Best Practices

Dell EMC SC Series: Red Hat Enterprise Linux Best Practices

Abstract

This paper provides guidelines for configuring volume discovery, multipath, filesystem, and queue-depth management for Red Hat® Enterprise Linux® (RHEL) 6.x to 8.x with Dell™ Storage Center OS (SCOS) 7.x.x.

August 2019

Revisions

Date	Description
October 2013	Initial release
December 2013	Refreshed for RHEL 6.4
May 2015	Introduce connectivity to Dell Storage SCv2x00
July 2017	Refreshed for RHEL 6.9
January 2018	Edits to address SCSI device timeouts
August 2019	Content merged for RHEL 6.x, 7.x, and 8.x

Acknowledgements

Author: Henry Wong

Table of contents

Executive summary

Red Hat® Enterprise Linux® (RHEL) is an extremely robust and scalable enterprise-class operating system. Correctly configured using the best practices presented in this paper, RHEL provides an optimized experience for use with the Dell EMC™ SC Series storage. These best practices include guidelines for configuring volume discovery, multipath, file system, and queue depth management.

This paper presents RHEL versions 6.x to 8.x with the Dell™ Storage Center Operating System (SCOS) version 7.x.x. There are various methods for accomplishing the described tasks, and this paper provides a starting point for end users and system administrators.

This paper focuses on using bash shell commands and scripts wherever possible on the command line interface (CLI) because it is the most universally applicable across UNIX® and Linux distributions.

1 Overview

SC Series storage provides Linux-compatible and SCSI-3 compliant disk volumes that remove the complexity of allocating, administering, using, and protecting mission-critical data. A properly configured SC Series array removes the need for cumbersome physical disk configuration exercises and management along with complex RAID configuration practices. The SC Series array also provides RAID 10 speed and reliability at the storage layer so that volumes do not need to be further RAID-managed within the Linux operating system.

The full range of Linux utilities such as mirroring, backup, multiple file system types, multipath, boot from SAN, and disaster recovery can be used with SC Series volumes.

Each major Red Hat release introduces new capabilities, features, performance improvements, and better interoperability. This paper focuses mainly on storage features and management. For a full list and descriptions of new features, refer to the applicable release notes on the Red Hat Enterprise Linux Document Portal.

2 Server configuration

This section discusses configuring the I/O stack on a Linux system. How the stack behaves and performs can be precisely configured to the needs of the environment where the Linux systems operate. The configuration aspects include tuning the HBA port retry count, the queue depth, the SCSI device timeout values, and many more. Depending on the types and models of the HBAs, the parameters might be set on the HBA BIOS. Consult the vendor documentation for information about the BIOS settings. The following subsections focus on setting the parameters on the OS HBA driver. This provides a starting point and outlines a few configuration parameters which should be considered as they are adapted for individual environment use.

The recommended settings for the HBAs, including the values to use with the OS HBA drivers, are documented in the Dell EMC Storage Compatibility Matrix .

2.1 Managing the HBA driver module

The **modprobe** Linux facility provides a means to manage and configure the HBA driver operating parameters (such as QLogic or Emulex) on a Linux system. Commonly, the queue depth and portdown timeout are defined or overwritten in the HBA driver module configuration files that reside in the following locations:

```
/etc/modprobe.d/<file> or /etc/modprobe.conf
/etc/modprobe.d/qla2xxx.conf  #For QLogic
/etc/modprobe.d/lpfc.conf     #For Emulex
```

2.1.1 Managing the queue depth

The default LUN queue depth can vary depending on the HBAs. The default value is usually around 32. A higher queue depth of 64 or 128 might benefit environments that generate a high volume of I/Os. The setting can be adjusted accordingly to meet individual environment needs. Increasing the queue depth might improve performance without impacting the overall latency but performance testing should be done to determine the exact benefit.

Show the current LUN queue depth setting:

```
# lsscsi -l
[14:0:2:0]   disk    COMPELNT Compellent Vol   0703  /dev/sdb
  state=running queue_depth=32 scsi_level=6 type=0 device_blocked=0 timeout=60
```

Show the HBA queue depth setting:

- For QLogic:

  ```
  # cat /sys/module/qla2xxx/parameters/ql2xmaxqdepth
  ```

- For Emulex:

  Global settings that apply to all Emulex HBAs:

  ```
  # cat /sys/module/lpfc/parameters/lpfc_lun_queue_depth
  # cat /sys/module/lpfc/parameters/lpfc_hba_queue_depth
  ```

Queue depth settings for a specific Emulex HBA:

```
# cat /sys/class/scsi_host/host12/lpfc_lun_queue_depth
# cat /sys/class/scsi_host/host12/lpfc_hba_queue_depth
```

Note: the HBA settings might not match the global settings if the HBA does not support them.

To change the HBA queue depth settings, include the following line to the HBA driver module configuration file (section 2.1) and reload the driver module (section 2.1.3).

```
options qla2xxx ql2xmaxqdepth=<value>  #For Qlogic

options lpfc lpfc_lun_queue_depth=<value> lpfc_hba_queue_depth=<value>  #For
Emulex
```

2.1.2 Setting the HBA port timeout

During an SC Series **path** or **controller** failover while operating in **legacy port mode,** failing ports trigger their respective World Wide Port Name (WWPN) identities to momentarily disappear from the SAN fabric. The WWPN identities then relocate to a reserve port within the same fault domain on the alternate active controller.

During a **path** failover while operating in **virtual port mode**, the WWPN identities of failing N_Port ID virtualization (NPIV) ports are relocated to another active NPIV port within the same fault domain on the same controller. During a **controller** failover, the WWPN identities of any failing NPIV ports are relocated to the active NPIV ports within the same fault domain on the alternate active controller.

In either failover scenario, the SC Series storage system may take up to 60 seconds to propagate these changes through the SAN fabric.

In order to mitigate I/O disruption, it is recommended to instruct the HBA driver to wait up to 60 seconds before marking a port as down or failed. For volumes with multiple access paths managed by **multipathd**, if all paths become inaccessible, **multipathd** starts queuing I/O until one or more paths have recovered. This allows the storage system sufficient time to relocate the WWPN identities of the failed ports to active ports and propagate the changes through the SAN fabric.

The HBA PortDown timeout parameter dictates how long a Linux system waits before destroying a connection after losing its connectivity with the port. The PortDown timeout is generally set to 60 seconds for the SC Series storage system. Consult the Dell EMC Storage Compatibility Matrix for the most up-to-date recommended values.

To configure the PortDown timeout in the HBA driver module, add or append the following options to the appropriate HBA driver module configuration file (section 2.1) and reload the driver module (section 2.1.3).

```
options qla2xxx qlport_down_retry=60   #For QLogic

options lpfc lpfc_devloss_tmo=60       #For Emulex
```

2.1.3 Reloading modprobe

After updating the HBA driver module configuration file, reload the modprobe facility for the new or updated configuration to take effect.

For local-boot systems, it is recommended to unmount all SAN volumes and reload the module. The module should be unloaded from memory before it is reloaded as shown in the following.

```
# modprobe –r qla2xxx
# modprobe qla2xxx
```

Replace **qla2xxx** with **lpfc** if working with Emulex hardware; SAN volumes can be remounted subsequently.

2.1.4 Rebuilding the initial ramdisk image

For configuration persistence, rebuild the **initramfs-*.img** file so that the new configurations are incorporated during boot time. The following methods demonstrate rebuilding the **initramfs-*.img** file. It is recommended to copy and back up the existing **initramfs-*.img** file before applying this procedure.

```
# cp /boot/initramfs-$(uname -r).img /boot/initramfs-$(uname -r).img.$(date
+%Y%m%d-%H%m)

# dracut --force  #this replaces the existing version of the initramfs file
```

Ensure that the grub configuration file points to the correct **initramfs-*.img** file and reboot the system. Find the grub configuration file in one of the following locations:

```
/boot/grub/grub.conf  #RHEL 6 and older
/boot/grub2/grub.conf #RHEL 7 and newer
```

2.1.5 Verifying the parameters

To verify that the configuration changes have taken effect, examine the driver module parameter values in their respective files.

- QLogic:

  ```
  # cat /sys/module/qla2xxx/parameters/qlport_down_retry
  # cat /sys/module/qla2xxx/parameters/ql2xmaxqdepth
  ```

- Emulex:

  ```
  # cat /sys/module/lpfc/parameters/lpfc_devloss_tmo
  # cat /sys/module/lpfc/parameters/lpfc_lun_queue_depth
  # cat /sys/module/lpfc/parameters/lpfc_hba_queue_depth

  # cat /sys/class/scsi_host/hostX/lpfc_devloss_tmo  #where X is the HBA no.
  # cat /sys/class/scsi_host/hostX/lpfc_lun_queue_depth
  # cat /sys/class/scsi_host/hostX/lpfc_hba_queue_depth
  ```

2.2 SCSI device command timeout

On Linux, the SCSI device command timeout has a default value of 30 seconds. Verify the setting with the following command.

```
# cat /sys/block/sdX/device/timeout
```

Typically, this value does not need to be changed unless instructed or recommended to do so by support or application-specific directives. If necessary, the value can be changed using the following process.

1. Change the timeout in seconds for a SCSI device by writing to the device's **sysfs** file. This does not persist after system reboots.

   ```
   # echo 60 > /sys/block/sdX/device/timeout
   ```

2. Change and persist the timeout in seconds by the **udev** rule.

   Location of the **udev** rule files:

   ```
   /etc/udev/rules.d/60-raw.rules   #RHEL 6 and older
   /lib/udev/rules.d/60-raw.rules   #RHEL 7 and newer
   ```

   Content of the udev rule:

   ```
   ACTION=="add", SUBSYSTEMS=="scsi", ATTRS{vendor}=="COMPELNT*",
   ATTRS{model}=="Compellent Vol*", RUN+="/bin/sh -c 'echo 60
   >/sys$DEVPATH/device/timeout'"
   ```

3. Reload **udev** and **multipathd** (if applicable) for the changes to take effect immediately. The 2-second sleep period between commands ensures **udev** has time to complete before reloading the **multipathd** service, picking up the new timeout value of the respective child paths.

   ```
   # udevadm trigger --action=add
   # sleep 2
   # multipath -r > /dev/null
   ```

2.3 Serial-attached SCSI

Certain SC and SCv Series models offer 12Gbps serial-attached SCSI (SAS) front-end connectivity. The front-end SAS support was introduced in SCOS 6.6.x. A Linux system equipped with a compatible SAS HBA can direct-attach an SC Series storage system through the SAS protocol. Consult the Dell EMC Storage Compatibility Matrix for the HBA compatibility information.

In direct SAS-connected environments, paths from both controllers are presented to the connected Linux system (active/optimized and standby). However, only the active/optimized path is used for all active I/O at any one time. When the active/optimized path becomes unavailable, the storage system dynamically determines which remaining standby path is the active/optimized path and continues to stream active I/O to the new active/optimized path.

2.3.1 SAS drivers

Since RHEL 6.5, SAS drivers are preloaded into certain Linux kernels. The existence of the SAS drivers can be validated with the following commands. As a best practice, validate the driver versions with the Dell EMC Storage Compatibility Matrix and use the latest supported driver as indicated.

```
# lsmod | grep sas
mpt3sas               188001  4
scsi_transport_sas     35588  1 mpt3sas
raid_class              4388  1 mpt3sas
megaraid_sas           96205  2

# modinfo mpt3sas | grep description
description:    LSI MPT Fusion SAS 3.0 Device Driver
```

2.3.2 FC/iSCSI and SAS

The use of a mixed FC/iSCSI and SAS-connected environment on the same Linux system is not validated nor supported. Maintain and operate a SAS-connected Linux system separately from other Linux systems using FC/iSCSI connectivity.

2.3.3 SAS queue depth

Currently supported SAS HBAs default to a queue depth of 254 for each volume. Keep this factory-default value unless directed by support- or application-specific directives.

```
# lsscsi -l
 [1:0:0:1]    disk    COMPELNT Compellent Vol    0606  /dev/sdb
  queue_depth=254
 [1:0:1:1]    disk    COMPELNT Compellent Vol    0606  /dev/sdc
  queue_depth=254
```

2.3.4 Boot-from-SAS

Boot-from-SAS functionality is not validated nor supported in use with Dell EMC 12Gbps SAS HBAs on Linux.

2.4 Software iSCSI

Most major Linux distributions support iSCSI and include a software iSCSI initiator. RHEL systems utilize the open-iSCSI implementation of software iSCSI on the Linux platform. While iSCSI is a mature technology that allows organizations to economically scale, it has grown in complexity at both the hardware and software layers. The scope of this document 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) consult the vendor documentation and services.

For instructions on setting up an iSCSI network topology, consult the SC Series *Deployment Guide* for the respective model at the SC Series customer portal (login required).

2.4.1 Configuration

The Linux system being configured requires an Ethernet port, preferably a dedicated one that can communicate with the iSCSI ports on the SC Series storage system.

The most important consideration when configuring an iSCSI volume is the network path. Due consideration is recommended to determine the confidentiality, security, and latency required for the iSCSI traffic. These needs will drive and define the network topology of the iSCSI architecture (for example: dedicated physical ports, VLAN usage, multipath, and redundancy).

It is a best practice to separate and isolate iSCSI traffic from routine network traffic by the use of dedicated ports, switches, and infrastructure. If the physical topology is constrained, it is recommended to separate and isolate iSCSI traffic by the use of VLAN subnets. It is also a best practice to always use iSCSI in a multipath configuration to create proper path redundancy.

If VLAN subnets are not possible, two further options that can be explored are as follows:

- Route traffic at the network layer by defining static routes.
- Route traffic at the iSCSI level with the configuration.

The following directions assume that a network port has already been configured to communicate with the SC Series iSCSI ports.

2.4.2 Initializing and configuring iSCSI

1. Install the software package.
2. The iSCSI software and tools are provided by the **iscsi-initiator-utils** package. Install the package with **yum** if it is not already available on the system.

   ```
   # yum install iscsi-initiator-utils
   ```

3. Start and enable iSCSI daemon.
4. The iSCSI software initiator consists of two main components: the daemon that runs in the background to handle connections and traffic, and the administration utility used to configure and modify connections. If the daemon does not start automatically during boot, it must be started before beginning the configuration.

   - For RHEL 6 and older:

   ```
   # service iscsid start
   # chkconfig --levels 345 iscsid on
   ```

   - For RHEL 7 and newer:

   ```
   # systemctl start iscsid
   # systemctl enable iscsid
   ```

5. Discover the IQN names for the SC Series ports.

6. In the following example, the iSCSI ports on the SC Series storage system have the IP addresses 172.16.26.180 and 10.10.140.180, respectively.

   a. Configure the first path:

```
# iscsiadm -m discovery -t sendtargets -p 172.16.26.180
172.16.26.180:3260,0 iqn.2002-03.com.compellent:5000d3100000677c
172.16.26.180:3260,0 iqn.2002-03.com.compellent:5000d3100000677e
172.16.26.180:3260,0 iqn.2002-03.com.compellent:5000d31000006780
172.16.26.180:3260,0 iqn.2002-03.com.compellent:5000d31000006782
```

   b. Configure the second path:

```
# iscsiadm -m discovery -t sendtargets -p 10.10.140.180
10.10.140.180:3260,0 iqn.2002-03.com.compellent:5000d3100000677d
10.10.140.180:3260,0 iqn.2002-03.com.compellent:5000d3100000677f
10.10.140.180:3260,0 iqn.2002-03.com.compellent:5000d31000006781
10.10.140.180:3260,0 iqn.2002-03.com.compellent:5000d31000006783
```

7. Log in to the target ports. The iSCSI daemon will save the nodes in **/var/lib/iscsi** and automatically log into them when the daemon starts. Use the following commands to instruct the software to log into all known nodes.

```
# iscsiadm -m node --login
Logging in to [iface: default, target: iqn.2002-
03.com.compellent:5000d31000006782, portal: 172.16.26.180,3260] (multiple)
Logging in to [iface: default, target: iqn.2002-
03.com.compellent:5000d31000006783, portal: 10.10.140.180,3260] (multiple)
[snip]
```

8. Launch the DSM GUI and create a server object on the SC Series storage system.
9. Create and map the SC Series volumes to the server.
10. On the Linux system, rescan the HBAs to discover new volumes presented to the system. See section 3.

As long as the iSCSI daemon automatically starts during boot, the system logs in to the SC Series iSCSI targets and discovers the iSCSI-based volumes.

2.4.3 Scanning for new volumes

The process for scanning the iSCSI software initiator for new volumes is identical to that used for scanning the Fibre Channel **hostX** devices as discussed in section 3.1.

2.4.4 iSCSI timeout

The pre-configured iSCSI settings, defined in **/etc/iscsi/iscsi.conf,** work with SC Series storage system when the Linux system is also configured to use **dm-multipath.** It is a best practice to use **dm-multipath** for path redundancy and failover. It is important to note the following timeout settings because they directly affect the storage availability and failure recovery for a SAN storage system. When an SC Series storage system controller fails, services and volumes fail over to the surviving controller and this failover might take up to 60 seconds. Therefore, ensure the iSCSI timeout settings are sufficient to accommodate this kind of event. These preconfigured timeout settings listed below work well with SC Series storage systems.

- `node.conn[0].timeo.noop_out_interval = 5`
- `node.conn[0].timeo.noop_out_timeout = 5`
- `node.session.timeo.replacement_timeout = 120`

noop_out_interval controls how often to send iSCSI NOP-Out requests. **noop_out_timeout** controls the number of seconds to wait for a NOP-Out. Together, they control how quickly the iSCSI layer detects network problems. The **replacement_timeout** controls the number of seconds to wait for session re-establishment before failing pending SCSI commands and commands worked on by the SCSI error handler.

When a network problem is detected (a NOP-Out times out), the iSCSI layer fails the running commands to the SCSI layer. If **dm-multipath** is used, the SCSI layer fails them to the multipath layer to handle the queueing and retry on other paths according to the policy in **/etc/multipath.conf** (see section 3.6). If **dm-multipath** is not used, the SCSI layer retries the commands 5 times before failing them. The SCSI device command timeout is defined in **/dev/block/sdX/device/timeout** (see section 2.2).

When the SCSI layer error handler kicks in (triggered by the timeout in **/dev/block/sdX/device/timeout**) and is running, the commands do not fail immediately. Instead, they wait **node.session.timeo.replacement_timeout** seconds before failing them. After the **replacement_timeout** seconds, the commands are failed to the multipath layer. To verify the error handler is running, run **iscsiadm** command and look for **State: Recovery** for any **Host Number.**

```
# iscsiadm -m session -P 3 |egrep "Host Number"
Host Number: X State: Recovery
```

The iSCSI configuration should be thoroughly tested in the environment against all perceivable failover scenarios (such as switch, port, fabric, and controller) before deploying them into a production environment.

2.4.5 Filesystem on iSCSI volume

Since iSCSI is dependent on a running network, filesystems created on iSCSI volumes should be designated as network-dependent in **/etc/fstab**. In other words, do not attempt to mount an iSCSI volume until the network layer services have fully started. The following example demonstrates how to create this network dependency to the iSCSI mount using the **_netdev** mount option in **/etc/fstab**.

```
LABEL=iscsiVOL /mnt/iscsi  ext4  _netdev     0 0
```

3 Volume management

Understanding how volumes are managed in Linux requires basic understanding of the /sys pseudo-filesystem. The **/sys** filesystem is a structure of files that allow interaction with various elements of the kernel and modules. While the read-only files store current values, read/write files trigger events with the correct commands. Generally, the cat and echo commands are used with a redirect as STDIN instead of being opened with a traditional text editor.

To interact with the HBAs (FC, iSCSI, and SAS), commands are issued against special files located in the **/sys/class/scsi_host/** folder. Each port on a multiport card represents a unique HBA, and each HBA has its own hostX folder containing files for issuing scans and reading HBA parameters. The folder layout, files, and functionality can vary depending on the HBA vendor or type. Example HBAs include QLogic® Fibre Channel, Emulex® Fibre Channel, software-iSCSI based HBAs, or Dell EMC 12Gbps SAS HBAs.

RHEL systems also provide several utilities to simplify the volume management. To access these utilities, install the **sg3_utils** and **lsscsi** packages.

```
# yum install sg3 utils lsscsi
```

3.1 Identifying HBAs on the Linux system

Before the SC Series volumes can be presented/mapped to a Linux system, first create a server object in DSM using the information of the Linux system HBAs. This section demonstrates how to identify the HBA information to be used with this process.

3.1.1 FC HBAs

SC Series storage identifies a Linux system FC HBA by the WWPN. To display the WWPN, display the content of **/sys/class/fc_host/hostX/port_name.**

```
# cat /sys/class/fc_host/host11/port_name
```

The following example script displays the WWPN and other useful information for all HBAs.

```
#!/bin/bash
# Script name: fcshow.sh

printf "%-10s %-20s %-10s %-s\n" "Host-Port" "WWN" "State" "Speed"
printf "%50s\n" |tr ' ' -

ls -1d /sys/class/fc_host/host* | while read host
do
   (echo ${host##*/}: ; cat $host/port_name $host/port_state $host/speed)| xargs
-n 5 printf "%-10s %-20s %-10s %s %s\n"
done
```

The script produces the following results. The WWPN field will be used during the creation of server object in DSM (see section 3.1.4).

```
# ./fcshow.sh

Host-Port  WWPN                  State       Speed
-------------------------------------------------
host14:    0x21000024ff1f5820    Online      16 Gbit
host15:    0x21000024ff1f5821    Online      16 Gbit
```

3.1.2 iSCSI HBAs

SC Series storage identifies a Linux system iSCSI HBA by the iSCSI initiator name. Locate this information in **/etc/iscsi/initiatorname.iscsi.**

```
# cat /etc/iscsi/initiatorname.iscsi
InitiatorName=iqn.1994-05.com.redhat:7cec2084e215
```

3.1.3 SAS HBAs

Each SAS HBA has a unique SAS address. SC Series storage uses this address to establish the connectivity to the Linux system. To identify the SAS addresses, display the content of each **host_sas_address** attribute file in the **/sys/devices** filesystem.

The following example finds and displays the contents of **host_sas_address**. The output shows the full path of the **host_sas_address** file and the actual value delimited by a colon. The value displayed will be used during the creation of server object in DSM (see section 3.1.4).

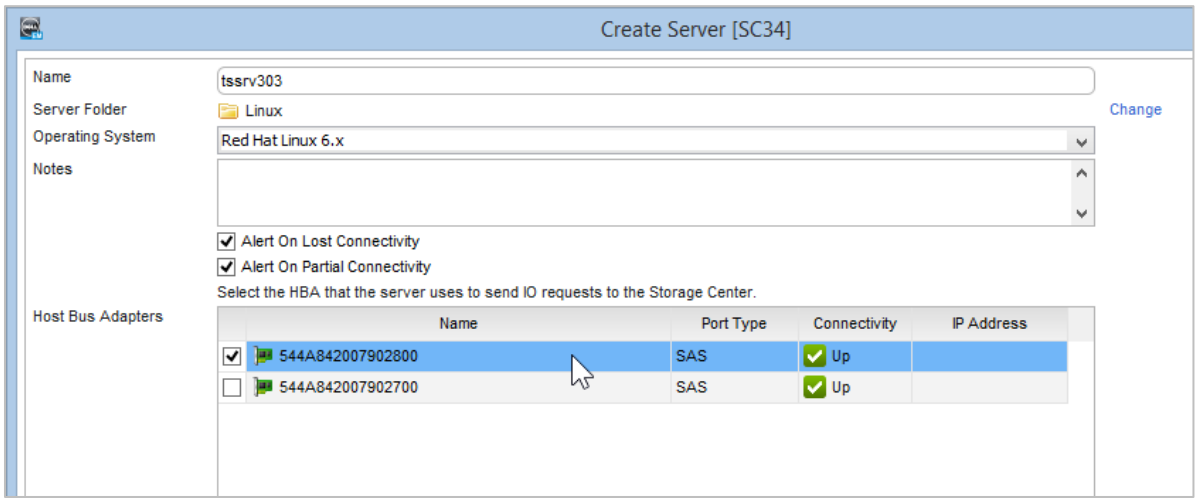
```
# find /sys/devices -name host_sas_address –exec egrep –H "*" {} \;

/sys/devices/pci0000:40/0000:40:01.0/0000:41:00.0/host1/scsi_host/host1/host_sas
_address:0x544a842007902800
/sys/devices/pci0000:40/0000:40:03.0/0000:42:00.0/host2/scsi_host/host2/host_sas
_address:0x544a84200792ce00
```

3.1.4 Creating a server object in DSM using the Linux HBA information

The following sequence outlines the procedure to create a new server object on SC Series storage.

1. Right-click the **Servers** folder tree and click **Create Server**.
2. On the **Create Server** window, available HBAs that are not associated with any existing server objects are selectable in a table. DSM discovers these HBAs in the following cases:
   - Supported FC HBAs are properly configured, zoned, and can establish SAN fabric connectivity between the Linux system and the SC Series storage system.
   - Supported iSCSI HBAs are properly configured and on a network that have network connectivity between the Linux system and the SC Series storage system.
   - Supported SAS HBAs are properly configured, directly connected to the SC Series storage system, and can establish connectivity between the Linux system and SC Series storage system.

3. To identify the correct Linux system HBAs, obtain the HBA information on the Linux system (see sections above), and cross-reference them with the list displayed.



3.2 Discovering and identifying SC Series volumes on a Linux system

Once the server object for the Linux system is created, present/map the SC Series volumes to the Linux system. This section provides information on how to discover and identify these mapped volumes on a Linux system.

The driver modules for the QLogic 24xx/25xx Series HBAs and the Emulex HBAs come with the base kernel code on RHEL 6 and newer systems. The following instructions apply to the default HBA driver modules. If the vendor (QLogic, Emulex) proprietary driver has been used, consult the vendor specific documentation for instructions and details.

3.2.1 Scanning for LUNs

The easiest way to scan for new volumes is to use the **rescan-scsi-bus.sh** command provided by the sg3_utils package. The command scans all SCSI buses (FC/iSCSI or SAS), then discovers and creates device files including the multipath devices if multipath is enabled.

Scan and discover new LUNS on all SCSI buses:

```
# /usr/bin/rescan-scsi-bus.sh -a
```

If **sg3_utils** package is not available to the system, the following sample script can be adopted to scan for new volumes. The sample script performs the following tasks. This script can be used to discover both FC and iSCSI devices presented to the host.

- Identifies the major revision number of the Linux operating system
- Applies the echo command to the respective hostX devices within the **/sys/class/scsi_host/** folder
- Scans the HBA ports, and discovers and identifies existing and new volumes presented to the host from the storage system

Note: STDOUT is not generated from this script. Check the contents of **/var/log/messages** or the output from the **dmesg** or **lsscsi** commands to identify any newly discovered volumes.

Rescanning the HBAs while mapping and discovering new volumes does not have any negative impact on the host.

```
#!/bin/bash

OSMajor=`uname -r |sed -rn 's/^.*(el.).*$/\1/p'`
echo "INFO: OS Major rev. ${OSMajor} detected!"

if [[ "$OSMajor" =~ !(el5|el6|el7|el8) ]];then
   echo "WARN: OSMajor parameter of unknown value, exit 1"
   exit 1
fi

for i in /sys/class/scsi_host/*
do
   if [[ $OSMajor == "el5" ]];then
         echo 1 >> ${i}/issue_lip
   fi

   echo "- - -" >> ${i}/scan
done
```

Note: Linux hosts cannot discover LUN ID 0 on demand. LUN ID 0 is typically reserved for the OS boot volume in boot from SAN environment. The **rescan-scsi-bus.sh** provides the **--forcerescan** option and is a possible method to discover LUN 0. All other volumes should be associated to LUN ID 1 or greater.

3.2.2 Identifying LUNs by WWNs

The most accurate way to identify a LUN on the host operating system is by its WWN, also known as WWID. The SC Series array assigns a unique WWN for each LUN. The WWN contains the SC Series volume Device Number or Serial ID which can be found in the DSM GUI under the general information section of the volumes (see Figure 1). See section 5.4 on how to decode the WWN.

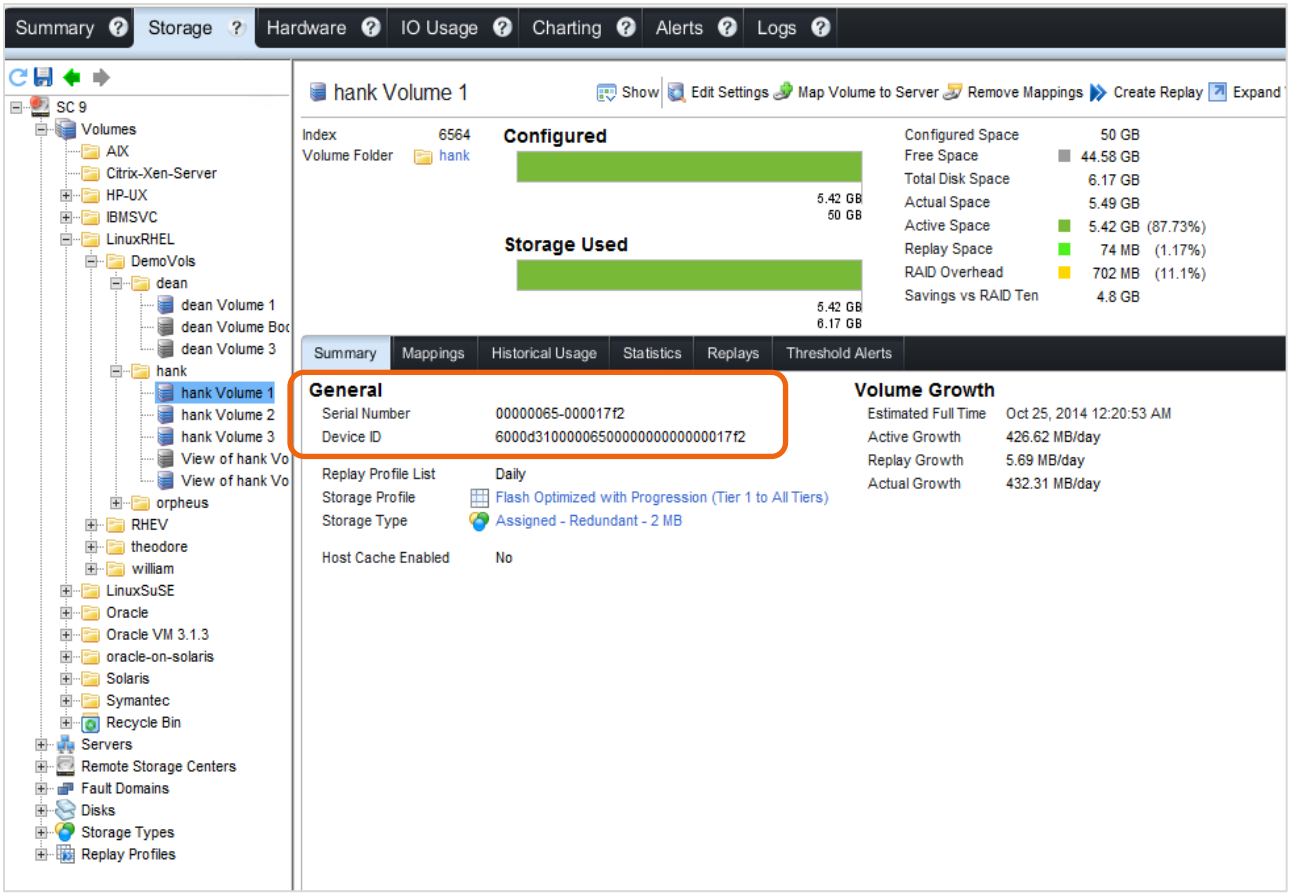


Figure 1 SC Series Volume Serial Number and Device ID fields

3.2.3 Querying WWNs using scsi_id command

The **scsi_id** command (located in the **/lib/udev** folder) can be used to report the WWID of volumes. This WWID can then be used to correlate volumes to their respective SC Series storage devices.

```
# /lib/udev/scsi_id --page=0x83 --whitelisted --device=<dev_device>  # RHEL 6
                                                                     # and newer

# /lib/udev/scsi_id -g -u -s <block_device>  # For older RHEL
```

<dev_device> can be one of the following:

- Single path device (/dev/sdX)
- Linux multipath device (/dev/mapper/mpathX)
- Dell EMC PowerPath device (/dev/emcpowerX)

<block_device> is for `/block/sdX` devices.

The string returned by the **scsi_id** command indicates the WWN of the SC Series volume as shown below.

```
36000d31000006500000000000000017f2
```

To query many volumes for their WWIDs, see a sample script that applies the scsi_id command on all the volumes in section 5.3.

3.2.4 Querying WWNs using multipath command

If the system has Linux **device-mapper-multipath** software enabled, the multipath command displays the multipath device properties including the WWN. Section 3.6 provides more information on multipathing.

```
# multipath -ll
appdata_001 (36000d31000006500000000000000017f2) dm-3 COMPELNT,Compellent Vol
size=50G features='1 queue_if_no_path' hwhandler='0' wp=rw
`-+- policy='service-time 0' prio=1 status=active
  |- 1:0:3:1 sdf 8:80  active ready running
  |- 1:0:5:1 sdk 8:160 active ready running
  |- 4:0:4:1 sdg 8:96  active ready running
  `- 4:0:5:1 sdj 8:144 active ready running
```

3.3 Managing persistent devices with volume labels and UUIDs

Volumes discovered in Linux are given device designations such as **/dev/sdd** and **/dev/sdf** depending on the Linux discovery method used by the HBA ports connecting the server to the SAN.

Among other uses, these **/dev/sdX** device names designate the volumes for mount commands including entries in the **/etc/fstab** file. In static disk environments, **/dev/sdX** device names work well for entries in the **/etc/fstab** file. However, the dynamic nature of Fibre Channel or iSCSI connectivity inhibits Linux from tracking these disk designations persistently across reboots.

There are multiple ways to ensure that these volumes are assigned and referenced by a persistent naming scheme. This section presents using volume labels or universally unique identifiers (UUIDs) with /dev/sdX-referenced volumes. Volume labels are exceptionally useful when scripting SC Series snapshot recovery. An example involves mounting a snapshot view volume of a production SC Series snapshot to a backup server. In this case, the snapshot view volume may be referenced by its volume label without needing to explicitly identify the associated **/dev/sdX** device. The volume label is metadata stored within the volume and is inherited by snapshot view volumes created from the SC Series snapshot.

Volume labels or UUIDs can also be used in multipath environments. That being said, multipath **/dev/mapper/mpathX** device names (or multipath aliases as described in section 3.6.7) are persistent by default and will not change across reboots. Volume labels, UUIDs, or multipath device names can be used interchangeably for entries in the **/etc/fstab** file of a local Linux host.

If snapshot view volumes are used for recovery or relocation of volumes to an alternate Linux system, the use of volume labels or UUIDs is recommended. These UUID values uniquely identify the volumes, in contrast to multipath names which may differ (depending on the configuration of the **/etc/multipath.conf** file).

3.4 Volume size

SC Series storage supports a maximum volume size of 500 TB or the maximum addressable space of the page pool, whichever is smaller. It also supports a maximum of 2000 volumes in a single system. Many modern filesystems and applications can now handle volumes larger than 2 TB. Large volumes have no discernible performance difference compared to small volumes. However, using more volumes can improve performance by allowing more concurrent streams and I/O queues to the storage system. Also, volumes are balanced across both SC Series storage controllers. It is a best practice to use two or more volumes on a Linux system to increase performance.

3.5 Thin-provisioned volumes

SC Series storage provides the flexibility to create large volumes upfront without consuming much space thanks to the thin provisioning feature. All SC Series volumes are thin-provisioned by default. No space is consumed until the applications start writing data to the volumes. When volumes eventually approach full capacity, administrators can easily and dynamically expand the volumes without affecting the Linux operating system and applications. Monitor the actual volume space usage in the volume summary and volume statistics sections in the DSM GUI.

While it is not possible to shrink an SC Series volume size, it is possible to reclaim unused space to reduce the actual volume space consumption after data/files have been deleted from the volume (see sections 3.8.2 and 3.9.3).

3.6 Multipathing

Multipathing is a software solution implemented at the host-operating-system level. While multipathing is optional, it provides path redundancy, failover, and performance-enhancing capabilities. Therefore, it is highly recommended to deploy the solution in a production environment or environments where availability and performance are critical.

The main benefits of using an MPIO solution include the following:

- Increase application availability by providing automatic path failover and failback.
- Enhance application I/O performance by providing automatic load balancing and capabilities for multiple parallel I/O paths.
- Ease administration by providing persistent user-friendly names for the storage devices across cluster nodes.

3.6.1 Presenting SC Series volumes with multiple access paths

To allow multiple access paths to the SC Series storage system, the following requirements should be met.

- Have at least two FC/iSCSI or SAS HBAs on the Linux system to provide path redundancy.
- Use multiple switches to provide switch redundancy.
- Use virtual ports and create two fault domains on the SC Series storage system.
- Configure proper FC zoning or a network VLAN (see the SC Series *Owner's Manual* on http://support.dell.com).
- Ensure all paths to the same volume use the same transport protocol and have similar performance characteristics. While it is possible to mix the FC and iSCSI access paths to the same volume, it is not recommended due to the difference in performance.
- If a Linux system provides both FC and iSCSI access, create one server object per transport protocol in DSM GUI to represent the Linux system. Do not include both FC and iSCSI access points in the same server object. For example, a server object with two FC access points to HostA and another server object with two iSCSI access points to the same HostA (see Figure 2)
- Map an SC Series volume to either an FC or iSCSI server object. This way, a Linux system can access both FC volumes and iSCSI volumes on the same SC Series storage system.

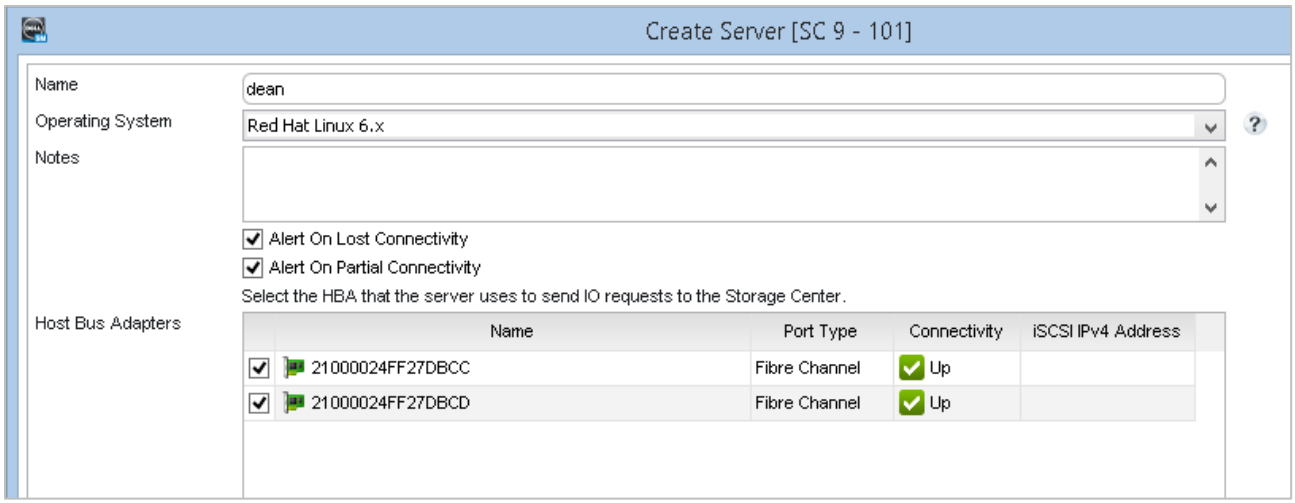


Figure 2 Name the Server object, define the Operating System, and select the corresponding FC HBA WWPNs associated with the Server object

3.6.2 Multipath software solutions

There are a few multipath software choices, and it is up to the administrator's to decide which software solution is best for the environment. The following list provides a brief description of some of these solutions.

- Native Linux multipath (device-mapper-multipath)
- Dell EMC PowerPath™
- Symantec™ Veritas Dynamic Multipathing (VxDMP)

The native Linux multipath solution is supported and bundled with most popular Linux distributions in use today. Because the software is widely and readily available at no additional cost, many administrators prefer using it compared to other third-party solutions.

Unlike the native Linux multipath solution, PowerPath and Symantec VxDMP provide extended capabilities for some storage platforms and software integration. Both solutions also offer support for numerous operating systems in addition to Linux.

Only one multipath software solution should be enabled on the host and the same solution should be deployed in a cluster on all cluster hosts.

Refer to the vendor's multipath solution documentation for more information. For information on operating systems supported by PowerPath, see the Dell EMC Simple Support Matrix. Appendix **Error! Reference source not found.** provides links to additional resources to these solutions.

3.6.3 Configuring the native Linux multipath software

If the **device-mapper-multipath** package is not installed on the system, install the package from the installation CD or the vendor's software repositories.

```
# yum install device-mapper-multipath
```

Enable multipathd to autostart during system boot:

```
# systemctl enable multipathd          #For RHEL 7 and newer
# chkconfig –levels 345 multipathd on  #For RHEL 6 and older
```

Start multipathd:

```
# systemctl start multipathd           #For RHEL 7 and newer
# service multipathd start             #For RHEL 6 and older
```

Check the multipathd status:

```
# systemctl status multipathd          #For RHEL 7 and newer
# service multipathd status            #For RHEL 6 and older
```

3.6.4 FC/iSCSI volume multipath configuration

To ease the deployment of native Linux multipath, the software has default settings for an extensive list of storage models including SC Series storage. The default settings allow the software to work with SC Series storage right out of the box. However, these settings might not be optimal for all situations and should be reviewed and modified if necessary.

Create the multipath daemon configuration file on newly installed systems. Copy a basic template from **/usr/share/doc/device-mapper-multipath-<version>/multipath.conf** to **/etc/multipath.conf** as a starting point. Any settings that are not defined explicitly in the file would assume the default values. Obtain the full list of these settings using the following command. Specific SC Series settings can be found by searching for **COMPELNT** from the output.

```
# multipathd -k"show config"
```

To avoid confusion and as a best practice, it is recommended to always explicitly define the SC Series storage in **/etc/multipath.conf**. The following example shows multipath configurations for SC Series volumes on RHEL 6 and newer. These configurations apply only in FC and iSCSI implementations. The SAS multipath configuration is discussed in section 3.6.5 and should be used instead in SAS implementations.

```
defaults {
        user_friendly_names yes
        find_multipaths yes
}

devices {
        device {
                vendor "COMPELNT"
                product "Compellent Vol"
                path_grouping_policy multibus
                path_selector "service-time 0"     # See note below
                path_checker tur
                features "0"
                hardware_handler "0"
                prio const
                failback immediate
                rr_weight uniform
                no_path_retry queue
        }
}

multipaths {
```

```
        multipath {
                wwid "36000d31000006700000000000000a68"
                alias boot-vol
                uid 0
                gid 0
                mode 0600
        }
        multipath {
                wwid "36000d31000006500000000000000017f2"
                alias appdata_001
                uid 0
                gid 0
                mode 0600
       }
}
```

Note: **round-robin 0** is the default for RHEL 6 and older. **service-time 0** is the default for RHEL 7 and newer. Both are acceptable values for SC Series FC/iSCSI volumes.

Once the configuration file is in place, reload **multipathd** with the updated configuration.

```
# multipath -r
# multipath –ll
```

3.6.5 SAS volume multipath configuration

SAS connectivity to a Linux system requires a specific configuration schema in the **/etc/multipath.conf** file. Add the following device section to **/etc/multipath.conf** used exclusively for SAS-connected Linux systems. The following example defines the configuration parameters for all devices identified by **vendor="COMPELNT"** and **product="Compellent Vol".**

```
devices {
   device {
       vendor COMPELNT
       product "Compellent Vol"
       path_checker tur
       prio alua
       path_selector "service-time 0"
       path_grouping_policy group_by_prio
       no_path_retry 24
       hardware_handler "1 alua"
       failback immediate
       rr_weight priorities
   }
}
```

A properly configured SAS volume will return the following **multipath -ll** output. This SC Series volume is discovered as a multipath ALUA-capable volume, where each path is capable of I/O. The path represented by **prio=50** (active/optimized path) is used for all active I/O requests. The path represented by **prio=1** (standby path) is a highly available, redundant path and is used when the active/optimized path becomes unavailable.

```
# multipath -ll
Compelnt_0016 (36000d31000feb3000000000000000016) dm-3 COMPELNT,Compellent Vol
size=100G features='1 queue_if_no_path' hwhandler='1 alua' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| `- 1:0:0:1 sdb 8:16 active ready running
`-+- policy='service-time 0' prio=1 status=enabled
  `- 1:0:1:1 sdc 8:32 active ready running
Compelnt_001a (36000d31000feb3000000000000001a) dm-4 COMPELNT,Compellent Vol
size=100G features='1 queue_if_no_path' hwhandler='1 alua' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| `- 1:0:1:2 sdd 8:48 active ready running
`-+- policy='service-time 0' prio=1 status=enabled
  `- 1:0:2:2 sde 8:64 active ready running
```

3.6.6 Multipath aliases

multipathd creates multipath device names in the form of **/dev/mapper/mpathX** as it discovers new devices. It is recommended to assign meaningful names (aliases) for the multipath devices though it is not mandatory. For example, create aliases based on the application type and environment it is in. The following snippet in the multipaths section assigns an alias of **appdata_001** to the SC Series volume with the WWN **36000d31000006500000000000000017f2.**

```
multipaths {
        multipath {
         wwid "36000d31000006500000000000000017f2"
         alias appdata_001
       }
}
```

3.6.7 Multipath queueing

The default **dm-multipath** settings of SC Series volumes enable the **no_path_retry queue** option that instructs **dm-multipath** to queue all I/O requests indefinitely should all paths become unavailable. After the paths are recovered, pending I/Os resume execution. This also prevents I/O errors from being propagated to the applications. The setting is shown in the multipath output. It is recommended to leave this in its default.

```
# multipath -ll
[snip]
appdata_001 (36000d31000006500000000000000017f2) dm-3 COMPELNT,Compellent Vol
size=50G features='1 queue_if_no_path' hwhandler='0' wp=rw
`-+- policy='service-time 0' prio=1 status=active
  |- 1:0:3:1 sdf 8:80  active ready running
  |- 1:0:5:1 sdk 8:160 active ready running
  |- 4:0:4:1 sdg 8:96  active ready running
  `- 4:0:5:1 sdj 8:144 active ready running
```

no_path_retry can be set to **fail** if it is preferable to fail the I/Os and let the applications to handle the errors. In which case, I/Os fail immediately and are not queued when all paths become unavailable.

Within the **devices** section, locate the **COMPELNT** device definition and change the following attributes.

```
features 0
no_path_retry fail
```

To implement this to specific device at runtime, issue the following command for the desired multipath device.

```
# dmsetup message /dev/mapper/vol_001 0 "fail_if_no_path"
```

3.7 LUN partition

Partition tables are not required for volumes other than the boot drive. It is recommended to use SC Series provisioned volumes as whole drives unless the applications requires the use of partitions. Filesystems, such as **ext4** and **xfs**, and applications, such as Logical Volume Manager and Oracle ASM, can manage whole drives more efficiently without partitions. This leverages the native strengths of the SC Series array in wide-striping volumes across all disks in the tier from which the volume is provisioned. Additionally, the use of partition tables may pose challenges when expanding the volumes capacity. Either **fdisk** or **parted** can be used to create the partition. However, only **parted** can create partitions larger than 2 TB.

3.7.1 Partition alignment

When partitioning a LUN, it is recommended to align the partition on the 1M boundary for optimal performance. Misalignment can cause excessive I/Os and degrade overall performance of the system.

3.7.2 Creating and validating partition using parted

Specify the partition offset at 2048 sector (1M). The following command creates a single partition that takes up the entire LUN. Once the partition is created, the partition file **/dev/mapper/testvol1** becomes available for creating file system or application volume.

Create a partition:

```
# parted -s /dev/mapper/testvol mklabel gpt mkpart primary xfs 2048s 100%
```

Validate the partition:

```
# parted /dev/mapper/testvol align-check opt 1
1 aligned
```

List partitions:

```
# parted /dev/mapper/testvol print
Model: Linux device-mapper (multipath) (dm)
Disk /dev/mapper/testvol: 10.7GB
Sector size (logical/physical): 512B/4096B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name     Flags
 1      1049kB  10.7GB  10.7GB               primary
```

Create and label a filesystem:

```
# mkfs.ext4 –L testfs /dev/mapper/testvol1
```

3.8 Linux LVM

Linux Logical Volume Manager (LVM) is a common general-purpose storage manager included in all popular Linux distributions. It provides application-level mirroring and RAID striping, but they are not necessary when the SC Series volumes are used because storage protection is already provided by the SC Series array. Data is periodically rebalanced on the SC Series array automatically to optimize performance and availability. LVM-level mirroring and RAID striping can be done manually upon creation. However, data is not rebalanced when the volume groups and logical volumes are expanded. Administrators must manually perform the rebalance on LVM.

Multiple LUNs can be grouped into a single LVM volume group. Then, logical volumes must be created that span across these LUNs. When taking SC Series snapshots on a multi-LUN volume group, ensure the LUNs are configured in a consistency group.

A filesystem is typically created on a logical volume where the application is then installed in the filesystem. Additional space can be added to the volume groups, logical volumes, and filesystems either by adding new LUNs or by expanding existing LUNS in the volume groups. Once volume groups and logical volumes are expanded, the file systems can be resized to the newly added space. LVM and many popular file systems such as ext4 and xfs allow on-demand expansion without taking down the applications.

The following LVM guidelines are recommended:

- Use whole LUNs for volume groups.
- Create a dedicated volume group for storing each copy or version of application. This simplifies management and allows greater flexibility on array-based snapshots.
- Use two or more LUNs in a volume group when performance is of concern. This allows more I/O streams and the use of both SC Series controllers.
- Configure all LUNs with the same size in the same volume group and group them in the same consistency group.

3.8.1 Physical volume data alignment

The data alignment detection is enabled by default in the **/etc/lvm/lvm.conf** file. If the Linux operating system has trouble determining the data alignment, use the **--dataalignment** argument to indicate the alignment starts at 1M when initializing LUNs in LVM.

The following example shows the tasks to create a file system on LVM:

```
# pvcreate --dataalignment 1m /dev/mapper/testvol
# vgcreate vgapp /dev/mapper/testvol
# lvcreate –L 5g –n lvapp vgapp
# mkfs.xfs –L appfs /dev/vgapp/lvapp (for xfs)
# mkfs.ext4 –L appfs /dev/vgapp/lvapp (for ext4)
```

3.8.2 LVM space reclamation and SCSI UNMAP/TRIM

If LVM is used with SC Series volumes, LVM can be configured to respect SCSI UNMAP/TRIM command recovered space when logical volumes are removed from the member volume group. The space recovered from this removed logical volume varies depending on whether data reduction is enabled on the SC Series volumes. This feature applies to RHEL 6 and newer.

To enable the feature, edit the **/etc/lvm/lvm.conf** file and change the key value pair **issue_discards = 0** to **issue_discards = 1.**

3.9 Filesystems

The SC Series array supports a wide range of filesystems on Linux. This section focuses on two popular and stable filesystems: ext4 and xfs. The filesystem can be created on top of a LUN, a LUN partition, or a logical volume in LVM. Dell EMC recommends using the whole LUN without partition or a logical volume in LVM for ease of management.

For additional information on supported filesystems and feature limitations, consult the Red Hat Enterprise Linux Administration Guide on the Red Hat Enterprise Linux Document Portal.

3.9.1 Filesystem label and universally unique identifiers

Filesystem volume labels can be applied when creating a filesystem on the volume or by subsequently using differing commands. Different filesystems (for example, ext4 or xfs) have different filesystem metadata schemas and use different commands to view, manage, and change this data.

The Linux operating system generates a universally unique identifier (UUID) for many objects and entities, including filesystems, on the system. These UUIDs are persistent and do not change. Therefore, they can be used reliably to reference the various objects including filesystems. The UUID is created automatically during the filesystem creation.

Create a filesystem and label at the same time:

```
# mkfs.ext4 -L appfs /dev/sdX                  #For ext4
# mkfs.xfs -L appfs /dev/mapper/mpathX         #For xfs
```

Apply a volume label to an existing filesystem:

```
# tune2fs -L appfs /dev/sdX                    #For ext4
# xfs_admin -L appfs /dev/mapper/mpathX        #For xfs
```

Remove a volume label from an existing filesystem:

```
# tune2fs -L "" /dev/sdX                       #For ext4
# xfs_admin -L "--" /dev/mapper/mpathX         #For xfs
```

Show volume labels and UUIDs on a filesystem using one of the following commands:

```
# blkid /dev/mapper/vgapp-lvapp
/dev/mapper/vgapp-lvapp: LABEL="appfs2" UUID="b060087c-fba6-4517-a386-
24dee0844f55" TYPE="xfs"

# lsblk -o name,mountpoint,label,size,uuid /dev/mapper/vgapp-lvapp
NAME        MOUNTPOINT LABEL  SIZE UUID
vgapp-lvapp /testfs    appfs2   5G b060087c-fba6-4517-a386-24dee0844f55

# xfs_admin -lu /dev/mapper/vgapp-lvapp      #For xfs
label = "appfs2"
UUID = b060087c-fba6-4517-a386-24dee0844f55
```

3.9.2 Persistent naming in /etc/fstab

The **LABEL=** or **UUID=** syntax can be used to reference volumes in a variety of places including **mount** commands and entries in the **/etc/fstab** file. This provides the liberty of uniquely identifying the volumes regardless of their discovery device name designations, multipath configurations, or predefined multipath aliases.

Reference a filesystem using UUID or label in /etc/fstab:

```
UUID="b060087c-fba6-4517-a386-24dee0844f55"  /testfs xfs defaults,discard 0 0

LABEL=appfs    /testfs xfs defaults,discard 0 0
```

3.9.3 Filesystem space reclamation and SCSI UNMAP

For filesystem types that support the online SCSI TRIM/UNMAP command, such as **ext4** and **xfs**, enable the **discard** mount option in **/etc/fstab** or include **–o discard** to the manual mount command. For example**, mount -o discard /dev/mapper/vgapp-lvapp /testfs.**

This allows space to be released back to the storage pool in the SC Series system when deleting files in the filesystem. Administrators should review the filesystem documentation to confirm the availability of the features.

As new data is written to the filesystem, actual space is consumed in SC Series storage. When files are deleted from the filesystem, the operating system informs the SC Series storage system which data blocks can be released. The release of storage is automatic and requires no additional steps. To confirm the release of space in SC Series storage, monitor the **Disk Usage** on the LUN Summary page in the DSM GUI.

3.9.4 Filesystem mount options

When mounting a filesystem, consider the following options and guidelines:

- Identify the file system by its UUID, label, or LVM LV device in the **/etc/fstab** file. Avoid using any non-persistent device paths such as **/dev/sdX.**
- Include **discard** in the mount option to enable space reclamation support for the filesystem.
- Include **nofail** in the mount option if the Linux operating system experiences mount issue during system boot. This prevents interruption during the boot process which requires manual intervention.

- For the **xfs** file system, the filesystem check (**fsck** option) should be disabled in **/etc/fstab** because it does not perform any check or repair automatically during boot time. The xfs journaling feature ensures the file system integrity and data is in a consistent state after abrupt shutdown. If a manual repair or check is necessary, use the **xfs_repair** utility to repair damaged file system.
- Set a value of **0** in the sixth field to disable fsck check. Here is an example of an xfs file system entry in **/etc/fstab:**

```
UUID="b060087c-fba6-4517-a386-24dee0844f55"  /testfs xfs defaults,discard
0 0
```

- For a filesystem on iSCSI volumes, use _**netdev** mount option (see section 2.4.5).

3.9.5 Expanding storage for the filesystem

Certain filesystem types, such as ext4 and xfs, support the online resize operation. The following outlines the general steps to resize a filesystem online assuming non-partition LUNs are used.

1. Take manual snapshots of LUNs that are going to be expanded. If a filesystem spans across multiple LUNs, make sure to take a consistent snapshot of all the LUNs in the same filesystem.
2. Expand the size of the existing LUNs in the DSM GUI.
3. Perform a SCSI scan on the host systems, refresh the partition table on each LUN path, and reload multipath devices.

   ```
   # rescan-scsi-bus.sh –resize
   ```

   or

   ```
   # echo 1 >> /sys/block/sdX/device/rescan
   ```

   Reload the multipath devices:

   ```
   # multipathd -k"resize map testvol"
   ```

   For PowerPath, the new size is automatically updated.

4. Expand the physical volume and logical volume if the file system is on top of LVM.

   ```
   # pvresize /dev/mapper/testvol
   # lvresize –L $NEW_SIZE_in_MB /dev/vgapp/lvapp
   ```

5. Extend the filesystem size to the maximum size, automatically and online.

   ```
   # xfs_growfs –d /testfs          #For xfs
   # resize2fs /dev/mapper/vgapp-lvapp #For ext4
   ```

Note: As with any partition or filesystem operation, there is some risk of data loss. Dell EMC recommends capturing a snapshot of the volume and ensuring good backups exist prior to executing the steps.

3.10 Removing volumes

The Linux OS stores information about each volume presented to it. Even if a volume is in an unmapped state on SC Series storage, the Linux OS will retain information about that volume until the next reboot. If the Linux OS is presented with a volume from the same target using the same LUN ID prior to any reboot, it will reuse the old data about that volume. This may result in complications, misinformation and mismanagement of the

volumes, and potentially cause data loss in the environment. Therefore, it is a best practice to always unmount, remove, and delete all volume information on the Linux OS after the volume is deemed no longer in use. This is non-destructive to any data stored on the actual volume itself, just the metadata about the volume stored by the OS (such as volume size or type).

To remove an SC Series volume from a Linux host that is using **DM-multipath,** use the following steps:

1. Stop the applications and unmount filesystems associated with the volumes.
2. Remove the DM-multipath volume information

   ```
   # multipath –f /dev/mapper/testvol
   ```

3. If LVM is involved, choose one of the options below.

   a. To keep the data on the volumes, deactivate and export the volume group that is associated with the volumes.

   ```
   # vgchange –a n /dev/vgapp
   # vgexport /dev/vgapp
   ```

   b. If the data is no longer needed, delete the logical volume and the volume group.

   ```
   # lvremove /dev/vgapp/lvapp
   # vgremove /dev/vgapp
   ```

4. Remove the **/dev/sdX** devices.

   ```
   # rescan-scsi-bus.sh –r
   ```

   or

   ```
   # echo 1 > /sys/block/sdX/device/delete
   ```

5. Unmap the volume from the SC Series array.
6. Clean up all volume references in **/etc/fstab** and **/etc/multipath.conf.**

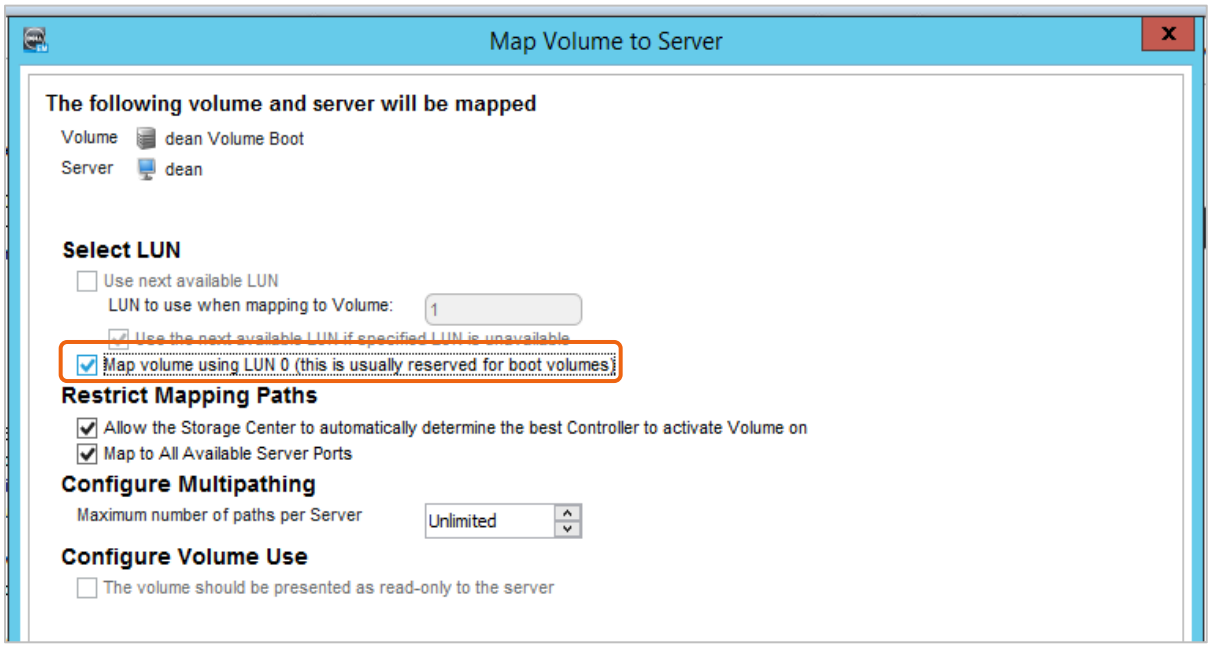
Note: Edit the **/etc/fstab** file to remove any references to the volume and associated filesystem as required. This ensures that no attempt is made to mount this filesystem at the next boot.

3.11 Boot from SAN

Using SC Series volumes as bootable volumes in Linux allows administrators to further leverage the strengths of SC Series snapshot and snapshot view volume technologies. Two uses for snapshots of Linux boot volumes include the following: a backup/recovery mechanism, or to preserve the state of an operating system at a point in time prior to upgrades.

To use an SC Series volume as a bootable volume, the target volume needs to be presented/mapped to the target Linux system as **LUN ID 0**. A volume LUN ID 0 mapping with SC Series volume is performed as follows.

1. Right-click the volume and select **Map Volume to Server**, click **Next**, and click **Advanced Options**.
2. Select the **Map volume using LUN 0** check box and click **Finish** to map this volume.

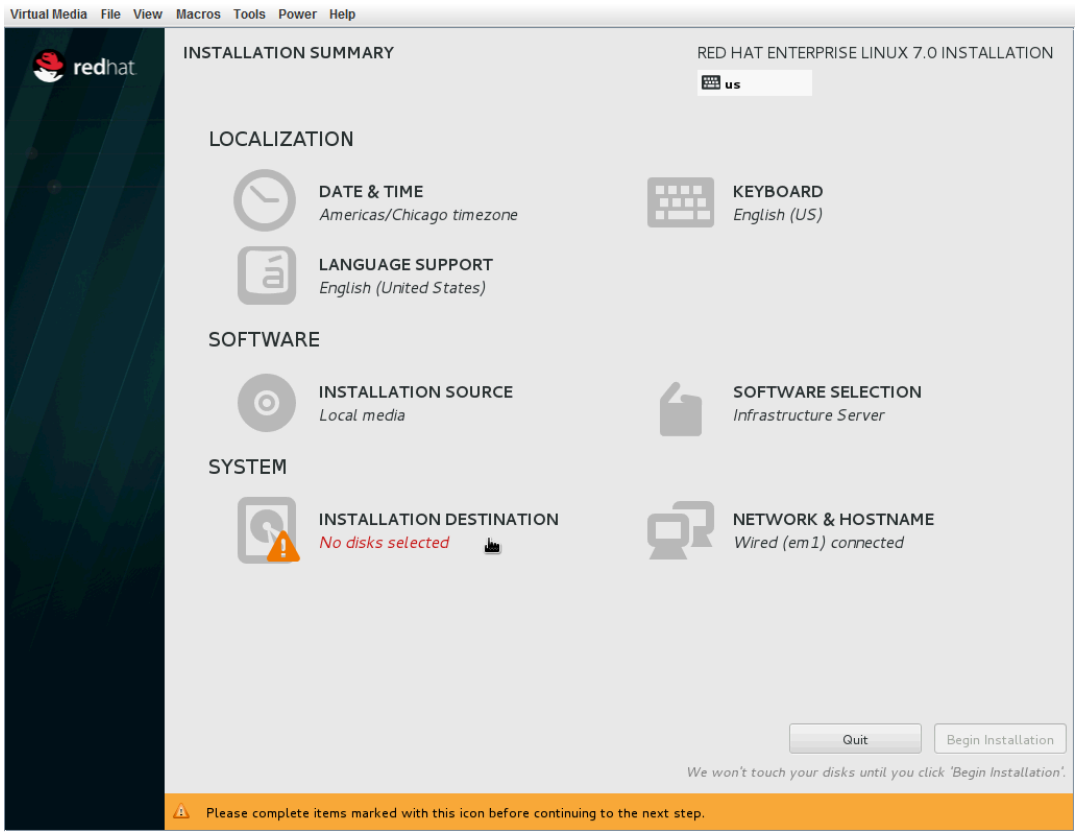


3. During the Linux system boot, interrupt the HBA BIOS boot process and identify the SC Series array WWPNs as the preferred boot devices in the boot device order.

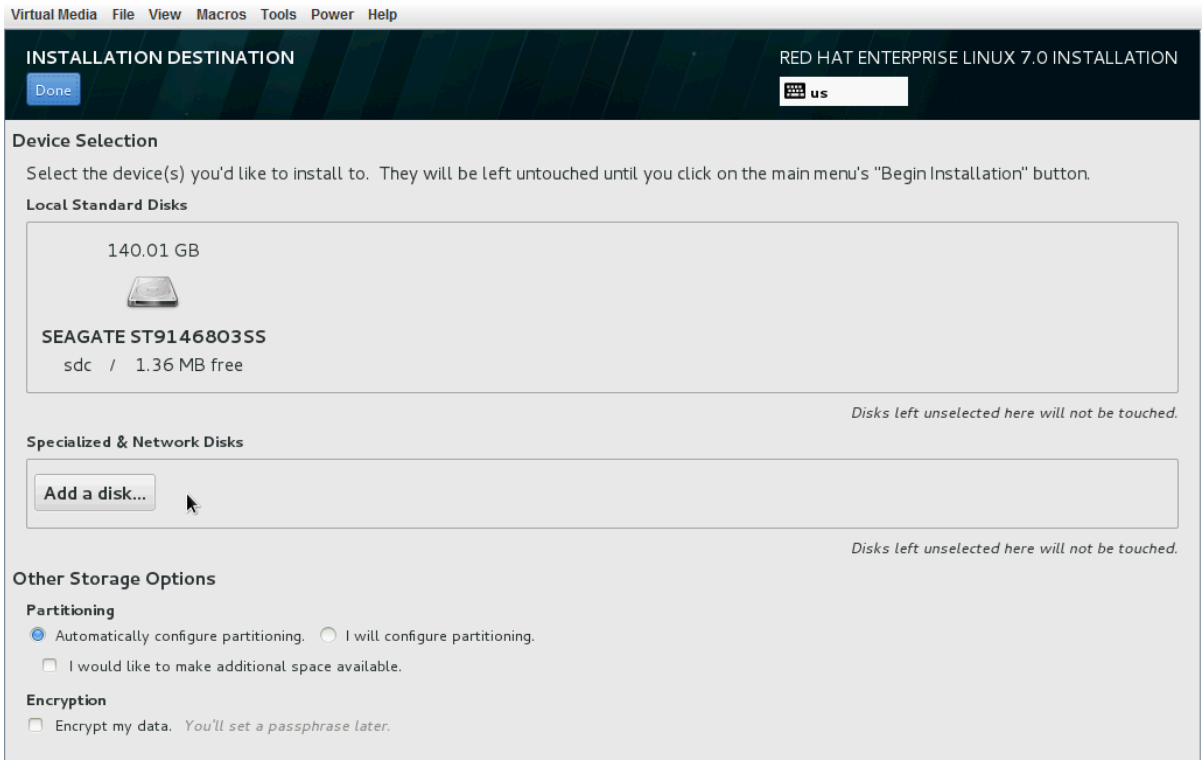
Note: The HBA BIOS configuration should identify all SC Series storage controller WWPNs of all the active and alternate controllers. The configuration of the HBA BIOS in this manner ensures that any boot from SAN volumes will remain visible and accessible on all fabric-zoned controller paths in the event of any SC Series storage controller failover or other path failure events.

4. Mount the Linux installation DVD/ISO and reboot the system.
5. During the Linux installation process, this SC Series volume is identified and selected as the installation target.

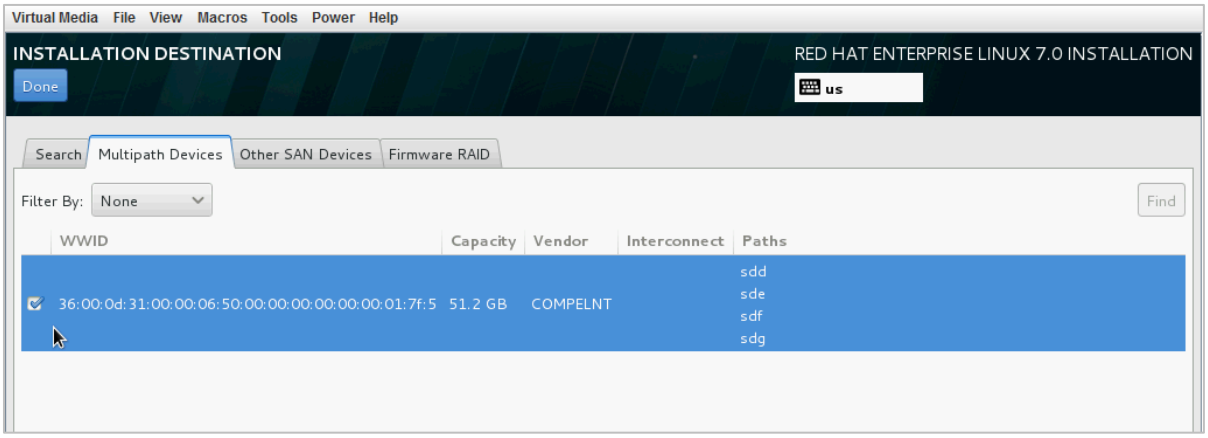
6. In the Installation main menu, click **Installation Destination**.



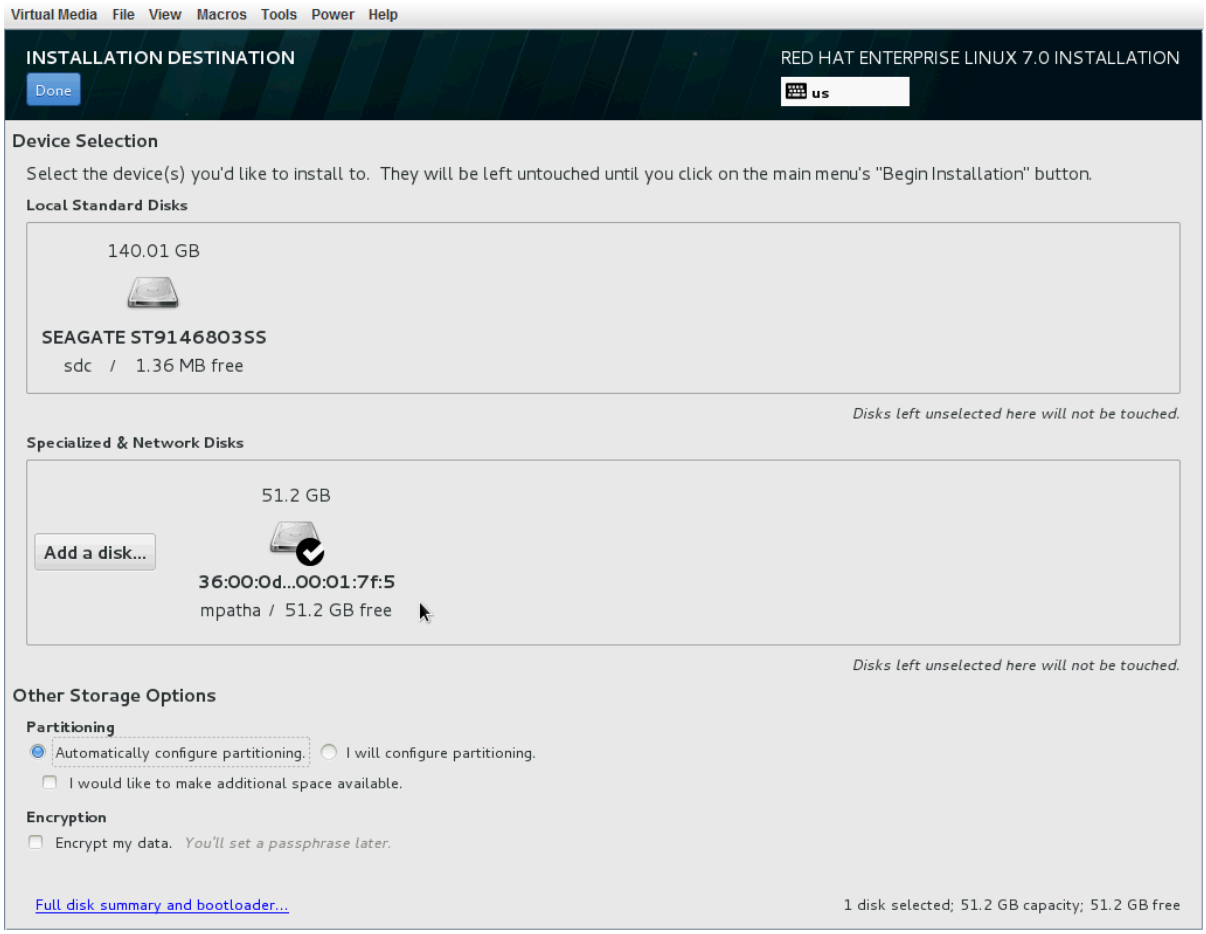
7. Click **Add a disk**.



8. Identify the SC Series volume in the **Multipath Devices** tab and select the check box accordingly.



9. Click **Done**. The SC Series volume shown is identified and selected as the single installation target for the Linux operating system.



10. Click **Done** to return to the installation main menu.
11. Complete the installation and then reboot. Ensure that the **multipathd** service is started during boot time. See section 3.6 for enabling **dm-multipath.**

On RHEL 6 systems, **dm-multipath** might not get installed and configured during the OS installation. This prevents **dm-multipath** to capture and manage the Boot-From-SAN volume after the OS is installed. To correct this:

1. Install and configure **device-mapper-multipath** package. See section 3.6.
2. Rebuild **initramfs** image with multipath.

   ```
   # dracut –v –f –a multipath –include /etc/multipath /etc/multipath
   ```

3. Reboot the system.
4. **multipath –ll** now shows the boot-from-SAN volume. The **pvs** command shows the multipath device instead of the **sdX** device for the OS volume group.

3.12 Snapshots

A snapshot is a point-in-time copy of a volume. Taking a snapshot on a volume makes the pages on the volume read-only. Any changes to the existing page (data) get written to new pages. Unlike other storage systems, snapshots do not require any space reservation on the SC Series system. It provides a fast and space-efficient way to protect data and applications on Linux systems.

3.12.1 Snapshot guidelines

When using snapshots, consider the following guidelines:

- All volumes of an application or database must be protected as a set using the consistency group feature. The consistency group will ensure that the snapshot is taken at the exact same time on all volumes in that group.
- Snapshots do not replace application-level backups. However, they can be used in conjunction with other backup solutions to offer additional protection to the data. They also can offload backup processing to an alternate system. For example, an administrator can take snapshots on the primary system, create view volumes from the snapshots, and mount them on an alternate system to run backup jobs or other resource-intensive processing.
- Snapshots can be taken on demand manually or automatically based on a single or multiple schedules.
- To ensure data integrity on the snapshots, quiesce the application to achieve a consistent state before taking the snapshot. See the following example to quiesce an xfs filesystem:

  ```
  # xfs_freeze –f /          #Freeze a xfs system mounted on /
  # xfs_freeze –u /          #Unfreeze the / file system
  ```

  For a database application such as Oracle, it is recommended to put the database in hot backup mode before taking a snapshot and end backup mode after the snapshot is taken. Consult the corresponding application documentation for details, or the Dell EMC SC Series Arrays and Oracle best practices document.

- Set an expiration policy on snapshots that meets the recovery point objective. Expired snapshots are deleted automatically to reclaim space. However, the snapshots are protected from deletion if there are existing view volumes created from them. These expired snapshots are retained until all the associated view volumes are deleted.
- Applications can access the point-in-time snapshot data only via view volumes. A view volume is a volume created from an existing snapshot.

- When view volumes are first created, they consume no storage because they share the same blocks as their parent snapshot at the beginning. As new data is written or changes are made to the existing data, new data blocks are allocated and tracked separately from the parent. Changes made to the view volumes do not affect the data on the parent volumes.
- Similar to regular volumes, many of the attributes, such as snapshot profiles, replications, data reduction profile, and QoS profiles, are also available to view volumes.
- At the time the view volumes are created, the data on them are exactly the same as the parent volumes. This includes any filesystem labels, volume partitions, UUIDs, and LVM metadata. However, they have different SC Series volume serial IDs, LUN IDs, and WWNs.
- View volumes are read-writeable by default but can be restricted to read-only if desired.

3.12.2 Snapshot use cases

Snapshots (and view volumes) are suitable for various uses:

- Create full-size development and test environments from production
- Test new code, patches, or data changes in a production replica
- Offload backup and restore processing
- Enable fast application/data protection before major changes happen and gain the ability to quickly roll back changes if necessary

3.12.3 Restoring data from a snapshot

Once snapshots are taken, data on these snapshots can be accessed through the view volumes. The following subsections outline several ways to restore the data back to the Linux system.

3.12.3.1 Replace parent volumes with view volumes

In this scenario, the current data is replaced with a point-in-time copy of the data.

1. Create view volumes from the desired snapshots in the DSM GUI.
2. If applicable, stop the applications, unmount the filesystems, and export the LVM volume groups associated with the parent volumes.
3. Remove and unmap the parent volumes from the Linux system. See section 3.10.
4. Map the view volumes to the Linux system.
5. Scan for the view volumes.
6. Since the view volumes contain the exact same metadata, including the labels, UUIDs, and LVM names, there are no further changes needed. Simply activate the LVM volume group and mount the filesystem.
7. Validate the restored data and start the application.
8. To roll back the process in the event of an issue, unmap the view volumes and remap the parent volumes.
9. Do not delete the parent volumes until the restore is successfully and fully validated.

3.12.3.2 Mount view volumes on an alternate system to extract data

In this scenario, an alternate system that has a similar configuration as the parent system is used to access the data on the view volumes.

1. Create view volumes from the desired snapshots in the DSM GUI.
2. Map the view volumes to an alternate Linux system.
3. Scan for the view volumes.
4. Import the LVM volume group and mount the filesystem.

5. Copy/export the data from the alternate system and import the data back to the parent system.
6. Validate the restored data.

3.12.3.3 Mount view volumes on the same system as the parent volumes to extract data

The view volumes contain exactly the same data as the parent volumes including any filesystem labels, disk partitions, UUIDs, LVM metadata, and other application metadata (such as Oracle ASM labels). It is recommended to mount the view volumes on an alternate system to avoid mixing up the parent volumes and the view volumes. If the view volumes must be mounted along with the parent volumes on the same system, additional steps must be taken to modify the metadata on the view volumes to avoid volume conflicts and duplicates.

1. Create view volumes from the desired snapshots in the DSM GUI.
2. Capture information of parent volumes (includes device files, WWN, UUID, labels, multipath, and LVM information) on the parent system before mapping view volumes to the same system.
   - multipath –ll
   - vgdisplay -v $VGNAME
   - lsscsi –is
   - blkid
   - lsblk -aso name,min-io,opt-io,phy-sec,log-sec,sched,disc-max,serial
   - scsi_id
3. Map the view volumes to the parent system.
4. Scan for the view volumes.
5. Identify the view volumes on the parent system and modify the metadata on the view volumes.

Note: **/dev/mapper/mpathX** represents the view volume multipath device file.

   a. Change LVM volume group name and the UUIDs for the physical volumes and volume groups.

```
# vgimportclone --basevgname $NEW_VGNAME /dev/mapper/mpathX
```

   b. Change the filesystem label if applicable

```
# tune2fs -L $NEW_LABEL /dev/mapper/mpathX (for ext4)
# xfs_admin -L $NEW_LABEL /dev/mapper/mpathX (for xfs)
```

   c. Change the filesystem UUID.

```
# tune2fs –U random /dev/mapper/mpathX (for ext4)
# xfs_admin –U generate /dev/mapper/mpathX (for xfs)
```

6. Mount the filesystems on alternate mount points.
7. Copy/export the data from the view volumes and import the data back to the parent volumes.
8. Validate the restored data.
9. Remove and unmap the view volumes after the restore is successful and validated.

3.12.4 Recovering from a boot-from-SAN snapshot view volume

A snapshot view volume of a boot-from-SAN volume, when presented back to the same Linux system, is automatically bootable as long as the boot partition and volume are referenced by their UUIDs and LVM names inside of **/etc/fstab**, **/boot/grub2/grub.cfg** or **/boot/grub/grub.conf.** Additionally, the entries and

aliases within **/etc/multipath/bindings** and **/etc/multipath/wwids** are also automatically updated to reflect any boot device changes.

Note: Ensure that the original boot-from-SAN volume is no longer mapped to the server object and the snapshot view volume is mapped to the Linux system as **LUN ID 0**.

If the OS fails to boot, the reason might be that old **sdX** devices or old WWIDs are still being referenced. Therefore, additional steps must be taken to allow the snapshot view volume to boot. The following procedure demonstrates a method to examine and modify the configuration files on the snapshot view volume on a RHEL 6 system.

1. Create a snapshot view volume from the boot-from-SAN snapshot.
2. Map the snapshot view volume to another Linux system that is able to interpret and mount the base file system type.
3. Mount the /boot and /devices of this snapshot view volume (/dev/sdc in this case).

```
# mkdir /mnt/snapshot-boot
# mkdir /mnt/snapshot-root
# mount /dev/sdc1 /mnt/snapshot-boot
# mount /dev/sdc3 /mnt/snapshot-root
```

4. Determine the WWID of the snapshot view volume. See section 3.1 for details.
5. Update the **/etc/multipath.conf** file to use the new WWID value.

```
# vi /mnt/snapshot-root/etc/multipath.conf
```

6. Review and update the **/mnt/snapshot-root/etc/fstab** file if necessary.
7. Review and update the **/boot/grub/grub.conf** if necessary
8. Map the volume to the original system as LUN ID 0 and boot the system.

Note: The system boot might fail with the **invalid multipath tables** error because of the obsolete WWNs. Continue performing these steps to remove/update them.

9. Mount the OS ISO and boot into rescue mode.
10. Mount the boot-from-SAN volume.

```
# chroot /mnt/sysimage
```

11. Resolve the multipath error by commenting out the old entries and adding new entries in the bindings and wwids files.

```
# vi /mnt/sysimage/etc/multipath/bindings
# vi /mnt/sysimage/etc/multpath/wwids
```

12. Create a new **initramfs** file. Substitute $kernel-release with the correct version of your system.

```
# /mnt/sysimage/sbin/dracut -v /mnt/sysimage/boot/<initramfs-$kernel-
release.img>
```

13. Update **/mnt/sysimage/boot/grub/grub.conf** to use the new **initrd/initramfs** file. Optionally, create a new entry and retain the old entry for a recovery option.
14. Reboot the system using the new initrd/initramfs file. Verify that the dm-multipath is working as normal.

4 Performance tuning

This section provides general information and guidance pertaining to some of the more common performance tuning options and variables available to Linux. This information is not intended to be all-encompassing and the values used should not be considered final. This section provides a starting point for Linux and storage administrators to use when fine-tuning their Linux installation to achieve optimal performance.

Prior to making any changes to the following parameters, establish a good understanding of the current environment and I/O workload. There are numerous methods to accomplish this, including evaluating the perception of the system or storage administrators based on day-to-day experience with supporting the environment. Dell EMC Live Optics™ is a free software that collects and analyzes the various aspects of the environment. For more information, visit https://www.liveoptics.com/.

Dell EMC CloudIQ is a free, software as a service (SaaS) offering that provides continuous monitoring performance, capacity, configuration, and data protection, and enables administrators to manage storage proactively by receiving advanced notification for potential issues. Find additional information in the CloudIQ overview on https://www.dellemc.com/en-us/storage/cloudiq.htm.

Some general guidelines to consider for performance tuning with Linux are as follows:

- Performance tuning is as much an art as it is a science. Since there are a number of variables that impact performance (I/O in particular), there are no specific values that can be recommended for every environment. Begin with a few variables and add more variables or layers as the system is tuned. For example, start with single path, tune, and then add multipath.
- Make one change at a time and then test, measure, and assess the impact on performance with a performance monitoring tool before making any subsequent changes.
- Make sure the original settings are recorded so the changes can be reverted to a known state if needed.
- Apply system tuning principles (such as failover) in a non-production environment first (where able) and validate the changes with as many environmental conditions as possible before propagating these changes into production environments.
- If performance needs are being met with the current configuration settings, it is generally a best practice to leave the settings alone to avoid introducing changes that may make the system less stable.
- Establish an understanding of the differences between block- and file-level data in order to target the tunable settings that will have the most impact on performance. Although the SC Series array is a block-based storage device, the support for the iSCSI transport mechanism introduces performance considerations that are typically associated with network- and file-level tuning.
- When validating whether a change is having an impact on performance, leverage the charting feature of DSM to track the performance. In addition, be sure to make singular changes between iterations in order to better track what variables have the most impact (positive or negative) on I/O performance.

4.1 Tuning profiles

RHEL 7.x introduced a new set of tools to assist administrators and storage professionals with tuning RHEL hosts. This is achieved using two new commands, **tuned** and **tuned-adm,** which manage a set of predefined, performance-tuned profiles.

The following output presents the active tuning profile (default) as well as the list of alternative tuning profiles that can be applied. It is recommended to use the default tuning profile, **throughput-performance,** with any SC Series storage implementation. The discussion of each tuning profile and its merits are outside the scope of this paper, and further discussion of this topic can be found in Red Hat documentation.

Show currently active tuning profile:

```
# tuned-adm active
Current active profile: throughput-performance
```

Show all available tuning profiles:

```
# tuned-adm list
Available profiles:
- balanced
- desktop
- latency-performance
- network-latency
- network-throughput
- powersave
- sap
- throughput-performance
- virtual-guest
- virtual-host
```

Show the contents of tuning profiles:

The contents of the tuning profiles are located in **/usr/lib/tuned/<PROFILE_NAME>/tuned.conf**.

```
# cat /usr/lib/tuned/throughput-performance/tuned.conf
```

Apply a tuned profile:

```
# tuned-adm profile latency-performance
```

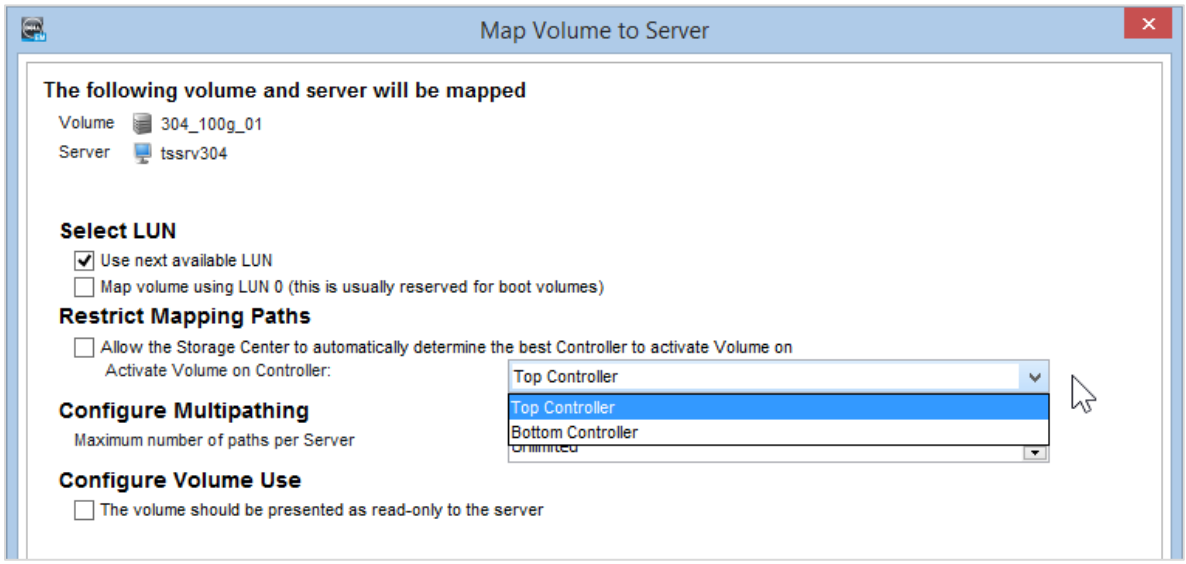
4.2 Using multiple volumes

A volume is only active on one SC Series controller at any one time. Where possible, distribute the volume workload evenly across both SC Series storage controllers to most effectively leverage simultaneous I/O processing. A larger number of smaller-sized volumes will often result in better performance than fewer larger-sized volumes. From a Linux perspective, having multiple target volumes can result in performance improvements by leveraging the kernel to process I/O in parallel to addressing multiple paths and SCSI devices.

In SAS-connected environments, paths from both controllers are presented to the connected host (active/optimized and standby). However, only the active/optimized path is used for all active I/O at any one time. When the active/optimized path becomes unavailable, the SC Series array will dynamically determine which one of the remaining standby paths will assume the role of the active/optimized path and continue to stream active I/O to the new active/optimized path. This is accomplished by explicitly pinning volumes to a different controller when mapping these volumes to the server object. This feature is accessible using the Advanced Options on the mapping dialog shown as follows.

1. Click **Advanced Options**.

2. Uncheck the **Restrict Mapping Paths** checkbox and select the controller in which to pin the volume for mapping to the server object.



4.3 Understanding HBA queue depth

Queue depth refers to the number of pending I/O requests. Modifying this value can lead to an improvement in I/O performance in some workloads. Generally, increasing queue depth can increase throughput, but caution should be taken since increasing this value can also lead to higher latency. Different applications may benefit from increasing this value, such as environments where the bulk of I/O consists of small reads and writes. In environments defined by lower IOPS requirements that need higher throughput, having a lower queue depth setting might be sufficient to meet and achieve the optimal levels of performance.

Consult section 2 for details on modifying this value for the particular HBA model being used.

4.4 SCSI UNMAP/TRIM

Since RHEL 6, the storage stack supports the SCSI UNMAP/TRIM specifications. The application and use of **SCSI UNMAP/TRIM** in filesystem use cases can be very effective towards storage and cost management in the business enterprise. However, consideration should also be made in regard to how this function is implemented.

The **discard mount** parameter would enable the filesystem to perform real-time, on-demand **SCSI UNMAP** commands to the SC Series array. In high-I/O-based application landscapes, this may introduce

unnecessarily high levels of I/O control traffic as well as raised CPU loads, causing an increase in I/O latency and potentially impacting business-critical I/O.

An alternative to this implementation is using the **fstrim** command. The **fstrim** command is part of the **util-linux** package and allows a one-time request to discard used blocks from a mounted filesystem. This command can be scripted (shown in the following sample script), injected into a scheduled cron job, and then applied to a set of mounted filesystems in batch during a time of day that would have less impact to business-critical functions.

```
#!/bin/bash

FSTrim=/usr/bin/fstrim
MntPoints="u01 u02 u03 u04 u05"

for i in ${MntPoints}
do
        echo "INFO: Applying ${FSTrim} to mount point ${i}"
        ${FSTrim} -v /${i}
done
```

4.5 SCSI device queue settings

Several Linux SCSI device queue settings can be applied to tune performance. The more common ones are listed in the following sections with a brief explanation of what each parameter does with regard to I/O. These values are found in the **/sys/block/dm-X/queue** directory (multipath devices) and **/sys/block/sdX/queue** directory (block devices) and should be modified for each path device for the intended multipath volume.

4.5.1 I/O scheduler

The **/sys/block/<device>/queue/schedule** parameter and its contents define the I/O scheduler in use by the Linux kernel for SCSI devices. Some application vendors (such as Oracle) provide specific recommendations for which I/O scheduler to use in order to achieve optimal performance with the application platform. The default scheduler varies on different versions of RHEL. Table 1 summaries the available schedulers and the preferred schedulers to use with SC Series storage system. The scheduler in use is denoted by the [] bracket. It is recommended to use **deadline** or **mq-deadline** scheduler for all SC Series array implementations. The scheduler can be further configured by writing directly to the devices' **sysfs** file (such as **fifo_batch, read_expire**, and **write_expire**) that are discussed in the *Performance Tuning Guide* on the Red Hat Enterprise Linux Document Portal.

Table 1 Preferred I/O schedulers

RHEL 6 and older	RHEL 7	RHEL 8
noop anticipatory [deadline] cfq	noop [deadline] cfq	none [mq-deadline] kyber bfq

Note: RHEL 8 supports a new multi-queue deadline scheduling on block devices. Multi-queue deadline scheduling allows better block layer performance with SSDs and multi-core systems.

Show the current I/O schedule:

- Show all **sd** devices:

```
# egrep "*" /sys/block/sd*/queue/scheduler

/sys/block/sda/queue/scheduler:noop anticipatory deadline [cfq]
/sys/block/sdb/queue/scheduler:noop anticipatory deadline [cfq]
/sys/block/sdc/queue/scheduler:noop anticipatory deadline [cfq]
/sys/block/sdd/queue/scheduler:noop anticipatory deadline [cfq]
/sys/block/sde/queue/scheduler:noop anticipatory deadline [cfq]
```

- Show all **dm-multipath** devices and the associated **sd** devices:

```
# multipath -ll | sed -rn 's/.*(dm-[[:digit:]]+|sd[[:alpha:]]+).*/\1/p'
|xargs -I % echo egrep -H \"*\" /sys/block/%/queue/scheduler | bash

/sys/block/dm-2/queue/scheduler:noop anticipatory deadline [cfq]
/sys/block/sdb/queue/scheduler:noop anticipatory deadline [cfq]
/sys/block/sdd/queue/scheduler:noop anticipatory deadline [cfq]
/sys/block/sdc/queue/scheduler:noop anticipatory deadline [cfq]
/sys/block/sde/queue/scheduler:noop anticipatory deadline [cfq]
```

Dynamically change the I/O schedule by writing to the device's scheduler sysfs file:

The change does not persist after system reboots.

- Change a single **sd** device:

```
# echo deadline > /sys/block/sda/queue/scheduler
```

- Change all **sd** devices:

```
# ls -1 /sys/block/sd[a-z]/queue/scheduler |xargs -I % echo echo deadline
">" %|bash
```

- Change a single **dm-multipath** device and the associated **sd** devices:

```
# multipath -ll $MPATH_NAME | sed -rn 's/.*(dm-
[[:digit:]]+|sd[[:alpha:]]+).*/\1/p' | xargs -I % echo echo deadline ">"
/sys/block/%/queue/scheduler | bash
```

- Change all **dm-multipath** devices and the associated **sd** devices:

  This command is the same as the single dm-multipath option above except the **$MPATH_NAME** argument is removed.

The new scheduler is in effect only for the current running instance of the OS. A script could be used to make this change persistent on all required SCSI (sd) devices (on a device-dependent basis) during boot time. Alternatively, this change can also be applied system-wide during boot time by appending the **elevator=** key value option to the end of the kernel string reflected in the **/boot/grub/grub.conf** (RHEL 6 and older) or **/boot/grub2/grub.cfg** (RHEL 7 and newer) configuration file as shown. Make sure to use the appropriate methods to update the files based on the version of the RHEL.

```
linux16 /vmlinuz-3.10.0-957.el7.x86_64 root=/dev/mapper/rhel-root ro
crashkernel=auto rd.lvm.lv=rhel/root rd.lvm.lv=rhel/swap rhgb quiet
LANG=en_US.UTF-8 elevator=deadline
```

4.5.2 read_ahead_kb

This parameter is used when the kernel detects it is sequentially reading from a block device and defines how many kilobytes of I/O the Linux kernel will read. Increasing this value can have a noticeable effect on performance in heavy sequential read workloads. The parameter defaults to 128 for RHEL 6, and 4096 for RHEL 7 and newer.

Dynamically change the read_ahead_kb setting by writing to the device's read_ahead_kb sysfs file:

The change does not persist after system reboots.

- Change a single **sd** device:

  ```
  # echo 4096 > /sys/block/sda/queue/read_ahead_kb
  ```

- Change all **sd** devices:

  ```
  # ls -1 /sys/block/sd[a-z]/queue/read_ahead_kb |xargs -I % echo echo 4096
  ">" %|bash
  ```

- Change a single **dm-multipath** device and the associated **sd** devices:

  ```
  # multipath -ll $MPATH_NAME | sed -rn 's/.*(dm-
  [[:digit:]]+|sd[[:alpha:]]+).*/\1/p' | xargs -I % echo echo 4096 ">"
  /sys/block/%/queue/read_ahead_kb | bash
  ```

- Change all **dm-multipath** devices and the associated **sd** devices:

  This command is the same as the single dm-multipath command above except the $MPATH_NAME argument is removed.

Change the read_ahead_kb setting using a udev rule:

- Create a **udev** rule in **/etc/udev/rules.d/50-compelnt.rules** with the following entry:

  ```
  ACTION=="add|change", KERNEL=="sd[a-z]", ATTRS{vendor}=="COMPELNT*",
  ATTRS{model}=="Compellent Vol*", ATTR{queue/read_ahead_kb}="4096"
  ```

- Activate the **udev** rule:

  ```
  # udevadm trigger && sleep 3 && systemctl reload multipathd.service >
  /dev/null
  ```

4.5.3 nr_requests

This value is used by the Linux kernel to define the depth of the request queue and is often used in conjunction with changes to the HBA queue depth configuration. The default for RHEL 6 and 7 is 128 while 256 is the new default in RHEL 8. The value can be adjusted using the procedures similar to section 4.5.2.

It is recommended to lower the value for latency-sensitive applications. Consult the Red Hat Performance Tuning Guide on Red Hat Enterprise Linux Document Portal for more information.
https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/

4.5.4 max_sectors_kb

max_sectors_kb determines the number of kilobytes that the block layer will allow and accept according to the filesystem request. This value must remain equal to or smaller than the maximum allowable size dictated by the hardware layer. Typically, **max_sector_kb** is automatically set to the page size in the SC Series storage system which is the recommended setting. An SC Series page size can be 512k, 2M, or 4M.

Show the current max_sectors_kb value:

```
# multipath -ll | sed -rn 's/.*(dm-[[:digit:]]+|sd[[:alpha:]]+).*/\1/p' | xargs
-I % echo egrep -H \"*\" /sys/block/%/queue/max_sectors_kb |bash

/sys/block/dm-2/queue/max_sectors_kb:512
/sys/block/sdb/queue/max_sectors_kb:512
/sys/block/sdd/queue/max_sectors_kb:512
/sys/block/sdc/queue/max_sectors_kb:512
/sys/block/sde/queue/max_sectors_kb:512
```

Show the SC page size in DSM GUI:

In DSM, under the **Storage** tab, select **Storage Types** on the left pane and expand the tree. See Figure 3.

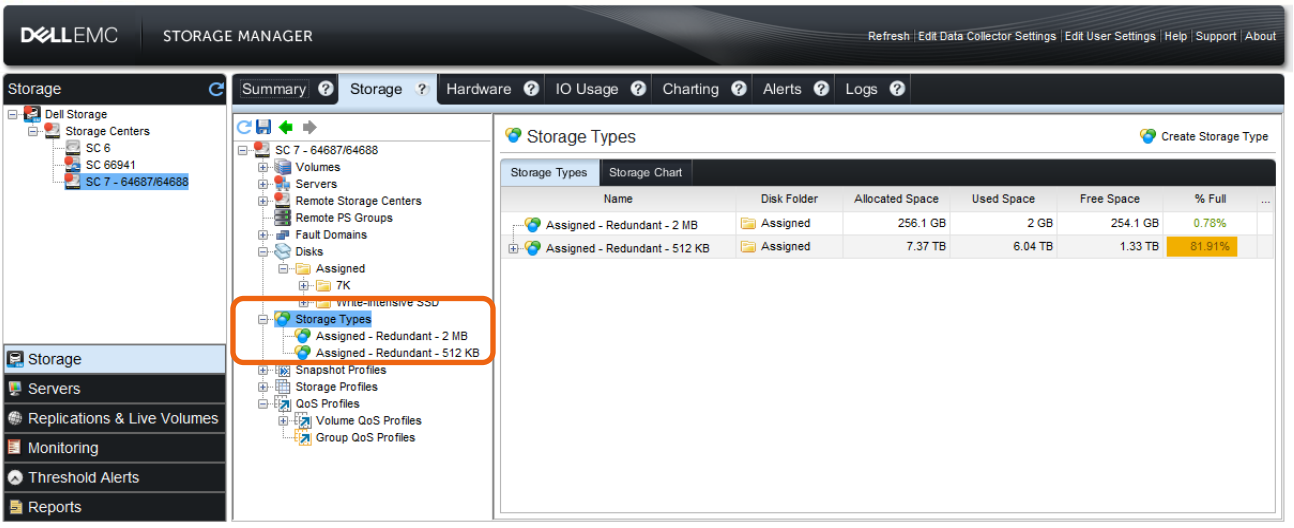


Figure 3 SC Series page sizes

4.6 iSCSI considerations

Tuning performance for iSCSI is as much an effort in Ethernet network tuning as it is block-level tuning. Evaluate the common Ethernet kernel tunable parameters in order to determine the settings that provide the optimal performance gain with iSCSI. The use of Jumbo frames can lead to improved iSCSI performance and is a common recommendation for a large database environment. To use Jumbo frames, all devices on the network path must support setting a large MTU value, typically 9000. SC Series storage allows setting the MTU value on each network interface. Like Fibre Channel, iSCSI changes should be made individually, incrementally, and evaluated against multiple workload types in order to fully understand the effects on overall performance.

In other words, tuning performance for iSCSI is often more time consuming because of the block-level subsystem tuning considerations in addition to network (Ethernet) tuning. A solid understanding of the various Linux subsystem layers involved is necessary to effectively tune the system.

Kernel parameters that can be tuned for performance are found in the **/proc/sys/net/core** and **/proc/sys/net/ipv4** kernel parameters. Once optimal values are determined, permanently set these in the **/etc/sysctl.conf** file. Like most other modern operating system platforms, Linux can efficiently auto-tune TCP buffers. However, by default, some of the settings are conservatively low. Experimenting with the following kernel parameters can lead to improved network performance, and subsequently improve iSCSI performance. For a full list of network tuning parameters, consult the Red Hat Enterprise Linux Network Performance Tuning Guide that contains details information about many parameters and their effects.

Table 2 TCP tuning parameters

Parameters	Description
net.core.rmem_max	Max receive socket buffer size
net.core.wmem_max	Max send socket buffer size
net.ipv4.tcp_rmem	Min, initial, and max TCP receive buffer size
net.ipv4.tcp_wmem	Min, initial, and max buffer space allocated
net.ipv4.tcp_moderate_rcvbuf	Enable/disable auto tuning of the TCP receive buffer size
net.ipv4.tcp_window_scaling	Allows a larger TCP Receive Window

5 Useful tools

The following native Linux tools can be used to identify and correlate volumes to their respective SC Series devices.

5.1 The lsscsi command

The **lsscsi** command is a tool that parses information from the **/proc** and **/sys** pseudo filesystems into human readable output. The **lsscsi** tool is installed using the following command.

```
# yum –y install lsscsi
```

The following **lsscsi** example filters and displays only SC Series LUNs in alphabetically order. The first column displays the **[host:channel:target:lun]** designation for each volume. The host number corresponds to the local HBA **hostX** device file that the volume is mapped to. The channel number is the SCSI bus address and is always zero (0). The target number correlates to the SC Series front-end ports (targets). The LUN number represents the LUN ID of the volume on the SC Series controller where it is mapped.

```
# lsscsi | grep COMPELNT | sort -k7
[snip]
[1:0:0:1]    disk    COMPELNT Compellent Vol   0605  /dev/sdb
[1:0:2:1]    disk    COMPELNT Compellent Vol   0605  /dev/sdc
[1:0:3:1]    disk    COMPELNT Compellent Vol   0605  /dev/sdd
[1:0:3:2]    disk    COMPELNT Compellent Vol   0605  /dev/sde
[1:0:5:1]    disk    COMPELNT Compellent Vol   0605  /dev/sdf
[snip]
```

For RHEL 7 and newer, new options offer additional useful information about the LUNs such as LUN size and WWPN. The WWPN contains the **Serial Number** or **Device ID,** prepended with a 3, of the SC Series volume that can be found in the summary section of the volume in DSM GUI. See Figure 4.

```
# lsscsi –is | grep COMPELNT
[6:0:0:0]    disk    COMPELNT Compellent Vol   0703  /dev/sdb
36000d31000fcaf0000000000000002c0   107GB
[6:0:1:0]    disk    COMPELNT Compellent Vol   0703  /dev/sdc
36000d31000fcaf0000000000000002c0   107GB
 [15:0:0:0]   disk    COMPELNT Compellent Vol   0703  /dev/sdd
36000d31000fcaf0000000000000002c0   107GB
[15:0:1:0]   disk    COMPELNT Compellent Vol   0703  /dev/sde
36000d31000fcaf0000000000000002c0   107GB
```

5.2 /usr/bin/rescan-scsi-bus.sh

The **rescan-scsi-bus.sh** script is installed with the **sg3_util** package. The script provides an easy way to scan, discover, add and remove SCSI devices on the system.

Note: The options vary on different Red Hat Linux versions. Consult the vendor documentation for details.

Scan all targets on the system:

```
# rescan-scsi-bus.sh –a
```

Remove obsolete targets on the system:

```
# rescan-scsi-bus.sh –r
```

Scan for LUN size changes (after increasing it on SC Series storage system) and adjust the LUN size accordingly on the system:

```
# rescan-scsi-bus.sh -s
```

5.3 The scsi_id command

The **scsi_id** command can be applied on each volume one at a time. When there are many volumes on a Linux system, it is easier to query the volumes with a script. The following sample script applies the **scsi_id** command and correlates Linux device names (**/block/sdX** or **/dev/sdX**) to their respective WWID values.

```
#!/bin/bash
# Script name get_WWID.sh

OSMajor=`uname -r |sed -rn 's/^.*(el.).*$/\1/p'`
echo "INFO: OS Major rev. ${OSMajor} detected!"

if [[ "${OSMajor}" =~ el5 ]]; then
   cmd="/lib/udev/scsi_id -g -u -s /block/"
elif [[ "${OSMajor}" =~ (el6|el7|el8) ]]; then
   cmd="/lib/udev/scsi_id --page=0x83 --whitelisted --device=/dev/"
else
   echo "WARN: OSMajor parameter of unknown value, Exiting"
   exit 1
fi

for i in `cat /proc/partitions | awk '/sd/ {print $4}'`
do
   echo "Device: $i WWID: `${cmd}${i}`"
done | sort -k4
```

The script returns the following results.

```
# ./get_WWID.sh
INFO: OS Major Rev. el7 detected!
[snip]
Device: sdd WWID: 36000d31000065000000000000000017f2
Device: sdf WWID: 36000d31000065000000000000000017f2
Device: sdj WWID: 36000d31000065000000000000000017f2
Device: sdl WWID: 36000d31000065000000000000000017f2
[snip]
```

These WWID values in turn, correlate to the Serial Number or the Device ID value of the SC Series storage devices.

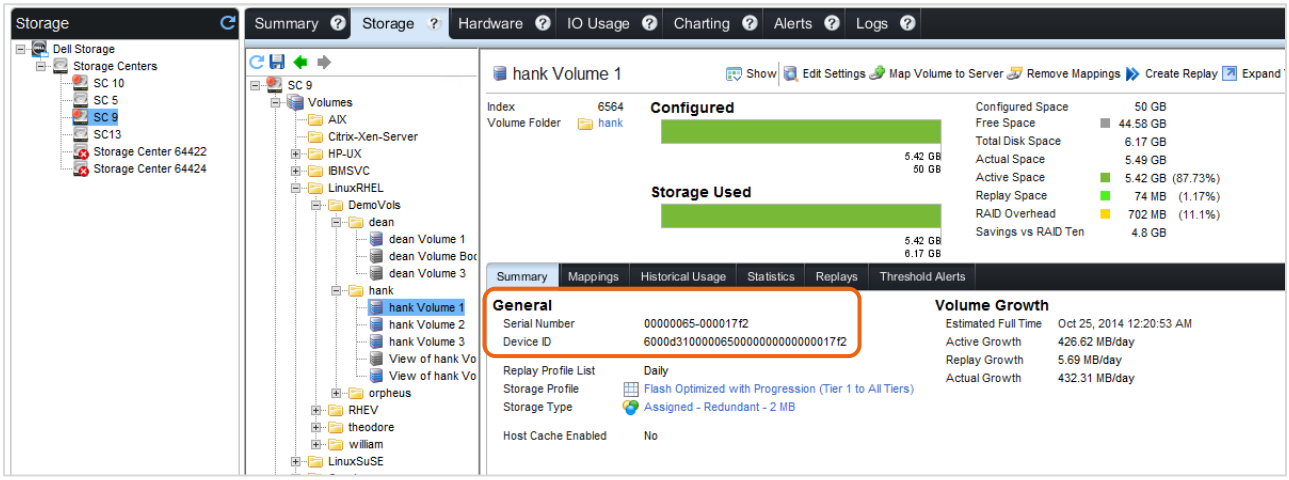


Figure 4 SC Series Volume Serial Number and Device ID fields

5.4 Decoding the WWID

The WWID returned by lsscsi, scsi_id, and multipath commands is a string of 33 characters. The string can be broken down into different parts that identify the vendor, make, serial number and device. The following example shows a breakdown of a SC volume WWID and their translation.

3 **6** 000d31 000fcaf 0000000000 000002d9	Network Address Authority (NAA) format identifier
3 6 **000d31** 000fcaf 0000000000 000002d9	IEEE Organizationally Unique Identifier (OUI) for Compellent
3 6 000d31 **000fcaf** 0000000000 000002d9	SC Series array serial number in hex
3 6 000d31 **000fcaf** 0000000000 **000002d9**	SC Series volume serial number

The list of IEEE OUI is located at http://standards-oui.ieee.org/oui.txt.

Convert SC Series array serial number from hex to decimal:

```
# echo $(( 16#000fcaf ))
64687
```

5.5 The dmesg command

The **dmesg** command is useful for discovering which device names are assigned to recently discovered volumes. The following output demonstrates the discovery and assignment of a new SC Series volume to the **/dev/sdh** device file.

```
# dmesg
[snip]
[1203830.267568] scsi 4:0:0:1: Direct-Access     COMPELNT Compellent Vol   0605
PQ: 0 ANSI: 5
[1203830.267996] sd 4:0:0:1: Attached scsi generic sg8 type 0
[1203830.268089] sd 4:0:0:1: [sdh] 20971520 512-byte logical blocks: (10.7
GB/10.0 GiB)
[1203830.268093] sd 4:0:0:1: [sdh] 4096-byte physical blocks
[1203830.268847] sd 4:0:0:1: [sdh] Write Protect is off
[1203830.268858] sd 4:0:0:1: [sdh] Mode Sense: 8f 00 00 08
[1203830.269033] sd 4:0:0:1: [sdh] Write cache: disabled, read cache: enabled,
doesn't support DPO or FUA
[snip]
```

6 Dell Storage REST API

The Dell Storage REST API interface is available with the installation of Dell Storage Manager (DSM) 2015 R3 or newer. The REST API is recommended when intending to interact with SC Series storage managed by the DSM installation, either programmatically or in a command line interface. For more information, refer to the Dell Storage Manager REST API Cookbook.

The use of the Compellent Command Utility (CompCU) has been deprecated.

A Technical support and resources

Dell.com/support is focused on meeting customer needs with proven services and support.

Storage technical documents and videos provide expertise that helps to ensure customer success on Dell EMC storage platforms.

A.1 Additional resources

- Dell Storage Manager 2016 R3 Administrator's Guide
  https://topics-cdn.dell.com/pdf/storage-sc2000_administrator-guide_en-us.pdf
- Dell Storage Manager 2016 R3 Web UI Administrator's Guide
  https://topics-cdn.dell.com/pdf/storage-sc2000_administrator-guide5_en-us.pdf
- Additional Dell Storage manuals and documents
  https://www.dell.com/support/home/us/en/04/product-support/product/storage-sc5020f/manuals
- Dell EMC Simple Support Matrix
  https://support.emc.com
- PowerPath Installation and Administration Guide
  https://support.emc.com
- Red Hat Enterprise Linux Document Portal
  https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/
- Red Hat Labs
  https://access.redhat.com/labs/
- Red Hat Enterprise Linux Release Dates
  https://access.redhat.com/articles/3078
  https://en.wikipedia.org/wiki/Red_Hat_Enterprise_Linux