protocol C;
  startup
  {
    degr-wfc-timeout 60;
    wfc-timeout 120;
  }
  disk
  {
    on-io-error detach;
  }
  net
  {
    #Auto sync from the node that was primary before the split brain situation happened.
    after-sb-0pri discard-younger-primary;
    # Always honour the outcome of the after-sb-0pri algorithm
    after-sb-1pri discard-secondary;
    # reboot after sb pri lost
    after-sb-2pri call-pri-lost-after-sb;
    max-epoch-size 8192;
    max-buffers 8192;
    unplug-watermark 128;
  }
}
resource drbd_disk1
{
  on mds01
  {
    device    /dev/drbd0;
    disk      /dev/sdb;
    address   mds01_drbd:7789;
    meta-disk internal;
  }
  on mds02
  {
    device    /dev/drbd0;
    disk      /dev/sdb;
    address   mds01_drbd:7789;
    meta-disk internal;
  }
}
```

12.0 Apendix D

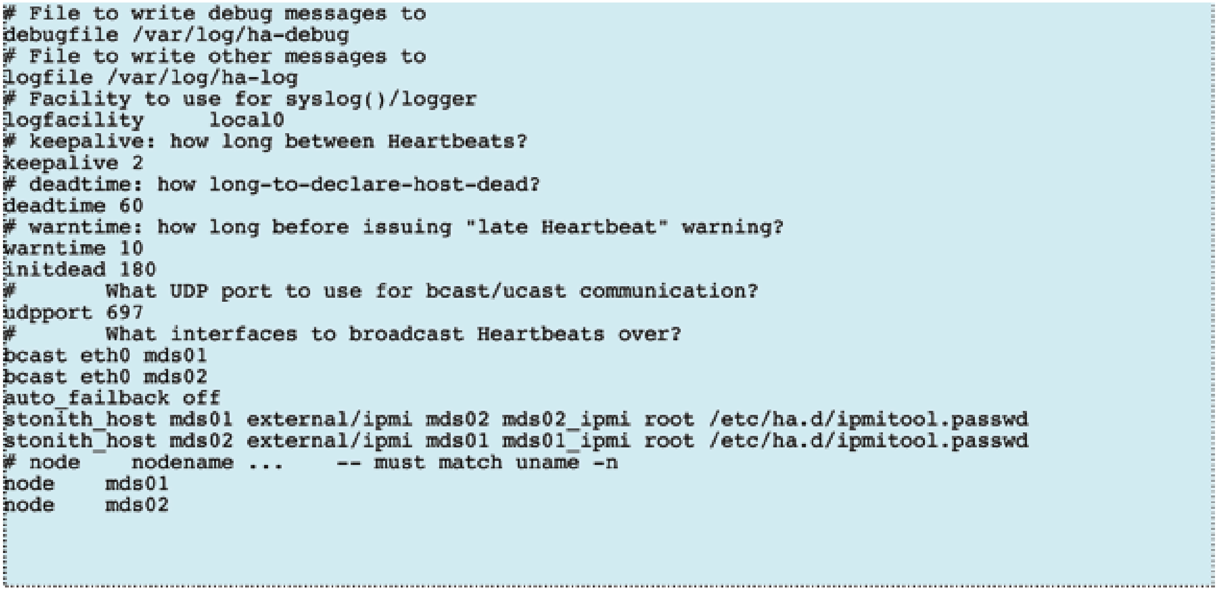
HA-Linux Heartbeat configuration

/etc/ha.d/authkeys



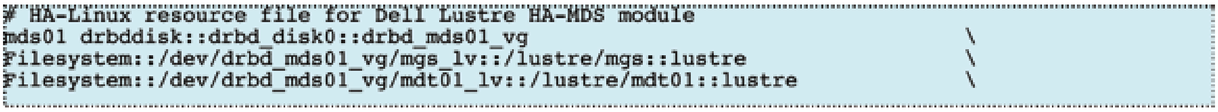
```
auth 2
#1 crc
2 sha1 puthereyourownkey
#3 md5 Hello!
```

/etc/ha.d/ha.cf



```
# File to write debug messages to
debugfile /var/log/ha-debug
# File to write other messages to
logfile /var/log/ha-log
# Facility to use for syslog()/logger
logfacility     local0
# keepalive: how long between Heartbeats?
keepalive 2
# deadtime: how long-to-declare-host-dead?
deadtime 60
# warntime: how long before issuing "late Heartbeat" warning?
warntime 10
initdead 180
#       What UDP port to use for bcast/ucast communication?
udpport 697
#       What interfaces to broadcast Heartbeats over?
bcast eth0 mds01
bcast eth0 mds02
auto_failback off
stonith_host mds01 external/ipmi mds02 mds02_ipmi root /etc/ha.d/ipmitool.passwd
stonith_host mds02 external/ipmi mds01 mds01_ipmi root /etc/ha.d/ipmitool.passwd
# node    nodename ...    -- must match uname -n
node    mds01
node    mds02
```

/etc/ha.d/ha.cf



```
# HA-Linux resource file for Dell Lustre HA-MDS module
mds01 drbddisk::drbd_disk0::drbd_mds01_vg                                \
Filesystem::/dev/drbd_mds01_vg/mgs_lv::/lustre/mgs::lustre               \
Filesystem::/dev/drbd_mds01_vg/mdt01_lv::/lustre/mdt01::lustre           \
```