DELL Compellent Best Practices: Storage Center with SUSE Linux Enterprise Server 11

A Dell Compellent Best Practices Guide

This document has been archived and will no longer be maintained or updated. For more information go to the Storage Solutions Technical Documents page on Dell TechCenter or contact support.
Adding Volumes post install ................................................................. 25
Adding Multipathed Volumes post install ............................................ 26
Resizing a volume ................................................................................. 27
# /etc/init.d/multipathd reload ............................................................ 27
iSCSI Timeout Values ........................................................................... 27
Mounting iSCSI targets at boot time .................................................... 29
10GB Ethernet and iSCSI ...................................................................... 29
Multipathing: ......................................................................................... 31
   Dell Compellent Storage Center Device Definition Specifics .............. 32
   path_checker.tur ................................................................................. 32
   Configuring Multipath Settings per specific device ......................... 32
   Multipathing Performance Considerations ......................................... 33
Working with attached volumes ............................................................ 34
   LVM ....................................................................................................... 34
Scalable file systems ............................................................................ 34
   EXT3/EXT4 .......................................................................................... 34
   XFS ........................................................................................................ 34
Volumes over 2TB .................................................................................. 36
Cloning Volumes ................................................................................... 36
   LVM ....................................................................................................... 36
Cloning Single Path Volumes ................................................................. 36
Cloning Multipathed Data Volumes ......................................................... 36
Cloning Multipathed Boot/Root Volumes ................................................. 37

Tables
Table 1. Conventions .............................................................................. 6
Table 2. Revision History ......................................................................... 6

Figures
Figure 1. File system creation examples .................................................. 8
Figure 2. Discovering a partition label using /etc/fstab .............................. 8
Figure 3. Discovering a UUID using tunefs ............................................. 9
Figure 4. Discovering a UUID using /dev/disk/by-uuid ............................. 9
Figure 5. Using a UUID for persistent disk mappings ................................................................. 9
Figure 6. Configuring GRUB to reference a label or UUID ............................................................ 10
Figure 7. Using lsscsi to get disk and volume information ............................................................. 10
Figure 8. Using scsi_id to get a volume WWID ............................................................................. 11
Figure 9. Correlating a WWID to a Serial Number in the Storage Center GUI .................................. 11
Figure 10. Viewing LUNs and targets in /proc/scsi/scsi ................................................................ 12
Figure 11. Viewing file system options in /proc/mounts ................................................................ 12
Figure 12. Viewing available disk device settings in /sys/block/sdX/queue ................................. 13
Figure 13. Viewing HBA WWN through /sys/class/fc_host/hostX ............................................... 13
Figure 14. Sample dmesg output showing assigned device names .............................................. 13
Figure 15. SLES 11 “Installation Settings” page .............................................................................. 14
Figure 16. Preparing Hard Disk (SLES 11 installer) ...................................................................... 15
Figure 17. Enabling Multipath (SLES 11 installer) ....................................................................... 16
Figure 18. Activating Multipath ..................................................................................................... 17
Figure 19. Multipath Activated ....................................................................................................... 18
Figure 20. Select fstab options ....................................................................................................... 19
Figure 21. Select Mount by Device Name ...................................................................................... 20
Figure 22. Confirming MBR location ............................................................................................. 21
Figure 23. Using the multipath command ...................................................................................... 21
Figure 24. Using dmesg to verify new volumes ............................................................................. 22
Figure 25. Checking expanded volume size ................................................................................... 22
Figure 26. Using modinfo to display available module parameters ........................................... 23
Figure 27. Example of an edited grub menu.lst file .................................................................. 25
Figure 28. Using iscsiadm to discover iSCSI routes ................................................................... 26
Figure 29. Using iscsiadm to list all available iSCSI routes .......................................................... 27
Figure 30. Simple multipath.conf file ........................................................................................... 32
Figure 31. Before multipath.conf .................................................................................................. 33
Figure 32. After multipath.conf ..................................................................................................... 33
Figure 33. Confirming a multipathed volume .................................................................................. 35
Figure 34. Creating an XFS file system on a multipathed volume ............................................. 35
Figure 35. Expanding an XFS file system on an expanded Storage Center volume .................... 35
Preface

This document is intended to provide an overview of specific information required for administrating storage on SUSE Linux Enterprise Server (SLES) 11 servers connected to a Dell Compellent Storage Center. This document is intended for administrators who have a confident understanding of Linux systems, specifically general tasks around managing disk partitions and file systems.

It is important to note that as is common in Linux, there are many ways to do what is covered in this document. This guide does not contain every possible way, and the way covered might not be the best for all situations. This documentation is brief and intended as a starting point of reference for Systems Administrators. Also note that this guide will focus almost exclusively on the command line.

Customer Support
For support, email Dell Compellent at support@compellent.com. Dell Compellent responds to emails during normal business hours.

General Syntax

Table 1. Conventions

<table>
<thead>
<tr>
<th>Item</th>
<th>Convention</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console or file trace. (Commands are shown with system prompt ‘#’ in console traces.)</td>
<td>Monospace Font</td>
</tr>
<tr>
<td>Other user input (file names and paths, field names, keys)</td>
<td>Monospace Font</td>
</tr>
<tr>
<td>Website addresses</td>
<td><a href="http://www.compellent.com">http://www.compellent.com</a></td>
</tr>
<tr>
<td>Email addresses</td>
<td><a href="mailto:name@dell.com">name@dell.com</a></td>
</tr>
</tbody>
</table>

Document Revision

Table 2. Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>August 26, 2011</td>
<td>A</td>
<td>Initial draft.</td>
</tr>
</tbody>
</table>

General

Patch and Service Pack levels
Due to known issues with various patch levels of SLES 11, it is strongly recommended that any SLES 11 install based on Dell Compellent storage be patched up through the most recent patch set of SLES 11 SP1. In particular, known issues in SLES 11 (pre-SP1) with several major HBA vendor drivers can cause erratic or unreliable behavior in a Storage Center (or any vendor SAN) based environment.
sysfs pseudo file system

The /sys file system is a synthetic file system within Linux that is meant to provide the administrator with information and configurable details about the running kernel. Usually each disk block device will have an entry under /sys/block/, and each Host Bus Adapter (HBA) will have an entry under /sys/class/scsi_host/hostX where X is the number (starting at 0) of the HBA in the server.

Disk Labels and UUIDs for Persistence

All modern Linux operating systems are capable of discovering multiple volumes from the Dell Compellent Storage Center. These new disks are given a device designation of /dev/sda, /dev/sdb, etc., depending upon how they are discovered by the Linux operating system via the various interfaces connecting the server to the storage.

The /dev/sdx names are used to designate the volumes in a myriad of commands and files, but most importantly, mount commands and the file system configuration file, /etc/fstab. In a static disk environment, the /dev/sdx name works well for entries in the /etc/fstab file.

However, in the dynamic environment of fiber channel or iSCSI connectivity, the Linux operating system lacks the ability to track these disk designations persistently through reboots and dynamic additions of new volumes via rescans of the storage subsystems.

There are multiple ways to ensure that disks are referenced by persistent names. This guide will cover using Disk Labels and Universally Unique Identifiers (UUIDs). Disk Labels or UUIDs should be used with all single path volumes.

Disk labels are also exceptionally useful when scripting Storage Center replay recovery. In the case where a view of a production volume is mapped to a backup server, it is not necessary to know what drive letter the view volume is assigned. Since the label is written to the file system, the label goes with the view and can easily be mounted or manipulated.

Note:

Disk labels will not work in a multipathed environment, and should not be used; multipath device names are persistent by default and will not change. Multipathing does support aliasing the multipath device names for human readable names.

Creating a New File system and Volume Label

This will format the volume destroying all data on that volume.

The mke2fs, mkfs.xfs and mkfs.reiserfs commands with the -L and -l LabelName added to the standard file system creation commands, erases any previous file system tables, destroys the pointers to existing files, creates a new file system and a new label on the disk.

The examples below create a new file system with the label FileShare for the various major file systems types.
Figure 1. File system creation examples

```
# mke2fs -j -L FileShare /dev/sdc
# mkfs -t ext3 -L FileShare /dev/sdc
# mkfs -t ext4 -L FileShare /dev/sdc
# mkfs.xfs -L FileShare /dev/sdc
# mkfs.reiserfs -l FileShare /dev/sdc
```

Adding or Changing the Volume Label of an Existing File system

To add or change the volume label without destroying data on the disk, use the following command. These commands can be performed while the file system is mounted.

```
# e2label /dev/sdb FileShare
```

It is also possible to set the file system label using the `-L` option of `tune2fs`.

```
# tune2fs -L FileShare /dev/sdb
```

Discover Existing Labels

To discover the label of an existing partition, the following simple command can be used.

```
# e2label /dev/sde FileShare
```

In this output, ‘FileShare’ is the volume label.

Figure 2. Discovering a partition label using `/etc/fstab`

```
LABEL=root     /ext3     defaults 1 1
LABEL=boot     /boot    ext3    defaults 1 2
LABEL=FileShare /share   ext3    defaults 1 2
```

The `LABEL=` syntax can be used in a variety of places including mount commands and GRUB boot loader configuration. Disk labels can also be referenced as a path for applications that do not recognize the `LABEL=` syntax. For example, the volume designated by the label `FileShare` can be accessed at the path ‘/dev/disk/by-label/FileShare’.

Swap Space

Swap space can also be labeled, however only at the time of creation. This isn’t a problem since no static data is stored in swap. To label an existing swap partition, follow these steps.

```
# swapoff /dev/sda1
# mkswap -L swapLabel /dev/sda1
# swapon LABEL=swapLabel
```

The new swap label can be used in `/etc/fstab` just like any volume label.

UUIDs

An alternative to disk labels is UUIDs. They are static and safe for use anywhere, however, their long length can make them awkward to work with. A UUID is assigned at file system creation.
Please note that it is Dell Compellent suggested best practice to not use UUIDs for the `fstab` entries responsible for mounting the boot, root and swap partitions with SLES 11.

A UUID for a specific file system can be discovered using `tune2fs -l` or `xfs_admin -u`.

**Figure 3. Discovering a UUID using `tune2fs`**

```
[root@local ~]# tune2fs -l /dev/sdc

File system volume name: dataVol
Last mounted on: <not available>
File system UUID: 5458d975-8f38-4702-9df2-46a64a638e07
```

Another simple way to discover the UUID of a device or partition is to do a long list on the `/dev/disk/by-uuid` directory.

**Figure 4. Discovering a UUID using `/dev/disk/by-uuid`**

```
[root@local ~]# ls -l /dev/disk/by-uuid
total 0
lrwxrwxrwx 1 root root 10 Sep 15 14:11 5458d975-8f38-4702-9df2-46a64a638e07 -> ../../sdc
```

From the output above, we discover that the UUID is ‘5458d975-8f38-4702-9df2-46a64a638e07’

Disk UUIDs can be used in `/etc/fstab` or any place were persistent mappings is required. Below is an example of its use in `/etc/fstab`.

**Figure 5. Using a UUID for persistent disk mappings**

```
/dev/VolGroup00/LogVol00 /ext3 defaults 1 1
LABEL=/boot /boot ext3 defaults 1 2
UUID=8284393c-18aa-46ff-9dc4-0357a5ef742d /var xfs defaults 0 0
```

As with disk labels, if an application requires an absolute path, the links created in `/dev/disk/by-uuid` should work in almost all situations.

**GRUB**

In addition to `/etc/fstab`, the GRUB configuration file `/boot/grub/grub.conf` should also be reconfigured to reference LABEL or UUID. The example below shows using a label for the root volume. UUID can be used the same way. Labels or UUIDs can also be used for “resume” if needed.
Unmapping Volumes

Linux systems store information on each volume presented to it. Even if a volume is unmapped on the Storage Center side, the Linux system will retain information about that volume until the next reboot. If the Linux system is presented with a volume from the same target using the same LUN again, it will reuse the old data on the volume. This can result in complications and misinformation.

Therefore, it is best practice to always delete the volume information on the Linux side after the volume has been unmapped. This will not delete any data stored on the volume itself, just the information about the volume stored by the OS (volume size, type, etc.).

- Determine the device name of the volume that will be unmapped. For example, /dev/sdc.
- Unmap the volume from the Storage Center GUI.
- Delete the volume information on the Linux OS with the following command replacing sdc with the correct device name.
  ```
echo 1 > /sys/block/sdc/device/delete
  ```

Useful Tools

Determining which Storage Center volume correlates to a specific Linux device can be tricky, but the following tools can be useful and many are included in the base install.

**lsscsi**

lsscsi is a tool that parses information from the /proc and /sys virtual file systems into a simple human readable output.

**Figure 7.** Using lsscsi to get disk and volume information

```bash
[root@local ~]# lsscsi
[0:0:0:0] disk    COMPELNT Compellent Vol 0504 -
[0:0:1:0] disk    COMPELNT Compellent Vol 0504 -
[0:0:2:0] disk    COMPELNT Compellent Vol 0504 /dev/sda
[0:0:3:0] disk    COMPELNT Compellent Vol 0504 -
[0:0:3:5] disk    COMPELNT Compellent Vol 0504 /dev/sdc
```

This output shows two drives from the Storage Center. It also shows that three front end ports are visible but are not presenting a LUN 0, as identified by the lines with a “-“ in the final column. This is the expected behavior. There are multiple options for lsscsi that provide even more detailed information.

The first column above shows the [host:channel:target:lun] designation for the volume. The first number corresponds to the local HBA hostX that the volume is mapped to. Channel is the SCSI bus address which will always be zero. The third number correlates to the Storage Center front end ports (targets). The last number is the LUN that the volume is mapped on.
*lspci*

*lspci* provides information regarding currently attached PCI devices. This can be useful for identifying not only what is currently visible to the server, but what variables/features have been associated with that device.

*scsi_id*

*scsi_id* can be used to report the World Wide Identifier (WWID) of a volume and is available in all base installations. This WWID can be matched to the volume serial number reported in the Storage Center GUI for accurate correlation.

**Figure 8. Using scsi_id to get a volume WWID**

```
[root@local ~]# /lib/udev/scsi_id -u -g /dev/sda
36000d31000006700000000000000668
```

**Figure 9. Correlating a WWID to a Serial Number in the Storage Center GUI**

The first part of the WWID is Storage Center’s unique ID, the middle part is made up of the controller number in hex, and the last part is the serial number of the volume. To ensure correct correlation in environments with multiple Dell Compellent Storage Centers, be sure to check the controller number as well.

The only situation where the two numbers would not correlate is if a Copy Migrate had been performed. In this case, a new serial number is assigned on the Storage Center side, but the old WWID must still be presented to the server so that the path to the server is not disrupted.

*PROC/scsi/scsi*

Viewing the contents of the *PROC/scsi/scsi* file can provide information about LUNs and targets on systems that do not have *lsscsi* installed. However, it is not easy to correlate them to a specific device.
Figure 10. Viewing LUNs and targets in /proc/scsi/scsi

```
[root@local ~]# cat /proc/scsi/scsi
Attached devices:
Host: scsi0 Channel: 00 Id: 02 Lun: 00
  Vendor: COMPELLNT Model: Compellent Vol   Rev: 0504
  Type:   Direct-Access                    ANSI  SCSI revision: 05
Host: scsi4 Channel: 00 Id: 00 Lun: 00
  Vendor: TEAC Model: DW-224E-V        Rev: C.CA
  Type:   CD-ROM                            ANSI  SCSI revision: 05
Host: scsi1 Channel: 00 Id: 05 Lun: 00
  Vendor: COMPELLNT Model: Compellent Vol   Rev: 0504
  Type:   Direct-Access                    ANSI  SCSI revision: 05
Host: scsi2 Channel: 00 Id: 02 Lun: 01
  Vendor: COMPELLNT Model: Compellent Vol   Rev: 0505
  Type:   Direct-Access                    ANSI  SCSI revision: 05
Host: scsi3 Channel: 00 Id: 02 Lun: 01
  Vendor: COMPELLNT Model: Compellent Vol   Rev: 0505
  Type:   Direct-Access                    ANSI  SCSI revision: 05
Host: scsi10 Channel: 00 Id: 00 Lun: 01
  Vendor: COMPELLNT Model: Compellent Vol   Rev: 0504
  Type:   Direct-Access                    ANSI  SCSI revision: 05
Host: scsi8 Channel: 00 Id: 00 Lun: 01
  Vendor: COMPELLNT Model: Compellent Vol   Rev: 0504
  Type:   Direct-Access                    ANSI  SCSI revision: 05
```

/proc/mounts

The /proc/mounts contains valuable information about the currently mounted file systems, as well as the flags used to mount the file system. This is especially useful because it exposes the default options, so options will appear that were not specifically called by Linux at boot time or by the mount command.

Figure 11. Viewing file system options in /proc/mounts

```
# cat /proc/mounts
rootfs / rootfs rw 0 0
udev /dev/tmpfs rw,relatime,nr_inodes=0,mode=755 0 0
tmpfs /dev/shm tmpfs rw,relatime 0 0
/dev/dm-3 / xfs rw,relatime,attr2,noquota 0 0
proc /proc proc rw,relatime 0 0
sysfs /sys sysfs rw,relatime 0 0
debugfs /sys/kernel/debug debugfs rw,relatime 0 0
devpts /dev/pts devpts rw,relatime,gid=5,mode=620,ptmxmode=000 0 0
/dev/mapper/36000d31000006700000000000000000668_part1 /boot ext2 rw,relatime,errors=continue,user_xattr 0 0
```
Each Linux disk device, as identified by the sdX identifier has a set of individually managed parameters and exposed settings. The majority of these are exposed through the `/sys/block/sdX/queue` file set.

**Figure 12.** Viewing available disk device settings in `/sys/block/sdX/queue`

```bash
# ls /sys/block/sdd/queue/

hw_sector_size  max_hw_sectors_kb  nr_requests  rotational
iosched          max_sectors_kb    optimal_io_size  rq_affinity
iostats          minimum_io_size   physical_block_size  scheduler
logical_block_size  nomerges    read_ahead_kb
```

Each Fibre Channel class HBA is associated with an entry in `/sys/class/fc_host/hostX` where X is the incremented number of visible HBAs. This file set exposes a wealth of information about the status and HBA visibility into the fabric.

**Figure 13.** Viewing HBA WWN through `/sys/class/fc_host/hostX`

```bash
# cat /sys/class/fc_host/host6/port_name

0x2100001b329a0261
```

dmesg

The output from `dmesg` can be useful for discovering what device name was assigned to a recently discovered volume.

**Figure 14.** Sample dmesg output showing assigned device names

```bash
SCSI device sdf: 587202560 512-byte hdwr sectors (300648 MB)
sdf: Write Protect is off
sdf: Mode Sense: 87 00 00 00
SCSI device sdf: drive cache: write through
SCSI device sdf: 587202560 512-byte hdwr sectors (300648 MB)
sdf: Write Protect is off
sdf: Mode Sense: 87 00 00 00
SCSI device sdf: drive cache: write through
sdf: unknown partition table
sd 0:0:3:15: Attached scsi disk sdf
sd 0:0:3:15: Attached scsi generic sg13 type 0
```

The above output is taken just after a host rescan and shows that a 300 GB volume has been discovered and assigned as `/dev/sdf`. 
Fibre Channel

For Fibre Channel connectivity instructions, please see the Storage Center Connectivity Guide. For more information about multipathing in SLES 11, please see the Multipathing section of this document.

Installing to a Multipathed Storage Center Volume

The SLES 11 installer is capable of detecting when a Storage Center volume is presented to the server over multiple paths, and automatically configures the Linux Device Mapper to present the paths as a single, multipathed disk device.

To install SLES 11 to a multipathed volume, choose to enter the partition manager menu when at the “Installation Settings” landing page.

Figure 15. SLES 11 “Installtion Settings” page

Choose the “Custom Partitioning” option at the “Preparing Hard Disk: Step 1”: 
At the Expert Partitioner landing page, in the bottom right hand of the window, is a button for “Configure” which has a sub option for “Configure Multipath”. Do this to enable device mapper multipathing.
Figure 17. Enabling Multipath (SLES 11 installer)

When asked by YaST to activate Multipath, choose “Yes”.

Figure 18. Activating Multipath
When adding partitions that will be mounted at boot time in a multipath, boot from SAN environment, it is considered Dell Compellent best practices to override the fstab defaults created by the SLES 11 installer. This stands in contrast to some established SLES Storage Best Practices, which suggest using “Mount by Device ID” (the default option) in the fstab configuration. For stability (particularly during kernel upgrades in SLES 11) and management reasons, Dell Compellent strongly suggests using “Mount by Device Name”. “Mount by Device Name” instructs the SLES 11 installer to create fstab and mkinitrd entries using the names of storage objects as they exist within the Linux Device Mapper, which provides for both LVM and Multipathing. Each partition created in the installer must be changed to “Mount by Device Name”.

Figure 19. Multipath Activated
Figure 20. Select fstab options
Once all of the needed partitions have been configured and accepted, the installer will revert back to “Installation settings” landing page. YaST will ask if the location of the MBR should be updated to comply with the changes, choose “Yes”.

Once the rest of the installer is complete and the server has rebooted into its new environment, the multipath command can verify that SLES has been installed to a multipath volume.

Figure 23. Using the multipath command

```
# multipath -l
mpatha (36000d3100000650000000000000019d33) dm-0 COMPELNT,Compellent Vol
  size=16G features='1 queue_if_no_path' hwhandler='0' wp=rw
  `-+ policy='round-robin 0' prio=1 status=active
   | `- 0:0:3:0  sda 8:0  active ready running
   `- 1:0:3:0  sdb 8:16 active ready running
```

Adding Volumes post install
Should more volumes need to be attached post install, the process of rescanning Fiber Channel HBA’s in SLESS 11 (Kernel version 2.6.32) is nearly identical to SLES 10. To force the HBAs to rescan for a new volume:
The rescan-scsi-bus.sh script provided with SLES 11 can also be used to rescan all HBAs automatically in a sequential fashion.

The discovery of new volumes can be seen in dmesg.

**Figure 24. Using dmesg to verify new volumes**

```bash
# dmesg | tail
scsi 0:0:1:4: Direct-Access COMPELNT Compellent Vol 0504 PQ: 0
ANSI: 5
sd 0:0:1:4: [sdj] 20971520 512-byte logical blocks: (10.7 GB/10.0 GiB)
sd 0:0:1:4: [sdj] 0-byte physical blocks
sd 0:0:1:4: [sdj] Write Protect is off
sd 0:0:1:4: [sdj] Mode Sense: 8f 00 00 08
sd 0:0:1:4: Attached scsi generic sg10 type 0
sd 0:0:1:4: [sdj] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
sdj: unknown partition table
sd 0:0:1:4: [sdj] Attached SCSI disk
```

Note that for adding volumes with multiple paths, each host adapter with a visible path must be rescanned, more detail can be found in the “Multipathing” section.

**Resizing a volume**

If a volume is expanded, the size increase will not be detected by the server until the volumes have been rescanned to detect the new size.

```bash
# echo 1 > /sys/block/sdj/device/rescan
```

This should cause dmesg to produce output similar to the following:

**Figure 25. Checking expanded volume size**

```bash
# dmesg | tail
...,
sd 0:0:1:4: [sdj] 31457280 512-byte logical blocks: (16.1 GB/15.0 GiB)
sd 0:0:1:4: [sdj] 0-byte physical blocks
sdj: detected capacity change from 10737418240 to 16106127360
```

It is important to note that multipath uses the smallest disk size it sees through all available paths, so all path devices must be rescanned, and multipathd must be reloaded before multipath will recognize the size increase.

**Module Settings**

Properly configuring the appropriate tuneables for a vendor specific HBA module is a critical step in achieving optimal host stability and performance. Included in this section are the required or consequentially valuable module settings for a Storage Center based environment. Because each major
vendor provides their own module and best practices, consulting the vendors own module
documentation should be considered a requirement. Along with the vendor’s documentation, the use of
the modinfo command can provide a brief description of available parameters.

Figure 26. Using modinfo to display available module parameters

```
# modinfo qla2xxx
...
parm: ql2xlogintimeout:Login timeout value in seconds. (int)
parm: qlport_down_retry:Maximum number of command retries to a port that returns a PORT-DOWN status. (int)
...
```

**Port Connectivity Timeout**

An important module parameter for the QLogic HBAs is qlport_down_retry, and for the Emulex HBAs it is lpfc_nodev_tmo. These settings determine how long the system waits to destroy a connection after losing connectivity at the port level. During a controller failover, the World Wide Name (WWN) for the active port will disappear from the fabric momentarily before returning on the reserve port on the other controller. This process can take anywhere from 5 to 60 seconds to fully propagate through a fabric. As a result, the default timeout of 30 seconds is insufficient and the value must be changed to 60. To view the current value:

QLogic:

```
# cat /sys/module/qla2xxx/parameters/qlport_down_retry
60
```

Emulex:

```
# cat /sys/class/scsi_host/host0/lpfc_nodev_tmo
60
```

Note that in a multipathing environment, a 60 second timeout can be significantly longer than is required for optimal availability, and the timeout value can actually be lowered to improve availability, more on this can be found in the Multipathing section of this document.

**Queue Depth**

Some workloads might benefit from tuning of the queue depth settings. Increasing the queue depth will generally increase the sustained throughput the host is able to achieve at the expense of observed latency. The goal of tuning the queue depth should be finding the setting that provides the most available throughput at an acceptable latency level. It’s also important to note that in environments where many servers are accessing the same Storage Center (or any other vendor SAN), that the Storage Center is has a finite ability queue incoming I/O, so while increasing a queue may improve a single servers workload performance, setting many servers to a high queue depth on an single Storage Center may cause a negative impact across all servers. Queue settings are governed by both the driver layer and card settings in its BIOS. The settings in the BIOS are authoritative in that settings in the BIOS will supersede settings made at the driver level. The relevant options for each driver are:
DELL Compellent Best Practices: Storage Center with SUSE Linux Enterprise Server 11

**QLogic:** ql2xmaxqdepth

**Emulex:** lpfc_lun_queue_depth and lpfc_hba_queue_depth

**Extended Logging**
Instructing the driver to verbosely describe observed topology changes or artifacts can be extremely useful when troubleshooting Fabric connectivity issues. To enable verbose or extended logging, the following options can be specified:

**QLogic:** ql2xextended_error_logging

**Emulex:** lpfc_log_verbose

**Applying Module Settings in SLES 11**
For servers that do not require a HBA module to be active in order to support the root file system or any other active file system I.E non boot from SAN servers, module settings can be changed by editing the `/etc/modprobe.conf.local` or `/etc/modprobe.d/DRIVERNAME` files with the appropriate settings, and unloading and reloading the module with the `rmmod` and `modprobe` commands.

For servers deployed in a boot from SAN environment, the settings for the module are configured as part of the ramdisk used to boot the system from the bootloader. This means that some driver settings, such as `lpfc_lun_queue_depth` remain static until reboots. There are two generally supported methods for configuring these settings in a SLES 11 server.

The first is rebuilding the ramdisk with the newly configured settings. The `mkinitrd` command is the primary tool used for rebuilding the ramdisk, and it will incorporate settings specified in the `modprobe.conf.local` or `modprobe.d/DRIVERNAME` files. The advantage of configuring module settings in this method is that it leverages the established configuration files and maintenance processes, it is likely to be best supported through routine maintenance, such as kernel upgrades or distribution upgrades.

The other option is to alter the grub menu boot options, where module settings can be specified before the kernel is loaded and the boot process continues. This method can be advantageous in that it does not require the ramdisk to be rebuilt to accommodate a simple module settings change, when rebuilding the ramdisk could be considered are sensitive or risky activity. The options can be specified by either editing the `/boot/grub/menu.lst` file directly, or they can be added via the `yast2 wizard under System -> Boot Loader -> Edit.`
Software iSCSI

Disclaimer: While iSCSI is considered to be a mature technology that allows organizations to economically scale into the world of enterprise storage, it has grown in complexity at both the hardware and software layers. The scope of this documented is limited to the default Linux iSCSI software initiator (open-iscsi). For more advanced implementations (for example leveraging iSCSI HBAs, or drivers that make use of iSCSI offload engines) please consult the associated vendor’s documentation and support services.

For instructions on setting up an iSCSI network topology, please consult the Storage Center Connectivity Guide.

Adding Volumes post install

In order to begin the process of using the open-iSCSI initiator, the open-iSCSI daemon must be started. The open-iSCSI daemon is controlled by the /etc/init.d/open-iscsi control script, which along with managing the open-iSCSI daemon, also manages its required module dependencies. The yast command menu can also be used to persistently enable or disable the iSCSI service.
Once the service has been started, the `iscsiadm` command can be used to target an iSCSI IP address on the Storage Center:

```
iscsiadm -m discovery -t sendtargets -p 10.10.64.1
```

This should return iSCSI name information for the discovered Storage Center, similar to:

```
10.10.64.1:3260,0 iqn.2002-03.com.compellent:5000d31000036011
```

The following will allow Storage Center to identify the server initiator:

```
iscsiadm -m node --login
```

The HBA should now be identifiable in Storage Center. It should have the same iSCSI name as is configured in “/etc/iscsi/initiatorname.iscsi”. Once the iSCSI HBA has been mapped to the appropriate server and volume objects in Storage Center, use `iscsiadm` to rescan the SAN:

```
iscsiadm -m node -R
```

The disk should now be mapped to a `/dev/sdX` name, and can be used as if it were a local disk.

**Adding Multipathed Volumes post install**

Use the `iscsiadm` command to target an iSCSI IP address on the Storage Center. Run the command once for each route/destination available to the SAN. Each command should look like and have a return value of:

```
# iscsiadm -m discovery -t sendtargets -p 10.10.64.1
10.10.64.1:3260,0 iqn.2002-03.com.compellent:5000d31000036011
```

Storage Center version 5 added iSCSI Control Ports, which simplify multipathed iSCSI discovery. If the Storage Center has been configured with control ports, targeting the control port will discover all available routes to the Storage Center on that control node.

**Figure 28. Using iscsiadm to discover iSCSI routes**

```
# iscsiadm -m discovery -t sendtargets -p 172.16.27.1
172.16.27.1:3260,0 iqn.2002-03.com.compellent:5000d31000038e4d
172.16.27.1:3260,0 iqn.2002-03.com.compellent:5000d31000038e4e
172.16.27.1:3260,0 iqn.2002-03.com.compellent:5000d31000038e4f
172.16.27.1:3260,0 iqn.2002-03.com.compellent:5000d31000038e50
# iscsiadm -m discovery -t sendtargets -p 10.10.1
10.10.1:3260,0 iqn.2002-03.com.compellent:5000d31000038e42
10.10.1:3260,0 iqn.2002-03.com.compellent:5000d31000038e43
10.10.1:3260,0 iqn.2002-03.com.compellent:5000d31000038e44
10.10.1:3260,0 iqn.2002-03.com.compellent:5000d31000038e45
```

When all the routes have been discovered, each route should be listed as a return value from the command:
DELL Compellent Best Practices: Storage Center with SUSE Linux Enterprise Server 11

Figure 29. Using iscsiadm to list all available iSCSI routes

```
# iscsiadm -m node
10.10.10.1:3260,0 iqn.2002-03.com.compellent:5000d3100038e43
10.10.10.1:3260,0 iqn.2002-03.com.compellent:5000d3100038e45
172.16.27.1:3260,0 iqn.2002-03.com.compellent:5000d3100038e4f
10.10.10.1:3260,0 iqn.2002-03.com.compellent:5000d3100038e42
172.16.27.1:3260,0 iqn.2002-03.com.compellent:5000d3100038e4e
10.10.10.1:3260,0 iqn.2002-03.com.compellent:5000d3100038e44
172.16.27.1:3260,0 iqn.2002-03.com.compellent:5000d3100038e50
172.16.27.1:3260,0 iqn.2002-03.com.compellent:5000d3100038e4d
```

The following will allow Storage Center to indentify the server initiator:

```
iscsiadm -m node --login
```

The HBA should now be identifiable in Storage Center. It should have the same iSCSI name as is configured in “/etc/iscsi/initiatorname.iscsi”. Once the iSCSI HBA has been mapped to the appropriate server and volume objects in Storage Center, use iscsiadm to rescan the SAN:

```
iscsiadm -m node -R
```

### Resizing a volume

Once the volume has been resized on the Storage Center, the iscsiadm command can easily rescan the size of connected disks:

```
# iscsiadm -m node -R
```

Once the iSCSI subsystem has been resized, the multipath daemon must also be reloaded to detect and apply the new changes to the Multipath device:

```
# /etc/init.d/multipathd reload
```

### iSCSI Timeout Values

In the event of a failure in the SAN environment that causes a head unit fail over, the iSCSI daemon needs to be configured to wait for a sufficient window as to allow the failure recovery to occur. A fail over between Storage Center controllers takes approximately 30 seconds to complete, so it is considered best practice to configure the iSCSI initiator to queue for 60 seconds before failing.

When using iSCSI in a Multipathed environment, however, you can configure the iSCSI daemon to fail a path very quickly. It will then pass outstanding I/O back to the multipathing layer. If dm-multipath still has an available route, the I/O will be resubmitted to the live route. If all available routes are down, dm-multipath will queue I/O until a route becomes available. This allows an environment to sustain failures at the network and storage levels.

For the iscsi daemon, the following configuration settings directly affect iSCSI connection timeouts:

To control how often a NOP-Out request is sent to each target, the following value can be set:

```
node.conn[0].timeo.noop_out_interval = X
```

Where X is in seconds and the default is 10 seconds.
DELL Compellent Best Practices: Storage Center with SUSE Linux Enterprise Server 11

To control the time out for the NOP-Out, the `noop_out_timeout` value can be used:

```bash
node.conn[0].timeo.noop_out_timeout = X
```

Again X is in seconds and the default is 5 seconds.

The next iSCSI timer that will need to be tweaked is:

```bash
node.session.timeo.replacement_timeout = X
```

Again X is in seconds.

`replacement_timeout` will control how long to wait for session re-establishment before failing pending SCSI commands and commands that are being operated on by the SCSI layer’s error handler up to a higher level like multipath, or to an application if multipath is not being used.

Remember, from the NOP-Out section that if a network problem is detected, the running commands are failed immediately. There is one exception to this and that is when the SCSI layer’s error handler is running. To check if the SCSI Error Handler is running, `iscsiadm` can be run as:

```bash
iscsiadm -m session -P 3
```

You will then see:

**Host Number: X State: Recovery**

When the SCSI Error Handler is running, commands will not be failed until `node.session.timeo.replacement_timeout` seconds.

To modify the timer that starts the SCSI Error Handler, you can either write directly to the device’s `sysfs` file:

```bash
echo X > /sys/block/sdX/device/timeout
```

where X is in seconds, or on most distributions, you can modify the `udev` rule.

To modify the `udev` rule, open `/etc/udev/rules.d/60-raw.rules`, and add the following lines:

```bash
ACTION=="add", SUBSYSTEM=="scsi", SYSFS{type}=="0|7|14", \ 
RUN="/bin/sh -c 'echo 60 > /sys$$DEVPATH/timeout'"
```

For `multipath.conf`:

By default, the new multipath should be created using the defaults built into SLES 11, for more information, please see the Multipath section of this document.

*Tip:*  
When modifying iSCSI timeout values, it is important to test and verify as many failure scenarios as possible before entering a system into production. Timeout and other connection settings are statically created during the discovery step and written to configuration files in `/var/lib/open-iscsi/*`. 

Page 28
When troubleshooting complex iSCSI environments, enabling debug level verbosity from the `iscsi` daemon can provide information that is significantly valuable. This can be done by changing the following line from:

```
ARGS="-c $CONFIG_FILE -n"
```

to:

```
ARGS="-d 3 -c $CONFIG_FILE -n"
```

Where `"-d 3"` enables the debug flag and 3 represents the level of debug verbosity. Note that the service must be restarted for any changes to take effect.

**Mounting iSCSI targets at boot time**

If you need to mount an iSCSI target at boot time, add the `"_netdev"` flag to `/etc/fstab` to ensure it is mounted after network dependencies. Example:

```
/dev/sdb /mnt/iscsi ext3 _netdev 0 0
```

**10GB Ethernet and iSCSI**

While 10GB Ethernet is becoming widely adopted, it is still a relatively new technology, and many layers in a network stack may not be configured for best performance with 10GB Ethernet. An example of this is the default network settings for the Linux kernel, which are intended for optimal performance over slower link speeds. A default stock installation of Linux will likely not be able to reach the full performance of a 10GB link, and considerable thought and planning can be required to get the most out of a 10GB network. Below are suggestions regarding the tuning and configuring of a SLES 11 server to make the best use of a 10GB link.

A network interface uses CPU interrupts to request time from the CPU. On a symmetrical multi-processing (SMP) system, the performance of the CPU handling the interruptions is vital to overall performance on the network interface. Generally each device has at least one line to communicate interrupts with each CPU, and a round robin mechanism is used to choose which CPU will handle an interrupt request. The `/proc/interrupt` pseudo file can be used to observe the behavior of interrupts on the server. Contrary to expectations, round robin IRQ does not lead to optimal performance. When interrupts round robin across CPUs, the IRQ handler has to be brought fresh into that processor’s cache, whereas if interrupts from the same device arrive at the same CPU for each interrupt, the IRQ handler is likely to be found in cache. It is because of this that IRQ manipulation/balancing mechanisms should be disabled for 10GB Ethernet capable servers. It is also imperative that TX and RX affinities match, so that TX requests are run on the same CPU as RX, once again this is to avoid cache misses.

IRQ balancing can also be disabled for individual CPUs in `/etc/sysconfig/irqbalance`. Binding of an interrupt line can be done through `/proc/irq/$IRQ#/smp_affinity`, and editing of this file can be done online.

It is important to note that these changes will affect other devices on the server, and that the network performance benefits garnered by these changes may not outweigh the penalties observed on other devices.
Most 10GB Ethernet NICs are also capable of coalescing Ethernet frames. By coalescing requests, they are bundled up before an interrupt is sent, and then passed after the interrupt has been acknowledged. This has the benefit of reducing the IRQs created by activity, but can also create artificial latency. Control of NIC coalescing is a function of the driver, and can likely be controlled with `ethtool` depending on the NIC vendor. Some NICs may also support “New API” (NAPI), which is a relatively new feature that removes interruptions when under load, and instead packets are passed on a polling interval, which means minimal latency impact while the interface is under load. NAPI is included in the Linux 2.6 kernel, and most vendor drivers also support NAPI.

TCP Offload was originally implemented to overcome limitations in loss prone or unreliable networks. TCP Offload has proven to be useful in reducing load on CPUs by managing repetitive tasks on the NIC instead of passing them up to the CPU, and can have significant performance impact on 10GB capable servers. RX/TX TCP Offload for most NIC/drivers can be managed using `ethtool`.

As with TCP Offload, RX/TX Checksum processing can be offloaded to most modern NICs, further reducing the stress of TCP traffic on CPUs. It is also important to note the RX checksums must be offloaded to the NIC to also take advantage of Large Receive Offloading (LRO). The benefits of TX/RX offloading are highlighted when using Jumbo Frames, it is estimated that with an MTU of 9000, offloading can offer a CPU savings of 15%.

TCP was originally designed to only support window fields 16 bits wide, which has become a major bottleneck in higher speed networks. This was overcome with RFC 1323 which provided a mechanism for scaling the TCP window as needed, this can be enabled with `sysctl net.ipv4.tcp_window_scaling`. It is worth noting that RFC 1323 also introduced the ability to remove the TCP Timestamp from the TCP header. While this can have positive performance impacts at the networking layer, it has the potential to severely break open-iscsi, and because of this it is strongly urged not to alter TCP Timestamp behavior.

TCP Selective Acknowledgement (SACK) is a feature of TCP/IP introduced with RFC 2018 that sends TCP ACKs for delivered segments, and the sender only needs to retransmit lost segments. As most 10GB networks will be lossless networks, disabling TCP SACK is likely to improve overall throughput. This can be done by setting `net.ipv4.tcp_sack` to 0.

As of Linux kernel; version 2.6.17 (SLES 11 runs 2.6.32), the kernel can automatically adjust its own network memory buffers as needed. Confirming this can be done with `sysctl net.ipv4.tcp_moderate_rcvbuf`. However while the kernel can adjust the amount of memory consumed for network buffers, the lower limit, initial size and upper limit do need to be adjusted in a 10Gb environment. Read buffers can be adjusted with `sysctl net.ipv4.tcp_rmem x y z` where x y z represent the lower limit, initial size, and upper limit in bytes respectively. Write buffers are managed nearly identically to read buffers, and are set under the `sysctl` key `net.ipv4.tcp_wmem`. Overall network buffers limits are set by the `sysctl` key `net.ipv4.tcp_mem`. It is important to note that `net.ipv4.tcp_mem` is in system pages where as the read/write buffers are in bytes. For most x86/x64 systems this will be 4k and on PPC systems this should be 64k. This can be confirmed with `getconf PAGESIZE`. As a best practice, setting `net.ipv4.tcp_mem` to at least twice the delay bandwidth is a good place to start. This is a good example of the performance capacity requirements needed to fully take advantage of 10GB Ethernet.
Along with `net.ipv4` buffer settings, the `net.core` settings must be similarly adjusted:

- `net.core.rmem_max` - max size of rx socket buffer
- `net.core.wmem_max` - max size of tx socket buffer
- `net.core.rmem_default` - default rx size of socket buffer
- `net.core.wmem_default` - default tx size of socket buffer
- `net.core.optmem_max` - maximum amount of option memory buffers
- `net.core.netdev_max_backlog` - how many unprocessed rx packets before kernel starts to drop them

For a complete list of settings that can be managed by `sysctl`, please consult the output of the `sysctl -a` command. Changes made by `sysctl` are not consistent across reboots, so persistent changes must be written to `/etc/sysctl.conf.`

TCP Congestion Protocol: As lossy WAN TCP/IP networks began to carry more and more data, algorithms were built to allow the kernel to dynamically adjust congestion behaviors as needed. In a stable, lossless high speed network, these mechanisms are no longer relevant. Altering the algorithm introduces another variable to be managed, and for that reason it is suggested to leave it at the default "cubic" algorithm.

With PCI-E, interrupts are managed in line rather than out of band through a feature called Message Signal Interrupts (MSI). MSI offers numerous improvements over legacy PCI implementations and can provide overall lower latency and CPU interruptions. An enhancement to MSI has also been implemented, which also improves performance. The `lspci` command can be used to verify if MSI or MSI-X is enabled per device.

For systems with both Intel NICs and Intel Dual or Quad Core CPUs, the `ioatdma` kernel module can assist in more efficiently moving networking tasks between CPUs. It is important to note that while the module can be loaded online, offlining it while powered on is not suggested, as the TCP stack will hold static references in and out of the `ioatdma` module. To load the module online:

```
# modprobe ioatdma
```

**Multipathing:**

The Linux Device-Mapper is a flexible yet generic framework for providing interconnected virtual block devices on top of physical storage devices. One of its most powerful feature sets is the ability to detect, create and monitor block devices with multiple paths to a backing storage device. This feature is referred to as Device-Mapper Multipath, but is frequently abbreviated to names such as `dm-multipath`, `dm_mpath` or even just `multipath`.

Device-Mapper Multipath is largely storage protocol agnostic, I.E Device-Mapper Multipath can manage Directly Attached, Fibre Channel, Fibre Channel over Ethernet and iSCSI storage. Aside from the
specifics in configuring each protocol, the concepts of Device-Mapper Multipath can be applied to any of the previously mentioned Storage Platforms.

**Dell Compellent Storage Center Device Definition Specifics**

Because the Dell Compellent device definition is included by default with the kernel version used by SLES 11, relatively little is required to configure simple multipath environments. After a Storage Center volume has detected by the SCSI subsystem, SLES Multipath will automatically create a multipath device, based on the SCSI ID of the volume, and automatically apply the defaults for a multipathed Compellent volume. The notable default options are:

```bash
path_checker tur
    This specifies that the SCSI “Test Unit Ready” command be used to monitor end device health.

no_path_retry queue
    This specifies that should multipath be in a state where no paths to a lun are in a healthy state, that I/O be queued until an available path is visible. This setting is so successful in allowing a Linux platform to survive Storage Center failover events, that it can trivialize the concern.
```

**Configuring Multipath Settings per specific device**

Multipath creates a device per observed unique SCSI ID. Each device that it creates can have a certain subset of settings configured, which are isolated from the settings of any other device. These devices are managed through the `/etc/multipath.conf` configuration file, which must be created after a fresh installation. An example of a configured device can be as simple as assigning a user readable alias, which configures a device to use an easy to read name instead of the long SCSI ID string.

**Figure 30. Simple multipath.conf file**

```bash
multipaths {
    multipath {
        wwid 36000d310000069000000000000000f28
        alias mpathiscsimysql
    }
}
```

After reloading the Multipath service, a multipath device that previously appeared as:
DELL Compellent Best Practices: Storage Center with SUSE Linux Enterprise Server 11

Figure 31. Before multipath.conf

```
# multipath -ll
36000d3100000690000000000000000f28 dm-4 COMPELNT,Compellent Vol
size=20G features='1 queue_if_no_path' hwhandler='0' wp=rw
`+-+- policy='round-robin 0' prio=1 status=active
  |- 11:0:0:2 sdc 8:32 active ready running
  `- 9:0:0:2 sde 8:64 active ready running
```

Now appears as:

Figure 32. After multipath.conf

```
# multipath -ll
mpathiscsimysql (36000d3100000690000000000000000f28) dm-4
COMPELNT,Compellent Vol
size=20G features='1 queue_if_no_path' hwhandler='0' wp=rw
`+-+- policy='round-robin 0' prio=1 status=enabled
  |- 11:0:0:2 sdc 8:32 active ready running
  `- 9:0:0:2 sde 8:64 active ready running
```

It should be noted that because this will change the name of the multipath device, and fstab entries or other places that directly reference the device name will need to be updated.

**Multipathing Performance Considerations**

Multipathing is a complex layer of abstraction above, and introduces performance concerns and characteristics that might not always be immediately intuitive. When trying to identify the best settings for performance in an environment, it is strongly encouraged that optimal performance be identified before introducing multipathing into the equation. Once an optimal non-multipath configuration has been identified, multipathing can be configured and further performance tuning performed.

“More Lanes on the Highway”, in reference to multiple volumes or storage protocol paths, is a common expression when considering Linux performance with Storage Center. In order to maximize multipathing’s performance, additional volumes provide additional logical queues which offer the best chance for Multipathing to distribute I/O down as many paths as possible.

`rr min io` is a setting which can be specified within the multipath.conf which designates how many I/O operations are submitted per path before moving to the next available path. Finding the correct value is very environment specific, but tuning should be performed with the intention of saturating available paths without impacting latency from the applications perspective.
It should also be noted that for applications with CPU bottlenecks, route-robin multipathing can actually cause performance regressions, as the CPU cycles needed to manage the context switches between paths can negatively impact application performance. In environments such as this, using failover multipathing instead of round-robin multipathing is suggested.

Working with attached volumes

**LVM**
LVM acts as an abstraction layer that allows for more complex disk management than was previously possible in Linux, and using LVM is an accepted best practice in many server environments. However in environments based on Storage Centers, LVM is largely duplicating feature functionality already present in the Storage Centers themselves, and is therefore an unnecessary layer of abstraction that adds performance overhead and administrative complexity. It is Dell Compellent best practice to not use LVM on any volume that can be presented as a whole disk to the server.

**Scalable file systems**
SLES 11 provides a wide variety of file systems, each with its own strengths and use-cases. Unless strictly necessary for an application or other requirement, it is suggested that EXT3, EXT4 and XFS be considered the best practices file systems.

**EXT3/EXT4**
The EXT3 and EXT4 file systems are the main stays of Linux file systems. For this reason, it is encouraged that EXT3 or EXT4 be used for common purpose file sets (such as root mountpoint file sets), or file sets that need to be easily transportable between various Linux platforms. Where performance is a concern, it is strongly suggested to use EXT4.

Creating an ext4 file system on top of a whole block device (as opposed to partitioning a block device and putting a file system on each partition) can be done with:

```
# mkfs.ext4 -L compellentvol01 /dev/sdj
```

Resizing an ext4 file system can be done mounted and online. Once the Storage Center volume has been expanded and the disk rescanned, the following will expand the file system to occupy the entire expanded volume:

```
# resize2fs /dev/sdj
```

**XFS**
XFS is a stable, high performance file system specifically designed to work with high performance storage and massive file sets. For truly high performance file sets requiring a file system, XFS is the suggested file system. To create a XFS file system across a multipathed volume:
Figure 33. Confiming a multipathed volume

```
# multipath -v2 -l
mpathd (360000d310000065000000000000000004d) dm-8 COMPELNT,Compellent Vol
size=125G features='1 queue_if_no_path' hwhandler='0' wp=rw
`+-' policy='round-robin 0' prio=1 status=active
 | - 0:0:1:3 sdg 8:96 active ready running
 `-- 1:0:1:3 sdh 8:112 active ready running
```

This shows that device `mpathd' is a multipathed volume, on top of which a XFS file system can be created.

Figure 34. Creating an XFS file system on a multipathed volume

```
# mkfs.xfs -L compellvol02 /dev/mapper/mpathd
log stripe unit (2097152 bytes) is too large (maximum is 256KiB)
log stripe unit adjusted to 32KiB
```

```
meta-data=/dev/mapper/mpathd isize=256 agcount=17,
agsize=2047488 blk
sectsz=512 attr=2
bsize=4096 blocks=32768000,
imaxpct=25
sunit=512 swidth=512 blks
naming=version 2 ascii-ci=0
log=internal log bsize=4096 blocks=16000, version=2
sectsz=512 sunit=8 blks, lazy-
count=1
realtime=none extsz=4096 blocks=0, rtextents=0
```

To expand a XFS file system on top of an expanded Storage Center volume (that has already been rescanned):

Figure 35. Expanding an XFS file system on an expanded Storage Center volume

```
# xfs_growfs /dev/mapper/mpathd
```

```
meta-data=/dev/mapper/mpathd isize=256 agcount=17,
agsize=2047488 blk
sectsz=512 attr=2
bsize=4096 blocks=32768000,
imaxpct=25
sunit=512 swidth=512 blks
naming=version 2 ascii-ci=0
log=internal bsize=4096 blocks=16000, version=2
sectsz=512 sunit=8 blks, lazy-
count=1
realtime=none extsz=4096 blocks=0, rtextents=0
data blocks changed from 32768000 to 39321600
```
Volumes over 2TB
Historically, partitions have been created using the MBR scheme, which has a capacity limit of 2TB. This limit can complicate disk layouts in larger sized systems. The two most common workarounds are:

A. Keeping the boot Volume below 2TB. Given that the OS file structure is likely to be only a handful of GBs, a small volume can be created to accommodate the system files. Then a larger 2TB+ volume can be created, and a file system laid out across that entire volume.

B. The GPT partition scheme is a part of the EFI standard which overcomes many of the limits of MBR, including the 2TB limit, and is supported by most major OSs, including Linux. It is important to note that while SLES 11 and GRUB may support GPT partitions, not all classic utilities do. A prominent example of this is fdisk.

To create a GPT partition:

After the volume has been created and mapped, rescan for the new volume. Then follow the example below to create a new volume. In this case, the volume is 5TB in size and is represented by /dev/sdb.

Invoke the parted command:

# parted /dev/sdb

Run the following two commands inside of parted replacing 5000G with the volume size needed:

> mklabel gpt
> mkpart primary 0 5000G

Finally format and label the new partition:

# mkfs.ext4 -L VolumeName /dev/sdb1

Cloning Volumes

LVM
As was detailed earlier in this document, it is considered best practice to not use LVM with a Dell Compellent Storage Center, as the Storage Center offers the majority of the LVM feature set at the SAN level, and LVM can add a level of complexity that reduces much of the flexibility offered by Storage Center. One example of this is cloned volumes containing LVM signatures. When a volume with an existing LVM signature is cloned and then remounted (say for development or restore functionality), the duplicate LVM signature will make the volume initially inaccessible. If needed, this can be circumvented by use of the command “vgimportclone”. While the vgimportclone command is not included as part of the SLES 11 distribution, it can still be acquired from the LVM project web site.

Cloning Single Path Volumes
Cloning volumes (including boot volumes) which are presented over a single path to a server is a relatively simple operation that requires very little unexpected effort. Once a volume is created from a replay in Storage Center, it can be mapped to a server object and booted and mounted as if it were the original volume.

Cloning Multipathed Data Volumes
Cloning and mounting data volumes (I.E not boot volumes) that are presented via multiple paths requires some manual effort. Once the volume has been created from a replay and mapped, the same
steps as detailed in “Adding a Multipath Volume post install” can be used to mount the volume within a server.

**Cloning Multipathed Boot/Root Volumes**
As part of the process to create the initial RAMDISK used to boot into a multipathed volume, some details such as the unique ID of the volume are packed into the RAMDISK. Because of this, cloning multipathed boot volumes can become a difficult task. As this is a limitation of SLES 11 (and most Linux distributions), we are unable to directly support cloning multipathed boot volumes for the purpose of booting from that cloned volume into a new server environment.