



# **XC™ Series Software Installation and Configuration Guide**

**(CLE 6.0.UP07)**

**S-2559 Rev D**

# Contents

1 About XC™ Series Software Installation and Configuration Guide (S-2559).....	8
1.1 Related Publications.....	13
2 Distribution Media.....	14
3 Introduction to Installation and Configuration of Cray XC™ Software.....	15
3.1 About Cray Scalable Services.....	16
3.2 About Service Nodes.....	17
3.3 About Config Sets.....	19
3.4 About Variable Names in the Configurator and Configuration Worksheets.....	20
3.5 About Snapshots and Config Set Backups.....	21
3.6 About Config Set Caching.....	22
3.7 About Node Groups.....	23
3.8 About Simple Sync.....	27
3.9 About Secure Shell Configuration.....	32
3.10 About Boot Automation Files.....	34
3.11 About the Admin Image.....	36
3.12 Where to Place the Root File System: tmpfs versus netroot.....	36
3.13 About Image Groups and How to Customize Them.....	38
3.14 About Parallel Image Creation.....	40
3.15 About Image Pushes: push versus sqpush.....	42
3.16 About the /etc/hosts File.....	43
4 Install and Configure SMW/CLE Software.....	46
4.1 Prepare for an SMW/CLE Fresh Install.....	46
4.1.1 Information to Collect Before Installation.....	46
4.1.2 Network Connections.....	49
4.1.3 SMW Internal Disk Requirements.....	49
4.1.4 Configuration Values.....	50
4.1.5 Passwords.....	51
4.2 Install the Base Operating System on the SMW.....	52
4.2.1 Prepare to Install the Base Linux Distribution.....	52
4.2.2 Install the SLES 12 SP3 Base Linux Distribution on the SMW.....	78
4.2.3 Configure Boot RAID Devices.....	83
4.2.4 Make a Snapshot Manually.....	103
4.3 Install the SMW and CLE Software.....	104
4.3.1 Start a Typescript File.....	105
4.3.2 Collect Software Media.....	105

---

4.3.3 Mount Software Media and Prepare install.cle.conf during a Fresh Install.....	106
4.3.4 Determine the Persistent Device Name for a LUN.....	108
4.3.5 RAID Disk Space Requirements.....	111
4.3.6 Bootstrap the SMW Installation.....	112
4.3.7 Provision SMW Storage.....	120
4.3.8 Run the Installer for an Initial Installation.....	121
4.3.9 Set Default Snapshot and Boot the SMW.....	123
4.4 Configure SMW for CLE System Hardware during a Fresh Install.....	124
4.4.1 Set or Change the HSS Data Store (MariaDB) Root Password.....	125
4.4.2 Start a Typescript File.....	127
4.4.3 Make a Post-install Snapshot using snaputil.....	127
4.4.4 Make a Post-install Backup of Current Global and CLE Config Sets.....	128
4.4.5 Update install.cle.conf for Software Updates.....	128
4.4.6 Prepare and Update the Global Config Set.....	129
4.4.7 Prepare the CLE Configuration Worksheets.....	140
4.4.8 Bootstrap Hardware Discovery.....	141
4.4.9 Discover Hardware and HSN Routing, Prepare STONITH .....	143
4.4.10 Update Firmware.....	145
4.4.11 (Optional) Configure Partitions.....	146
4.4.12 Repurpose a Compute or Service Node.....	147
4.4.13 Finish Configuring the SMW for the CLE System Hardware.....	148
4.5 Configure CLE.....	149
4.5.1 Update CLE Configuration Worksheets.....	150
4.5.2 Create New CLE Config Set from Worksheets.....	210
4.5.3 Update Unconfigured CLE Configuration Services.....	211
4.5.4 Change Group Ownership or Permissions for Log Files and Directories in the CLE Config Set.....	217
4.5.5 Add or Migrate Site Data to /etc/hosts File.....	219
4.5.6 Update CLE Config Set after a Fresh Install.....	220
4.5.7 Check CLE Host Names in /etc/hosts File.....	222
4.5.8 Set Up Advanced RSIP Configuration Before Booting the System.....	223
4.5.9 Update /etc/motd for Nodes.....	227
4.5.10 Copy Files for External Lustre Fine-grained Routing.....	228
4.5.11 Configure Files for Cray Simple Sync Service.....	228
4.5.12 Display and Capture all Config Set Information.....	229
4.5.13 Validate Config Sets.....	230
4.5.14 Make a Post-config Snapshot using snaputil.....	231
4.5.15 Make a Post-config Backup of Current Global and CLE Config Sets.....	231

---

4.6 Prepare Boot Images and Boot the CLE System during a Fresh Install.....	232
4.6.1 Create a NIMS Map.....	232
4.6.2 Build Boot Images for a Fresh Install.....	233
4.6.3 Configure Boot and/or SDB Node Failover.....	239
4.6.4 Set the Turbo Boost Limit.....	244
4.6.5 Check NIMS Information during a Fresh Install.....	245
4.6.6 Boot the System using a Boot Automation File.....	246
4.6.7 Check Cabinet Cooling Parameters for an Air-Cooled XC System.....	249
4.6.8 Run Tests after Boot is Complete.....	250
4.6.9 Prepare Site and Software Revision Information Reporting using xtgetrev and xtshowrev.....	251
4.6.10 Test xtdumpsys and cdump.....	252
4.6.11 Make a Post-boot Snapshot using snaputil.....	254
4.6.12 Make a Post-boot Backup of Current Global and CLE Config Sets.....	255
4.7 Configure Other Features and Services.....	255
4.7.1 Configure Power Management Disk.....	256
4.7.2 Push Diag Image to Boot Node and Update the Diags Bind Mount Profile.....	260
4.7.3 Configure Netroot.....	262
4.7.4 Configure the SEC and check_xt Monitoring and Notification Utilities.....	268
4.7.5 Configure Direct-attached Lustre (DAL).....	269
4.7.6 LMT Configuration for DAL.....	277
4.7.7 Reduce Impact of Btrfs Periodic Maintenance on SMW Performance .....	282
4.7.8 Configure Cray NAT Masquerading Service.....	283
4.7.9 Prevent Unintentional Re-creation of Mail Configuration Files.....	291
4.7.10 About System Environmental Data Collections (SEDC).....	292
4.8 Install Additional Software.....	292
4.8.1 Install the Dell Systems Management Tools and Documentation DVD.....	292
4.8.2 Install and Configure DataWarp.....	293
4.8.3 Install Cray Programming Environment (PE) Software.....	294
4.8.4 Install and Configure a Workload Manager (WLM).....	306
4.8.5 Install and Configure eLogin.....	306
5 Update SMW/CLE Software.....	308
5.1 Prepare for an SMW/CLE Software Update.....	308
5.1.1 Start a Typescript File.....	309
5.1.2 Show Current HSS Partition and Back Up PMDB Configuration.....	310
5.1.3 Prepare for a DataWarp Software Update.....	310
5.1.4 Prepare for an eLogin Software Update.....	311
5.1.5 Set Variable for Release Snapshot Name.....	312
5.1.6 Make a Pre-update Release Snapshot using snaputil.....	312

---

5.1.7 Make a Pre-update Backup of Current Global and CLE Config Sets.....	313
5.1.8 Clean Up SMW Disk Space Before a Software Update.....	313
5.1.9 Rename Existing Cray Image Groups File.....	316
5.1.10 Collect Software Media.....	317
5.2 Install the SMW and CLE Software Update.....	318
5.2.1 Mount Software Media and Prepare install.cle.conf during a Software Update.....	318
5.2.2 Make a Release Snapshot using snaputil.....	321
5.2.3 Install the SLES, SMW, and CLE Software.....	321
5.2.4 Shut Down CLE and Reboot SMW to Release Snapshot.....	322
5.3 Build Boot Images.....	323
5.3.1 Start a Typescript File.....	324
5.3.2 Make a Post-install Snapshot using snaputil.....	325
5.3.3 Make a Post-install Backup of Current Global and CLE Config Sets.....	325
5.3.4 Update Image Groups and Recipes.....	326
5.3.5 Build Boot Images for a Software Update.....	329
5.4 Configure the Software Update.....	330
5.4.1 Check for New Config Set Default Values after a Software Update.....	331
5.4.2 Merge Installed Configuration Template Content.....	333
5.4.3 Configure Fields that were New or Changed in CLE 6.0.UP06.....	334
5.4.4 Configure Fields that are New or Changed in CLE 6.0.UP07.....	339
5.4.5 Update Diags Bind Mount Profile.....	340
5.4.6 Restore Compute Node Volume Group to cray_bootraid.....	341
5.4.7 Remove Xeon Phi Plugins from NHC Configuration Service.....	344
5.4.8 Update and Validate the CLE and Global Config Sets after a Software Update.....	345
5.4.9 Display All Config Set Information.....	346
5.4.10 Make a Post-config Snapshot using snaputil.....	347
5.4.11 Make a Post-config Backup of Current Global and CLE Config Sets.....	348
5.5 Update Cray Programming Environment (PE) Software on x86-64.....	348
5.6 Configure SMW for CLE System Hardware during a Software Update.....	352
5.6.1 Start a Typescript File.....	353
5.6.2 Discover Cray Hardware.....	354
5.6.3 Update Firmware.....	356
5.6.4 Update Config Sets.....	358
5.6.5 Validate Config Sets.....	358
5.6.6 Restore PMDB Customizations after a Software Update.....	359
5.6.7 Finish Configuring the SMW for the CLE System Hardware.....	360
5.7 Install Patches.....	361
5.8 Boot the CLE System during a Software Update.....	362

---

5.8.1 Check NIMS Information during a Software Update.....	363
5.8.2 Update Site Automation Files for System Boot and Shutdown.....	365
5.8.3 Push New Images and Boot the CLE System.....	365
5.8.4 Check Cabinet Cooling Parameters for an Air-Cooled XC System.....	367
5.8.5 Run Tests after Boot is Complete.....	368
5.8.6 Test xtdumpsys and cdump.....	369
5.8.7 Make a Post-boot Snapshot for a Software Update.....	371
5.8.8 Make a Post-boot Backup of Current Global and CLE Config Sets.....	372
5.9 Update Other Features after an SMW/CLE Software Update.....	373
6 Customize Preinstalled SMW/CLE Software.....	375
6.1 Update Site Information and Install Needed Patches.....	376
6.2 Change the Default System Management Workstation (SMW) Passwords.....	378
6.3 Change the Time Zone.....	379
6.4 Configure the SMW Firewall.....	382
6.5 Configure LAN on the SMW.....	383
6.6 Change Networks, IP Addresses in Global Config Set.....	383
6.7 Change Networks and IP Addresses in CLE Config Set.....	385
6.8 Set Up iDRAC for a Dell R630 or R640 SMW.....	388
6.9 Set Up iDRAC for a Dell R815 SMW.....	391
6.10 Change the Default iDRAC Password.....	395
6.11 Enable Write Cache on SMW Boot RAID Volume.....	395
6.12 Configure the SEC and check_xt Monitoring and Notification Utilities.....	396
6.13 Configure Site Lightweight Log Manager (LLM).....	397
6.14 Prevent Unintentional Re-creation of Mail Configuration Files.....	399
6.15 Check Cabinet Cooling Parameters for an Air-Cooled XC System.....	400
6.16 Make a Post-customize Snapshot using snaputil.....	400
6.17 Make a Post-customize Backup of Current Global and CLE Config Sets.....	401
7 Troubleshoot SMW/CLE Software Installation.....	403
8 Miscellaneous Installation and Configuration Procedures.....	404
8.1 Back Up Site Data.....	404
8.2 Back Up Current Global and CLE Config Sets.....	406
8.3 Back Up or Restore User, Group, and Permissions Information Files.....	406
8.4 Set Default Config Set for a NIMS Map.....	408
8.5 Set Config Set for a Node.....	409
8.6 Rename a NIMS Map.....	409
8.7 Modify a Config Set for Use with Advanced Authentication Configurations.....	410
8.8 Install Third-Party Software with a Custom Image Recipe.....	417
8.9 Remove an Undesired RPM After Building From a Cray Recipe.....	425



---

8.10 Enable Multipath on an Installed XC System.....	427
8.11 Change the Time Zone.....	433
8.12 Initialize zsh for Non-interactive Jobs.....	437
8.13 Prepare Site and Software Revision Information Reporting using xtgetrev and xtshowrev.....	437
8.14 Shut Down the CLE System.....	439
9 Checklists for XC™ Series Software Installation.....	440
9.1 Master Checklist: Install and Configure New SMW/CLE Software.....	440
9.2 Installation Checklist 1: Install the Base Operating System on the SMW.....	440
9.3 Installation Checklist 2: Install the SMW and CLE Software.....	441
9.4 Installation Checklist 3: Configure SMW for CLE Hardware during a Fresh Install.....	442
9.5 Installation Checklist 4: Configure CLE.....	443
9.6 Installation Checklist 5: Update CLE Configuration Services.....	444
9.7 Installation Checklist 6: Prepare Boot Images and Boot the CLE System during a Fresh Install.....	445
9.8 Installation Checklist 7: Configure Other Features and Services.....	445
9.9 Installation Checklist 8: Install Additional Software.....	447
9.10 Installation Checklist 9: Customize Preinstalled SMW/CLE Software.....	447

# 1 About XC™ Series Software Installation and Configuration Guide (S-2559)

---

## Scope and Audience

The *XC™ Series Software Installation and Configuration Guide (S-2559)* provides overview information and detailed procedures to install, update, and customize software on a Cray XC™ Series system, including Cray System Management Workstation (SMW) software, Cray Linux Environment (CLE) software, and the SMW base operating system, SUSE® Linux Enterprise Server version 12 SP3 (SLES® 12 SP3).

This publication does not include procedures for administering a Cray XC Series system; for those, see *XC™ Series System Administration Guide (S-2393)*.

This publication is intended for system installers, administrators, and anyone who installs and configures software on a Cray XC™ Series system. It assumes some familiarity with standard Linux and open source tools (e.g., zypper/yum for RPMs, Ansible, YAML/JSON configuration data).

## CLE 6.0.UP07 / SMW 8.0.UP07 Release

*XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP07) S-2559 Rev D* supports Cray software release CLE 6.0.UP07 / SMW 8.0.UP07 for Cray XC™ Series systems, released on 02 Jan 2019.

### New in Revision D

- Updated compiler information in the PE procedures for installing on AArch64 nodes. See [Install Cray Programming Environment \(PE\) Software on AArch64](#) on page 300 for more information.

### New in Revision C

- Installation support for Dell R640 SMWs. A new `slebase` and patch ISO are required when performing a fresh installation. A new procedure is required for configuring RAID virtual disks when using Dell R640 hardware. See [Dell R640 SMW: Configure the RAID Virtual Disks](#) on page 64 for more information.

### New in Revision B

- Improved PE procedures that describe how to install on AArch64 nodes. See [Install Cray Programming Environment \(PE\) Software on AArch64](#) on page 300.

### New in Revision A

- Errors have been corrected in [Reduce Impact of Btrfs Periodic Maintenance on SMW Performance](#) on page 282.
- Minor updates to clarify the PE procedures for installing on AArch64 nodes. See [Install Cray Programming Environment \(PE\) Software on AArch64](#) on page 300.
- Various corrections and editorial changes were made.

### Changes Published in the CLE 6.0.UP07 / SMW 8.0.UP07 Release



- A new configuration service, Cray NAT (network address translation) Masquerading (`cray_nat_masq`), defines a gateway node to serve as the default gateway for one or more client nodes, and it provides the option to configure `SuSEfirewall12` to permit network traffic from a client node to a network outside the Cray high-speed network (HSN). It is typically used for AArch64 login nodes. See [Configure Cray NAT Masquerading Service](#) on page 283.
- New PE procedures describe how to install on AArch64 nodes. See the following for an overview of changes: [Install Cray Programming Environment \(PE\) Software](#) on page 294
- A new topic describes how images can be built in parallel by the installer or on the command line. See [About Parallel Image Creation](#) on page 40. Example commands are provided in the appropriate places in fresh install and software update procedures.
- A new procedure backs up and restores user, group, and permissions information contained in config set directories. See [Back Up or Restore User, Group, and Permissions Information Files](#) on page 406.
- CLE repository, package collection, recipe, and image names now have a "dot" between 6.0 and `upxx` in the CLE version. For example, all instances of 6.0`up07` in CLE repo/package/recipe/image names have changed to 6.0.`up07`. This enables the use of variable substitution for the CLE release name in recipe and image names in `cray_image_groups.yaml`. Instructions to update `cray_image_groups.yaml` and all custom repos, package collections, and recipes that reference CLE repos, package collections, and recipes are included in the appropriate places in fresh install and software update procedures.
- New architecture-specific image group stanzas and NIMS groups:
  - A description of the architecture field and new architecture-specific stanzas for `cray_image_groups.yaml` are included in [About Image Groups and How to Customize Them](#) on page 38.
  - Instructions for creating architecture-specific NIMS groups are included in [Build Boot Images for a Fresh Install](#) on page 233 and [Configure Netroot Images](#) on page 262.
- A new step in [Prepare and Update the Global Config Set](#) on page 129 adds another host definition for the second SMW of an SMW HA pair, which is necessary for SMW-managed eLogin in an SMW HA system.
- In the software update process, image building now occurs before the software update is configured. This enables administrators to begin building images in one window, and then in a different window, work through the configuration procedures while images are building.
- Procedures to make a snapshot and back up current config sets have been added or moved:
  - A post-install backup procedure has been added to the fresh install process.
  - Post-install snapshot and backup procedures in the software update process have been moved to just after the SMW has been rebooted to the release snapshot.
  - Post-config snapshot and backup procedures have been added to the software update process.
- References to `xtpmdbconfig` have been removed from [Show Current HSS Partition and Back Up PMDB Configuration](#) on page 310 and [Restore PMDB Customizations after a Software Update](#) on page 359.

The `xtpmdbconfig` utility was used to configure partitions in the power management database, but because it was no longer needed, it was removed in CLE 6.0.UP06. For more information, see "About the XC™ Series Power Management and SEDC Administration Guide" in *XC™ Series Power Management and SEDC Administration Guide (CLE 6.0.UP06) S-0043*.
- If the `cmap create` command reports an error that the default config set is missing, use the workaround documented in [Build Boot Images for a Software Update](#) on page 329 or [Rename a NIMS Map](#) on page 409.

- Errors have been corrected in [Set Up Advanced RSIP Configuration Before Booting the System](#) on page 223.
- Various corrections and editorial changes were made.

Table 1. Record of Revision

Publication Title	Date	Release
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP07) S-2559 Rev D	02 Jan 2019	CLE 6.0.UP07 / SMW 8.0.UP07
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP07) S-2559 Rev C	04 Dec 2018	CLE 6.0.UP07 / SMW 8.0.UP07
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP07) S-2559 Rev B	09 Nov 2018	CLE 6.0.UP07 / SMW 8.0.UP07
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP07) S-2559 Rev A	18 Oct 2018	CLE 6.0.UP07 / SMW 8.0.UP07
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP07) S-2559	12 Jul 2018	CLE 6.0.UP07 / SMW 8.0.UP07
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP06) S-2559 Rev B	27 Mar 2018	CLE 6.0.UP06 / SMW 8.0.UP06
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP06) S-2559 Rev A	12 Mar 2018	CLE 6.0.UP06 / SMW 8.0.UP06
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP06) S-2559	01 Mar 2018	CLE 6.0.UP06 / SMW 8.0.UP06
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP05) S-2559	05 Oct 2017	CLE 6.0.UP05 / SMW 8.0.UP05
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP04) S-2559 Rev B	10 Jul 2017	CLE 6.0.UP04 / SMW 8.0.UP04
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP04) S-2559 Rev A	05 Jul 2017	CLE 6.0.UP04 / SMW 8.0.UP04
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP04) S-2559	22 Jun 2017	CLE 6.0.UP04 / SMW 8.0.UP04
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP03) S-2559 Rev B	04 May 2017	CLE 6.0.UP03 / SMW 8.0.UP03
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP03) S-2559 Rev A	03 Apr 2017	CLE 6.0.UP03 / SMW 8.0.UP03
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP03) S-2559	16 Feb 2017	CLE 6.0.UP03 / SMW 8.0.UP03
XC™ Series Software Installation and Configuration Guide (CLE 6.0.UP02) S-2559 Note title change.	03 Nov 2016	CLE 6.0.UP02 / SMW 8.0.UP02

Publication Title	Date	Release
<i>XC™ Series Software Initial Installation and Configuration Guide (CLE 6.0.UP01) S-2559 Rev A</i>	26 Aug 2016	CLE 6.0.UP01 / SMW 8.0.UP01
<i>XC™ Series Software Initial Installation and Configuration Guide (CLE 6.0.UP01) S-2559</i> Note that S-2559 combines SMW and CLE installation and supersedes both S-2480 and S-2444.	20 Jun 2016	CLE 6.0.UP01 / SMW 8.0.UP01
<i>CLE Installation and Configuration Guide S-2444-5204</i>	24 Sep 2015	CLE 5.2.UP04
<i>System Management Workstation (SMW) Software Installation Guide S-2480-7204a</i>	30 Oct 2015	SMW 7.2.UP04
<i>System Management Workstation (SMW) Software Installation Guide S-2480-7204</i>	24 Sep 2015	SMW 7.2.UP04

## Command Prompt Conventions

**Host name and account in command prompts** The host name in a command prompt indicates where the command must be run. The account that must run the command is also indicated in the prompt.

- The `root` or super-user account always has the `#` character at the end of the prompt.
- Any non-`root` account is indicated with `account@hostname>`. A user account that is neither `root` nor `crayadm` is referred to as `user`.

<code>smw#</code>	Run the command on the SMW as <code>root</code> .
<code>cmc#</code>	Run the command on the CMC as <code>root</code> .
<code>sdb#</code>	Run the command on the SDB node as <code>root</code> .
<code>crayadm@boot&gt;</code>	Run the command on the boot node as the <code>crayadm</code> user.
<code>user@login&gt;</code>	Run the command on any login node as any non- <code>root</code> user.
<code>hostname#</code>	Run the command on the specified system as <code>root</code> .
<code>user@hostname&gt;</code>	Run the command on the specified system as any non- <code>root</code> user.
<code>smw1#</code> <code>smw2#</code>	For a system configured with the SMW failover feature there are two SMWs—one in an active role and the other in a passive role. The SMW that is active at the start of a procedure is <code>smw1</code> . The SMW that is passive is <code>smw2</code> .

```
smwactive#
smwpassive#
```

In some scenarios, the active SMW is smw1 at the start of a procedure—then the procedure requires a failover to the other SMW. In this case, the documentation will continue to refer to the formerly active SMW as smw1, even though smw2 is now the active SMW. If further clarification is needed in a procedure, the active SMW will be called smwactive and the passive SMW will be called smwpassive.

#### Command prompt inside chroot

If the `chroot` command is used, the prompt changes to indicate that it is inside a chroot environment on the system.

```
smw# chroot /path/to/chroot
chroot-smw#
```

#### Directory path in command prompt

Example prompts do not include the directory path, because long paths can reduce the clarity of examples. Most of the time, the command can be executed from any directory. When it matters which directory the command is invoked within, the `cd` command is used to change into the directory, and the directory is referenced with a period (.) to indicate the current directory.

For example, here are actual prompts as they appear on the system:

```
smw:~ # cd /etc
smw:/etc# cd /var/tmp
smw:/var/tmp# ls ./file
smw:/var/tmp# su - crayadm
crayadm@smw:~> cd /usr/bin
crayadm@smw:/usr/bin> ./command
```

And here are the same prompts as they appear in this publication:

```
smw# cd /etc
smw# cd /var/tmp
smw# ls ./file
smw# su - crayadm
crayadm@smw> cd /usr/bin
crayadm@smw> ./command
```

## Typographic Conventions

Monospace	Indicates program code, reserved words, library functions, command-line prompts, screen output, file/path names, and other software constructs.
<b>Monospaced Bold</b>	Indicates commands that must be entered on a command line or in response to an interactive prompt.
<i>Oblique or Italics</i>	Indicates user-supplied values in commands or syntax definitions.
<b>Proportional Bold</b>	Indicates a graphical user interface window or element and key strokes (e.g., <b>Enter</b> , <b>Alt-Ctrl-F</b> ).
\ (backslash)	At the end of a command line, indicates the Linux® shell line continuation character (lines joined by a backslash are parsed as a single line). Do not type anything after the backslash or the continuation feature will not work correctly.

## Trademarks

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, Urika-GX, and YARCDATA. The following are trademarks of Cray Inc.: APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYDOC, CRAYPAT, CRAYPORT, DATAWARP, ECOPHLEX, LIBSCI, NODEKARE. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.

## 1.1 Related Publications

This publication supersedes *System Management Workstation (SMW) Software Installation Guide (S-2480)* and *CLE Installation and Configuration Guide (S-2444)*.

Although this publication is all that is necessary for installing SMW and CLE software, the following publications contain additional information that may be helpful. The release errata and readme files are available on CrayPort, and the rest of these publications (and other Cray publications) can be found at <http://pubs.cray.com>.

- *SMW Release Errata* (includes notice of any patches) and the *SMW README*, which are provided with the SMW release software
- *CLE Release Errata* and the *CLE README*, which are provided with the CLE release software
- *XC™ Series Configurator User Guide (S-2560)*
- *XC™ Series Ansible Play Writing Guide (S-2582)*
- *XC™ Series System Administration Guide (S-2393)*
- *XC™ Series Boot Troubleshooting Guide (S-2565)*
- *XC™ Series Lustre® Administration Guide (S-2648)*
- *XC™ Series Power Management and SEDC Administration Guide (S-0043)*
- *XC™ Series DataWarp™ Installation and Administration Guide (S-2564)*, which supersedes *DataWarp Installation Guide (S-2547)*
- *Cray Compiling Environment Release Overview and Installation Guide*
- For a system that will have eLogin nodes:
  - *XC™ Series SMW-managed eLogin Installation Guide (S-3020)*
  - *XC™ Series SMW-managed eLogin Administration Guide (S-3021)*
- *XC™ Series SEC and check\_xt Software Configuration Guide (S-2542)*, which describes the Cray Simple Event Correlator and the `check_xt` utility
- *XC™ Series Aries™ Network Resiliency Guide (S-0041)*
- For a system that will use DVS for projecting external file systems:
  - *XC™ Series DVS Administration Guide (S-0005)*
  - *XC™ Series GPFS Software Installation Guide (S-2569)*
- For a system that will be configured for SMW high availability (HA):
  - *XC™ Series SMW HA Installation Guide (S-0044)*
  - *XC™ Series SMW HA Administration Guide (S-2551)*

## 2 Distribution Media

The Cray CLE 6.0.UP07 / SMW 8.0.UP07 release distribution media consist of one DVD and several other pieces of media that may be DVDs or ISO files.

Configuration worksheets for CLE config sets and the global config set are also included in this distribution, so that sites can begin entering site-specific configuration data in them before and during the installation process.

This table shows all installation media included with this release.

bootable SMW SLES12 media	<ul style="list-style-type: none"> <li>For Dell R815 and R630: Cray-slebase-12-SP3-201709141039.iso DVD</li> <li>For Dell R640: Cray-slebase-12-SP3-201806150923.iso DVD</li> </ul>
SMW release	<ul style="list-style-type: none"> <li>smw-8.0.7128-201806242300.iso</li> </ul>
CLE release	<ul style="list-style-type: none"> <li>cle-6.0.7128-201806242300.iso</li> </ul>
SLES release	<ul style="list-style-type: none"> <li>SLE-12-SP3-Server-DVD-x86_64-GM-DVD1.iso</li> <li>SLE-12-SP3-Server-DVD-aarch64-GM-DVD1.iso</li> <li>SLE-12-SP3-SDK-DVD-x86_64-GM-DVD1.iso</li> <li>SLE-12-SP3-SDK-DVD-aarch64-GM-DVD1.iso</li> <li>SLE-12-SP3-WE-DVD-x86_64-GM-DVD1.iso</li> <li>SLE-12-Modules-v3.iso</li> </ul>
SLE update	<ul style="list-style-type: none"> <li>sleupdate-12sp3+180209-201802091328.iso</li> </ul>
CentOS software	<ul style="list-style-type: none"> <li>CentOS-6.5-x86_64-bin-DVD1.iso</li> </ul>
CLE and global configuration worksheets	<ul style="list-style-type: none"> <li>cle-MMDD-worksheets.tar</li> <li>global-MMDD-worksheets.tar</li> </ul>
<b>(SMW HA only)</b> SLE HA software	<ul style="list-style-type: none"> <li>SLE-12-SP3-HA-DVD-x86_64-GM-CD1.iso</li> <li>slehaupdate-12sp3+180423+7-201804261316.iso</li> </ul>
<b>(SMW HA only)</b> SMW HA release	<ul style="list-style-type: none"> <li>smwha-12.0.7128-201806242300.iso</li> </ul>



## 3 Introduction to Installation and Configuration of Cray XC™ Software

---

This guide provides information and instructions to perform an initial installation of System Management Workstation (SMW) and Cray Linux Environment (CLE) software release packages on a Cray XC Series system, update SMW and CLE software, and customize a preinstalled system.

With the SMW 8.0 / CLE 6.0 release, Cray has changed the way software is installed, configured, and managed on XC Series systems. The changes that most affect installation and configuration are summarized here.

The new Cray Management System (CMS)

- uses a common installation process for SMW and CLE (which is why there is now a single installation guide for XC systems—this one—instead of separate guides for SMW and CLE)
- leverages standard Linux and common open source tools (e.g., zypper/yum for RPMs, Ansible, YAML/JSON configuration data)
- keeps software images and configuration separate until boot
  - prescriptive image creation using recipes
  - centralized configuration
  - configuration applied at boot time or after configuration adjusted

The core elements of this new management system are:

**IMPS** Image Management and Provisioning System (IMPS) is responsible for creating and distributing repository content and for prescriptive image creation. Note that although filepaths for configuration data and tools include `imps`, this is an artifact of an early implementation that grouped both image and configuration management under IMPS. IMPS is now image management only.

**CMF** Configuration Management Framework (CMF) comprises the configuration data (stored in config sets on the SMW), tools to manage and distribute that data (e.g., the configurator and the IMPS Distribution System (IDS)), and software to apply the configuration data to the running image (Ansible plays).

**NIMS** Node Image Mapping Service (NIMS) is responsible for keeping track of which images get booted on which nodes, what additional kernel parameters to pass to nodes at boot time, and which load file to use within a boot image.

What else is new?

- New base operating system for the SMW/CLE: SUSE® Linux Enterprise Server version 12 SP3 (SLES® 12 SP3) for x86\_64
- New base operating system for HSS (Hardware Supervisory System) controllers: OpenSUSE 13.2 for 32 bit
- New modular installer

Much of the software remains unchanged, for the most part, such as Application-level Placement Scheduler (ALPS), Node Health Checker (NHC), and Resource Utilization Reporting (RUR), among others.

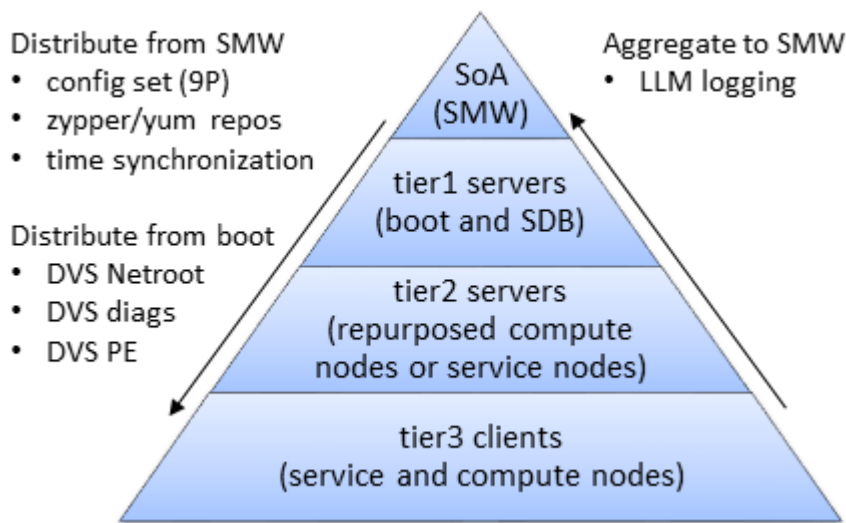
This guide includes procedures for installing the following software: the SMW base operating system, Cray SMW software, CLE software, SLES security updates, Cray Programming Environment (PE), and direct-attached Lustre (DAL), if needed.

## 3.1 About Cray Scalable Services

Cray Scalable Services is an essential part of the Cray Management System that is used to both distribute and aggregate information. Within Cray Scalable Services, nodes are designated as SoA (server of authority), tier1, tier2, or tier3. A node can be a member of only one of these groups. Tier1 nodes are clients of the SoA and servers for tier2 nodes. Tier2 nodes are clients of tier1 nodes and servers for tier3 nodes. Tier3 nodes are clients of tier2 nodes. Configuration of nodes as SoA, tier1, and tier2 is defined in the `cray_scalable_services` configuration service, which must be configured properly for the system to function.

As indicated in this figure, the SMW is the designated SoA in Cray XC systems. The boot and SDB nodes are designated tier1 nodes, and they must have direct network connectivity to the SMW via Ethernet. Typically, tier2 nodes are service nodes or repurposed compute nodes that have no other duties beyond being part of the Scalable Services. All other nodes are tier3 nodes.

Figure 1. Cray Scalable Services



This table shows what gets distributed or aggregated using Cray Scalable Services.

<b>from SMW to rest of system</b>	<ul style="list-style-type: none"> <li>• config set data is shared using a 9P file system and <code>diod</code> (distributed I/O daemon)</li> <li>• <code>zypper</code> software repositories can be used from any node with the LiveUpdate feature (HTTP forwarding from the SMW through the tiers)</li> <li>• time synchronization (NTP)</li> </ul>
<b>from boot node to rest of system</b>	<ul style="list-style-type: none"> <li>• PE (Programming Environment) image root</li> <li>• diag (online diagnostics) image root</li> </ul>

	<ul style="list-style-type: none"> <li>• netroot image roots<sup>1</sup></li> </ul>
<b>from rest of system to SMW</b>	<ul style="list-style-type: none"> <li>• Lightweight Logging Manager (LLM) logging</li> </ul>

Here is an example of how Scalable Services works with Live Updates to distribute software out to nodes. Any tier3 node can run zypper to access the repositories on the SMW because it has an entry in `/etc/zypp/repos.d/liveupdates.repo` that points to the tier2 nodes by means of a base URL, which uses HTTP protocol listing all of the tier2 nodes. The tier2 nodes, in turn, have an entry in `/etc/zypp/repos.d/liveupdates.repo` that lists at least one tier1 node. All tier1 nodes have an entry in `/etc/zypp/repos.d/liveupdates.repo` that lists the SMW.

## Services that Depend on Cray Scalable Services

It is important to configure Cray Scalable Services correctly. The following features and services use data from the `cray_scalable_services` configuration service, and they may not be functional if `cray_scalable_services` is configured incorrectly.

<b>Node Image Mapping Service (NIMS) plugin</b>	Uses <code>cray_scalable_services</code> data to determine tier1 servers and adds the tier1 kernel command line parameter to each tier1 server.
<b>IMPS Distribution Service (IDS)</b>	Uses <code>cray_scalable_services</code> data to set the <code>ids</code> kernel command line parameter to the node's parent, from whom it will receive config set data.
<b>DVS Ansible configuration</b>	Uses <code>cray_scalable_services</code> data to determine which nodes should serve DVS file systems. This will also impact netroot functionality, which uses DVS.
<b>CLE LiveUpdates functionality</b>	Configured using <code>cray_scalable_services</code> data to determine the parent each node should contact en route to the package repos stored on the SMW.
<b>LLM Ansible configuration</b>	Uses <code>cray_scalable_services</code> data to determine the next server to which a node should send its log data, which depends on the node's tier.
<b>NFS Ansible configuration</b>	Uses <code>cray_scalable_services</code> data to determine which nodes should act as clients and servers.
<b>IP forwarding Ansible configuration</b>	Uses <code>cray_scalable_services</code> data to enable IP forwarding and configure servers' routes depending on their tier.

## 3.2 About Service Nodes

Service nodes perform the functions needed to support users, administrators, and applications running on compute nodes.

Service nodes, unlike compute nodes, are generally equipped with Peripheral Component Interconnect (PCI) protocol card slots to support external devices.

Service nodes run a full-featured version of the Linux operating system. Service node kernels are configured to enable non-uniform memory access (NUMA), which minimizes traffic between sockets by using socket-local memory whenever possible.

<sup>1</sup> Netroot is a mechanism that enables nodes booted with a minimal, local in-memory file system to execute within the context of a larger, full-featured root file system that is available to the node via a network mount.

System management tools are a combination of Linux commands and Cray system commands that are analogous to standard Linux commands but operate on more than one node. After the system is up, an administrator possessing the correct permissions can access any service node from any other service node.

## Boot Node

The boot node hosts the Boot service and is controlled by Ansible. It contains the home directories on the boot RAID for local accounts, nonvolatile storage, and image roots for PE, diagnostics, and netroot. It is a tier1 node, which means it has a direct private network connection to the SMW. An administrator logs on to the boot node through the SMW console. It is booted via PXE boot and is the first node to boot.

Two boot nodes can be configured per system or per partition: one primary and one secondary for backup. The two boot nodes must be located on different blades. When the primary boot node is booted, the backup boot node also begins to boot. But the backup boot node boot process is suspended until a primary boot-node failure event is detected. For more information, see [Configure Boot and/or SDB Node Failover](#) on page 239.

## Service Database (SDB) Node

The SDB node hosts the service database, a MySQL database that resides on a separate file system on the boot RAID. The SDB is accessible to every service node. The SDB provides a central location for storing information so that it does not need to be stored on each node. The SDB is accessible, with correct authorizations, from any service node after the system is booted.

The SDB stores the following information:

- Global state information of compute processors. This information is used by the Application Level Placement Scheduler (ALPS), which allocates compute processing elements for compute nodes.
- System configuration tables that list and describe processor attribute and service information.

The SDB node is the second node that is started during the boot process. It is a tier1 node, which means it has a direct private network connection to the SMW.

Two SDB nodes can be configured per system or per partition, one primary and one secondary for backup. The two SDB nodes must be located on different system blades. For more information, see [Configure Boot and/or SDB Node Failover](#) on page 239.

## Login Nodes

Users log on to a login node, which is the single point of control for applications that run on the compute nodes. Users do not log on to compute nodes.

## Other Service Nodes

Some common service nodes are listed below. Not every type of service node is in use on every system.

- MOM (machine-oriented miniserver) nodes

Note that Cray recommends treating MOM nodes (as long as they are not also SDB nodes) the same as login nodes so that they will have the same capabilities. This means that they should be assigned the same NIMS group, boot image, node group, and so forth.

- LNet router nodes
- Network gateway nodes
- DVS nodes

- Tier2 nodes for Cray Scalable Services
- DataWarp nodes
- DAL (direct-attached Lustre) nodes

## 3.3 About Config Sets

Users invoke the `cfgset` command to take configuration content delivered in service packages and combine it with site-specific configuration content gathered either interactively or through bulk import. The results are used by `cfgset` to create a config set, which is a central repository that stores all configuration information necessary to operate the system. Config sets reside on the management node (e.g., the SMW) in `/var/opt/cray/imps/config/sets` by default. The contents of each config set reside in the following subdirectories:

<b>ansible</b>	Local site-provided Ansible play content can be placed here for distribution with the config set. When the config set is created, <code>cfgset</code> copies Ansible content from service packages to this location. Whenever the config set is updated, <code>cfgset</code> copies Ansible content from service packages again, overwriting the previous service-package Ansible content and leaving the site-provided content unchanged.
<b>changelog</b>	YAML change logs from previous sessions with the configurator.
<b>config</b>	Configuration templates containing configuration information. When the config set is created, the configurator copies service package templates to this location. Users can modify the content of these templates using <code>cfgset</code> to invoke the configurator. Whenever the config set is updated, the configurator merges service package templates with the templates in this location.
<b>dist</b>	Other site-provided content, such as static files required for the configuration of a service, can be placed here for distribution with the config set. When the config set is created, <code>cfgset</code> copies dist content from service packages to this location. Whenever the config set is updated, <code>cfgset</code> copies dist content from service packages again, overwriting the previous service-package dist content and leaving the site-provided content unchanged.
<b>files</b>	Files necessary for system configuration that are distributed with the config set. They can be placed here by: <ul style="list-style-type: none"><li>• the <code>cfgset</code> command, which runs configuration callback scripts to generate some configuration files (e.g., <code>/etc/hosts</code>)</li><li>• the Simple Sync service</li><li>• local site administrators</li></ul>
<b>worksheets</b>	Configuration worksheets generated by the configurator using data stored in the configuration templates in the <code>config</code> subdirectory of the config set. Administrators copy these worksheets to a location outside the config set, edit them with site-specific configuration data, and then import them to create a new config set or update an existing one.

### Config Set Types

All config sets have a type associated with them that is specified upon creation. XC systems require both a `global` config set type and a `cle` config set type. After a config set of a given type is created, its type cannot be changed. A user may create multiple config sets to support partitioned systems or alternate configurations.

Typically a config set of type `cle` is created for each partition to store partition- and CLE-specific content, and another config set of type `global` is created to store configuration data that pertains to the management node domain as well as configuration data that can be easily shared among `cle` config sets. Config sets can be portable between partitions or to other systems if their partition-specific information is modified accordingly.

## Configuration Service Inheritance

When a config set is created or updated, only service package templates that match the type of the config set can be included in the config set. Cray provides several service package templates that match both types and can be included in both `cle` and `global` config sets. In such cases, the user can choose which template will be used to configure the service in question. When a `cle` config set is created, and a service that has a template of both types is ready for configuration, the configurator will inject an initial question for the user to choose between configuring the service (i.e., using the `cle` version of the template) or letting the service inherit configuration values from the `global` config set (i.e., inheriting values from the `global` version of the template). Configuration worksheets for such services also provide that choice by including an `inherit` field, which can be set to true or false. If the user sets it to true, the configuration data from the global config set version of the service will be used. When the Cray-provided `cray-ansible` service (part of the Cray Configuration Management Framework) is run at boot time or at the system administrator's discretion, it uses the value of the `inherit` field to determine which configuration template data (`global` or `cle`) to use.

Inheritance is useful for systems with multiple partitions where a subset of partitions need custom configuration of a service, but another subset of partitions can all share the same global configuration.

## 3.4 About Variable Names in the Configurator and Configuration Worksheets

In the configurator and configuration worksheets, variable names can be quite long because they are composed of a data structure hierarchy. Each variable name begins with the name of the service to which it belongs. The next part of each name is always 'settings' to indicate that what follows is a service setting, one of the available settings for that service. After 'settings' comes the name of the setting, which could be a simple data type (string, boolean, integer, etc.) or a more complex data type (list, multival, etc.). The next part after the name of the setting is always 'data' to indicate that what follows is one of the fields of that setting. For a full description of data types, see *XC™ Series Configurator User Guide (S-2560)*.

For example, here is the variable for the IP address of the high-speed network (HSN), one of several networks.

```
cray_net.settings.networks.data.hsn.ipv4_network
```

This variable belongs to the `cray_net` service and the `networks` setting of that service. The `networks` setting is of type `multival`, which means it can have multiple entries, and each entry can have multiple fields to set. This variable targets the `ipv4_network` field of the `hsn` network entry.

This example shows the variable for the IP address of the HSN SDB node alias interface (one of several interfaces) of the SDB node (one of several hosts).

```
cray_net.settings.hosts.data.sdbnode.interfaces.hsn_sdb_alias.ipv4_address
```



This variable belongs to the `cray_net` service and the `hosts` setting of that service. The `hosts` setting is of type `multival`, and this variable belongs to the `sdbnode` host entry. The `sdb_node` host has a field `interfaces`, which is also of type `multival`. This variable targets the `ipv4_address` field of the `hsn_sdb_alias` interface entry.

## 3.5 About Snapshots and Config Set Backups

Sites can make as few or as many snapshots and config set backups as they deem useful, but Cray recommends that sites make a snapshot and back up config sets at certain milestones during the installation and configuration process. Most of these will be for archival purposes, but snapshots and config set backups can be used to stage updates/upgrades and roll back to or switch between SMW and CLE releases as well.

### How are snapshots and config sets created?

- Snapshots are created and managed using `snaputil`, a Python utility delivered with the `cray-install-support` RPM that is installed by default on the SMW. However, the fresh install procedure makes the first snapshot manually, because at that point in the process, `snaputil` has not yet been installed.

Note that on a system with SMW 8.0.UP05 / CLE 6.0.UP05 or a later release installed, whenever that system is booted to a pre-UP05 snapshot, the full path to `snaputil` must be used to boot the system back to an UP05 or post-UP05 snapshot:

```
/boot/install-support/default/snaputil
```

Beginning with the SMW 8.0.UP05 / CLE 6.0.UP05 release, the `snaputil` utility is in that directory, which is exempt from snapshots.

- Config sets are created and managed using `cfgset`.

Procedures for creating snapshots and config set backups are included at each point in the process where they are needed.

**What does a snapshot contain?** Snapshots capture content in these three file systems on the SMW: root (`/`), `/var/lib/mysql`, and `/var/opt/cray/repos`. Used in conjunction with backups of config sets, they provide enough information to be able to re-create the state of the system at the time of the snapshot and config set backup.

**What does a config set contain?** See [About Config Sets](#) on page 19 for details about the contents of a config set.

**Best Practice.** Make a snapshot and back up config sets at the same time to keep them in sync. Name the snapshot and config set backup using the same suffix and date/time stamp to help administrators identify which snapshot and config set backup pairs belong together.

*Table 2. Suffixes and Corresponding Milestones for Snapshots and Config Set Backups*

Suffix	Description	Snapshot	Config Set
preupdate	before beginning any software update activities (software updates only)	yes	yes
postinstall	after installing a new software release (fresh install or software update) and before configuring that software	yes	yes
postconfig	after configuring SMW/CLE software and before booting the CLE system	yes	yes

Suffix	Description	Snapshot	Config Set
postboot	after booting the CLE system	yes	yes
postpe	after installing or updating Cray PE software	yes	yes
postcustomize	after customizing a preinstalled system (not for fresh installs or software updates)	yes	yes

## Other Snapshot-related Utilities: `dumphss` and `freshenhss`

Because the Hardware Supervisory System (HSS) database is local to a snapshot, for staged updates/upgrades, Cray provides these two additional utilities as well:

**`dumphss`** The `dumphss` utility dumps the current HSS database. When an administrator runs `snaputil` to set the default snapshot before rebooting to that snapshot, `snaputil` runs `dumphss` automatically to back up the database.

**`freshenhss`** The `freshenhss` utility updates the HSS database on the new snapshot after the SMW is rebooted to that snapshot. It syncs the snapshot-local HSS database with the changes made while the previously booted snapshot was active. The `freshenhss` command is not run automatically; it is run manually by an administrator, if needed.

The HSS database in a snapshot that has not been booted recently may no longer reflect the physical state (what components are where) or administrative state (which nodes are enabled, disabled, set-to-service, and so forth) of the XC system. In such cases, after the SMW is rebooted to that snapshot, run `freshenhss` in the snapshot to restore this information from the last-booted snapshot. Note that `freshenhss` will not take action if (1) the software versions are too different between the snapshots, or (2) hardware changes have occurred since the snapshot was last booted. In the case of hardware changes, run `xtdiscover` to manually update the HSS database.

When possible, it is usually preferable to run `freshenhss` instead of `xtdiscover`, because while `xtdiscover` can restore the physical state, it cannot detect administrative state changes made while another snapshot was booted. The `freshenhss` utility compares the last snapshot with the current one before taking any action, and depending on the software levels involved, an explicit `xtdiscover` may still be required as an additional step. See the `freshenhss` man page for details.

## 3.6 About Config Set Caching

Config sets are defined and reside on the Server of Authority, which on XC systems is the SMW. Config set content is made available to all nodes in the system by means of Cray Scalable Services.

To make the sharing of config set content both quick and reliable, the `cray-cfgset-cache` service was created. It caches config sets locally on nodes (compressed for a smaller footprint). On the SMW, it does the following:

- notices changes to config sets on the SMW
- refreshes the local caches dynamically
- detects failures and retries automatically

The `cray-cfgset-cache` service ensures that config set content gets refreshed on all nodes whenever config sets are created or updated on the SMW. It is triggered when `cray-ansible` is run on a node with the `start`, `restart`, or `link` commands.

**ATTENTION:** If the `cray-cfgset-cache` service is stopped, config set content in node-local memory will not get refreshed when `cray-ansible` is run. If that happens, nodes will continue to use the most recent compressed copy of the config set data created before the service was stopped.

## What Gets Cached

The `cray-cfgset-cache` service does not copy an entire config set to node-local memory. Instead, it uses the config set on the SMW to create these two files in the root of the config set:

- a compressed copy of the config set using SquashFS tools, (typically < 3 MB)
- a checksum of the compressed copy of the config set

The compressed copy is made available (effectively copied) to node-local RAM, and the checksum is used to know when the config set in node-local memory no longer matches the config set on the SMW. Even though Scalable Services makes the entire config set directory structure on the SMW available to the rest of the system, only the compressed copy and its associated checksum are used by nodes. They are the key to the performance, scalability, and reliability improvements provided by config set caching.

When `cray-ansible` is run on a node, the node will do the following:

1. Check to see if the cached node-local version of the compressed config set is out of date.
2. If it is stale, replace it with a newer version available on the SMW and start using that newer version.

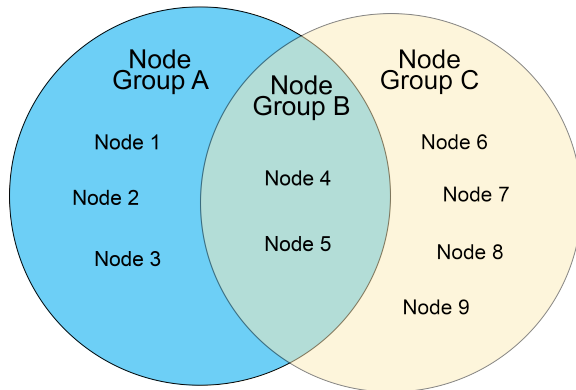
## 3.7 About Node Groups

The Cray Node Groups service (`cray_node_groups`) enables administrators to define and manage logical groupings of system nodes. Nodes can be grouped arbitrarily, though typically they are grouped by software functionality or hardware characteristics, such as login, compute, service, DVS servers, and RSIP servers.

Node groups that have been defined in a config set can be referenced by name within all CLE services in that config set, thereby eliminating the need to specify groups of nodes (often the same ones) for each service individually and greatly streamlining service configuration. Node groups are used in many Cray-provided Ansible configuration playbooks and roles and can be also used in site-local Ansible plays. Node groups are similar to but more powerful than the class specialization feature of releases prior to CLE 6.0. For example, a node can be a member of more than one node group but could belong to only one class.

The figure below demonstrates how several nodes may belong to more than one node group. In this example, node group A contains nodes 1-5, node group B contains nodes 4-5, and node group C contains nodes 4-9. Nodes 4 and 5 belong to node groups A, B, and C. In this example, if nodes 1-5 are the desired target for an Ansible play, the play can target node group A instead of specifying each node individually.

Figure 2. Node Group Member Overlap



Sites are encouraged to define their own node groups and specify their members. Administrators can define and manage node groups using any of these methods:

- Edit and upload the node groups configuration worksheet (`cray_node_groups_worksheet.yaml`).
- Use the `cfgset` command to view and modify node groups interactively with the configurator.
- Use the `cfgset get` and `cfgset modify` CLI commands to view and modify node groups at the command line. Note that CLI modifications must be followed by a config set update.

After using any of these methods, remember to validate the config set.

## Characteristics of Node Groups

- Node group membership is not exclusive, that is, a node may be a member of more than one node group.
- Node group membership is specified as a list of nodes:
  - use `cname` for a CLE node
  - use host ID (the output of the `hostid` command) for the SMW
  - use host name for an eLogin node
- All compute nodes and/or all service nodes can be added as node group members by including the keywords “platform:compute” and/or “platform:service” in a node group.
- Any CLE configuration service is able to reference any defined node group by name.
- The Configuration Management Framework (CMF) exposes node group membership of the current node through the local system “facts” provided by the Ansible runtime environment. This means that each node knows what node groups it belongs to, and that knowledge can be used in Cray and site-local Ansible playbooks.

## Pre-populated Node Groups

Pre-populated node groups are groups of nodes that

- are likely to be customized and used by many sites
- support useful default values for many of the configuration services

Several of the pre-populated node groups require customization by a site to provide the appropriate node membership information. This table lists the pre-populated groups and indicates which ones require site customization.

Note that beginning with CLE 6.0.UP06, Cray no longer supports a single node group for all login nodes. Instead, there are two architecture-specific login node groups: one for all login nodes with the x86-64 architecture and one for all login nodes with the AArch64 architecture. To specify all login nodes in the system, use both of those node groups.

Table 3. *cray\_node\_groups*

Pre-populated Node Group	Requires Customization?	Notes
compute_nodes	No	Contains all compute nodes in the given partition. The list of nodes is determined at runtime.
compute_nodes_x86_64	No	Contains all x86-64 compute nodes in the given partition. The list of nodes is determined at runtime.
compute_nodes_aarch64	No	Contains all AArch64 compute nodes in the given partition. The list of nodes is determined at runtime.
service_nodes	No	Contains all service nodes in the given partition. The list of nodes is determined at runtime.
service_nodes_x86_64	No	Contains all x86-64 service nodes in the given partition. The list of nodes is determined at runtime.
service_nodes_aarch64	No	Contains all AArch64 service nodes in the given partition. The list of nodes is determined at runtime.
smw_nodes	Yes	Add the host ID (output of the <code>hostid</code> command) of the SMW. For an SMW HA system, add the host ID of the second SMW also.
boot_nodes	Yes	Add the cname of the boot node. If there is a failover boot node, add its cname also.
sdb_nodes	Yes	Add the cname of the SDB node. If there is a failover SDB node, add its cname also.
login_nodes_x86_64	Yes	Add the cnames of all x86-64 internal login nodes on the system.
login_nodes_aarch64	Yes	Add the cnames of all AArch64 internal login nodes on the system. Leave empty (set to <code>[]</code> ) if there are none.
eloin_nodes	Yes	Add the host names of external login nodes on the system. Leave empty (set to <code>[]</code> ) if there are no eLogin nodes.
all_nodes	Maybe	Contains all compute nodes and service nodes on the system. Add external nodes (e.g., eLogin nodes), if needed.
all_nodes_x86_64	No	Contains all x86-64 nodes in the given partition. The list of nodes is determined at runtime.

Pre-populated Node Group	Requires Customization?	Notes
all_nodes_aarch64	No	Contains all AArch64 nodes in the given partition. The list of nodes is determined at runtime.
tier2_nodes	Yes	Add the cnames of nodes that will be used as tier2 servers in the <code>cray_scalable_services</code> configuration.

**Why is there no "tier1\_nodes" pre-populated node group?** Cray provides a pre-populated tier2\_nodes node group to support defaults in the `cray_simple_shares` service. Cray does not provide a tier1\_nodes node group because no default data in any service requires it. Because it is likely that tier1 nodes will consist of only the boot node and the SDB node, for which node groups already exist, Cray recommends using those groups to populate the `cray_scalable_services` tier1\_groups setting rather than defining a tier1\_nodes group.

**About eLogin nodes.** To add eLogin nodes to a node group, use their host names instead of cnames, because unlike CLE nodes, eLogin nodes do not have cname identifiers. If eLogin nodes are intended to receive configuration settings associated with the all\_nodes group, add them to that group, or change the relevant settings in other configuration services to include both all\_nodes and elogin\_nodes.

## Additional Platform Keywords

Cray uses these two platform keywords to create pre-populated node groups that contain all compute or all service nodes.

```
platform:compute
platform:service
```

Cray uses these keywords to create pre-populated node groups that contain all compute or service nodes with the x86-64 or AArch64 architecture.

```
platform:compute-X86
platform:service-X86
platform:compute-ARM
platform:service-ARM
```

**Disabled nodes.** All platform keywords, such as `platform:compute`, `platform:service-ARM`, and `platform:compute-HW12`, include nodes that have been disabled. To identify disabled nodes, use this keyword: `platform:disabled`

**Excluded nodes.** Groups of nodes can be excluded using a negation operator: `~` (the tilde symbol). For example, a custom node group that contains all enabled compute and service nodes would have the following list as its members. The ordering of the list does not matter: all non-negated keywords are resolved first, then negated ones are removed.

```
- platform:compute
- platform:service
- ~platform:disabled
```

Sites that need finer-grained groupings can use additional platform keywords to create custom node groups. For a node group that contains all compute or service nodes with a particular processor/core type, use one of the following platform keywords.



```
platform:compute-XX##
platform:service-XX##
```

For *XX##*, substitute a processor/core code, such as KL64 or KL68, which designate two Intel® Xeon Phi™ "Knights Landing" (KNL) processors with different core counts. To find the code associated with each node on a Cray system, use the `xtcli status p0` command and look in the "Core" column of the output, as shown in the following example.

```
smw# xtcli status p0
Network topology: class 0
Network type: Aries
Nodeid: Service Core Arch| Comp state [Flags]
-----
c0-0c0s0n0: service BW18 X86| ready [noflags|]
c0-0c0s0n1: service BW18 X86| ready [noflags|]
c0-0c0s0n2: service BW18 X86| ready [noflags|]
c0-0c0s0n3: service BW18 X86| ready [noflags|]
c0-0c0s1n0: service BW18 X86| ready [noflags|]
c0-0c0s1n1: service BW18 X86| ready [noflags|]
c0-0c0s1n2: service BW18 X86| ready [noflags|]
c0-0c0s1n3: service BW18 X86| ready [noflags|]
c0-0c0s2n0: - HW12 X86| ready [noflags|]
c0-0c0s2n1: - HW12 X86| ready [noflags|]
c0-0c0s2n2: - HW12 X86| ready [noflags|]
c0-0c0s2n3: - HW12 X86| ready [noflags|]
```

The following table lists some of the common processor/core codes supported by Cray.

*Table 4. Cray Supported Intel Processor/Core (XX##) Codes*

Processor (XX)	Core (##)	Intel Code Name
BW	12, 14, 16, 18, 20, 22, 24, 28, 32, 36, 40, 44	"Broadwell"
HW	04, 06, 08, 10, 12, 14, 16, 18, 20, 24, 28, 32, 36	"Haswell"
IV	02, 04, 06, 08, 10, 12, 16, 20, 24	"Ivy Bridge"
KL	60, 64, 66, 68, 72	"Knights Landing"
SB	04, 06, 08, 12, 16	"Sandy Bridge"
SK	04, 08, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 52, 56	"Skylake"

## 3.8 About Simple Sync

The Cray Simple Sync service (`cray_simple_sync`) provides a simple, generic mechanism for copying user-defined content to internal and external nodes in a Cray XC system. When executed, the service automatically copies files found in source directories in the config set to one or more target nodes. The Simple Sync service is enabled by default and has no additional configuration options. It can be enabled or disabled during the initial installation using worksheets or with the `cfgset` command at any time. For more information, see `man cfgset(8)`.

With regard to external nodes like eLogin nodes, the exclusions specified in the `cray_cfgset_exclude` configuration service are applied when the CLE config set is transferred to the node, and some portions of the

Simple Sync directory in the config set are excluded. The "Files Excluded from eLogin Nodes" section contains more details.

Simple Sync is a simple tool and not intended as the sole solution for making configuration changes to the system. Writing custom Ansible plays might provide better maintainability, flexibility, and scalability in the long term.

## How Simple Sync Works

When enabled, the Simple Sync service is executed on all internal CLE nodes and eLogin nodes at boot time and whenever the administrator executes `/etc/init.d/cray-ansible start` on a CLE node or eLogin node. When Simple Sync is executed, files placed in the following directory structure are copied to the root file system (/) on the target nodes.

The Simple Sync directory structure has this root:

```
smw:/var/opt/cray/imps/config/sets/<config_set>/files/simple_sync/
```

Below that root are the directories listed on the left. Files placed in those directories are copied to their associated target nodes.

<code>./common/files/</code>	Targets all nodes, both internal CLE nodes and eLogin nodes.
<code>./hardwareid/&lt;hardwareid&gt;/files/</code>	Targets a specific node with that hardware ID, which is the cname of a CLE node or the output of the <code>hostid</code> command (e.g., <code>1eac0b0c</code> ) on other nodes. An admin must create both the <code>&lt;hardwareid&gt;</code> directory and the <code>files</code> directory. Not applicable to eLogin nodes.
<code>./hostname/&lt;hostname&gt;/files/</code>	Used ONLY for eLogin nodes. Targets a node with the specified host name. An admin must create both the <code>&lt;hostname&gt;</code> directory and the <code>files</code> directory.
<code>./nodegroups/&lt;node_group_name&gt;/files/</code>	Targets all nodes in the specified node group. The directories for this <code>nodegroups</code> directory are automatically stubbed out when the config set is updated after node groups are defined and configured in the <code>cray_node_groups</code> service.
<code>./platform/[compute service]/files/</code>	Targets all compute nodes or all service nodes, depending on whether they are placed in <code>platform/compute/files</code> or <code>platform/service/files</code> . Each time the config set is updated, the HSS data store is queried to update which nodes are service and which are compute. Not applicable to eLogin nodes.
<code>./README</code>	Provides brief guidance on using Simple Sync and a list of existing node groups in the order in which files will be copied. This ordering enables an administrator to predict behavior in cases where a file may be duplicated within the Simple Sync directory structure.

Simple Sync copies content into place prior to the standard Linux startup (`systemd`) and before `cray-ansible` runs any other services.

The ownership and permissions of copied directories and files are preserved when they are copied to root on the target nodes. An administrator can run `cray-ansible` multiple times, as needed, and only the files that have changed will be copied to the target nodes.

Because of the way it works, Simple Sync can be used to configure services that have configuration parameters not currently supported by configuration templates and worksheets. An administrator can create a configuration file with the necessary settings and values, place it in the Simple Sync directory structure, and it will be distributed and applied to the target nodes.

## Files Excluded from eLogin Nodes

Because eLogin nodes use the `cray_cfgset_exclude` configuration service, some directories within the Simple Sync directory structure on the SMW can be excluded from transfer to eLogin nodes. The default “`eloin_security`” profile will exclude the following config set directories from being transferred to an eLogin node when the CLE config set is pushed to the node from the SMW.

- `files/simple_sync/common/files/etc/ssh`
- `files/simple_sync/common/files/root/.ssh`

To specify other areas within the Simple Sync directory structure that should not be transferred to eLogin nodes, create a customized site profile in `cray_cfgset_exclude`.

## Simple Sync and Configuration File Management

Configuration files can be managed in one of three ways:

- Managed entirely by a site system administrator

Such config files are considered non-conflicting because there is no potential conflict between administrator-provided content and Cray-managed content.

- Managed entirely by Cray configuration services

Where possible, such config files have a comment at the top indicating that the file is completely under the management of the Cray service. Files that have been changed by Cray services can be identified by checking the change logs on the running node in `/var/opt/cray/log/ansible`. Simple Sync does not provide a mechanism to override changes made by Cray services. To override changes made by Cray services, refer to the documentation for the specific service.

- Jointly managed by a system administrator and by Cray config services

These config files can contain both administrator-managed content and Cray-managed content, so there is potential for conflict. Administrator changes to Cray-managed content can be overridden. Content that is not managed by Cray is considered non-conflicting because any admin changes to it will not conflict with changes made by Cray services.

Because Simple Sync copies administrator-provided files into place before `cray-ansible` runs, any Cray services that make small changes to jointly managed files will operate on the administrator-provided files. Afterwards, that file will contain both non-conflicting administrator-provided content as well as the changes made by the Cray service. Because these changes happen prior to Linux startup, the changes will be in place when the services start up.

## Characteristics of Simple Sync

Simple Sync is:	Simple Sync is NOT:
for simple and straightforward use cases	a comprehensive system management solution
for copying a moderate number of moderately sized files*	intended to transfer large objects or a large volume of files
	an interface to configure Cray "turnkey" services such as ALPS, Node Health or Lightweight Log Manager (LLM)

\* Bear in mind that anything in the Simple Sync directory structure is part of a config set, and a SquashFS copy of the current config set is distributed to all nodes in the system. Even though it is a reduced-size config set that is distributed, it is good practice to not add very large files to a config set, hence the use of "moderate" here.

Simple Sync:

- runs as early in the Ansible execution sequence as possible (it runs BEFORE other `cray-ansible` plays, so it can be used to make changes to files that Cray updates, like `sshd_config`)
- runs during the netroot setup sequence, so it can be used to change LNet and DVS settings, if needed
- supports node groups for targeting which system nodes to copy files to (see [About Node Groups](#) on page 23)

Simple Sync does not support:

- removing files
- appending to files
- changing file ownership and permissions (the permissions of the file in the config set are mirrored on-node)
- backing up files
- overriding Cray-set values (it cannot be used to change files that Cray completely overwrites, such as `alps.conf`, or change values in files that Cray modifies such as `PermitRootLogin` in `/etc/ssh/sshd_config`)

## Cautions about the Use of Simple Sync

- Simple Sync copies files from the config set, which in the case of nodes without a persistent root file-system is cached in a compressed form, locally, in memory. As a result, each file stored in the config set uses some memory on the node. Therefore, using Simple Sync to copy binary files or large numbers of files is inadvisable.
- Be aware of differences in node environments when using Simple Sync. For example, systems configured with direct-attached Lustre (DAL) have nodes running CentOS instead of SLES. Administrators would have to be very careful to avoid putting an inappropriate configuration file into place when using the Simple Sync platform/service target in such a situation.
- Storage and distribution of verbatim config files through Simple Sync creates the potential for unintentional impact to the system when config files evolve due to software changes. Making minimal necessary changes through a site-local Ansible playbook provides more flexibility and minimizes the potential for unintended consequences.

## Use Cases

### Copy a non-conflicting file to all nodes

1. Place `etc/myfile` under `./common/files/` in the Simple Sync directory structure.
2. Simple Sync copies it to `/etc/myfile` on all nodes.

### Copy a non-conflicting file to a service node

1. Place `etc/servicefile` under `./platform/service/files/` in the Simple Sync directory structure.
2. Simple Sync copies it to `/etc/servicefile` on all service nodes.

### Copy a non-conflicting file to a compute node

1. Place `etc/computefile` under `./platform/compute/files/` in the Simple Sync directory structure.
2. Simple Sync copies it to `/etc/computefile` on all compute nodes.

### Copy a non-conflicting file to a specific node

1. Place `etc/mynode` under `./hardwareid/c0-0c0s0n0/files/` in the Simple Sync directory structure.
2. Simple Sync copies it to `/etc/mynode` on `c0-0c0s0n0`.

### Copy a non-conflicting file to a user-defined collection of nodes

1. Create a node group called "my\_nodes" containing a list of nodes.
2. Update the config set.

```
smw# cfgset update p0
```

3. Place `etc/mynodes` under `./nodegroups/my_nodes/files/` in the Simple Sync directory structure.
4. Simple Sync copies it to `/etc/mynodes` on all nodes listed in node group `my_nodes`.

### Copy to a node a file that has Cray-maintained content

To reduce the number of authentication tries from the default of six,

1. Place a version of `sshd_config` (entire file) that includes "MaxAuthTries 3" under `./nodegroups/login_nodes_x86_64/files/etc/ssh/` and `./nodegroups/login_nodes_aarch64/files/etc/ssh/` in the Simple Sync directory structure.
2. The booted system will contain both:
  - "MaxAuthTries 3" (from the files copied by Simple Sync)

- “PasswordAuthentication yes” (from modification of file by Cray)

### Copy to a node a file that is exclusively maintained by Cray

Files exclusively maintained by Cray such as `alps.conf` cannot be updated using Simple Sync. Please refer to the owning service (such as ALPS) for information on how to update the contents.

### Copy to a node a file that resides on a file system that will be mounted during Linux boot

No special operational changes are necessary. However, Simple Sync will put the file in place early in the boot sequence, and then it will be over-mounted by the file system. Because Simple Sync runs again later, it will copy the file into the mounted file system. Due to the ordering of operations, the file will not be present between the time the file system was mounted and the late execution of Ansible.

### On netroot login nodes, modify an LNet modprobe parameter

1. Generate a file `my_lnet.conf` containing `options lnet_router_ping_timeout=100`.
2. Place `my_lnet.conf` under `./nodegroups/login/files/etc/modprobe.d/` in the Simple Sync directory structure.
3. The `lnet_router_ping_timeout` value will be 100.

Note that normally Simple Sync does not allow the user to override Cray values, but this procedure takes advantage of the standard Linux mechanism to override Kernel module options.

### Copy a file with incompatible content to a node file that has Cray-maintained content

While Simple Sync allows an administrator to make changes to configuration files that are modified by Cray, be very careful to avoid introducing syntax errors or incompatible values that may cause the system to fail to operate correctly.

## 3.9 About Secure Shell Configuration

The Cray secure shell (SSH) configuration service, which generates and manages SSH keys, provides a turnkey environment that establishes SSH functionality quickly and easily and supports basic customer needs. SSH functionality can also be established in a variety of ways that support more complex SSH configurations for both CLE and eLogin nodes (for systems running CLE 6.0.UP04 and later):

- Automatic SSH key generation can be disabled to prevent interference with site-provided configuration.

The `cray_ssh` configuration service has a flag called `simple_ssh_keys`. It is set to `true` by default, which enables automatic SSH key generation/management. If this flag is set to `false`, that functionality is disabled, and the site assumes responsibility for providing a working SSH key configuration.

- eLogin nodes can have different SSH keys.



The `cray_login` configuration service has a setting that must be set on all systems: `elogin_groups`. It specifies which nodes will be used as external login nodes, and it is set to the pre-populated `elogin_nodes` node group by default.

**IMPORTANT: Action required.** Sites that DO NOT have eLogin nodes MUST set `elogin_groups` to an empty list (`[]`). Sites that DO have eLogin nodes must ensure that the node group(s) specified for `elogin_groups` contain ALL eLogin nodes in the system. Instructions are included in the appropriate fresh install and software update procedures.

- Simple Sync and node groups are used to synchronize SSH keys.

The location for all SSH keys is in the Simple Sync directory structure. For CLE nodes, common keys are located in the `common` subdirectory, and keys for specific node groups can be placed in the associated node group subdirectories. For eLogin nodes,

- Common key location for CLE nodes: `<Simple Sync path>/common/files`
  - Common key location for eLogin nodes: `<Simple Sync path>/nodegroups/elogin_nodes/files`
- `<Simple Sync path> = /var/opt/cray/imps/config/sets/<config_set>/files/simple_sync/`

## Basic Components

The following three basic components of SSH configuration can be combined in several ways to create a wide range of SSH functionality.

<b>SSH key generation</b>	<ul style="list-style-type: none"><li>• [default] generated automatically by Cray</li><li>• generated entirely by the site</li><li>• a mixture of Cray-generated and site-generated</li></ul>
<b>SSH key synchronization</b>	<ul style="list-style-type: none"><li>• [default] synchronized automatically by Cray using Simple Sync or the Cray SSH play (only if Simple Sync disabled)</li><li>• synchronized automatically using Simple Sync only</li><li>• synchronized entirely by the site</li></ul>
<b>sshd_config</b>	<ul style="list-style-type: none"><li>• [default] minimally modified by the Cray SSH play</li><li>• never modified by the Cray SSH play</li></ul>

The following use cases illustrate common combinations of these elements.

## Use Case 1: [Default] Automatic SSH Key Management

By default, the Cray SSH play and automatic key management are enabled. This means:

- **Generation.** System and root user SSH keys will be automatically generated (if none are present in the common key location) when the config set is updated.
- **Synchronization.** Keys will be copied automatically from the config set onto the nodes.
- **sshd\_config.** The Cray SSH play will make minimal changes to `sshd_config` to ensure that basic logins are enabled.

The behavior is identical to previous CLE 6.0 releases, except that the location in the config set of the SSH files is now in the Simple Sync directory.

## Use Case 2: Site Modifies SSH Content in Simple Sync Directories

The Cray SSH play and automatic key management are enabled, as in Use Case 1, but after installation or configuration, the site administrator adds new or different content in Simple Sync directories for SSH, such as different keys for login nodes. This use case illustrates that sites can leave automatic key generation in place but still customize SSH keys in Simple Sync.

- **Generation.** Automatic key generation is enabled, as in Use Case 1, but after the admin adds site-specific content to the common key SSH key location in the Simple Sync directory, no new keys will be generated.
- **Synchronization.** Same as Use Case 1.
- **sshd\_config.** Same as Use Case 1.

## Use Case 3: Automatic SSH Key Management Disabled

Disabling automatic key generation and synchronization (set `simple_ssh_keys` to 'false' in `cray_ssh` config service) enables sites to have complete control over key management. A site may wish to use a configuration that has no common SSH keys, and because the absence of keys in the common location triggers the generation of new keys, the site would need to disable automatic SSH key management.

**ATTENTION:** A site that disables automatic SSH key management assumes responsibility for providing a working SSH key configuration.

- **Generation.** No SSH keys will be automatically generated when the config set is updated, even if none are present in the common key location.
- **Synchronization.** No special synchronization will be performed for SSH keys beyond generic Simple Sync functionality.
- **sshd\_config.** Same as Use Case 1.

## Use Case 4: SSH Play Disabled

Disabling the Cray SSH play (set `cray_ssh.enabled: false` in `cray_ssh` config service) enables sites to completely replace Cray SSH configuration. The site must provide `sshd_config` as well as SSH keys. Keys may be synchronized using Simple Sync or a site-local Ansible play.

- **Generation.** Same as Use Case 3.
- **Synchronization.** Site will synchronize keys using Simple Sync or a site-local Ansible play.
- **sshd\_config.** No configuration of `sshd_config` will take place.

## Use Case 4-EZ: SSH Play Disabled after System Boot

Customers who wish total control over SSH and SSH keys can still leverage the Cray SSH infrastructure:

1. Boot the system with Cray SSH play and automatic key management are enabled (Use Case 1).
2. Copy `sshd_config` from the booted system into the Simple Sync directory.
3. Disable the Cray SSH play (Use Case 4).

## 3.10 About Boot Automation Files

The default boot behavior for Cray systems without direct-attached Lustre (DAL) nodes is to boot the boot and SDB nodes first, then boot all other service nodes and all compute nodes at the same time, thereby decreasing overall boot time. Systems with DAL must boot the compute nodes after the service nodes.

- Default for systems without DAL:
  1. Boot + SDB (if SDB image small enough to PXE boot)
  2. SDB (if SDB image too large to PXE boot)
  3. Service + Compute
- Default for systems with DAL:
  1. Boot + SDB (if SDB image small enough to PXE boot)
  2. SDB (if SDB image too large to PXE boot)
  3. Service
  4. Compute

Cray provides the following boot automation files with this release.

<b>auto.generic</b>	Used to boot the entire XC system.
<b>auto.xtshutdown</b>	Used to shut down the entire XC system.
<b>auto.bootnode</b>	Used to boot only the boot node(s).
<b>auto.bootnode+sdb</b>	Used to boot only the boot node(s) and SDB node(s).

During a fresh install, sites typically copy `auto.generic`, rename it with the host name of the system for which it will be used (`auto.hostname.start`), and customize it for that site and system. Likewise, sites typically copy `auto.xtshutdown`, rename it with the host name of the system for which it will be used (`auto.hostname.stop`), and customize it, as needed. The host name is included because different systems may have different software installed, resulting in different boot or shutdown requirements. For example, on a system with a workload manager (WLM) installed, extra commands may be needed in the `auto.hostname.stop` file to cleanly stop the WLM queues on SDB or MOM nodes before shutting down the nodes.

### When is customization of an automation file needed?

- For systems booting tmpfs images (instead of netroot) with no SDB node failover, no changes may be necessary.
- For systems with boot or SDB node failover, instructions for adding or enabling commands are provided at the appropriate place in the fresh install and update processes.
- For systems booting netroot images, instructions for making netroot-related changes after the first boot with tmpfs are provided at the appropriate place in the fresh install process.
- For systems booting direct-attached Lustre (DAL) images, instructions for making DAL-related changes are provided at the appropriate place in the fresh install process.
- For systems with added content in the recipe used for SDB nodes, if the resulting custom recipe produces a boot image too large for a PXE boot, changes to the boot automation file are necessary. If based on `auto.generic`, the system boot automation file will have an option (commented out by default) to boot the boot node via PXE boot and then boot the SDB node via the HSN.
- For systems with a workload manager (WLM) installed, WLM-related changes may be needed. Specific commands to add will vary based on the WLM.

## 3.11 About the Admin Image

**About the admin image.** The admin image can be used on boot and SDB nodes ("admin" nodes) instead of the general service node image. The admin recipe produces an image root that is smaller than that produced by the general service recipe, resulting in a boot image small enough for a PXE boot. Using the admin boot image on the boot and SDB nodes may enable them to PXE boot at the same time. And because the general service image is no longer used for nodes that are intended to PXE boot, content can be added to the general service image without regard for the PXE boot size limitation.

For sites with boot node failover and/or SDB node failover, if the admin image is used on the primary nodes, it should be used on the backup (failover) nodes as well.

### Should this site use the admin recipe for both boot nodes and SDB nodes?

**boot node(s)** Yes. This will enable a PXE boot of the boot node(s).

**SDB node(s)** It depends.

- Yes, if nothing needs to be added to the recipe for the SDB node. This will enable a PXE boot of the SDB node(s).
- Maybe, if the site needs to create a custom recipe for the SDB node (e.g., to add content for a workload manager), and the admin recipe can be used as a base. Create a custom recipe for the SDB node and add the admin recipe as a subrecipe. A PXE boot of the SDB node(s) may be possible if the resulting boot image size does not exceed the PXE boot size limit (the compressed initramfs must be 500MB or smaller).
- No, if the admin recipe is missing content that is needed for the custom SDB recipe. Use the service recipe as the base, instead. Create a custom recipe for the SDB node and add the service recipe as a subrecipe. A PXE boot of the SDB node(s) may be possible if the resulting boot image size does not exceed the PXE boot size limit.

For an example of creating and extending a recipe, see [Install Third-Party Software with a Custom Image Recipe](#) on page 417.

## 3.12 Where to Place the Root File System: tmpfs versus netroot

The Cray XC™ Series root file system for nodes can either reside in RAM (tmpfs) or be mounted from a network source (netroot), depending on the type of node. The boot and SDB nodes, all other service nodes (except login nodes), and all DAL (direct-attached Lustre) nodes must use tmpfs. Compute nodes and login nodes may use either tmpfs or netroot. Use the information provided here to decide whether to use netroot for some or all compute and login nodes at this site.

### About netroot and Dynamic Shared Objects and Libraries (DSL)

In releases prior to CLE 6.0 / SMW 8.0, the dynamic shared objects and libraries (DSL) feature was optional. It was necessary for many sites because it enabled both dynamic shared libraries and large network-based images, which were needed for systems with NVIDIA GPUs and for most production workloads.

In the current release, DSL is supported by default. Note, however, that the DSL feature no longer includes provision for large network-based images. That capability is now provided by netroot.

- Sites that require large network-based images and additional storage should use netroot.
- Sites using NVIDIA GPUs must use netroot.

## Comparison of tmpfs and netroot

**tmpfs** By default, the root file system on Cray XC™ Series systems resides in the memory resident file system, tmpfs.

tmpfs has these characteristics and limitations:

- always used for service nodes (except login nodes) and DAL (direct-attached Lustre) nodes
- efficient and fast root file system access
- large memory footprint
- file system content needs to be restricted to reduce memory footprint
- typically used when minimal commands and libraries required
- works well for compute nodes with well defined workloads and for service nodes that are used primarily for internal services

**netroot** netroot is an alternative approach that mounts the root file system from a network source. It is used only for compute and login nodes. It uses overlayfs to layer tmpfs on top of a read-only network file system.

Due to the reliance on overlayfs, the decision to use netroot should include consideration of the characteristics and limitations of overlayfs in addition to those of netroot listed here.

netroot has these characteristics and limitations:

- used only for compute and login nodes, never for service nodes (except login nodes)
- slower root file system access
- increased node boot time
- minimized memory footprint (mounted from network, so requires less disk space)
- no restriction on file system content
- typically used when a robust set of commands and libraries required (netroot enables large network-based images, formerly enabled through the DSL feature)
- works well for compute nodes with diverse workloads and for compute nodes with a high memory footprint
- always used for GPUs
- supports a SquashFS compressed image format for better boot performance (recommended)

This comparison of tmpfs and netroot memory footprints is based on a fresh install with nothing extra added. These numbers could be larger or smaller for a site depending on whether the Cray image recipes for tmpfs and netroot have been extended (by adding necessary RPMs) or reduced (by removing unnecessary RPMs).

Table 5. Comparison of tmpfs and netroot Memory Footprints

Image Type	Memory Consumption	Number of RPMs
Admin image root - tmpfs	1400 MB	600
Service image root – tmpfs	1700 MB	670
Login image root – tmpfs	3600 MB	1100
Compute image root – tmpfs	1500 MB	745
Login image root – netroot	125 MB	2500
Compute image root – netroot	150 MB	2380

### 3.13 About Image Groups and How to Customize Them

Image group configuration information is used by the `imgbuilder` command to build boot images. Image groups are defined in the global config set in the `cray_image_groups` configuration template (`/var/opt/cray/imps/config/sets/global/config/cray_image_groups.yaml`). This template is different than the other configuration templates in the following ways:

- It is the only configuration template that is not managed by the configurator. This means that when the configurator generates worksheets, it does not generate one for `cray_image_groups`. It also means that image group data is not accessible from within the configurator; to manage that data, `cray_image_groups.yaml` must be edited manually.
- It is included with the new set of templates in every SMW/CLE software installation and update; however, the installer will place the new `cray_image_groups.yaml` into the config set ONLY if an existing `cray_image_groups.yaml` is not already there. The existing one is never overwritten. This means that for a software update, the existing `cray_image_groups.yaml` must be renamed (moved) to make way for the new one, and the new one is then customized, using the renamed one as reference.

Here is an example of the contents of that file:

```
cray_image_groups:
  default:
    - recipe: "admin_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "admin{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-created{date}"
      export_format: "cpio"
      nims_group: "admin"
    - recipe: "compute_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "compute{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-x86_64-created{date}"
      arch: "x86_64"
      export_format: "cpio"
      nims_group: "compute"
    - recipe: "compute_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "compute{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-aarch64-created{date}"
      arch: "aarch64"
      export_format: "cpio"
      nims_group: "compute_aarch64"
    - recipe: "login_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "login{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-x86_64-created{date}"
      arch: "x86_64"
      export_format: "cpio"
      nims_group: "login"
    - recipe: "login_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "login{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-aarch64-created{date}"
      arch: "aarch64"
      export_format: "cpio"
      nims_group: "login_aarch64"
    - recipe: "service_cle_{cle_release_lowercase}_sles_12sp3_ari"
```

```

    dest: "service{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-created{date}"
    export_format: "cpio"
    nims_group: "service"
  - recipe: "diags_cle_{cle_release_lowercase}_sles_12sp3_ari"
    dest: "diags_cle_{cle_release_lowercase}_sles_12sp3_x86-64"
    arch: "x86_64"
  - recipe: "diags_cle_{cle_release_lowercase}_sles_12sp3_ari"
    dest: "diags_cle_{cle_release_lowercase}_sles_12sp3_aarch64"
    arch: "aarch64"
  - recipe: "eloin-smw_cle_{cle_release_lowercase}_sles_12sp3_ari"
    dest: "eloin-smw{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-created{date}"
    export_format: "squashfs"
...
testing:
  - recipe: "compute_cle_{cle_release_lowercase}_sles_12sp3_ari"
    dest: "{my_custom_prefix}_compute-TEST-{my_other_value}_{date}_{time}"
    arch: "x86_64"
    export_format: "cpio"
    nims_group: "compute-test"

```

The only way to modify this information to customize it for a site is to edit this YAML file directly.

```
smw# vi /var/opt/cray/imps/config/sets/global/config/cray_image_groups.yaml
```

The following sections describe important things to know to successfully customize the `cray_image_groups` configuration file.

## What image groups contain

- The `cray_image_groups` configuration file can contain multiple image groups (this example shows two: `default` and `testing`). When invoked, `imgbuilder` builds one of these. It builds `default` if no image group name is passed as a parameter.
- Each image group contains a list of image specifications that will be built: by default, the standard admin, compute, service, and login images.
- Each image specification is a stanza containing these fields:

<b>recipe</b>	An IMPS (Image Management and Provisioning System) image recipe name.
<b>dest</b>	The destination file name used for the IMPS image root that will be created from the recipe. This can be customized as described below.
<b>arch</b>	The architecture to be used for the image root. If not specified, the recipe's default architecture will be used. Values supported are <code>x86_64</code> and <code>aarch64</code> .
<b>export_format</b>	<p>The format in which the image root will be exported. Values supported are <code>cpio</code> (used for booting all internal CLE nodes) and <code>squashfs</code> (used for SMW-managed eLogin nodes).</p> <p>The <code>export_format</code> field is specified only for image roots that will be used to create boot images, so not all specifications have this field (for example, the <code>diags</code> image in the <code>default</code> image group does not). If omitted, the image will not be exported.</p>
<b>nims_group</b>	<p>The NIMS group to which the boot image is mapped.</p> <p>The <code>nims_group</code> field is specified only for internal CLE nodes with image roots that will be used to create boot images. Note that in the <code>default</code> image group, the <code>diags</code> image specification does not have a <code>nims_group</code> because that image is not used to boot a node, and the eLogin image specification does not have a <code>nims_group</code> because it is for external nodes, not internal CLE nodes.</p>

**NOTE:** The NIMS group specified here must be the same as the NIMS group assigned (using the `cnode` command) to the nodes that will use this image.



## How to customize an image root file name using placeholders

Placeholders like {date} can be used to customize an image root name. The `dest` values in the above example contain several such placeholders. At build time, relevant values are substituted for these placeholders. Currently, `imgbuilder` supports the following built-in placeholders for use in the `cray_image_groups` configuration file:

<b>{date}</b>	the current system date (e.g., 20140314)
<b>{time}</b>	the current system time (e.g., 134514)
<b>{host}</b>	the current system host name
<b>{user}</b>	the current username
<b>{cle_release}</b>	name of the currently active CLE release in uppercase
<b>{cle_release_lowercase}</b>	name of the currently active CLE release in lowercase
<b>{cle_build}</b>	the currently active CLE build
<b>{patch}</b>	the currently active patch

**IMPORTANT:** When adding one or more placeholders to `dest`, ensure that the whole expression is enclosed by double quotes. For example,

```
dest: "
service{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-created{date}"
```

User-defined placeholders (optional) are also supported for further customization of image names. An example of a user-defined placeholder is {note}, which Cray has added to the image root name in several of the standard image specifications. {note} does not need to be defined in order for the image specifications to work; however, if a site wishes to add something more to the image root file names that contain {note}, a value for {note} can be specified on the command line when running `imgbuilder`, and substitution occurs at runtime. For example, if a site wanted to add the string "favorite" to those image root names, the following command could be used.

```
(EXAMPLE ONLY - DO NOT USE) smw# imgbuilder --map -- note=favorite
```

Other custom placeholders can be defined as well. As with {note}, the key/value pair defining the placeholder would be added to the `imgbuilder` command on the command line. The syntax is two hyphens and a space (`--` ) followed by any number of placeholder definitions as `key=value` pairs separated by spaces.

For example, this command would tell `imgbuilder` to build the images in the `testing` image group, map them to the NIMS groups specified in that group, and substitute "foo" everywhere for "my\_custom\_prefix" and "bar" everywhere "my\_other\_value" appears.

```
(EXAMPLE ONLY - DO NOT USE) smw# imgbuilder --map --image-group testing \
-- my_custom_prefix=foo my_other_value=bar
```

## 3.14 About Parallel Image Creation

The `imgbuilder` command validates recipes and builds/exports images for all of the image specifications in the default image group in `cray_image_groups.yaml`. If an image group is specified on the command line, `imgbuilder` will validate recipes and build/export images for all of the image specifications in the specified image group. Depending on how many image specifications there are in the group, and how extensive the recipes are, this can take an hour or more.

Beginning with CLE 6.0.UP07, the recipe validation and image build/export tasks can be performed in parallel, which can greatly reduce the elapsed time required for those tasks. Use one or both of the following methods to do them in parallel:

- Method 1: If images are built automatically when the SMW/CLE installer is run, then in `install.cle.conf`, set the value of `build_images_processes` to the number of images to be built concurrently.
- Method 2: If images are built by running `imgbuilder` directly, add the `--processes=NUM` option on the command line, replacing `NUM` with the number of images to be built concurrently.

The number of concurrent processes is capped at the number of CPUs on the SMW, as reported by `lscpu`, even if the number specified exceeds that cap.

Note that the time saved by performing the recipe validation and image build/export tasks in parallel will depend on the configuration of the SMW (especially with regard to I/O) and the number of processes specified. Experiment to find the optimum number for this system, which may be less than the CPU cap.

## Method 1: Specify Number of Processes in `install.cle.conf`

If `build_images` is set to `yes` in the `install.cle.conf` file, as in the following example, then when the installer is run, it will invoke `imgbuilder` automatically during the SMW/CLE software installation.

```
build_images: yes
```

To validate recipes and build/export images in parallel when the installer is run, edit the `install.cle.conf` file and change the value of the following setting.

```
build_images_processes: 1
```

The default value is 1, which is equivalent to building images serially. Replace 1 with the number of images to be built concurrently (8 in the following example).

```
build_images_processes: 8
```

Note that this setting in `install.cle.conf` affects the behavior of `imgbuilder` ONLY when `imgbuilder` is invoked by the installer.

## Method 2: Specify Number of Processes with `imgbuilder` Command

To validate recipes and build/export images in parallel with `imgbuilder`, use the following command.

```
smw# imgbuilder --map --processes=NUM
```

Replace `NUM` with the number of images to be built concurrently. Omitting this option is equivalent to using `--processes=1`, resulting in recipe validation and image build/export tasks being performed serially.

The following example shows the output when `--processes=16` is specified. The lines with `###` were added to show how all of the recipes in the default image group are validated in parallel, and then all of the images for the default image group are created and exported in parallel. Actual output does not include the `###` lines.

```
smw# imgbuilder --map --processes=16
Full output will be captured in /var/adm/cray/logs/imgbuilder/imgbuilder.20180329.log
Using 16 process(es) for image building.
Using configuration in ./cray_image_groups.yaml

#####
### the following lines show recipes being validated in parallel
#####
Validating image recipes in group: 'default'.
Recipe diag_cle_6.0.up07_sles_i2sp3 took 0min 8sec to validate.
```

```

Recipe dal_cle_6.0.up07_centos_6.5_ari took 0min 8sec to validate.
Recipe initrd-compute-large_cle_6.0.up07_sles_12sp3_ari took 0min 11sec to validate.
Recipe initrd-login-large_cle_6.0.up07_sles_12sp3_ari took 0min 16sec to validate.
Recipe admin_cle_6.0.up07_sles_12sp3_ari took 0min 18sec to validate.
Recipe service_cle_6.0.up07_sles_12sp3_ari took 0min 26sec to validate.
Recipe compute-large_cle_6.0.up07_sles_12sp3_ari took 0min 29sec to validate.
Recipe elogin-smw-large_cle_6.0.up07_sles_12sp3_ari took 0min 32sec to validate.
Recipe login-large_cle_6.0.up07_sles_12sp3_ari took 0min 41sec to validate.

#####
### the following lines show images being created and exported in parallel
#####
Building images in group: 'default'.
Image diags_cle_rhine_sles_12_x86_64 took 8min 15sec to build.
Image initrd-compute-large_cle_6.0.up07-build6.0.7128_sles_12sp3-x86_64-created20180329 took 10min 48sec to build.
Image initrd-login-large_cle_6.0.up07-build6.0.7128_sles_12sp3-x86_64-created20180329 took 10min 50sec to build.
Image initrd-compute-large_cle_6.0.up07-build6.0.7128_sles_12sp3-x86_64-created20180329 took 0min 47sec to export.
Image initrd-login-large_cle_6.0.up07-build6.0.7128_sles_12sp3-x86_64-created20180329 took 1min 29sec to export.
Image dal_cle_6.0.up07-build6.0.7128_centos_6.5-created20180329 took 13min 55sec to build.
Image admin_cle_6.0.up07-build6.0.7128_sles_12sp3-created20180329 took 15min 12sec to build.
Image dal_cle_6.0.up07-build6.0.7128_centos_6.5-created20180329 took 2min 12sec to export.
Image service_cle_6.0.up07-build6.0.7128_sles_12sp3-created20180329 took 16min 41sec to build.
Image admin_cle_6.0.up07-build6.0.7128_sles_12sp3-created20180329 took 3min 37sec to export.

```

### 3.15 About Image Pushes: push versus sqpush

Netroot, diags, and PE image roots must be staged to the boot node before booting the XC system during a fresh install or software update. This staging is necessary because these image roots are projected from the boot node to tier2 nodes, and from tier2 nodes to the rest of system according to the tiers set up in Cray Scalable Services.

There are two commands that can be used to push an image root from the SMW to the boot node: `image push` and `image sqpush`. They differ in the following ways:

*Table 6. Comparison of image push versus image sqpush*

	image push	image sqpush
what is pushed	transfers image root as a file system (tree)	transfers image root as a single SquashFS file
how it is pushed	copies the image root; if changes made to image root, copies only the changed files/directories ( <code>rsync</code> )	copies the SquashFS file; if changes made to image root, copies entire new SquashFS file to replace the existing one
how it is accessed after push	entire file system is projected by DVS, resulting in many metadata operations and slower access	single SquashFS file is projected by DVS, resulting in many fewer metadata operations and quicker access
what space is needed on boot node	consumes space equal to size of image root file system	compressed file consumes less space

#### Which command should sites use?

Cray recommends using the `image sqpush` command in most cases. It is better for large deployments because the dramatically reduced number of metadata operations reduces overall boot times and enables faster file system access. In the case of SMW-managed eLogin, there is no choice. The `image sqpush` command **MUST** be used to push eLogin images to eLogin nodes.

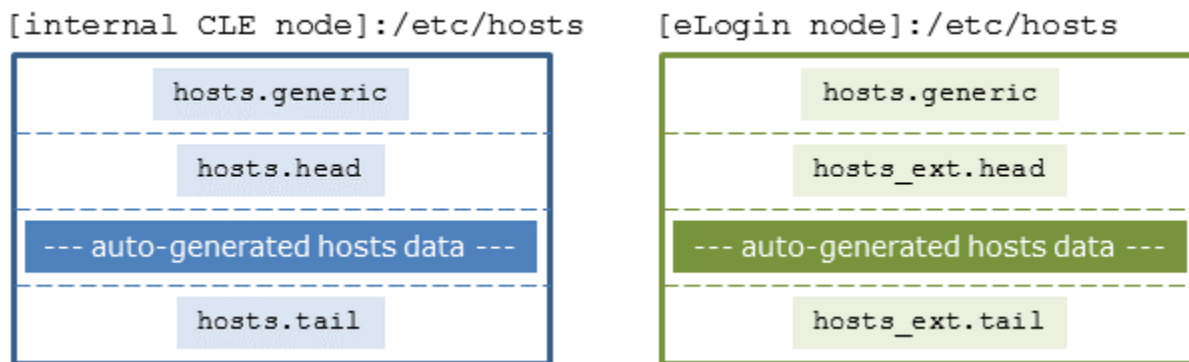
The `image push` command is still an option for development situations, where an image is modified and transferred repeatedly to test the modifications, and the ability to transfer only what has changed provides more rapid turnaround.

### 3.16 About the /etc/hosts File

Cray system management software uses Hardware Supervisory System (HSS) and config set data to automatically generate the `/etc/hosts` file with authoritative data for installation on internal CLE nodes in a Cray XC system and for installation on eLogin nodes connected to that system.

Sites often need to edit `/etc/hosts` to add their own data, but because these files are generated each time the CLE config set is updated, those additions could be overwritten. To address this issue for both internal CLE nodes and eLogin nodes, Cray provides a way to separate and preserve site data in the `/etc/hosts` file. That file is composed of four distinct components: the auto-generated hosts data and three additional files that can be modified by sites, as described in the following figure and list.

Figure 3. Components of the Hosts File for Internal CLE Nodes and eLogin Nodes



- `hosts.generic`  
Contains generic entries from Cray. The contents are placed at the top of the `hosts` file.
- `hosts.head`  
Contains site entries. The contents are placed above the auto-generated hosts data in the `hosts` file that will be installed on internal CLE nodes.
- `hosts_ext.head`  
Contains site entries. The contents are placed above the auto-generated hosts data in the `hosts` file that will be installed on eLogin nodes.
- Auto-generated hosts data  
Generated from HSS and configuration data. Note that site entries in this component of the `hosts` file (comments and other changes) are no longer preserved. To add extra labels to a host entry line, add them to the configured aliases list of that host definition in the `cray_net` configuration service.
- `hosts.tail`  
Contains site entries. The contents are placed below the auto-generated hosts data in the `hosts` file that will be installed on internal CLE nodes.
- `hosts_ext.tail`  
Contains site entries. The contents are placed below the auto-generated hosts data in the `hosts` file that will be installed on the eLogin node.

These component files are located in the following directory on the SMW:

```
/var/opt/cray/imps/config/sets/<CONFIG_SET>/files/roles/common/etc/
```

If the head and/or tail files are not present when the CLE config set is updated, the missing file(s) will be automatically generated and will be empty. Also, when the config set is updated, if there is no backup hosts file, the old hosts files for that config set will be automatically saved as `hosts.autobk`.

Cray recommends using the head and tail files to supply entries for nodes that cannot be defined in either the `cray_global_net` or `cray_net` configuration services. Do not add an entry to a head/tail file that may conflict with entries generated automatically from `cray_global_net` and `cray_net` configuration data.

## Action May be Needed

If a site manually entered data in `/etc/hosts` prior to CLE 6.0.UP06, those entries may need to be migrated to:

- `hosts.head` and/or `hosts.tail` for internal CLE nodes
- `hosts_ext.head` and/or `hosts_ext.tail` for eLogin nodes

Sites are prompted to do this at the appropriate places in the fresh install and software update processes.

## Examples

The following example shows the `hosts.generic` file provided by Cray. Sites can modify this data, and those modifications will be preserved during config set updates.

```
#
# hosts          This file describes a number of hostname-to-address
#                mappings for the TCP/IP subsystem.  It is mostly
#                used at boot time, when no name servers are running.
#                On small systems, this file can be used instead of a
#                "named" name server.
# Syntax:
#
# IP-Address    Full-Qualified-Hostname  Short-Hostname
#
127.0.0.1      localhost

# special IPv6 addresses
::1            localhost ipv6-localhost ipv6-loopback

fe00::0        ipv6-localnet

ff00::0        ipv6-mcastprefix
ff02::1        ipv6-allnodes
ff02::2        ipv6-allrouters
ff02::3        ipv6-allhosts
```

The following example shows typical auto-generated hosts data. This is regenerated each time the config set is updated. Sites can affect this content only through configuration data.

```
10.128.0.1      nid00000 c0-0c0s0n0 rsip rsip1 fireball-rsip
10.128.0.2      nid00001 c0-0c0s0n1
10.128.0.3      nid00002 c0-0c0s0n2
10.128.0.4      nid00003 c0-0c0s0n3 dvs dvs1
10.128.0.5      nid00004 c0-0c0s1n0 login fireball1 fireball-login
10.128.0.6      nid00005 c0-0c0s1n1 lnet lnet1
```

The following example shows the four components of an `/etc/hosts` file. The comments indicating the beginning and ending of each component are for illustrative purposes only; they will not appear in an actual `/etc/hosts` file.

```
# hosts.generic inserted here:
...
127.0.0.1      localhost
...
# hosts.generic ends here
# hosts.head inserted here:

172.10.1.123  myhost.us.cray.com

# hosts.head ends here

10.128.0.1      nid00000 c0-0c0s0n0 rsip rsip1 fireball-rsip
10.128.0.2      nid00001 c0-0c0s0n1
10.128.0.3      nid00002 c0-0c0s0n2
10.128.0.4      nid00003 c0-0c0s0n3 dvs dvs1
10.128.0.5      nid00004 c0-0c0s1n0 login fireball fireball-login
...

# hosts.tail inserted here
...
# hosts.tail ends here
```

## 4 Install and Configure SMW/CLE Software

---

Follow the procedures in this chapter to perform a fresh install of CLE 6.0.UP07 / SMW 8.0.UP07 on a Cray XC™ Series system.

Use [Master Checklist: Install and Configure New SMW/CLE Software](#) on page 440 to track progress through the fresh install process.



**WARNING:** When a fresh install is performed on a system, disks are wiped clean. To prevent loss of necessary data, before beginning any installation procedures, consider what configuration files, log files, or other files should be preserved, and save them in a location unaffected by the installation.

**SMW HA only:** For a system with two SMWs that will be configured for SMW high availability (HA), the process to install the first SMW is the same as for a system with a stand-alone SMW, with a few minor differences that are noted in this guide. However, installing the second SMW uses a completely different process. Do not use this guide for the second SMW. For more information, see *XC™ Series SMW HA Installation Guide* (S-0044).

### 4.1 Prepare for an SMW/CLE Fresh Install

In preparation for a fresh install, do the following:

- If there is any data that should be saved (configuration files, log files, etc.) before the SMW disks are wiped clean by this fresh install, use the procedure in [Back Up Site Data](#) on page 404 now.
- Extract the configuration worksheets in preparation for entering site data. Example global and CLE configuration worksheets are provided in the following file on Crayport:

```
/cray/css/release/package/release/CLE/6.0.UP07/up07.sample.worksheets.tgz
```

- Read the *SMW Release Errata* and the *SMW README* provided with the SMW release package for any additional installation-related requirements, corrections to this guide, and other relevant information about the release package.
- Read the *CLE Release Errata* and the *CLE README* provided with the CLE release package for any additional installation-related requirements, corrections to this guide, and other relevant information about the release package.
- Read the Field Notices (FN) related to kernel security fixes to identify any changes to this release package. Apply any needed changes before installing the new software.

When those preparation activities are done, complete preparation for the fresh install by collecting information, verifying network connections, and becoming familiar with disk space and other requirements using the topics that follow.



### 4.1.1 Information to Collect Before Installation

#### SMW Information

This information will be needed to update the global config set during configuration.

- Network base IP address for SMW eth0
- Netmask for SMW eth0
- Gateway IP address for SMW eth0
- List of IP addresses to use as DNS server
- List of domains to use in the DNS search path for hosts attached to SMW eth0 network
- List of NTP servers
- Host name of the SMW: both the short name and the fully qualified domain name (FQDN)
- IP address of SMW eth0

#### Hardware Information

When `xtdiscover` is used to discover XC system hardware, it will prompt for this information.

- Maximum cabinet size in the X dimension
- Maximum cabinet size in the Y dimension
- Network topology class (0 or 2 for Cray XC Series liquid-cooled systems, 0 for Cray XC Series air-cooled systems: XC30-AC, XC40-AC)
- Primary boot node and backup boot node (if enabling boot node failover)—must not be on same blade in case of blade failure
- Primary SDB node and backup SDB node (if enabling SDB node failover)—must not be on same blade in case of blade failure



**CAUTION:** The system will fail if a blade containing both the boot node and the SDB node fails, because Cray does not support concurrent failover of boot and SDB nodes. Therefore, the boot and SDB nodes and their backups (for boot/SDB node failover) must be on different blades.

#### Service Node Roles

The XC system being installed and configured must have service nodes designated to function in some or all of the following roles. A node may have more than one role (e.g., boot and tier1). The system at this site may not require all of these roles.

- boot
- SDB
- login
- tier1 (boot node and SDB node)
- tier2 (see Tier2 Node FAQ)
- LNet router to external Lustre server
- realm-specific IP (RSIP) nodes
- DataWarp-managed nodes with SSD hardware
- DataWarp API gateway nodes

nodes providing a role for a workload manager (WLM)  
DVS servers to an external file system  
Direct-attached Lustre (DAL) MGS, MDS, or OSS nodes  
compute node repurposed to be a service node

## Tier2 Node FAQ

- Q. How many tier2 nodes are needed?** **A.** At least one server must be provided, and for resiliency, two nodes placed on different blades. The recommended ratio of tier2 nodes (servers) to tier3 nodes (clients) is 1 to 400.
- Q. Will adding more tier2 nodes help performance?** **A.** Adding more tier2 nodes does not always yield additional performance and is subject to diminishing returns.
- Q. What kind of node can be used as a tier2 node?** **A.** Use these:
- OPTIMAL: dedicated repurposed compute nodes (RCN)
  - dedicated service nodes
  - nodes with uniform light to moderate load
  - nodes with relatively homogeneous single core speeds to reduce resource contention disparity during periods of partial availability
- AVOID these (will result in sub-optimal performance):
- nodes with slower individual CPU cores, such as Intel® Xeon Phi™ "Knights Landing" (KNL) processors
  - direct-attached Lustre (DAL) servers
  - RSIP (realm-specific IP) servers
  - login nodes
- Q. Can a tier2 node have more than one role?** **A.** Small test and development systems (TDS) may use tier2 nodes that have additional roles, but generally, it is better for tier2 nodes to be dedicated.
- Q. Where should tier2 nodes be placed?** **A.** Distribute nodes throughout the system (on different blades) for resiliency in the event of hardware failure.

## Service Node Network Information

For each service node with a network interface, either Ethernet or InfiniBand, collect this information.

- For each network defined:**
- unique identifier for the network (management, login, lnet)
  - description or notes about the network (e.g. "Network to external Lustre")
  - network base IP address
  - netmask
  - gateway IP address
- For each network**
- unique identifier for each interface (primary\_ethernet, eth0, eth1, eth2, eth3, ib0, ib1, etc.) on this host

- interface added to a host**
- device name for the interface (eth0, ib1, etc.)
  - description or notes about the interface (e.g., "Ethernet connecting boot node to SMW")
  - any host name aliases by which this node should be known
  - name of the network to which this interface belongs (see list of networks defined above)
  - IPv4 network address for the interface

### 4.1.2 Network Connections

The following network connections are required.

- A stand-alone SMW with a single quad-ethernet card has these private network connections:
  - eth0 - To the customer/management network
  - eth1 - To the Hardware Supervisory System (HSS) network
  - eth2 - Used for SMW HA (failover) heartbeat 1 network
  - eth3 - To the boot and SDB nodes (the admin network)
- An SMW configured for SMW failover (SMW HA) has a second quad-ethernet card with these connections:
  - eth4 - Used for SMW HA heartbeat 2 network
  - eth5 - Used for SMW HA PostgreSQL hot-standby replication
  - eth6 - Used for SMW-managed eLogin (external-ipmi-net)
  - eth7 - Used for SMW-managed eLogin (external-management-net)

Things to note about network connections:

- Ethernet port assignments are valid only after the SMW software installation completes.
- The SMW must have a Fibre Channel or serial attached SCSI (SAS) connection to the boot RAID.
- A boot node must have a Fibre Channel or SAS connection to the boot RAID. If boot node failover is enabled or there are multiple logical CLE partitions, then each boot node should have such a connection to the boot RAID.
- A service database (SDB) node must have a Fibre Channel or SAS connection to the boot RAID. If SDB node failover is enabled or there are multiple logical CLE partitions, then each SDB node should have such a connection to the boot RAID.
- Each boot node and SDB node (including backup nodes) must have an Ethernet connection to the network shared with the SMW in order to PXE boot and transfer data as a tier1 node.

**IMPORTANT:** The SMW must be disconnected from the boot RAID before the initial installation of the SLES software.

**IMPORTANT:** Ensure that the Fibre Channel optic cable connectors or SAS cable connectors have protective covers when disconnected from the SMW, boot node, SDB node, or boot RAID.

### 4.1.3 SMW Internal Disk Requirements

Internal SMW disks are used for the boot disk (with optional RAID1 mirroring between two boot drives) and the power management disk (PMDISK).

The PMDISK requires a minimum of 500 GB. This may be a fresh disk or a repurposed disk on an existing SMW. The PMDISK will be allocated to `/var/lib/pgsql` in an ext4 file system.

The boot disk (or pair of boot disks in RAID1 configuration) requires a minimum of 160 GB, but may be larger. If a RAID1 mirror is enabled, the drives in the RAID1 configuration must be the same size. The boot disk has 4 GB allocated to `/boot` in an ext3 file system, 32 GB for swap, and the rest of the disk for the root (`/`) file system in a Btrfs file system.

*Table 7. SMW Internal Disk Requirements*

Mount Point	FS Type	Disk	Size	Description
<code>/boot</code>	ext3	boot	4 GB	Boot information
<code>swap</code>	swap	boot	32 GB	SMW swap
<code>/</code>	Btrfs	boot	120+ GB	root file system of SMW with Btrfs subvolumes
<code>/var/lib/pgsql</code>	ext4	power management	1000+ GB	Power Management disk

#### 4.1.4 Configuration Values

The following IP addresses are set by default and are not site dependent.

*Table 8. Default IP Addresses*

IP Address	Description
10.1.0.1	Primary boot RAID controller
10.1.0.2	Secondary boot RAID controller
10.1.0.15	Storage RAID controller
10.1.1.1	SMW eth1 - HSS network
10.2.1.1	(SMW HA only) SMW eth2 - SMW HA heartbeat 1
10.3.1.1	SMW eth3 - admin network
10.3.1.253	SDB node
10.3.1.254	boot node
10.4.1.1	(SMW HA only) SMW eth4 - SMW HA heartbeat 2
10.5.1.1	(SMW HA only) SMW eth5 - SMW HA PostgreSQL hot-standby replication
127.0.0.1	localhost (loopback)

The following configuration values are site dependent. Record the actual values for the installation site in the third column.

Table 9. Site-dependent Configuration Values

Description	Example Value	Actual Value
SMW hostname	smw	
Domain	cray.com	
Aliases	cray-smw smw1	
Customer network IP address	192.168.78.68	
Customer network netmask	255.255.255.0	
Default gateway	192.168.78.1	
Domain names to search	us.cray.com mw.cray.com	
Nameserver IP address	10.0.73.30 10.0.17.16	
iDRAC hostname	cray-drac	
iDRAC IP address	192.168.78.69	
iDRAC Subnet Mask	255.255.255.0	
iDRAC Default GW	192.168.78.1	
Timezone	US/Central	
NTP servers	ntpghost1 ntpghost2	
X dimension	1-64	
Y dimension	1-32	
Topology Class	0, 2 (see note below)	

**NOTE:** Regardless of the number of cabinets in the system, Cray XC Series air-cooled systems must be set to topology class 0. Cray XC Series liquid-cooled systems can be topology class 0 or 2.

### 4.1.5 Passwords

The following default account names and passwords are used throughout the initial software installation process. Cray recommends changing these default passwords during the installation and configuration process at appropriate times before the SMW or network CLE nodes are connected to networks that are external to the XC system.

Table 10. Default System Passwords

Account Name	Password
root	initial0
crayadm	crayadm
mysql	None; a password must be created
root (iDRAC)	initial0

## 4.2 Install the Base Operating System on the SMW

The base operating system must be installed on the SMW before the Cray SMW and CLE software release packages can be installed. To install the base OS and configure the boot RAID, use the procedures and reference topics in this section, beginning with [Prepare to Install the Base Linux Distribution](#) on page 52.

Use [Installation Checklist 1: Install the Base Operating System on the SMW](#) on page 440 to track progress through this part of the fresh install process.

Note that Cray provides two rack-mount SMW models: the Dell PowerEdge™ R815 Rack Server and the Dell PowerEdge™ R630 Rack Server. Earlier desktide SMW hardware is not supported. The figure below shows an easy way to distinguish between the two rack-mount models when viewing them from the front.

*Figure 4. Distinguishing Features of Dell R815 and R630 Servers*



Dell R815: 2U high and 6 drive bays



Dell R630: 1U high and 8 drive bays

### 4.2.1 Prepare to Install the Base Linux Distribution

#### About this task

A full initial installation begins with installing the base operating system. This procedure provides initial steps that are common to installing the base OS on both Dell R815 and R630 SMW models.

#### Procedure

1. Disconnect the SMW connection to the boot RAID.  
Disconnect the data cables and place protective covers on the fibre optic cable connectors (if present).
2. Connect the SMW keyboard, monitor, and mouse.  
Connect a keyboard, monitor, and mouse to the USB and monitor connectors on the SMW, if not already connected. Once the iDRAC has been configured, the keyboard, monitor, and mouse can be connected to the iDRAC for remote console activities instead of being directly connected to the SMW console.

As the next step in preparing to install the base OS, do one of the following, depending on the SMW model:

- **Dell R815 SMW.** Configure the BIOS and iDRAC. Proceed to [Dell R815 SMW: Change the BIOS and iDRAC Settings](#) on page 53.

- **Dell R630 SMW.** First configure the SMW RAID, then configure the BIOS and iDRAC. Proceed to [Dell R630 SMW: Configure the RAID Virtual Disks](#) on page 60.

#### 4.2.1.1 Dell R815 SMW: Change the BIOS and iDRAC Settings

### Prerequisites

This procedure assumes the following:

- The SMW is disconnected from the boot RAID.
- The SMW is connected to a keyboard, monitor, and mouse (without this direct connection, some procedure instructions will not work as intended).

### About this task

This procedure changes the system setup for a Dell R815 SMW: the network connections, remote power control, and the remote console. Depending on the server model and version of BIOS configuration utility, there may be minor differences in the steps to configure the system. For more information, refer to the documentation for the Dell server used at this site. Because Cray ships systems with most of the installation and configuration completed, some of these steps may have been done already.

For a Dell R630 SMW, see [Dell R630 or R640 SMW: Change the BIOS and iDRAC Settings](#) on page 68.

### Procedure

1. Remove SMW non-boot internal drives.

Eject all the internal disk drives from the SMW except for the primary boot disk in slot 0 and the secondary boot disk in slot 1.

2. Power up the SMW. When the BIOS power-on self-test (POST) process begins, **quickly press the F2 key** after the following messages appear in the upper-right of the screen.

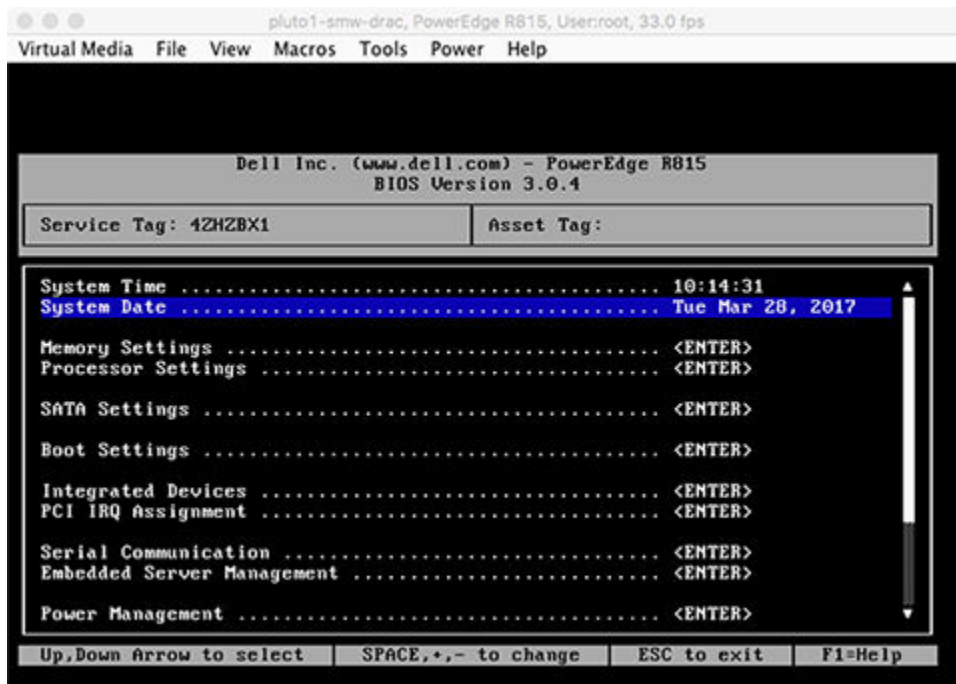
```
F2 = System Setup
F10 = System Services
F11 = BIOS Boot Manager
F12 = PXE Boot
```

When the **F2** keypress is recognized, the **F2 = System Setup** line changes to **Entering System Setup**.

After the POST process completes and all disk and network controllers have been initialized, the BIOS **System Setup** menu appears.

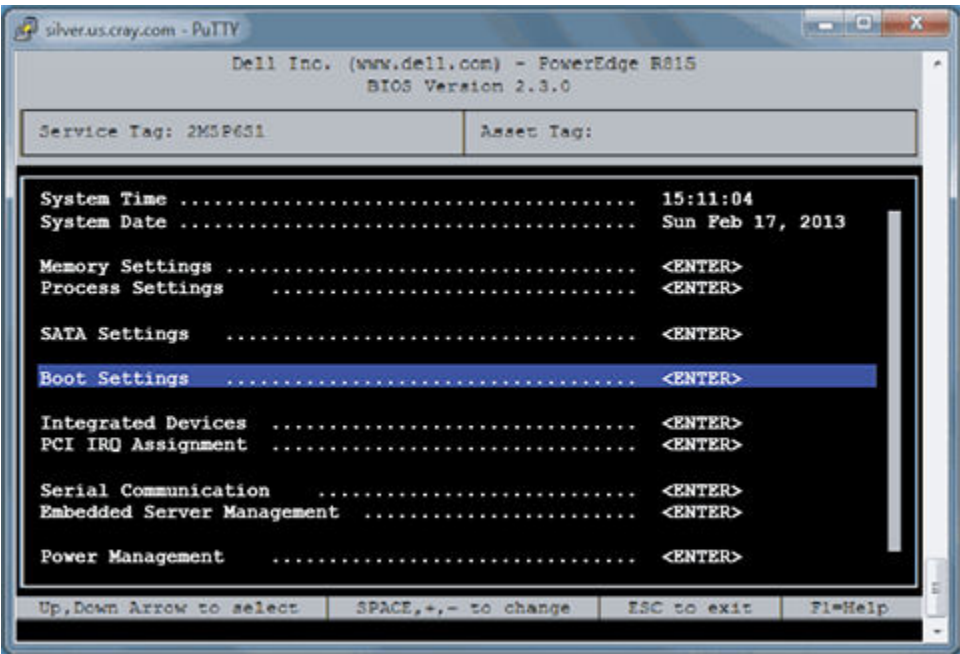


Figure 5. Dell R815 SMW BIOS System Setup Menu



3. Change system time, if necessary.
  - a. Select **System Time** in the **System Setup** menu.  
The hours will be highlighted in blue.
  - b. Set the correct time in UTC, not in the local time zone.
    1. Press the space key to change hours.
    2. Use the right-arrow key to select minutes, then change minutes with the space key.
    3. Use the right-arrow key to select seconds, then change seconds with the space key.
  - c. Press **Esc** when the correct time is set.
4. Change boot settings.
  - a. Select **Boot Settings** in the **System Setup** menu, then press **Enter**.

Figure 6. Dell R815 SMW Boot Settings Menu

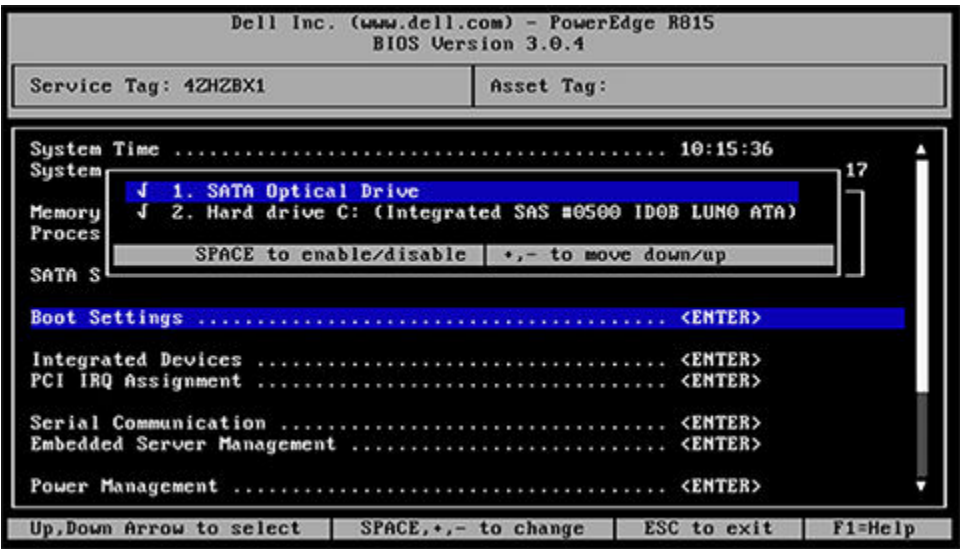


A pop-up menu with the following list appears:

```
Boot Mode ..... BIOS
Boot Sequence ..... <ENTER>
USB Flash Drive Emulation Type..... <ENTER>
Boot Sequence Retry ..... <Disabled>
```

- b. Select **Boot Sequence**, then press **Enter**.

Figure 7. Dell R815 SMW Boot Sequence Settings



- c. Change the order of items in the **Boot Sequence** list so that the optical (DVD) drive appears first, then the hard drive. If **Embedded NIC** appears in the list, it should end up below the optical drive and hard drive in the list.

- d. Disable embedded NIC.  
If **Embedded NIC** is in the list, select it, and then use the space key to disable it.
- e. Press **Esc** to exit the **Boot Sequence** menu.
- f. Press **Esc** again to exit the **Boot Settings** menu.

5. Change serial communication.

- a. Select **Serial Communication** in the **System Setup** menu, then press **Enter**.
- b. Confirm these settings in the **Serial Communication** menu.
  - **Serial Communication** is set to **On with Console Redirection via COM2**
  - **Serial Port Address** is set to **Serial Device1=COM2, Serial Device2=COM1**
  - **External Serial Connector** is set to **Serial Device2**
  - **Failsafe Baud Rate** is set to **115200**
- c. Press **Esc** to exit the **Serial Communication** menu.

6. Select **Embedded Server Management** in the **System Setup** menu, then press **Enter**.

The **Embedded Server Management** pop-up menu with the following list appears:

```
Front-Panel LCD Options .....User-Defined String
User-Defined LCD String ..... <ENTER>
```

- a. Set **Front-Panel LCD Options** to **User-Defined String**.
- b. Set **User-Defined LCD String** to the login host name (e.g., cray-drac), then press **Enter**.
- c. Press **Esc** to exit the **Embedded Server Management** menu.

7. Insert base operating system DVD into SMW.

Insert the base OS DVD (Cray-slebase-12-SP3-201709141039.iso) into the DVD drive. (The DVD drive on the front of the SMW may be hidden by a removable decorative bezel.)

8. Save BIOS changes and exit.

- a. Press **Esc** to exit the BIOS **System Setup** menu.

A menu with a list of exit options appears.

```
Save changes and exit
Discard changes and exit
Return to Setup
```

- b. Ensure that **Save changes and exit** is selected, then press **Enter**.

The SMW resets automatically.

9. Enter BIOS boot manager.

- a. When the BIOS POST process begins again, **quickly press the F11 key** within 5 seconds of when the following messages appear in the upper-right of the screen.

```
F2 = System Setup
F10 = System Services
```

F11 = BIOS Boot Manager  
F12 = PXE Boot

When the **F11** keypress is recognized, the **F11 = BIOS Boot Manager** line changes to **Entering BIOS Boot Manager**.

###### 10. Change the integrated Dell Remote Access Controller (iDRAC) settings.

Watch the screen carefully as text scrolls until the **iDRAC6 Configuration Utility 1.57** line is visible. When the line **Press <Ctrl-E> for Remote Access Setup within 5 sec...** displays, press **Ctrl-E** within 5 seconds.

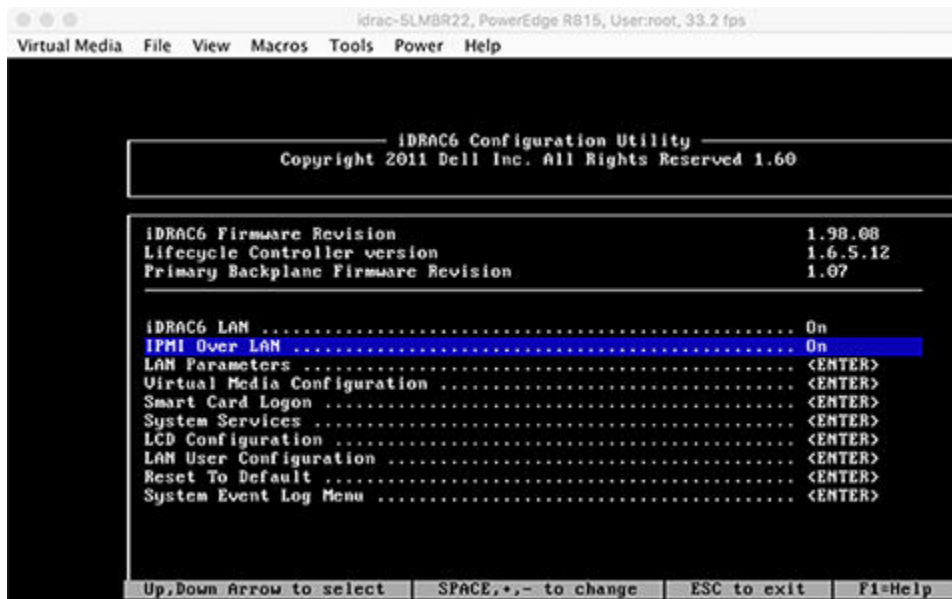
```

0 5 0 ATA WDC WD5000BPVT-0 1A01 465 GB
LSI Corporation MPT2 boot ROM successfully installed!
iDRAC6 Configuration Utility 1.57
Copyright 2010 Dell Inc. All Rights Reserved
iDRAC6 Firmware Revision version: 1.54.15
Primary Backplane Firmware Revision 1.07
-----
IPv6 Settings
-----
IPv6 Stack : Disabled
Address 1 : ::
Default Gateway : ::
-----
IPv4 Settings
-----
IPv4 Stack : Enabled
IP Address : 172. 31. 73.142
Subnet mask : 255.255.255. 0
Default Gateway : 172. 31. 73. 1
Press <Ctrl-E> for Remote Access Setup within 5 sec...
```

The **iDRAC6 Configuration Utility** menu appears.

###### 11. Set iDRAC6 LAN to ON.

*Figure 8. Dell R815 SMW iDRAC6 Configuration Utility Menu*



**12. Set IPMI Over LAN to ON.****13. Configure the iDRAC LAN parameters.**

Select **LAN Parameters**, then press **Enter**.

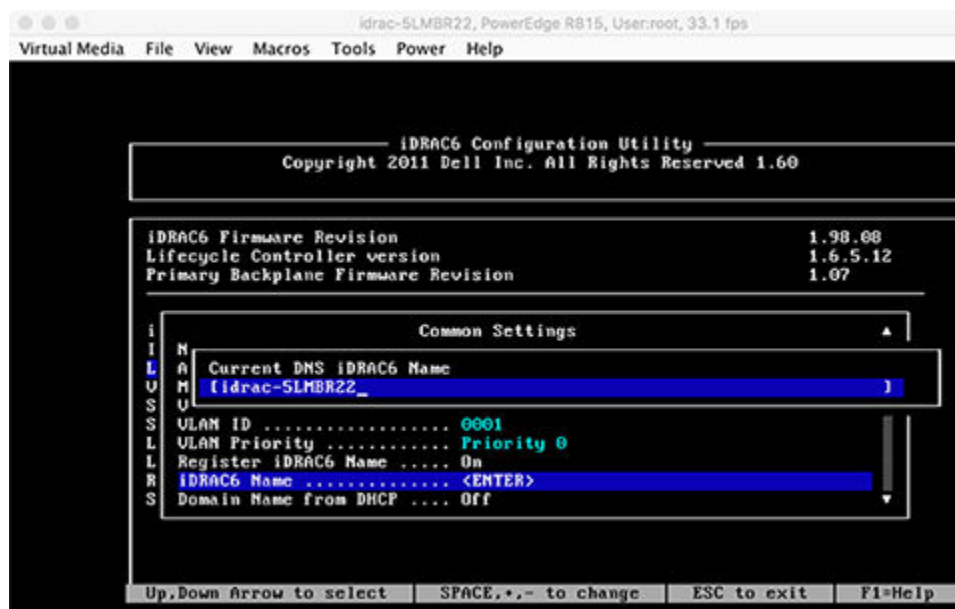
## a. Configure iDRAC6 name.

Use the arrow key to scroll down and select **iDRAC6 Name**, then press **Enter**. Enter a value for **Current DNS iDRAC6 Name** (e.g., smw-drac), then press **Esc**.

**Trouble?** If unable to set the iDRAC6 name, try this:

1. Temporarily set **Register iDRAC6 Name** to **On**.
2. Press **Enter** to set **iDRAC6 Name**. Select current or suggested name (edit enabled). When done, press **Esc**.
3. Return to **Register iDRAC6 Name** and set it to **Off**.

*Figure 9. Dell R815 SMW iDRAC6 LAN Parameters: iDRAC6 Name*



## b. Configure domain name.

Use the arrow key to scroll down and select **Domain Name**, then press **Enter**. Enter a value for **Current Domain Name** (e.g., us.cray.com), then press **Enter**.

## c. Configure host name string.

Use the arrow key to scroll down and select **Host Name String**, then press **Enter**. Enter a value for **Current Host Name String** (e.g., smw-drac), then press **Esc**.

## d. Configure IPv4 settings.

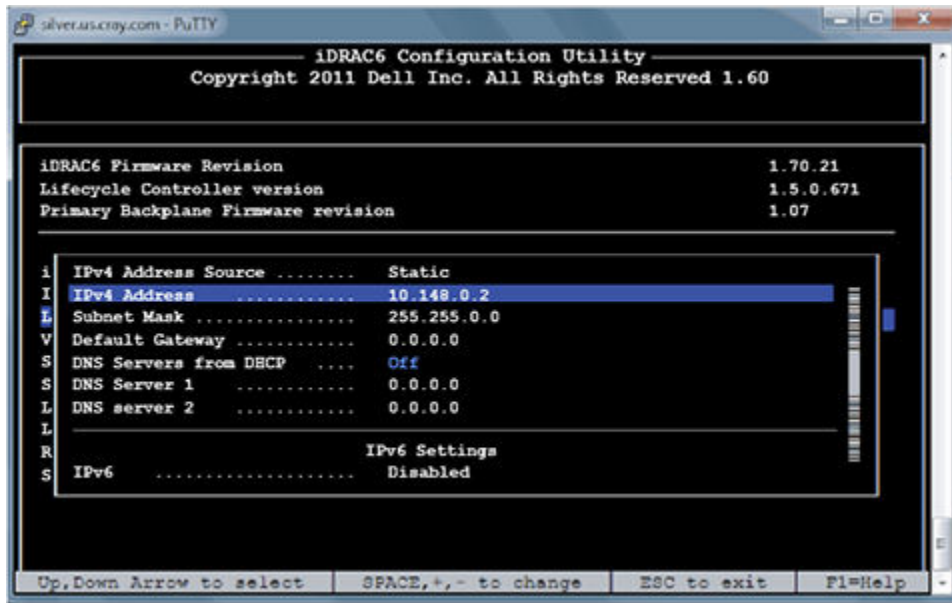
Use the arrow key to scroll down into the **IPv4 Settings** group and confirm that the **IPv4 Address Source** is set to **static**. Then enter values for the following:

- IPv4 Address** (the SMW DRAC IP address)
- Subnet Mask** (the SMW iDRAC subnet mask)
- Default Gateway** (the SMW iDRAC default gateway)

**DNS Server 1** (the first site DNS server)

**DNS Server 2** (the second site DNS server)

Figure 10. Dell R815 SMW iDRAC6 IPv4 Parameter Settings



e. Configure IPv6 settings.

Use the arrow key to scroll down into the **IPv6 Settings** group and ensure that **IPv6** is disabled.

f. Press **Esc** to exit **LAN Parameters** and return to the **iDRAC6 Configuration Utility** menu.

#### 14. Configure iDRAC virtual media.

a. Select **Virtual Media Configuration**, then press **Enter**.

b. Select the **Virtual Media** line and press the space key until it indicates **Detached**.

c. Press **Esc** to exit the **Virtual Media Configuration** menu.

#### 15. Set the password for the iDRAC LAN root account.

Using the arrow keys, select **LAN User Configuration**, then press **Enter**. The following configuration is for both SSH and web browser access to the iDRAC.

a. Select **Account User Name** and enter the account name `root`.

b. Select **Enter Password** and enter the intended password.

c. Select **Confirm Password** and enter the intended password again.

d. Press **Esc** to return to the **iDRAC6 Configuration Utility** menu.

#### 16. Exit the iDRAC configuration utility.

a. Press **Esc** to exit the **iDRAC6 Configuration Utility** menu.

b. Select **Save Changes and Exit**.

The **BIOS Boot Manager** menu appears.

**17.** Choose to boot from SATA Optical Drive.

Using the arrow keys, select the **SATA Optical Drive** entry, then press **Enter**.

Now that the Dell R815 SMW system setup (changing default BIOS and iDRAC settings) is complete, do the following:

1. Physically eject from SMW internal disk drive bays all SMW internal disks that are not to receive the base operating system.
2. Proceed to [Install the SLES 12 SP3 Base Linux Distribution on the SMW](#) on page 78.

#### 4.2.1.2 Dell R630 SMW: Configure the RAID Virtual Disks

##### Prerequisites

This procedure assumes the following:

- The SMW is disconnected from the boot RAID.
- The SMW is connected to a keyboard, monitor, and mouse.

##### About this task

Before installing and configuring SMW software, the base operating system needs to be installed on the SMW. And before the base operating system can be installed, the internal disk drives of the SMW must be configured as RAID virtual disks, as described in this procedure, and the default system setup for the R630 SMW node must be configured, as described in [Dell R630 or R640 SMW: Change the BIOS and iDRAC Settings](#) on page 68.

A Dell R630 SMW has five physical disks. The SMW node must be reconfigured so that the internal Dell Power Edge Expandable RAID Controller (PERC) treats four of these disks as RAID 5 with a hot spare and the fifth disk as non-RAID. This procedure describes how to do that. Because Cray ships systems with most of the installation and configuration completed, some of the steps may be needed only if changes are made to the configuration.

This procedure includes detailed steps for the Dell R630 server using the PERC H330 Mini BIOS Configuration Utility 4.03-0010. Depending on the server model and version of RAID configuration utility, there could be minor differences in the steps to configure this system. For more information, refer to the documentation for the Dell PERC controller or server RAID controller software.

##### Procedure

1. Connect a keyboard, monitor, and mouse to the front panel USB and monitor connectors on the SMW, if not already connected.
2. Ensure that all SMW internal disk drives are inserted into the SMW drive slots.
3. Power up the SMW. As the SMW node reboots, watch for the Power Edge Expandable RAID Controller section and be ready to press **Ctrl-R** when prompted.

Cray recommends using the RAID configuration utility (via **Ctrl-R**) to configure the RAID virtual disks instead of the **System Setup Device Settings** menu.

**TIP:** In the RAID configuration utility:



- Use the up-arrow or down-arrow key to highlight an item in a list.
- Press the **Enter** key to select an item.
- Press the **F2** key to display a dialog box with options for an item.
- Use the right-arrow, left-arrow, or **Tab** key to switch between the **Yes** and **No** buttons in a confirmation dialog box.

4. Delete existing/default disk group, if present.

If any disk groups are currently defined:

- a. Select **Disk Group 0**, then press **F2**.
- b. Select **Delete Disk Group**, then press **Enter**.
- c. Select **Yes** in the confirmation dialog box to confirm the changes.

5. Switch disk controller from HBA-Mode to RAID-Mode.

Some SMW hardware might be configured for HBA-Mode. If it is, then change it to RAID-Mode using the following substeps. If it is not, then skip these substeps.

- a. Switch disk controller from HBA-Mode to RAID-Mode.
  1. Press **Ctrl-N** (multiple times) to move to the **Ctrl Mgmt** tab.
  2. Press **Tab** (multiple times) to get to **Personality Mode**.
  3. Press **Enter** to see choice between **RAID-Mode** and **HBA-Mode**.
  4. Use the up-arrow or down-arrow key to select **RAID-Mode**, then press **Enter**.
  5. Press **Tab** (multiple times) to get to **Apply**, then press **Enter**. This message appears: "The operation has been performed successfully. Reboot the system for the change to take effect."
  6. Press **Enter**.
- b. Exit RAID configuration utility.
  1. Press **Esc** to exit the RAID configuration utility.
  2. Select **OK** to confirm, then press **Enter**.
- c. Reboot the SMW.

Press **Ctrl-Alt-Delete** at the prompt to reboot. The server will restart the boot process. Be prepared to press **Ctrl-R** when prompted.

- d. Enter RAID configuration utility.

As the SMW node reboots, enter the RAID controller configuration utility by pressing **Ctrl-R** when prompted. This will return to the point prior to switching from HBA-Mode to RAID-Mode.

6. Delete previous RAID configuration, if present.

If the drives to be configured have been used in a RAID configuration previously and have not been completely cleaned, the left pane of the **VD Mgmt** tab will indicate that with the text "Foreign Config Present." To delete the existing configuration:

- a. Select the PERC controller in the left pane and then press the **F2** key.
- b. Select **Foreign config**.
- c. Select **Foreign config** → **Clear**.

- d. Select **Yes** in the confirmation dialog box to confirm deletion of the existing RAID configuration.

The the **VD Mgmt** tab should now show four unconfigured physical disks in a Ready state.

— CONFIGURE MOST INTERNAL DISKS AS `/dev/sda` IN A RAID-5 VIRTUAL DISK —

The following steps configure most internal disks to appear as `/dev/sda` in a RAID-5 virtual disk. The R630 typically ships with five 1-TB drives. One of the 1-TB drives will be excluded from this RAID-5 configuration. If this SMW shipped with four 500-GB drives and one 1-TB drive instead, exclude the 1-TB drive from RAID-5 configuration. The excluded drive will be used to hold the PostgreSQL Power Management Database (PMDB).

7. Convert non-RAID disks to RAID-capable.

- a. Select **No Configuration Present!**, then press the **F2** key.

A pop-up screen appears.

- b. Select **Convert to RAID capable**, then press **Enter**.

The **Convert Non-RAID Disks to RAID capable** pop-up screen appears. If the disks have already been converted, only the PMDB disk (the 1-TB disk) will be listed in this pop-up screen. If the configuration already looks like the figure in the next step, skip the rest of this step.

- c. Press **Enter** to check the box for a physical disk, which selects it for this RAID-5 disk group (this action also advances the selection to the next disk).

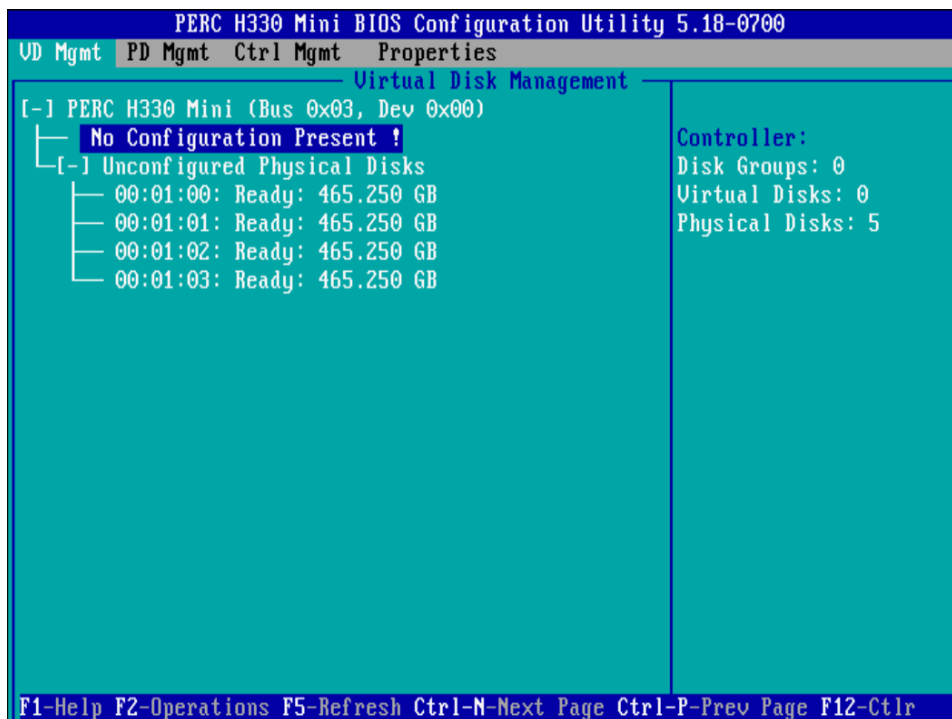
In this manner, select four of the 1-TB drives (or all four 500-GB drives, if applicable) but exclude one 1-TB drive (leave it unselected).

- d. Press **Tab** to move to **OK**, then press **Enter**.

8. Verify the virtual disk changes.

To verify the virtual disk changes, compare settings with those shown in the figure (note that this example shows four 500-GB drives).

Figure 11. Dell R630 RAID Disk Validation



9. Create new virtual disk (VD) sda.

- a. Use up-arrow to return to the **No Configuration Present!** item.
- b. Press **F2** to see a pop-up menu.
- c. Press **Enter** to choose **Create New VD**.

The **Convert Non - RAID Disks to RAID capable** screen appears. The only disk left on this screen should be the 1-TB disk that was excluded earlier. It should not be added to the RAID capable set of disks, so continue to exclude it.

- d. Press **Tab** to move from the list of disks to **Cancel**, then press **Enter**.

This cancels the conversion of non-RAID disks to RAID capable. The **Create New VD** screen appears.

- e. Press **Enter** to switch from **RAID-0** to other options.
- f. Use down-arrow to select **RAID-5**, then press **Enter**.
- g. Press **Tab** to move to the **Physical Disks** area.

- h. Press **Enter** to select each disk except one.

One disk should not be selected so that it can become the hot spare (configured later in this step).

- i. Press **Tab** to move to **VD Name**.

- j. Select name sda.

- k. Press **Tab** to move to **Advanced**, then press **Enter**.

The **Create Virtual Disk-Advanced** screen appears.

The remaining substeps configure one disk as the hot spare.

- l. Press **Tab** multiple times to move to **Initialize**, then press **Enter** to select it.  
A pop-up window with the following text appears: "Initialization will destroy data on the virtual disk. Are you sure you want to continue?"
- m. Press **Tab** or arrow keys to move to **OK**, then press **Enter** to confirm initialization.
- n. Press **Tab** to move to **Configure Hot Spare**, then press **Enter** to select it.
- o. Press **Tab** or arrow keys to move to **OK** on the **Create Virtual Disk-Advanced** screen, then press **Enter**.
- p. Press **Tab** or arrow keys to move to **OK** on the **Create New VD** screen, then press **Enter**.  
A pop-up window with the following text appears: "Virtual disk is successfully created and initialized."
- q. Press **Tab** or arrow keys to move to **OK**, then press **Enter**.  
A pop-up window with the following text appears: "Dedicated Hot Spare for Disk Group 0."
- r. Select the disk to be the hot spare, then press **Enter**.
- s. Press **Tab** or arrow keys to move to **OK**, then press **Enter**.  
A pop-up window with the following text appears: "Initialization complete on VD 0."
- t. Press **Tab** or arrow keys to move to **OK**, then press **Enter**.

**ATTENTION:** If the hot spare drive is not added and configured during the initial definition of the VD, delete the VD and repeat step 9 on page 63. The RAID configuration menu does not allow the addition of a hot spare drive later.

#### 10. Exit RAID configuration utility.

Exit the RAID configuration utility, reboot, and then begin installing the base operating system.

- a. Press the **Esc** key to exit the RAID configuration utility.
- b. Select **OK**, then press **Enter** to confirm.

#### 11. Reboot the system.

A message appears that prompts to reboot.

**ATTENTION:** Only the disk drives configured to be the RAID-5 virtual disk sda should be inserted into the SMW internal drive bays when installing the base OS.

- a. Eject from the SMW the 1-TB disk that was not added to the RAID-5 virtual disk sda.  
This will be re-inserted when the base OS installation is complete.
- b. Press **Ctrl-Alt-Delete**.

The server will restart the boot process and will not interrupt RAID initialization. During the system reboot, be prepared to press **F2** when prompted, to change the system setup.

RAID configuration on the Dell R630 SMW is now complete.

To continue preparation for installing the base operating system, proceed to [Dell R630 or R640 SMW: Change the BIOS and iDRAC Settings](#) on page 68.

### 4.2.1.3 Dell R640 SMW: Configure the RAID Virtual Disks

#### Prerequisites

This procedure assumes the following:

- The SMW is disconnected from the boot RAID.
- The iDRAC is configured and accessible via a web browser.
- The SMW is powered on.

#### About this task

Before installing and configuring SMW software, the base operating system needs to be installed on the SMW. And before the base operating system can be installed, the internal disk drives of the SMW must be configured as RAID virtual disks, as described in this procedure, and the default system setup for the R640 SMW node must be configured, as described in [Dell R630 or R640 SMW: Change the BIOS and iDRAC Settings](#) on page 68.

A Dell R640 SMW has five physical disks. The SMW node must be reconfigured so that the internal Dell Power Edge Expandable RAID Controller (PERC) treats four of these disks as RAID 5 with a hot spare, and the fifth disk as RAID 0. Because Cray ships systems with most of the installation and configuration completed, some of the steps may be necessary only if changes are made to the configuration.

This procedure includes detailed steps to configure the Dell R640 RAID virtual disks using the iDRAC web interface. Depending on the server model and version of the iDRAC, minor differences in the steps to configure this system may occur. For more information, refer to the Dell iDRAC documentation.

#### Procedure

1. Use a web browser to navigate to the iDRAC web interface at the IP address configured in [Dell R630 or R640 SMW: Change the BIOS and iDRAC Settings](#) on page 68. Log in using the credentials configured in the above procedure.
2. Under the **Configuration** tab near the top, navigate to the **Storage Configuration** panel.
3. Scroll to the bottom of the page and expand the **Virtual Disk Configuration** section.
4. Click **Create Virtual Disk**.
5. In the window that appears, enter the following details:
  - a. Enter **Name** as `sda`
  - b. Select **Raid 5** from the **Layout** dropdown. Once selected, scroll to the bottom of that window.
  - c. Mark the checkboxes for the first three physical disks:

Figure 12. Select Physical Disks in iDRAC

## Create Virtual Disk






Number of Physical Disks - Minimum : [ 3 ] Maximum : [ 32 ] Current Selection : [ 3 ]

Virtual Disk Size - Minimum : [ 100MB ] Maximum : [ 1.82 TB ] Specified Size : [ 1.82 TB ]

The *Span Count* can only be adjusted for *RAID 50 & 60* after selecting the physical disks.

A Red Diamond icon highlights the physical disks that are already supporting one or more virtual disks.

## Select Physical Disks

Status	Name	Capacity	Media Type
<input checked="" type="checkbox"/> 	Physical Disk 0:1:0	931.00 GB	HDD
<input checked="" type="checkbox"/> 	Physical Disk 0:1:1	931.00 GB	HDD
<input checked="" type="checkbox"/> 	Physical Disk 0:1:2	931.00 GB	HDD
<input type="checkbox"/> 	Physical Disk 0:1:3	931.00 GB	HDD
<input type="checkbox"/> 	Physical Disk 0:1:4	931.00 GB	HDD

Cancel

Add to Pending Operations

- d. Click the **Add to Pending Operations** button. Close the information window that appears.
  - e. Under the **Storage Configuration** panel, find the virtual drive (sda) that was just created. Under **Actions**, select **Assign Dedicated Hotspare Physical Disk 0:1:3** and click **Ok** in the confirmation window that appears.
  - f. Under **Actions**, select **Initialize: Fast**.
6. Click **Create Virtual Disk** for a second time.
  7. In the window that appears, enter the following details:
    - a. **Name** as `pmdb`
    - b. Expand the **Layout** dropdown and select `Raid 0`. Once selected, scroll to the bottom of that window.
    - c. Mark the checkbox for the only remaining physical disk (Physical Disk 0:1:4).
    - d. Click the **Add to Pending Operations** button. Close the confirmation window that appears.
    - e. In the **Storage Configuration** panel, find the virtual drive (pmdb) that was just created. Under **Actions**, select **initialize: Fast**. Select **Ok** in the confirmation window that appears.
  8. In the **Storage Configuration** panel, five pending operations should be shown at the top:

Figure 13. Storage Configuration Pending Operations

Pending Operations on the component group: PERC H740P Mini (Embedded)

Order ▾	Component Type	Component Name	Operation
0	Physical Disk	Physical Disk 0:1:3	Assign Dedicated Hotspare
1	Virtual Disk	sda	Create Virtual Disk
2	Virtual Disk	sda	Initialize: Fast
3	Virtual Disk	pmdb	Create Virtual Disk
4	Virtual Disk	pmdb	Initialize: Fast

9. Scroll to the bottom of the **Storage Configuration** panel and click **Apply Now**. Select **Ok** in the confirmation box that appears.
10. Monitor the progress of the above operations using the **Job Queue** button. Do not proceed until the job's status reads *Completed (100%)*.
11. (Optional) If the job does not enter the *Running* state after some time, it may be necessary to reboot the system. Navigate to the **Dashboard** using the links at the top of the page and select **Reset**:

Figure 14. Reset the System

The screenshot shows the 'Integrated Remote Access Controller 9 | Enterprise' dashboard. The top navigation bar includes links for Dashboard, System, Storage, Configuration, Maintenance, and iDRAC Settings. The main section is titled 'Dashboard' and features a 'Graceful Shutdown' dropdown menu with options: Power Off System, NMI(Non-Masking Interrupt), Reset System(warm boot), and Power Cycle System (cold boot). The 'Reset System(warm boot)' option is highlighted. Below the menu, there are two health status cards: 'System Health' and 'Storage Health', both showing a green checkmark and the word 'Healthy'. To the right, the 'System Information' panel displays various system details:

System Information	
Power State	ON
Model	
Host Name	
Operating System	
Operating System Version	
Service Tag	F5551S2
BIOS Version	1.5.4
iDRAC Firmware Version	3.21.21.21
iDRAC MAC Address	4c:d9:8f:0f:33:18

12. Once the job's status reads *Completed (100%)*, RAID configuration on the Dell R640 SMW is now complete.



#### 4.2.1.4 Dell R630 or R640 SMW: Change the BIOS and iDRAC Settings

### Prerequisites

This procedure assumes the following:

- The [Dell R630 SMW: Configure the RAID Virtual Disks](#) on page 60 procedure has been completed.
- The SMW is rebooting. If the SMW is not rebooting, press **Ctrl-Alt-Delete** to reboot when ready to begin this procedure.

### About this task

This procedure describes how to change the system setup for the SMW: the network connections, remote power control, and the remote console. This procedure includes detailed steps for the Dell R630 and R640 servers. Depending on the server model and version of BIOS configuration utility, there could be minor differences in the steps to configure the system. For more information, refer to the documentation for the Dell server used at this site. Because Cray ships systems with most of the installation and configuration completed, some of the steps may have been done already.

For a Dell R815 server, see [Dell R815 SMW: Change the BIOS and iDRAC Settings](#) on page 53.

### Procedure

Watch as the system reboots and the BIOS power-on self-test (POST) process begins. Be prepared to press **F2**, when prompted, to change the system setup.

1. Press the **F2** key immediately after the following messages appear in the upper-left of the screen:

```
F2 = System Setup
F10 = Lifecycle Controller (Config iDRAC, Update FW, Install OS)
F11 = Boot Manager
F12 = PXE Boot
```

When the **F2** keypress is recognized, the **F2 = System Setup** line changes color from white-on-black to white-on-blue.

After the POST process completes and all disk and network controllers have been initialized, the Dell **System Setup** screen appears. The following submenus are available on the **System Setup Main Menu** and will be used in subsequent steps: **System BIOS**, **iDRAC Settings**, and **Device Settings**.

*Figure 15. Dell R630/R640 System Setup Main Menu*

**TIP:** In system setup screens,

- Use the **Tab** key to move to different areas on the screen.
- Use the up-arrow and down-arrow keys to highlight or select an item in a list, then press the **Enter** key to enter or apply the item.
- Press the **Esc** key to exit a submenu and return to the previous screen.

2. Change the BIOS settings.

- a. Select **System BIOS** on the **System Setup Main Menu**, then press **Enter**.

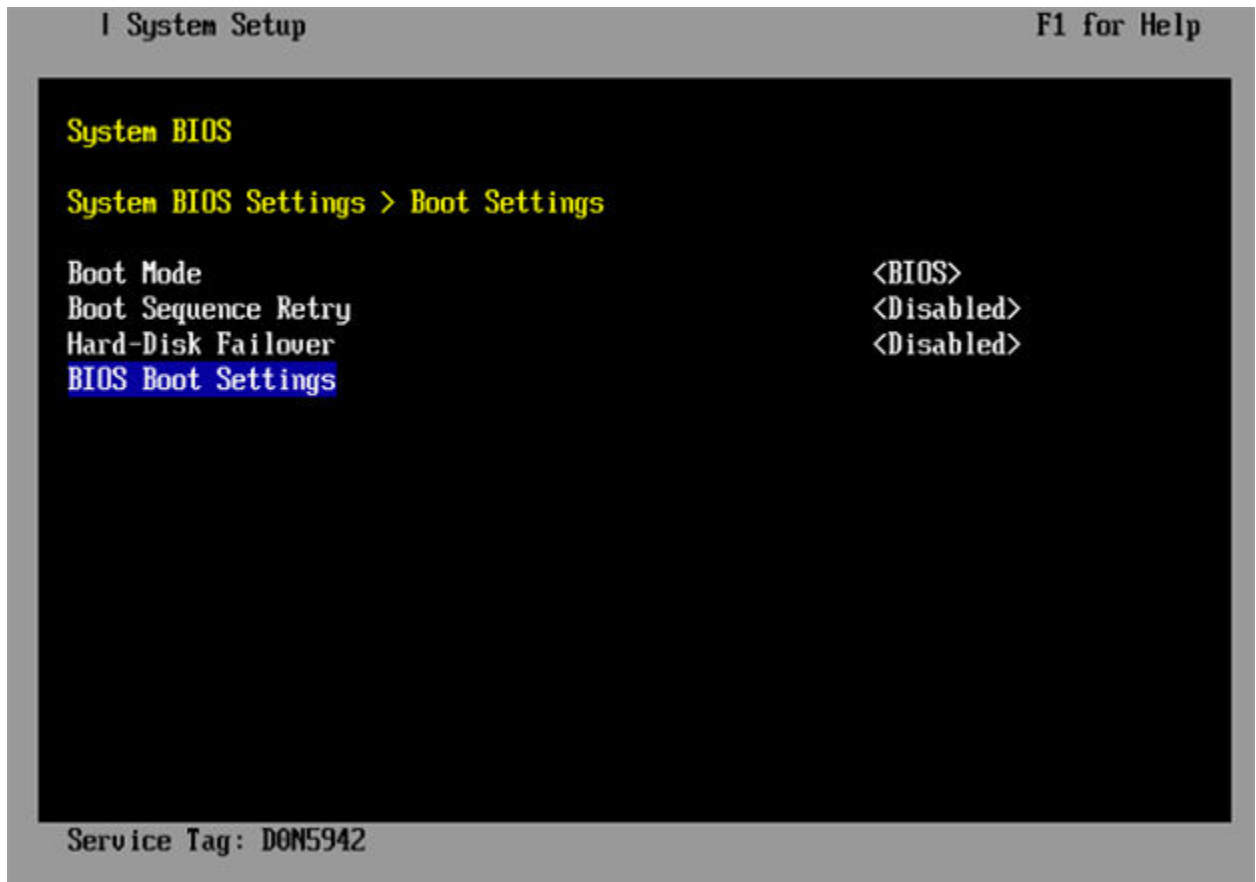
The **System BIOS Settings** screen appears.

*Figure 16. Dell R630/R640 System BIOS Settings Screen*

b. Change Boot Settings.

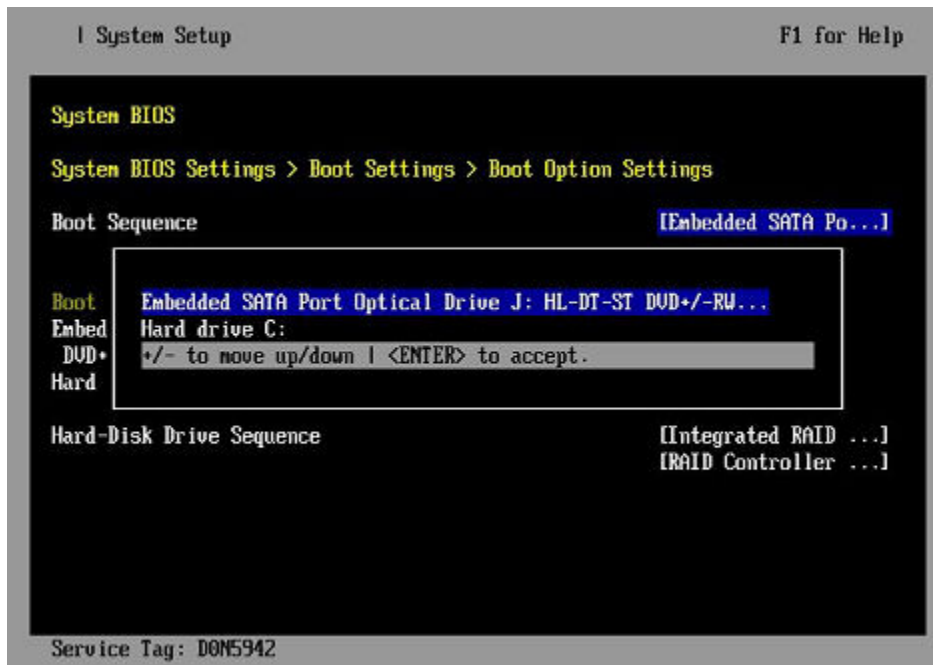
1. Select **Boot Settings** on the **System BIOS Settings** screen, then press **Enter**. The **Boot Settings** screen appears.

Figure 17. Dell R630/R640 Boot Settings Screen



2. Ensure that **Boot Mode** is **BIOS** and not **UEFI**.
3. Select **BIOS Boot Settings**, then press **Enter**.
4. Select **Boot Sequence** on the **Boot Option Settings** screen, then press **Enter** to view a pop-up window with the boot sequence.

Figure 18. Dell R630/R640 BIOS Boot Sequence

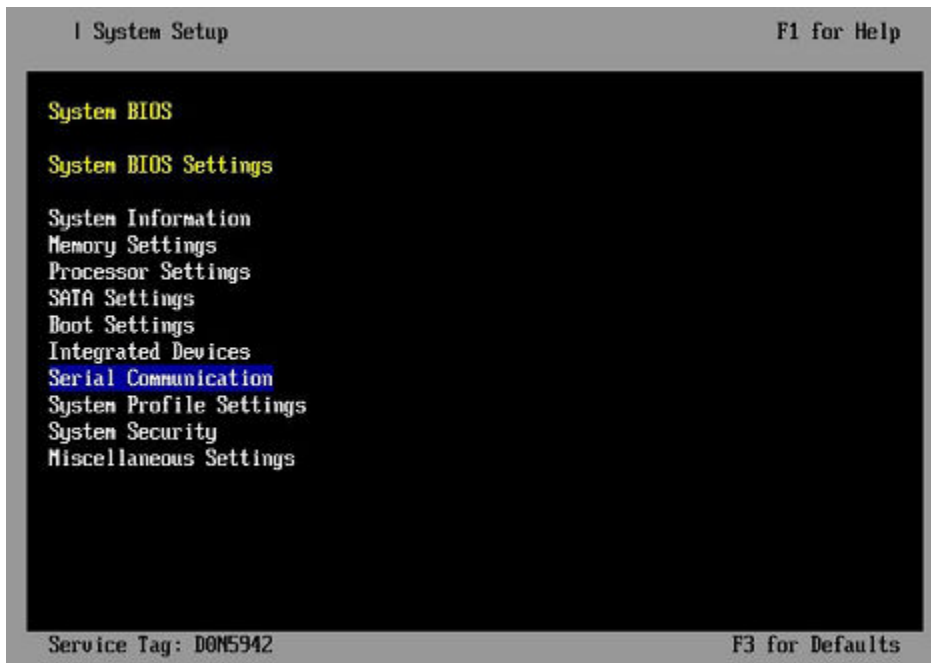


5. Change the boot order in the pop-up window so that the optical drive appears first, then the hard drive. If **Integrated NIC** appears in the list, it should end up below the optical drive and hard drive in the list.

**TIP:** Use the up-arrow or down-arrow key to highlight or select an item, then use the + and - keys to move the item up or down.

6. Select **OK**, then press **Enter** to accept the change.
  7. Select the box next to **Hard drive C:** under the **Boot Option Enable/Disable** section to enable it. Do the same for the optical drive, if necessary.
  8. Select **integrated NIC**, then press **Enter** to disable it.
  9. Press **Esc** to exit **Boot Option Settings**.
  10. Press **Esc** to exit **Boot Settings** and return to the **System BIOS Settings** screen.
- c. Change Serial Communication Settings.

Figure 19. Dell R630/R640 System BIOS Settings: Serial Communication



1. Select **Serial Communication** on the **System BIOS Settings** screen. The **Serial Communication** screen appears.

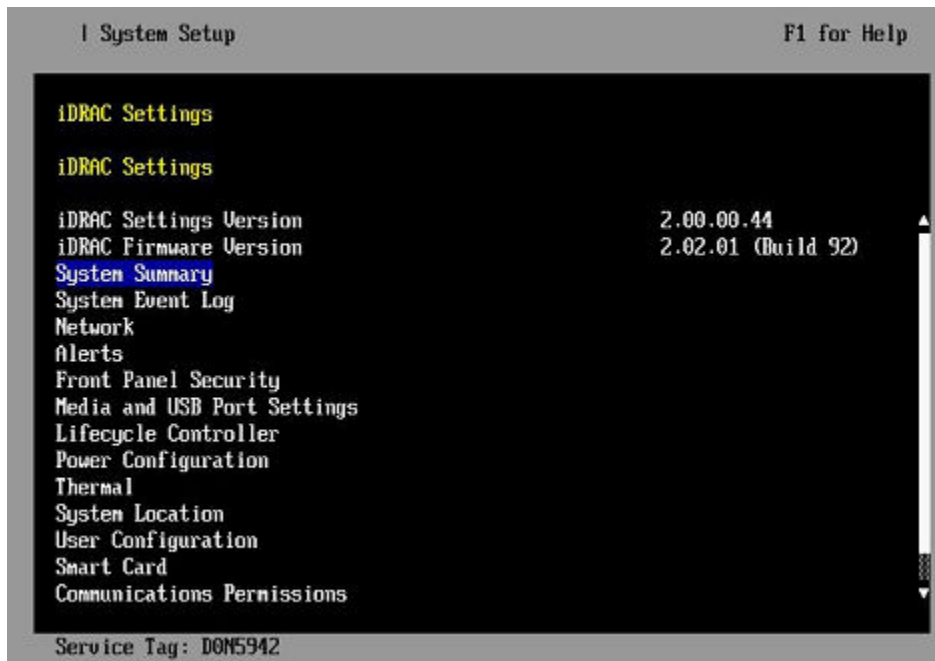
Figure 20. Dell R630/R640 Serial Communication Screen



2. Select **Serial Communication** on the **Serial Communication** screen, then press **Enter**. A pop-up window displays the available options.
3. Select **On with Console Redirection via COM2** in the pop-up window, then press **Enter** to accept the change.

4. Select **Serial Port Address**, then select **Serial Device1=COM1**, **Serial Device2=COM2**, then press **Enter**.
  5. Select **External Serial Connector**, then press **Enter**. A pop-up window displays the available options.
  6. Select **Remote Access Device** in the pop-up window, then press **Enter** to return to the previous screen.
  7. Select **Failsafe Baud Rate**, then press **Enter**. A pop-up window displays the available options.
  8. Select **115200** in the pop-up window, then press **Enter** to return to the previous screen.
  9. Press the **Esc** key to exit the **Serial Communication** screen.
  10. Press **Esc** to exit the **System BIOS Settings** screen. A "Settings have changed" message appears.
  11. Select **Yes** to save changes. A "Settings saved successfully" message appears.
  12. Select **Ok**.
3. Change the iDRAC (Integrated Dell Remote Access Controller) settings.  
Select **iDRAC Settings** on the **System Setup Main Menu**, then press **Enter**.  
The **iDRAC Settings** screen appears.

*Figure 21. Dell R630/R640 iDRAC6 Settings Screen*



4. Change the iDRAC network.
  - a. Select **Network** to display a long list of network settings.
  - b. Change the DNS DRAC name.  
Use the arrow key to scroll down to **DNS DRAC Name**, then enter an iDRAC host name that is similar to the SMW node host name (e.g., cray-drac).
  - c. Change the static DNS domain name.

Use the arrow key to scroll down to **Static DNS Domain Name**, then enter the DNS domain name and press **Enter**.

d. Change the IPv4 settings.

Use the arrow key to scroll down to the **IPV4 SETTINGS** list.

1. Ensure that IPv4 is enabled.
  - a. If necessary, select **Enable IPV4**, then press **Enter**.
  - b. Select **<Enabled>** in the pop-up window, then press **Enter** to return to the previous screen.
2. Ensure that DHCP is disabled.
  - a. If necessary, select **Enable DHCP**, then press **Enter**.
  - b. Select **<Disabled>** in the pop-up window, then press **Enter** to return to the previous screen.
3. Change the IP address.
  - a. Select **Static IP Address**.
  - b. Enter the IP address of the iDRAC interface (`ipmi0`) for the SMW, then press **Enter**.
4. Change the gateway.
  - a. Select **Static Gateway**.
  - b. Enter the appropriate value for the gateway of the network to which the iDRAC is connected, then press **Enter**.
5. Change the subnet mask.
  - a. Select **Subnet Mask**.
  - b. Enter the subnet mask for the network to which the iDRAC is connected (such as `255.255.255.0`), then press **Enter**.
6. Change the DNS server settings.
  - a. Select **Static Preferred DNS Server**, enter the IP address of the primary DNS server, then press **Enter**.
  - b. Select **Alternate DNS Server**, enter the IP address of the alternate DNS server, then press **Enter**.

e. Change the IPMI settings.

Change the IPMI settings to enable the Serial Over LAN (SOL) console.

1. Use the arrow key to scroll down to the **IPMI SETTINGS** list.
2. Ensure that **Enable IPMI over LAN** is selected.

**TIP:** Use the left-arrow or right-arrow to switch between two settings.

3. Ensure that **Channel Privilege Level Limit** is set to **Administrator**.

f. Exit Network screen.

Press the **Esc** key to exit the **Network** screen and return to the **iDRAC Settings** screen.

5. Change host name in iDRAC LCD display.

Change front panel security to show the host name in LCD display.

- a. Use the arrow key to scroll down and highlight **Front Panel Security** on the **iDRAC Settings** screen, then press **Enter**.



- b. Select **Set LCD message**, then press **Enter**.
- c. Select **User-Defined String**, then press **Enter**.
- d. Select **User-Defined String**, then enter the SMW host name and press **Enter**.
- e. Press the **Esc** key to exit the **Front Panel Security** screen.

6. (Optional) Change the iDRAC **System Location** fields.

Change the **System Location** configuration on the **iDRAC Settings** screen to set any of these fields: **Data Center Name**, **Aisle Name**, **Rack Name**, and **Rack Slot**.

7. Configure iDRAC virtual media.

- a. Select **Media and USB Port Settings**, then press **Enter**.
- b. Configure settings as needed for this system.
- c. Press **Esc** to exit the **Media and USB Port Settings** menu.

8. Set the password for the iDRAC root account.

- a. Use the arrow key to highlight **User Configuration** on the **iDRAC Settings** screen, then press **Enter**.
- b. Confirm that User Name is root. Select **User Name**, then enter the "root" user name.
- c. Select **Change Password**, then enter a new password.
- d. Reenter the new password in the next pop-up window to confirm it (the default password is "calvin").
- e. Press the **Esc** key to exit the **User Configuration** screen.

9. Exit iDRAC settings.

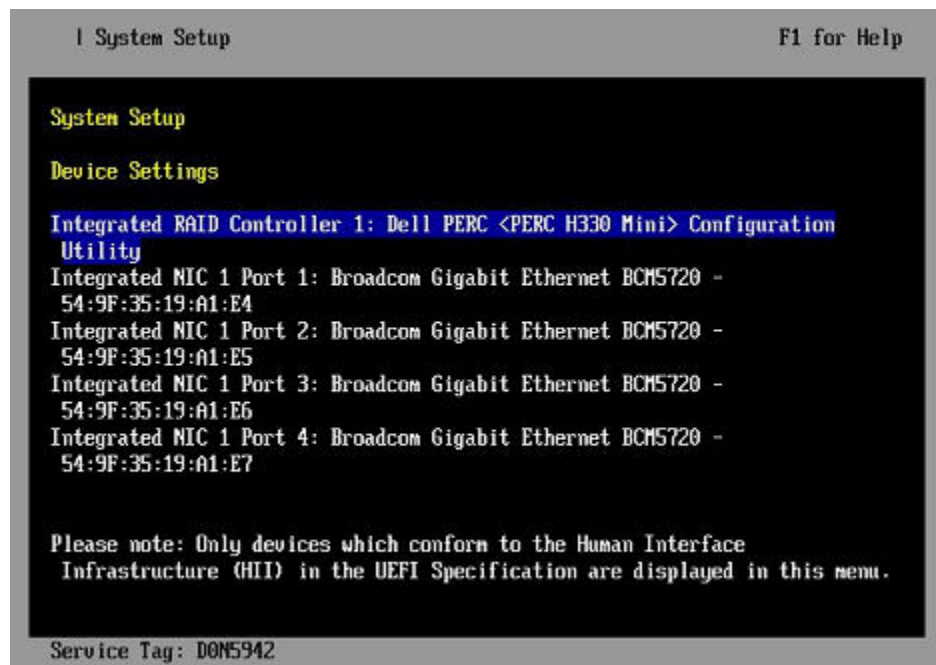
- a. Press the **Esc** key to exit the **iDRAC Settings** screen.  
A "Settings have changed" message appears.
- b. Select **Yes**, then press **Enter** to save the changes.  
A "Success" message appears.
- c. Select **Ok**, then press **Enter**.  
The main screen (**System Setup Main Menu**) appears.

10. Change device settings.

These steps disable an integrated NIC device by changing the setting for the integrated NIC on a port from **PXE** to **None**.

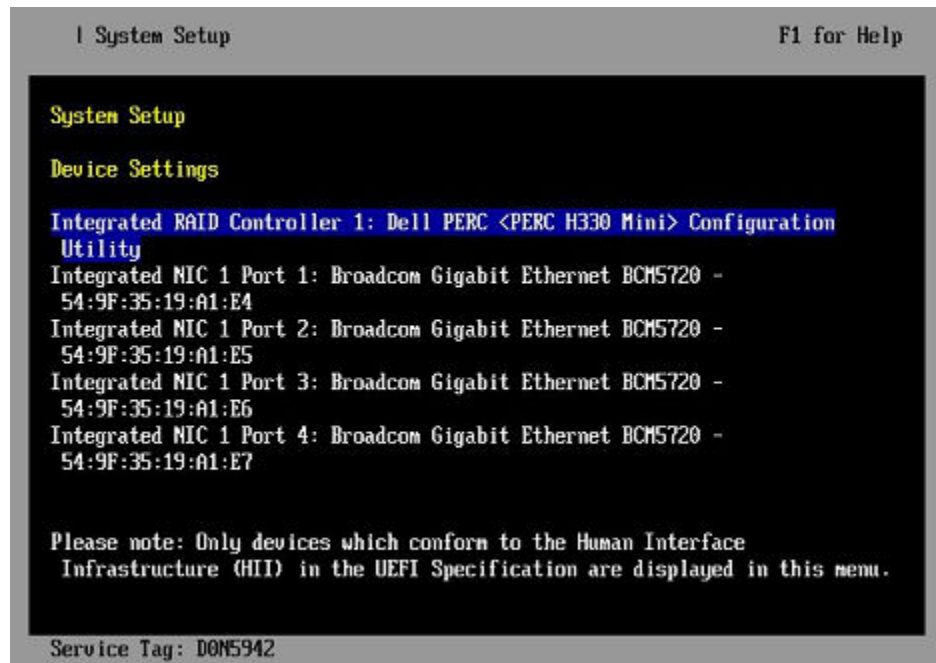
- a. Change Integrated NIC 1 Port 1
  - 1. Select **Device Settings** on the **System Setup Main Menu**, then press **Enter**. The **Device Settings** screen appears.

Figure 22. Dell R630/R640 Device Settings Screen



2. Select **Integrated NIC 1 Port 1: ...** on the **Device Settings** screen, then press **Enter**.
  3. Select **NIC Configuration** on the **Main Configuration Page** screen, then press **Enter**.
  4. Select **Legacy Boot Protocol** on the **NIC Configuration** screen, use the right-arrow or left-arrow key to highlight **None**, then press **Enter**.
  5. Press the **Esc** key to exit the **NIC Configuration** screen.
  6. Press **Esc** to exit the **Main Configuration Page** screen. A "Warning Saving Changes" message appears.
  7. Select **Yes**, then press **Enter** to save the changes. A "Success" message appears.
  8. Select **OK**, then press **Enter**. The **Device Settings** screen appears.
  9. Press **Esc** to exit the **Device Settings** screen. A "Settings have changed" message appears.
  10. Select **Yes**, then press **Enter** to save the changes. A "Settings saved successfully" message appears.
  11. Select **OK**, then press **Enter**. The main screen (**System Setup Main Menu**) appears.
- b. Change Integrated NIC 1 Port 2
1. Select **Device Settings** on the **System Setup Main Menu**, then press **Enter**. The **Device Settings** screen appears.

Figure 23. Dell R630/R640 Device Settings Screen



2. Select **Integrated NIC 1 Port 2: ...** on the **Device Settings** screen, then press **Enter**.
3. Select **NIC Configuration** on the **Main Configuration Page** screen, then press **Enter**.
4. Select **Legacy Boot Protocol** on the **NIC Configuration** screen, use the right-arrow or left-arrow key to highlight **None**, then press **Enter**.
5. Press the **Esc** key to exit the **NIC Configuration** screen.
6. Press **Esc** to exit the **Main Configuration Page** screen. A "Warning Saving Changes" message appears.
7. Select **Yes**, then press **Enter** to save the changes. A "Success" message appears.
8. Select **OK**, then press **Enter**. The **Device Settings** screen appears.
9. Press **Esc** to exit the **Device Settings** screen. A "Settings have changed" message appears.
10. Select **Yes**, then press **Enter** to save the changes. A "Settings saved successfully" message appears.
11. Select **OK**, then press **Enter**. The main screen (**System Setup Main Menu**) appears.

Now that the Dell R630 SMW system setup (changing default BIOS and iDRAC settings) is complete, do the following:

1. Physically eject from SMW internal disk drive bays all SMW internal disks that are not to receive the base operating system.
2. Proceed to [Install the SLES 12 SP3 Base Linux Distribution on the SMW](#) on page 78.

## 4.2.2 Install the SLES 12 SP3 Base Linux Distribution on the SMW

### Prerequisites

This procedure assumes the following:

- The BIOS and iDRAC settings have just been changed on the SMW and it is restarting the boot process.
- All SMW internal disks that are not to receive the operating system are physically ejected from SMW internal disk drive bays.
- All connections to the boot RAID are unplugged so that no disk devices from the boot RAID will inadvertently lose existing data or receive the operating system.

### About this task

This procedure describes the base operating system installation process. It provides detailed instructions for installing SLES 12 SP3 on the SMW (Dell R815, R630, and R640 models); configuring the SMW; and performing final steps: reconnect cables, reinsert drives, and reboot the SMW. For Dell R815 and R630 SMWs, install the base operating system using the Cray-slebase-12-SP3-201709141039.iso DVD, which contains SUSE Linux Enterprise Server version 12 SP3 (SLES 12 SP3). For Dell R640 SMWs, install the base operating system using the Cray-slebase-12-SP3-201806150923.iso DVD, which contains SUSE Linux Enterprise Server version 12 SP3 (SLES 12 SP3).

### Procedure

#### ———— SLES 12 SP3 SOFTWARE PACKAGE INSTALLATION ————

1. Select one of the **Cray SMW Initial Install** options.

Within 10 to 15 seconds after this **SUSE Linux Enterprise Server** boot menu displays, use the arrow key to scroll down and select one of the install options, then press **Enter**.

```

Boot from Hard Disk
Cray SMW Initial Install without software RAID
Cray SMW Initial Install with software RAID1
Cray SMW Initial Install with software RAID1 And Small Disks
Rescue System
Check Installation Media
Firmware Test
Memory Test

```

Select the option that is best for the SMW model:

- |                              |   |
|------------------------------|---|
| <b>For Dell R815</b>         | Select <b>Cray SMW Initial Install with software RAID1 And Small Disks</b> if the disk size is 250 GB or less; otherwise select <b>Cray SMW Initial Install with software RAID1</b> . Either one is a mirrored boot disk option that creates a software RAID1 mirror on the first two drives. These two options are best for a Dell R815 because the R815 should use two disk drives to become the software RAID1 mirror. |
| <b>For Dell R630 or R640</b> | Select <b>Cray SMW Initial Install without software RAID</b> , a non-mirrored boot disk option, for servers with a single disk or virtual disk. This option is best for a Dell R630 because the R630 should have the internal RAID controller configured to present four disk drives as a virtual disk.   |

**ATTENTION:** If the selection is not made in time, the system will boot from the default selection, which is **Boot from Hard Disk**. If that happens, shut down the SMW, then start the power-up sequence again.

Note: The upper left corner of the installation screen has a date/time stamp for when the bootable SLES 12 SP3 DVD was created.

As the base installation progresses, the following phases appear on the screen:

```
Starting ... Loading Linux kernel
Initializing
Preparing System for Automated Installation
Initializing the Installation Environment
System Probing
Installation Settings
```

2. Review installation settings while the installation pauses on the **Installation Settings** screen.
3. Confirm the language for the SMW.

English (US) is the primary language by default. To change the primary language:

  - a. Select the **Language** heading in the **Installation Settings** screen.

The **Languages** window opens.
  - b. Select a language (or multiple languages) from the drop-down menu, then select **Accept** at the bottom of the window.
4. Begin automated install.
  - a. On the **Installation Settings** screen, select **Install**.

The **Confirm Installation** pop-up window appears.
  - b. Select **Install**.

The installation of software packages runs for about 20–55 minutes.
  - c. In the **SUSE Linux Enterprise Server** boot menu, select the Boot from Hard Disk option so that the SMW will reboot from the hard disk.

The installation process continues with system configuration.

---

## SYSTEM CONFIGURATION

---

5. Log in to SMW as root.

When the login screen is displayed with the crayadm account as the account which will be logged in:

  - a. Select **Not listed?**, then enter `root` for the username.
  - b. Either press **Enter** or select **Sign In**.
  - c. Enter the password for root.

Default passwords are listed in [Passwords](#) on page 51.

To perform some of the steps that follow, a terminal window is necessary. To get a terminal window after logging in as root, click **Applications** in the lower-left of the screen, then navigate to **Utilities > Xterm**.

6. Change default passwords on the SMW by executing the following commands.

The SMW contains its own `/etc/passwd` file that is separate from the password file for the rest of the CLE system.

```
smw# passwd root
```

```
smw# passwd crayadm
```

```
smw# passwd mysql
```

7. Configure the SMW firewall.

The SUSE firewall settings may need to be adjusted to match site firewall policy and to customize for site IP addresses. These steps enable and configure the firewall.

**TIP:** It is not necessary to shut down the system before performing this task.

a. Save the SUSE firewall configuration.

Before modifying the SUSE firewall settings, make a copy of the configuration file.

```
smw# cp -p /etc/sysconfig/SuSEfirewall12 /etc/sysconfig/SuSEfirewall12.orig
```

b. Check current firewall settings.

Check current firewall settings and change to support any site requirements. During the process of configuring Cray SMW and CLE software, some of the firewall settings may be adjusted. SSH access is one of the protocols permitted through the firewall from the external network to the SMW.

```
smw# iptables -L
smw# vi /etc/sysconfig/SuSEfirewall12
```

c. Start the firewall immediately.

Invoke the modified configuration.

```
smw# systemctl start SuSEfirewall12_init.service
smw# systemctl start SuSEfirewall12.service
```

d. Ensure that the firewall will start at next boot.

Execute the following commands to start the firewall at boot time.

```
smw# systemctl enable SuSEfirewall12_init.service
smw# systemctl enable SuSEfirewall12.service
```

e. Verify firewall changes.

Verify the changes to the `iptables`.

```
smw# iptables -nvL
```

8. Configure LAN on the SMW.

Set network configuration for `eth0` and the host name for the SMW.

a. Execute this command:

```
smw# yast2 lan
```

The **Network Settings** screen appears with the **Overview** tab highlighted.

- b. Select the **eth0** line on the **Overview** tab, then select **Edit**.

The **Network Card Setup** screen appears with the **Address** tab highlighted.

- c. Select **Statically Assigned IP address** on the **Address** tab and enter values for IP address, subnet mask, and host name (including the domain name). Then select **Next**.
- d. Select the **Hostname/DNS** tab on the **Network Settings** screen.
  1. For the **Hostname and Domain Name** area, enter host name and domain name.
  2. For the **Name Servers and Domain Search List**, enter Name Server 1, Name Server 2, Name Server 3, and Domain Search.
- e. Select the **Routing** tab on the **Network Settings** screen, then enter the Default IPv4 Gateway (for the network connected to eth0) and set Device to eth0 using the dropdown menu.
- f. Click **OK** after all of the **Network Settings** have been prepared.

### FINAL STEPS

9. Reconnect boot RAID disk cables.

Remove the protective covers from the Fibre Channel or SAS cable connectors, clean the ends of the cable connectors, and reconnect the data cables that connect the SMW to the boot RAID.

10. Reinsert SMW non-boot internal drives.

Reinsert all of the SMW internal disk drives that were removed earlier.

**TIP:** It is not necessary to turn off the power for the SMW before inserting these drives—the operating system can be in a booted state.

11. Eject the base operating system DVD.

If the base operating system DVD (Cray-slebase-12-SP3-201709141039.iso or Cray-slebase-12-SP3-201806150923.iso) is still in the DVD drive, eject it.

```
smw# eject
```

12. Reboot the SMW.

Reboot the SMW to allow the SMW to discover the drives properly.

```
smw# reboot
```

If the SMW was configured with RAID1, then it may still be synchronizing the data between the two disks in the RAID1 mirror. This resync can take about 30 minutes when SLES 12 SP3 is freshly installed. If the SMW is rebooted at this point in the process, the resync will be interrupted. However, that is not a problem because as soon as the SMW is up again, the resync process will continue.

- a. (R815 SMW only) Check the status of RAID1 resync activities on a Dell R815 SMW.

Note that several RAID resyncs may occur. In this example, the resync of md127 is estimated to finish in 24.3 minutes.

```

smw# cat /proc/mdstat
Personalities : [raid1]
md125 : active raid1 sdc2[1] sda2[0]
      33559424 blocks super 1.0 [2/2] [UU]
      bitmap: 0/1 pages [0KB], 65536KB chunk

md126 : active raid1 sda1[0] sdc1[1]
      4200384 blocks super 1.0 [2/2] [UU]
      bitmap: 0/1 pages [0KB], 65536KB chunk

md127 : active raid1 sda3[0] sdc3[1]
      206437248 blocks super 1.0 [2/2] [UU]
      [=====>.....] resync = 33.7% (69700352/206437248)
      finish=24.3min speed=93748K/sec
      bitmap: 2/2 pages [8KB], 65536KB chunk

unused devices: <none>

```

- For a stand-alone SMW or the first SMW in an SMW HA system, the next step in the process is [Configure Boot RAID Devices](#) on page 83.
- (SMW HA only) For the second SMW in an SMW HA system, there is no need to configure the boot RAID because it is shared with the first SMW and has already been configured. The next step in the process is [Make a Snapshot Manually](#) on page 103.

### 4.2.3 Configure Boot RAID Devices

In typical system installations, the RAID provides the storage for file systems used by the SMW, boot node, and SDB node. These file systems are prepared from LVM volumes in LVM volume groups using the physical volumes that are created on the RAID LUNs (logical unit numbers) or volumes. Proper configuration software should be used when assigning RAID LUNs. Failure to use the proper management tools when assigning RAID LUNs can lead to undiscoverable disk devices or boot failure. RAID units also provide user and scratch space and can be configured to support a variety of file systems. For more information about configuring RAID, see *XC™ Series Lustre® Administration Guide (S-2648)*, which is provided with the CLE release package.

### Prerequisites and Assumptions for Configuring the Boot RAID

Sites that require a long distance between the SMW, XC, and the boot RAID will use Fibre Channel (FC) components, while sites that have the SMW, XC, and boot RAID in the same area (within 10 meters) will typically use SAS as the interface for the boot RAID.

- The SMW has an Ethernet connection to the Hardware Supervisory System (HSS) network.
- The SMW has a Fibre Channel (FC) or Serial Attached SCSI (SAS) connection to the boot RAID or to an FC or SAS switch.
- The boot nodes have an FC or SAS connection to the boot RAID or to an FC or SAS switch.
- The SDB nodes have an FC or SAS connection to the boot RAID or to an FC or SAS switch.

### Boot RAID Configuration Procedures

Cray provides support for system boot RAID from NetApp, Inc.

**NOTE:** Cray ships systems with much of this software installed and configured. Performing all of the steps in these boot RAID procedures may not be necessary unless the configuration needs to be changed.



1. Configure the boot RAID for a NetApp, Inc. storage system using the following procedures (reference [Recommended Boot RAID LUN Values](#) on page 84 as needed). The first one installs the SANtricity Storage Manager Utility, which is used to perform the other procedures.
  - a. [Install SANtricity Storage Manager for NetApp, Inc. Devices](#) on page 85
  - b. [Set Up Boot RAID Space for Direct-attached Lustre](#) on page 87
  - c. [Create Boot RAID Volume Group and Volumes for NetApp, Inc. Devices](#) on page 87
2. Zone the SAS (Serial Attached SCSI) or FC (Fibre Channel) switch. For FC storage, there will be an FC Switch to be configured. For SAS storage, there will be a SAS Switch to be configured. Use the applicable procedure(s):
  - [Zone the QLogic FC Switch](#) on page 90 and (recommended) [Create a Backup of the QLogic Switch Configuration](#) on page 91
  - [Zone the Brocade FC Switch](#) on page 92
  - [Zone the LSI SAS Switch](#) on page 99
3. [Reboot the SMW and Verify LUNs are Recognized](#) on page 102

#### 4.2.3.1 Recommended Boot RAID LUN Values

The recommended boot RAID LUN configuration is shown in these tables for different sizes of boot RAID: 4.5 TB, 9.0 TB, and 1.5 TB.

#### Boot RAID with 4.5 TB Available, Non-partitioned System

For a boot RAID with 4.5 TB available, use these values for a non-partitioned system. This is the default configuration installed in the factory.

LUN	Label	Size	Segment Size
0	smw0	3000 GB	256 KB
1	boot0	1000 GB	256 KB
2	sdb0	200 GB	256 KB

#### Boot RAID with 4.5 TB Available, Multiple Partitions

For a boot RAID with 4.5 TB available, use these values for a system with multiple CLE partitions.

- There must be one SMW LUN for the entire system with a size of at least 1000GB.
- There must be one boot LUN for each partition with a size of at least 500GB.
- There must be one SDB LUN for each partition with a size of at least 100GB.

This table shows example values for three CLE partitions.

LUN	Label	Size	Segment Size
0	smw1	2500 GB	256 KB
1	boot1	500	256 KB

LUN	Label	Size	Segment Size
2	sdb1	100 GB	256 KB
3	boot2	500 GB	256 KB
4	sdb2	100 GB	256 KB
5	boot3	500 GB	256 KB
6	sdb3	100 GB	256 KB

### Boot RAID with 9.0 TB Available, Non-partitioned System

For a boot RAID with 9.0 TB available, use these values for a non-partitioned system. Values for boot1 and sdb1 LUNs are shown also, because they can be added to volume groups for the boot node volume group and SDB node volume group, if needed. If added, they should be the same size as the boot0 and sdb0.

LUN	Label	Size	Segment Size
0	smw0	4000 GB	256 KB
1	boot0	1000	256 KB
2	sdb0	200 GB	256 KB
3	boot1	1000 GB	256 KB
4	sdb1	200 GB	256 KB

### Boot RAID with 1.5 TB Available, Non-partitioned System

For a boot RAID with only 1.5 TB available, use these values for a non-partitioned system.

LUN	Label	Size	Segment Size
0	smw0	1000 GB	256 KB
1	boot0	400 GB	256 KB
2	sdb0	100 GB	256 KB

## 4.2.3.2 Install SANtricity Storage Manager for NetApp, Inc. Devices

### About this task

The SANtricity Storage Manager software is generally preinstalled and the SANtricity media is shipped with the system. If the SANtricity software is installed, then the `SMclient` executable will be found in `/opt/SMgr/client`. If this Cray system does not have the software installed on the SMW, install it using this procedure.

### Procedure

1. Prepare X Windows for NetApp SANtricity Storage Manager.

The NetApp installation software will launch an X Windows application, so an X Windows server must be ready. There are many ways to prepare this: logging into SMW console as root, logging into SMW console as crayadm and then becoming root, or logging into SMW from a remote workstation with X Windows port forwarding enabled via ssh.

- If already logged in to the SMW as crayadm, su to root and enable X Windows port forwarding:

```
crayadm@smw> su -
smw# ssh -X localhost
```

- If not already logged on to the SMW, log in and enable X Windows port forwarding like this:

```
user@host> ssh -X root@smw
```

**Trouble?** If unable to ssh, edit `/etc/ssh/sshd_config` and ensure that the following line is uncommented:

```
#Port 22
```

## 2. Copy NetApp SANtricity Storage Manager installer to SMW.

- If installing from the SANtricity Storage Manager CD, insert it into the SMW CD drive and mount the CD.

```
smw# mount /dev/cdrom /media/cdrom
smw# mkdir -p /tmp/netapp
smw# cp -p /media/cdrom/SMIA-LINUX64-11.30.0A00.0010.bin /tmp/netapp
smw# umount /media/cdrom
smw# eject
```

- If installing from the SMIA-LINUX64-11.30.0A00.0010.bin file, copy that file to /tmp/netapp.

```
smw# mkdir -p /tmp/netapp
smw# cp ./SMIA-LINUX64-11.30.0A00.0010.bin /tmp/netapp
```

## 3. Run the NetAPP SANtricity Storage Manager installer.

```
smw# /tmp/netapp/SMIA-LINUX64-11.30.0A00.0010.bin
```

The **SANtricity Storage Manager Introduction** window displays. The following substeps provide guidance through the installation, but the exact steps may differ for newer versions of the NetApp software.

- Select **Next** in the **SANtricity Storage Manager Introduction** window.  
The **License Agreement** window displays.
- Select **I accept the terms of the License Agreement**, then select **Next**.  
The **Select Installation Type** window displays.
- Select **Typical (Full Installation)**, then select **Next**.  
The **Multi-Pathing Driver Warning** window displays.
- Select **OK**.  
The **Pre-Installation Summary** window displays.
- Select **Install**.

The **Installing SANtricity** window displays and shows the installation progress. When the installation completes, an **Install Complete** window displays.

- f. Select **Done** to acknowledge and finish.

The SANtricity client, `SMclient`, is installed in `/opt/SMgr/client`.

#### 4. Enable `crayadm` to run `SMclient`.

To be able to execute `SMclient` from the `crayadm` account, change the ownership and permissions for the executable files. If this step is skipped, only the `root` account will be able to run `SMclient`.

```
smw# chown -R crayadm /opt/SMgr
smw# chmod 775 /opt/SMgr
smw# chmod 755 /opt/SMgr/client/SMcli /opt/SMgr/client/SMclient
smw# chown -R crayadm:crayadm /var/opt/SM
smw# chmod -R ug+w /var/opt/SM
```

### 4.2.3.3 Set Up Boot RAID Space for Direct-attached Lustre

If the system will use direct-attached Lustre (DAL), create LUNs for DAL nodes to use for the MGT, MDT, and OST disk devices. This must be done before installing CLE and DAL.

If creating LUNs on the NetApp 2700 boot RAID device or external Netapp block storage device, use the SANtricity data management software installed on the SMW to create the DAL LUNs.

### 4.2.3.4 Create Boot RAID Volume Group and Volumes for NetApp, Inc. Devices

#### Prerequisites

This procedure assumes the following:

- the SANtricity Storage Manager has been installed
- the user is logged on to the SMW as `crayadm`

#### About this task

This procedure creates the 8+2 Volume Group and 2 Global Hot Spares for a 4.5 TB Volume Group (the amount of storage for this installation may be different). A standard new boot RAID has 2 hot spares; the number of hot spares depends on the number of available drives left over after configuration of the 8+2 RAID6.

#### Procedure

##### 1. Start the SANtricity Storage Manager.

```
crayadm@smw> /usr/bin/SMclient
```

The SANtricity Storage Manager window appears.

##### 2. Select a method for adding a volume group.

If the **Select Addition Method** window appears, choose one of the following options. Otherwise, continue with the next step.

- **Automatic.** Select this option if a serial connection was not used to assign IP addresses to the storage array controllers. The SANtricity software automatically detects the available controllers, in-band, using the Fibre Channel link.

- **Manual.** Select this option if IP addresses have already been assigned to the storage array controllers.

---

CREATE A VOLUME GROUP

---

**3. Create a volume group.**

The following substeps apply only if the **Select Addition Method** window did not display or if the **Manual** option was selected.

- Double-click the name for the storage array to be configured.  
The **Array Management** window displays.
- Select the **Logical/Physical** tab.
- Right-click **Unconfigured Capacity** and select **Create Volume**.  
The **Create Volume** wizard displays.
- Select **Next** on the **Introduction (Create Volume)** window.
- Select the **Manual** option on the **Specify Volume Group (Create Volume)** window.
- Select tray 99, slots 1-10, then select **Add**.
- Verify that the RAID level is set to 6.
- Select **Calculate Capacity**.
- Select **Next** on the **Specify Volume Group (Create Volume)** window.

The **Array Management** window should still be displayed after performing this step.

---

CREATE AND CONFIGURE VOLUMES

---

After creating the first volume group, create the first volume when prompted. Configure the boot RAID with enough LUNs to support the various system management file systems (Cray recommends a minimum of three LUNs).

**4. Create a volume.**

- Enter a new volume capacity. Specify units as GB or MB.
- Enter a name for the volume.
- Select the **Customize Settings** option.
- Select **Next** in the **Specify Capacity/Name (Create Volume)** window.
- Verify the settings on the **Customize Advanced Volume Parameters (Create Volume)** window.

These settings are used for the all of the LUNs.

- For **Volume I/O characteristics type**, verify that **File System** is selected.
  - For **Preferred Controller Ownership**, verify that **Slot A** is selected. This places the LUN on the A Controller.
- Select **Next** in the **Customize Advanced Volume Parameters (Create Volume)** window.
  - Select the **Default** mapping option in the **Specify Volume to LUN Mapping** window.

- h. For **Host** type, select **Linux** from the drop-down menu.
- i. Select **Finish** in the **Specify Volume to LUN Mapping** window.
- j. Select **Yes** when prompted to create more LUNs in the **Creation Successful (Create Volume)** window, unless this is the last volume to be created. If this is the last volume, select **No** and continue with the next step (skipping the rest of these substeps).
- k. Verify that **Free Capacity** is selected on **Volume Group 1 (RAID 6)** in the **Allocate Capacity (Create Volume)** window.
- l. Select **Next** in the **Allocate Capacity (Create Volume)** window.

Repeat this step to create all of the volumes (applicable to this system) described in [Recommended Boot RAID LUN Values](#) on page 84

5. Enable write caching on the volume with the ALPS shared file system.

To reduce risk to end-user data integrity, all SMW boot RAID storage arrays should be configured with write cache support disabled. However, ALPS requires write caching, so the volume that contains the ALPS shared file system will need to have write cache enabled. To mitigate the risk to data integrity, write caching should be used only with battery backup and controller mirroring. See Cray FN5338 "Write cache support statement" for more information.

- a. Select the storage array containing the volume to be write-cache enabled.
- b. Locate the volume to be write-cache enabled, right-click it, then select **Change > Cache Settings**.
- c. Enable the desired cache settings.

6. Indicate that volume creation and configuration is complete.

Select **OK** in the **Completed (Create Volume)** window.

7. Create a hot spare.

The hot spare provides a ready backup if any of the drives in the volume group fail.

- a. Right-click on the last drive in the slot 12 icon on the right portion of the window and select **Hot Spare Coverage**.
- b. Select the **Manually Assign Individual Drives** option.
- c. Select **OK**.
- d. Select **Close**.

8. Exit the tool.

9. (optional) Configure remote logging of the boot RAID messages.

The NetApp, Inc. storage system uses SNMP to provide boot RAID messages. Cray does not provide a procedure for this; see [NetApp, Inc. Storage System documentation](#) for information about how to configure remote logging.

The next step in the process is to zone the switches. Go to one of the following, depending on the switch type:

- [Zone the QLogic FC Switch](#) on page 90.
- [Zone the Brocade FC Switch](#) on page 92

- [Zone the LSI SAS Switch](#) on page 99

### 4.2.3.5 Zone the QLogic FC Switch

#### Prerequisites

This procedure assumes the following:

- The QLogic SANBox™ FC (Fibre Channel) switch has been configured and is on the HSS network.
- The disk device has four host ports connected to ports 0-3 for the QLogic SANbox switch, and the following connections have been made:
  - The SMW must be connected to port 10 on the SANBox.
  - The boot node must be connected to port 4 on the SANBox.
  - The SDB node must be connected to port 5 on the SANBox.

#### About this task

This procedure describes how to use the QuickTools utility to zone the LUNs on the QLogic SANBox FC switch. QuickTools is an application that is embedded in the QLogic switch and is accessible from a workstation browser with a compatible Java™ plug-in. It requires a Java browser plugin, version 1.4.2 or later.

Zoning is implemented by creating a zone set, adding one or more zones to the zone set, and selecting the ports to use in the zone.

**NOTE:** If a LUN is to be shared between failover host pairs, each host must be given access to the LUN. The SMW host port should be given access to all LUNs.

#### Procedure

1. Start a web browser.
2. Enter the IP address of the switch.

If the configuration has a single switch, the IP address is 10.1.0.250. The IP address of each RAID controller is preconfigured by Cray and is listed on a sticker on the back of the RAID controller.
3. Enter the login name and password when the **Add a New Fabric** window pops up and prompts for them.

The default administrative login name is `admin`, and the default password is `password`.

The QuickTools utility displays in the browser.
4. Select **Add Fabric**.

If a dialog box appears stating that the request failed to connect over a secured connection, select **Yes** and continue.
5. Double-click the switch icon when the switch is located and displayed in the window.

Information about the switch displays in the right panel.
6. Select the **Configured Zonesets** tab at the bottom of the panel.



7. Select **Zoning** and then **Edit Zoning** from the toolbar menu.

The **Edit Zoning** window displays.

8. Create a zone set.

- a. Select the **Zone Set** button.

The **Create a Zone Set** window displays.

- b. Create a new zone set.

In this example, assume that the zone set is named `XT0`.

9. Create a zone.

- a. Right-click the `XT0` zone and select **Create a Zone**.

- b. Create a new zone named `BOOT`.

10. Define the ports in the zone.

On the right panel, select the button in front of `BOOT` to open a view of the domain members. Ports 0, 4, 5, and 10 are added to the `BOOT` zone. Define the ports in the zone to ensure that the discovery of LUNs is consistent among the SMW, the boot node, and the SDB node.

- a. Using the mouse, left-click Port # 0 and drag it to the `BOOT` zone.
- b. Using the mouse, left-click Port # 4 and drag it to the `BOOT` zone. This port is for the boot node.
- c. Using the mouse, left-click Port # 5 and drag it to the `BOOT` zone. This port is for the SDB node.
- d. Using the mouse, left-click Port # 10 and drag it to the `BOOT` zone. This port is for the SMW.
- e. Select **Apply**. The error-checking window displays.
- f. Select **Perform Error Check** when prompted.
- g. Select **Save Zoning** after confirming that no errors were found.

11. Select **Yes** when prompted to activate a zone set, then select the appropriate `XT0` zone set.

At this point, Cray recommends creating a backup of the switch configuration ([Create a Backup of the QLogic Switch Configuration](#) on page 91) before closing and exiting the application. Otherwise, proceed to the next step in the boot RAID configuration process: [Reboot the SMW and Verify LUNs are Recognized](#) on page 102.

#### 4.2.3.6 Create a Backup of the QLogic Switch Configuration

##### About this task

Use the QuickTools utility to create a backup of the QLogic switch configuration. To use QuickTools, a Java browser plugin, version 1.4.2 or later is required.

To start a web browser and open the QuickTools utility, complete steps 1 through 4. If the QuickTools utility is already open, skip to step 5.

## Procedure

1. Start a web browser.

2. Enter the IP address of the switch.

The IP address of each RAID controller is preconfigured by Cray and is listed on a sticker on the back of the RAID controller. If the configuration has a single switch, the IP address is 10.1.0.250.

3. Enter the login name and password when the **Add a New Fabric** window pops up and prompts for them.

The default administrative login name is `admin`, and the default password is `password`.

The QuickTools utility displays in the browser.

4. Select **Add Fabric**.

If a dialog box appears stating that the request failed to connect over a secured connection, select **Yes** and continue.

The QuickTools utility opens.

5. Complete the configuration backup from within the QuickTools utility:

- a. Select **Switch** and then **Archive** from the top bar.

A **Save** window pops up with blanks for **Save in:** and **File Name:**.

- b. Enter the directory (for example, `crayadm`) and a file name (for example, `sanbox_archive`) for saving the QLogic switch configuration.

- c. Select the **Save** button.

6. Close and exit the application.

The QLogic FC switch is now zoned and backed up. Proceed to the next step in the boot RAID configuration process: [Reboot the SMW and Verify LUNs are Recognized](#) on page 102.

### 4.2.3.7 Zone the Brocade FC Switch

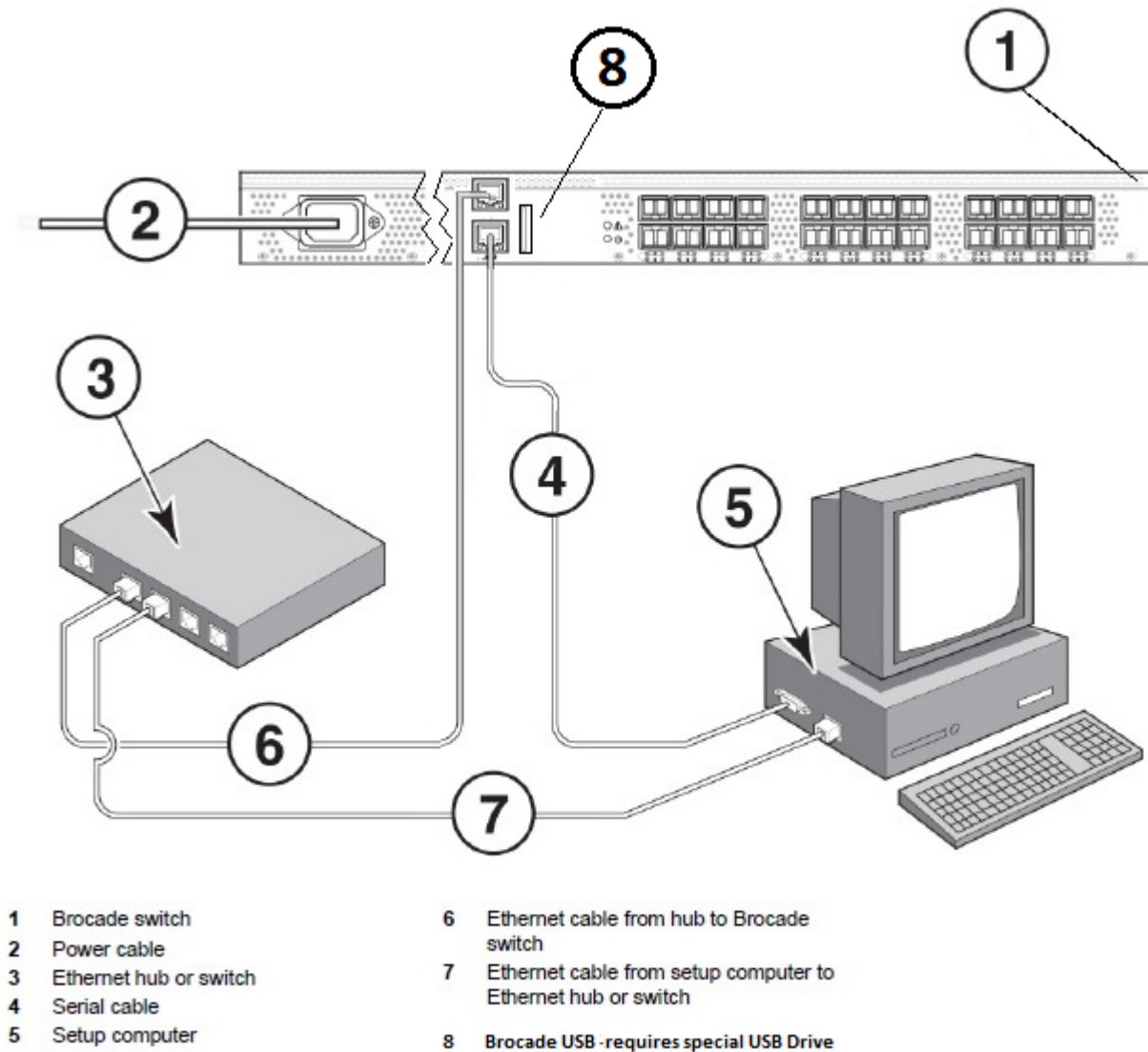
## Prerequisites

This procedure assumes the following:

- The Brocade FC (Fibre Channel) switch has been configured and is on the HSS network.
- The connections shown in the figure have been made.

**NOTE:** The SMW Ethernet port can be directly connected to the switch MGT port if no hub/switch (item 3 in figure) is available.

Figure 24. Brocade FC Switch Connections



## About this task

This procedure describes how to set up and use Web Tools, the embedded GUI application, to configure zoning of the Brocade 6505 Fibre Channel (FC) switch for standard boot RAID usage.

## Procedure

1. Set up the GUI.
  - a. Open a Firefox web browser on the SMW.

```
crayadm@smw> cd
crayadm@smw> firefox
```

- b. Enter the IP address of the switch (10.1.0.250) into the address bar.

A pop up window appears.

- c. Select **Save File** to save the file in the default Downloads directory.

The Java plugin starts the GUI and a security warning appears, stating that the certificate is not trusted and the browser will not let the application continue.

- d. Exit the browser.
- e. Run `javaws -viewer` in an SMW xterm window.

Substitute the correct version of java in the javaws path, if different from this example.

```
crayadm@smw> cd
crayadm@smw> cd /Downloads
crayadm@smw> /usr/lib64/jvm/java-1.7.1-ibm-1.7.1/jre/bin/javaws -viewer
```

The **Java Control Panel** appears.

- f. Add the switch to the **Exception Site List** in the **Security** tab.

Select the **Security** tab. Under **Exception Site List**, click **Edit Site List** and enter `http://10.1.0.250` as a trusted site, then click **Apply**.

- g. Exit the `javaws -viewer` application.
- h. Open `switchExplorer_installed.html` in an SMW xterm window.

Change directories to the `Downloads` directory, which is where the `switchExplorer_installed.html` file should be located (type `ls` to verify, if desired). Run `javaws -verbose switchExplorer_installed.html`.

Substitute the correct version of java in the javaws path, if different from this example.

```
crayadm@smw> cd /Downloads
crayadm@smw> /usr/lib64/jvm/java-1.7.1-ibm-1.7.1/jre/bin/javaws -verbose \
switchExplorer_installed.html
```

A **Verifying application** window appears. About three minutes later, a **Security Warning** window appears.

- i. Select **I accept the risk and want to run this application**, then click **Run**.

A **Login** window appears.

- j. Log in to the switch as `admin` with password `password`.

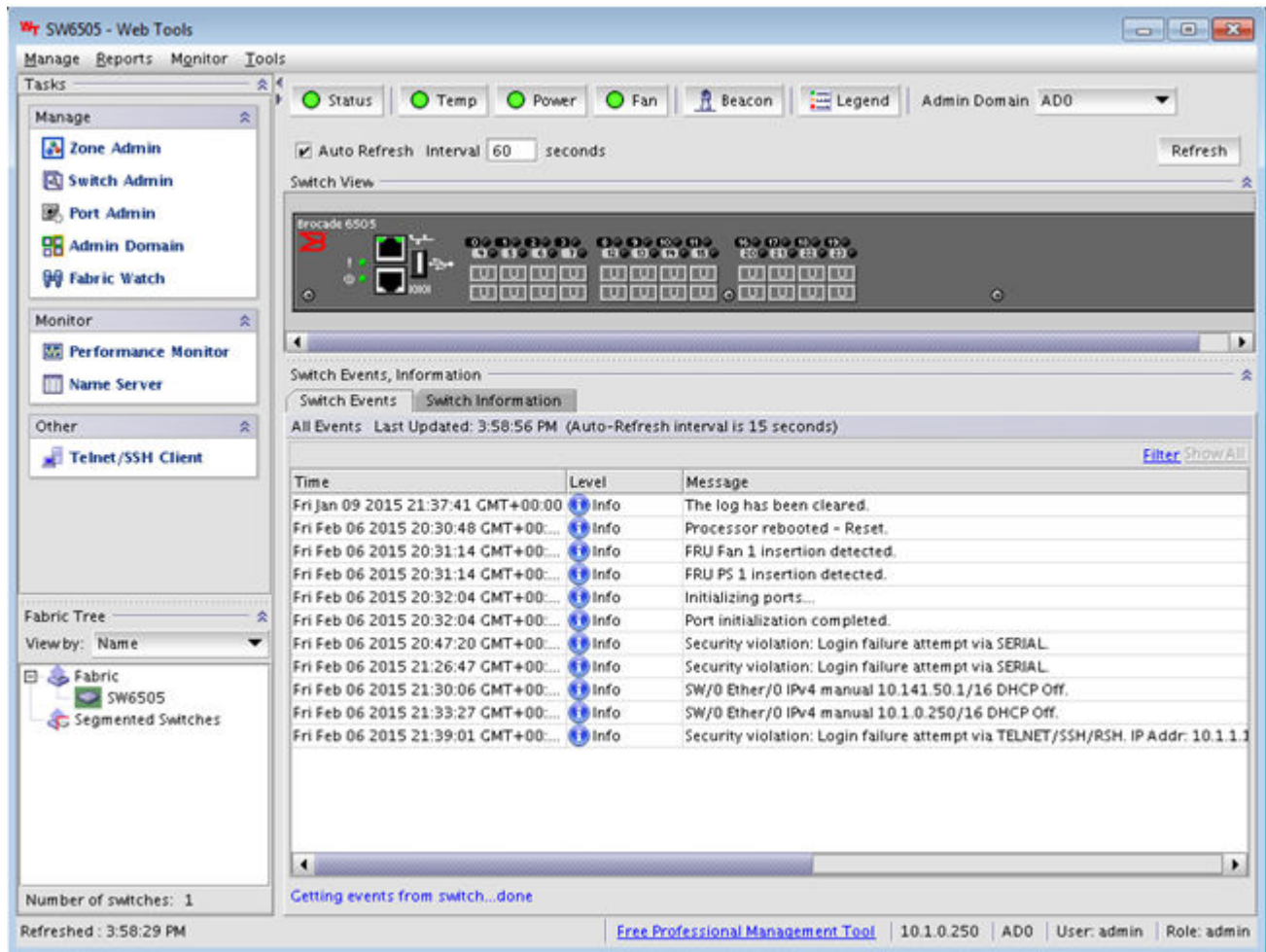
The main **Web Tools** switch window appears.

Use the Web Tools GUI to complete configuration of the Brocade FC switch.

2. Check the LED status of the tabs on the **Web Tools** window to ensure that there are no major issues.

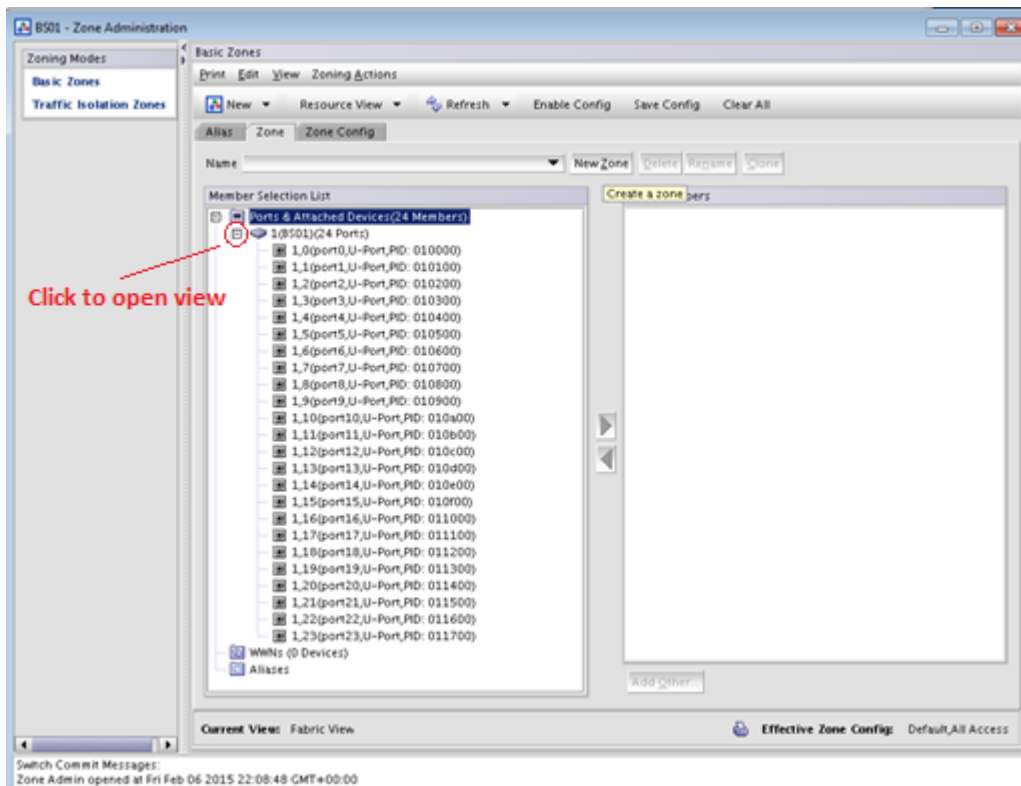
The tab LEDs should all be green.

Figure 25. Brocade FC Switch Web Tools Window



3. Change the name of the switch.
  - a. Click **Switch Admin** in the **Manage** pane (upper left).  
The **Switch Administration** window appears.
  - b. Change the name of the switch to BS#, where the # is the number of the switch being configured (e.g., BS01), then click **Apply** to save the name.  
A confirmation window appears.
  - c. Click **Yes** to confirm, then close the **Switch Administration** window to return to the main **Web Tools** window.
4. Set up a Boot zone for the switch.
  - a. Click **Zone Admin** in the **Manage** pane (upper left).  
The **Zone Administration** window appears.
  - b. Click the **Zone** tab in the **Zone Administration** window, then click the **New Zone** button (located just below tabs, to the right).

Figure 26. Brocade FC Switch Zone Administration Window



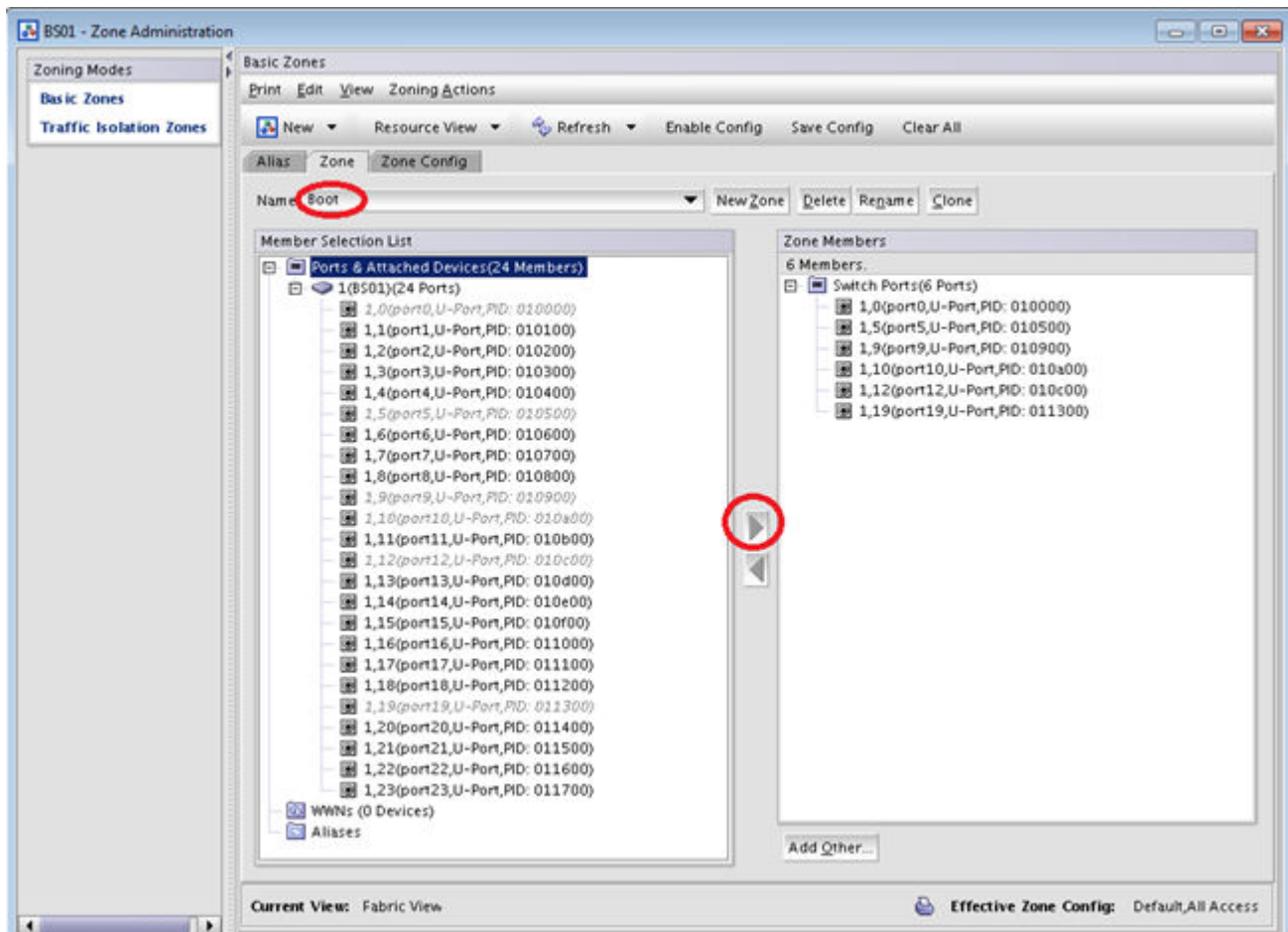
A **Create New Zone** window appears.

- c. Enter the name of the new zone as "Boot."

"Boot" is displayed in the **Name** field in the **Zone** tab of the **Zone Administration** window.

- d. Select port 0 in the **Member Selection List** (left pane in the **Zone** tab), then click the right arrow icon to move port 0 into **Zone Members** (right pane in the **Zone** tab). Repeat for ports 4, 5, 10, 11, 12, and 19.

Figure 27. Brocade FC Switch Boot Zone Members



Ports 0, 4, 5, 10, 11, 12, and 19 are in the **Zone Members** pane.

5. Configure the Boot zone.

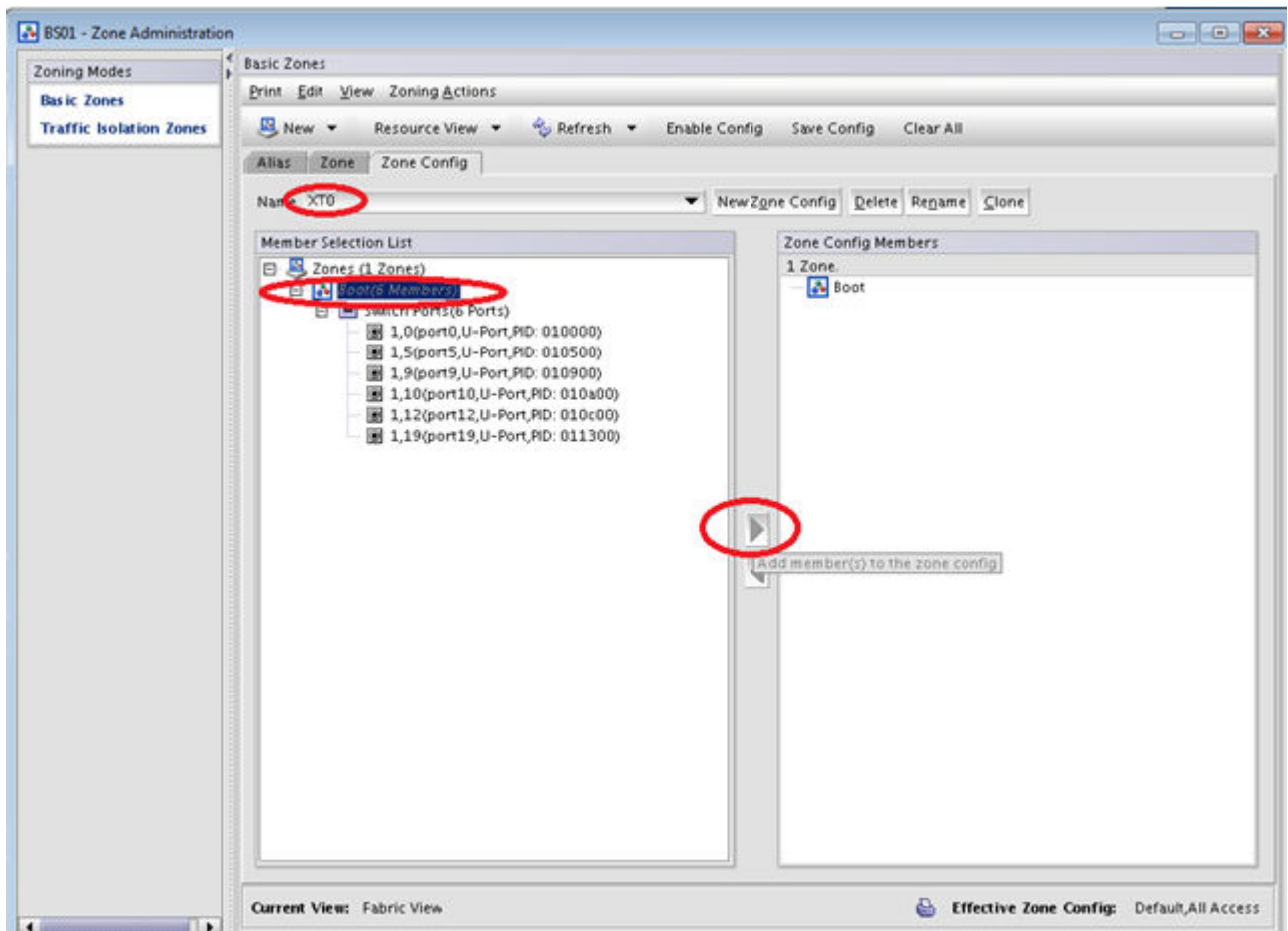
- a. Click the **Zone Config** tab in the **Zone Administration** window.
- b. Click all of the + icons in the **Member Selection List** to expand all of the entries.
- c. Click the **New Zone Config** button.

A **Create New Config** window appears.

- d. Enter the name "XT0" in the **Create New Config** window, then click **OK**.
- e. Select the Boot zone in the **Member Selection List** (left pane in the **Zone Config** tab), then click the right arrow icon to move the Boot zone into **Zone Config Members** (right pane in the **Zone Config** tab), which puts it in the XT0 group.



Figure 28. Brocade FC Switch XT0 Zone Config Members



f. Click the **Save Config** button (located just above tabs, to the right), then click **Yes** in the confirmation window that appears.

6. Enable the Boot zone configuration.

a. Click the **Enable Config** button (located just above tabs, to the right).

A **Choose Zone Config to be enabled** window appears.

b. Select the XT0 zone config from the menu, click **OK**, then click **Yes** in the **Enable Config XT0** confirmation window that appears.

At the bottom of the **Zone Administration** window, a status appears.

c. Click the **X** button at the top right to exit from the **Zone Administration** window when the status shows a "Commit succeeded" message, then click **Yes** in the exit confirmation window that appears.

The main **Web Tools** window appears.

7. Verify the configuration.

a. Remove power from the switch.

b. Re-apply power after 30 seconds, then wait for the Brocade FC switch to finish booting via the serial connection.



- c. Enter the following from the serial connection:

```
BS01:> enable
BS01:> zoneshow

Defined configuration:
cfg:   XT0      boot
zone:  boot     1,0; 1,4; 1,5; 1,10; 1,11; 1,12; 1,19

Effective configuration:
cfg:   XT0
zone:  boot 1,0
        1,4
        1,5
        1,10
        1,11
        1,12
        1,19
```

- d. Verify that the configuration matches the effective configuration.

The Brocade FC switch is now zoned. Proceed to the next step in the boot RAID configuration process: [Reboot the SMW and Verify LUNs are Recognized](#) on page 102.

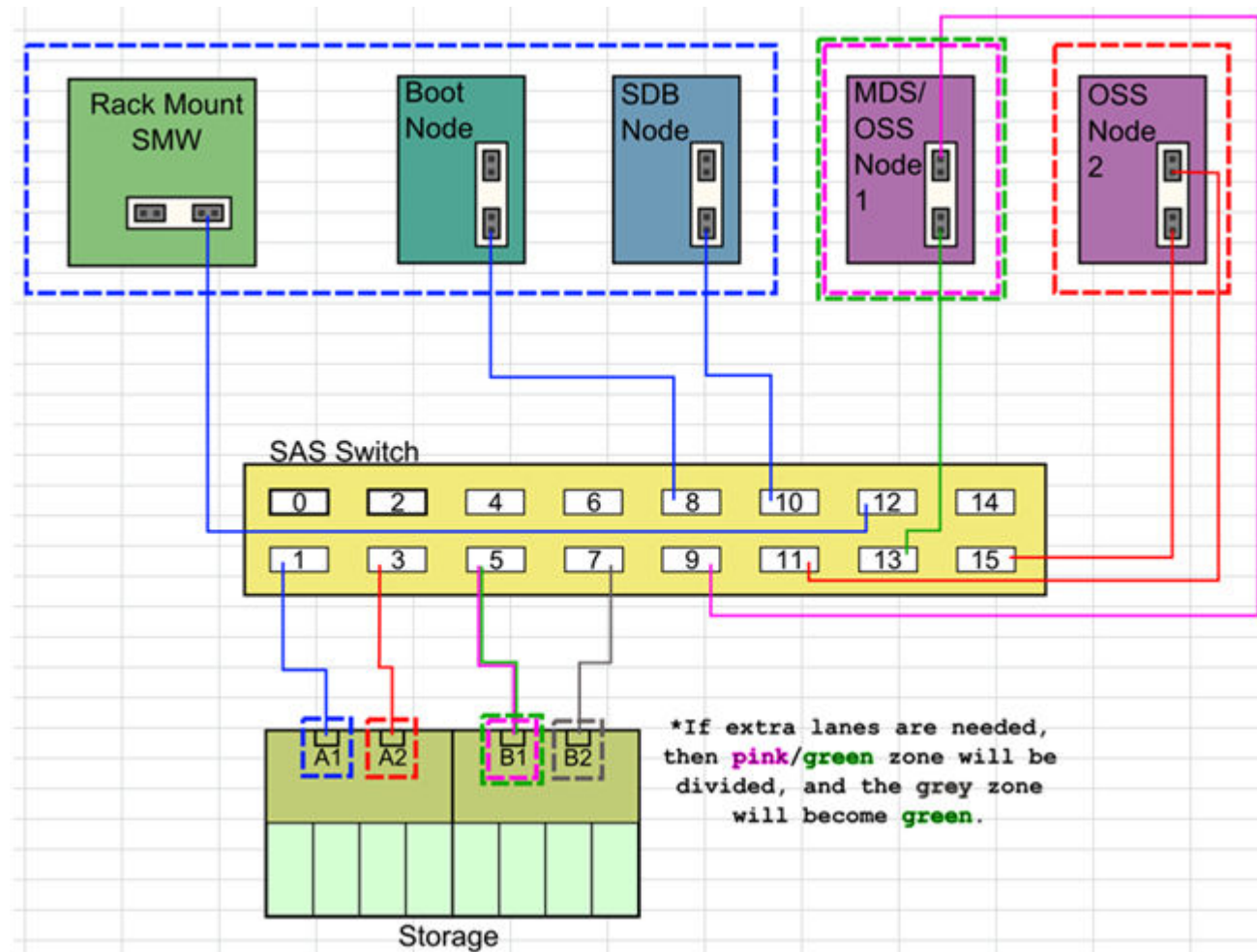
#### 4.2.3.8 Zone the LSI SAS Switch

### Prerequisites

This procedure assumes the following:

- The LSI 6160 SAS (Serial Attached SCSI) switch has been configured and is on the HSS network.
- The SMW is within 10 meters of the SAS switch (otherwise an FC switch is required).
- The following connections have been made (as shown in figure):
  - The SMW SAS card port 1 is connected by SAS cable to port 12 on the SAS switch.
  - The boot node is connected by SAS cable to port 8 on the SAS switch.
  - The SDB node is connected by SAS cable to port 10 on the SAS switch.

Figure 29. LSI SAS Switch Boot RAID Cable Connections



## About this task

This procedure describes how to use SAS Domain Manager, the embedded GUI application, to zone the LUNs on the LSI (now Broadcom) 6160 SAS switch for standard boot RAID usage.

## Procedure

### 1. Open the SAS Domain Manager GUI.

- Open a Firefox web browser on the SMW.

```
crayadm@smw> cd
crayadm@smw> firefox
```

- Enter the IP address of the switch (10.1.0.250) into the address bar.
- If Firefox displays the message "JRE 1.6 or higher required," do one of the following:
  - Select Tools > Add-ons > Plugins** from the Firefox menu bar, then click the green button marked "enable" for Java to enable Java and allow the switch GUI to work.

- Run the following command from the command line to bypass Firefox and open the GUI.

```
crayadm@smw> /usr/lib64/jvm/jre/bin/javaws http://10.1.0.250/sdmgui.jnlp
```

When the **Opening sdmgui.jnlp** window appears, click OK to open the file, then in the **Warning - Security** window that appears, click **Run** to run the application.

The **SAS Domain Manager GUI** login window appears.

- Log in to the switch as `admin` with password `admin`.

The **SAS Domain Manager GUI** main window appears.

## 2. Create the zone groups.

- Click the **Domain** tab in the **SAS Domain Manager GUI** main window, then click **Create Zone Group**.

Create the following zone groups and assign the ports as indicated. Note that it is the phys values that should be mapped to the zone group, not the port(s).

Zone Group Name	SAS Ports	Phys
XT-A1-Storage	1	4,5,6,7
XT-A2-Storage	3	12,13,14,15
XT-B1-Storage	5	20,21,22,23
XT-B2-Storage	7	28,29,30,31
XT-SysRaid	8 10 12 4 6	32,33,34,35 40,41,42,43 48,49,50,51 16,17,18,19 24,25,26,27
XT-MDS-OSSn1 <sup>(1)</sup>	9 13 14	36,37,38,39 52,53,54,55 56,57,58,59
XT-OSSn2 <sup>(1)</sup>	11 15	44,45,46,47 60,61,62,63

(1) If an external Lustre server will be used instead of an internal Lustre server (DAL), then the XT-MDS-OSSn1 and XT-OSSn2 zones are not necessary.

## 3. Create the zone set.

- Click the **Domain** tab in the **SAS Domain Manager GUI** main window, then click **Create Zone Set**.

Create the zone set XT0.

- b. Assign the zone groups to the zone set by clicking the empty boxes to match this layout:

Zone Group Name	1	2	3	4	5	6	7
XT-A1-Storage					X		
XT-A2-Storage							X
XT-B1-Storage						X	
XT-B2-Storage						*X*(2)	
XT-SysRaid	X						
XT-MDS-OSSn1 <sup>(1)</sup>			X	*X*(2)			
XT-OSSn2 <sup>(1)</sup>		X					

(1) If an external Lustre server will be used instead of an internal Lustre server (DAL), then the XT-MDS-OSSn1 and XT-OSSn2 zones are not necessary.

(2) Setting a box marked with \*X\* may add more SAS lanes for performance, but do not set unless instructed to.

4. Activate the zone set.
  - a. Click the **Domain** tab, then click **Activate/Deactivate Zone Set**.
  - b. Select XT0 from the menu and enter the Zone Password of `lynx` to activate the zone set.
  - c. Click the **Views** tab under **Active zone set** to verify that the zone groups and zone set are correct and active.

The SAS switch is now zoned. Proceed to the next step in the boot RAID configuration process: [Reboot the SMW and Verify LUNs are Recognized](#) on page 102.

#### 4.2.3.9 Reboot the SMW and Verify LUNs are Recognized

##### About this task

Use this procedure to make the SMW rediscover the LUNs (logical unit numbers) and zones that were created.

##### Procedure

1. Log on as the `root` user.

```
crayadm@smw> su - root
```

2. Reboot the SMW to ensure that the LUNs are recognized.

```
smw# reboot
```



**CAUTION:** Failure to reboot the SMW at this point could produce unexpected results later on.

3. When the SMW has finished rebooting, log on as the `root` user.

```
crayadm@smw> su - root
```

4. Execute the `lsscsi` command to verify that the LUNs (volumes) have been rediscovered.

```
smw# lsscsi
```

5. List the disk devices by using the `fdisk` command to verify that the LUNs (volumes) are configured according to the boot LUN configuration table in [Recommended Boot RAID LUN Values](#) on page 84.

```
smw# fdisk -l
```

## 4.2.4 Make a Snapshot Manually

### Prerequisites

The SLES 12 SP3 base operating system has been installed on the SMW and boot RAID devices have been configured, but no other software has been installed yet.

### About this task

Create a Btrfs snapshot of the SMW immediately after SLES 12 SP3 has been installed and before any files or directories have been modified by Cray's installation software or the rest of the installation process. With this snapshot, it will be possible to revert to this point if an initial/fresh install is repeated.

Snapshots are usually made using the `snaptail` program, but that program has not been installed at this point in the installation process. `snaptail` will be installed to the SMW with other Cray RPMs for the SMW and will be used for all Btrfs snapshot manipulations after this point.

For more information, see [About Snapshots and Config Set Backups](#) on page 21.

### Procedure

1. Determine the root subvolume.

It will be the string starting with "UUID." In this example it is "UUID=ffb0b613-2033-4835-87b5-6ca8ff1bacde."

```
smw# grep " / " /etc/fstab
UUID=ffb0b613-2033-4835-87b5-6ca8ff1bacde /          btrfs
defaults          0 0
```

2. Mount the root subvolume.

Substitute the correct subvolume string for the example string shown in this command.

```
smw# mount -o subvol=@ UUID=ffb0b613-2033-4835-87b5-6ca8ff1bacde /mnt
```

3. Create a subvolume for snapshots (if `/mnt/snapshots` does not already exist).

```
smw# btrfs sub create /mnt/snapshots
```

4. Create the snapshot (if `/mnt/snapshots/SLES12SP3` does not already exist).

```
smw# btrfs sub snap / /mnt/snapshots/SLES12SP3
```

5. Unmount the snapshot.

```
smw# umount /mnt
```

6. Make a new `/media/root-sv` directory.

```
smw# mkdir -p /media/root-sv
```

7. Mount root subvolume under `/media/root-sv` instead of `/mnt` as was used above.

Substitute the correct subvolume string for the example string shown in this command.

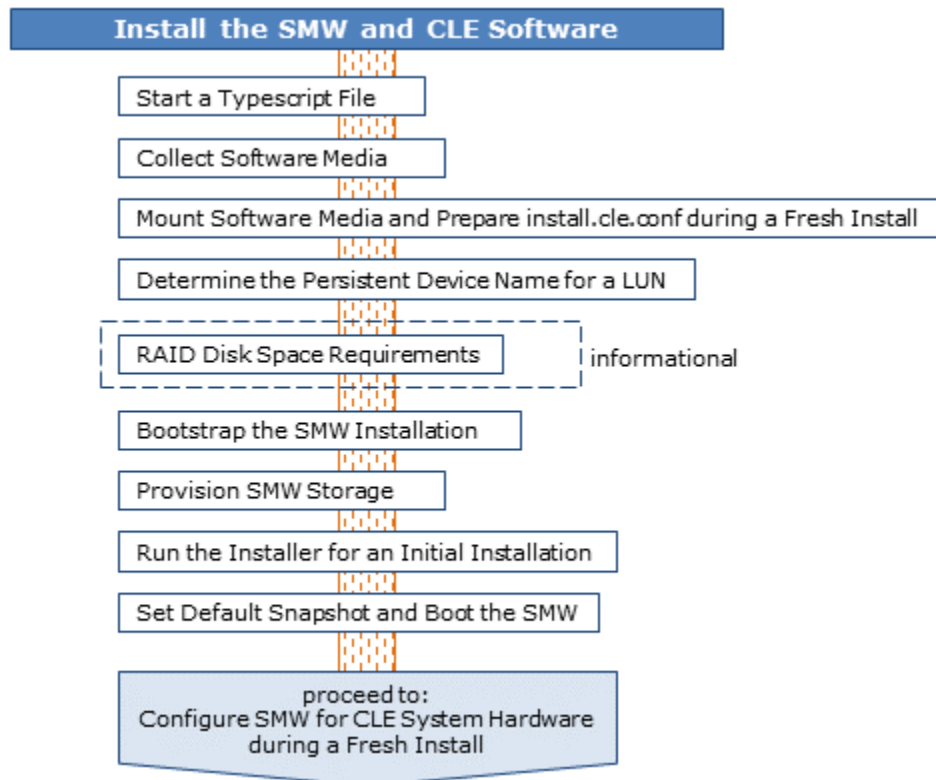
```
smw# mount -o subvol=@ UUID=ffb0b613-2033-4835-87b5-6ca8ff1bacde /media/root-sv
```

A "SLES12SP3" snapshot has been made. Reboot to this snapshot whenever it is necessary to restart a fresh software installation from this point.

## 4.3 Install the SMW and CLE Software

To install the SMW and CLE software, use the following procedures and reference topics in the order listed.

Figure 30. Visual Guide to Installing the SMW and CLE Software



Use [Installation Checklist 2: Install the SMW and CLE Software](#) on page 441 to track progress through this part of the fresh install process.

### 4.3.1 Start a Typescript File

#### About this task

Sites can make as few or as many typescripts as they deem useful. Cray recommends starting a typescript file at these milestones:

- just before installing a new software release
- just before configuring the newly installed software

#### Procedure

1. Log in as root to the SMW.
2. (First time only) Create a release directory for the typescript file.

```
smw# mkdir -p /var/adm/cray/release
```

3. Change to the release directory.

```
smw# cd /var/adm/cray/release
```

4. Set a variable equal to today's date.

```
smw# export TODAY=`date +%Y%m%d`  
smw# echo ${TODAY}
```

5. Start a typescript file.

```
smw# script -af ${TODAY}.suffix
```

For *suffix*, substitute a unique string to distinguish among typescript files, such as `install.1` or `update.2`.

6. Change prompt to include a timestamp.

```
smw# PS1="\[\e[1;31m\]\u@\h:\w \t # \[\e[0m\]\[\e[00m\]"
```

### 4.3.2 Collect Software Media

#### About this task

The Cray release distribution consists of one DVD and several other pieces of media that may be on DVDs or furnished as ISO files. These ISO files are available for download at CrayPort (<https://crayport.cray.com>).

The installer requires several ISO files to be available for setting up and installing packages from SLE repositories. The names of these ISOs are hard-coded in the installer configuration, but the containing directory can be anywhere that makes sense for this site.

**IMPORTANT:** The installer expects these ISO files to be located in the default location, `/root/isos`. If that default location is not used for this system, specify the correct location for the ISO files by using the `--iso-dir` option with the `SMWinstall` command.

## Procedure

1. Make an ISO directory on the SMW that is exempt from snapshots, and link it to the default ISO location.

Instead of placing the ISOs directly in `/root/isos`, create a directory in the Btrfs subvolume `/var/adm/cray`, which is exempt from snapshots, place the ISOs there, and link the two directories. This prevents the large ISO files from unnecessarily increasing the size of snapshots.

```
smw# mkdir -p /var/adm/cray/release/isos
smw# ln -s /var/adm/cray/release/isos /root/isos
```

The "ISO directory" referred to in the following steps is `/var/adm/cray/release/isos`.

2. Download the SLES 12 SP3 distribution ISOs to the ISO directory on the SMW.
  - `SLE-12-SP3-Server-DVD-x86_64-GM-DVD1.iso`
  - `SLE-12-SP3-Server-DVD-aarch64-GM-DVD1.iso`
  - `SLE-12-SP3-SDK-DVD-x86_64-GM-DVD1.iso`
  - `SLE-12-SP3-SDK-DVD-aarch64-GM-DVD1.iso`
  - `SLE-12-SP3-WE-DVD-x86_64-GM-DVD1.iso`
  - `SLE-12-Modules-v3.iso`
3. Download the CentOS 6.5 distribution ISOs to the ISO directory on the SMW.
  - `CentOS-6.5-x86_64-bin-DVD1.iso`
4. Download CLE 6.0 and SMW 8.0 ISOs to the ISO directory on the SMW.
  - SMW release: `smw-8.0.7128-201806242300.iso`
  - CLE release: `cle-6.0.7128-201806242300.iso`
5. Download the SLES 12 security updates ISO (`sleupdate-12sp3+180209-201802091328.iso`) to the ISO directory on the SMW.
6. Make a directory on the SMW (if it does not already exist) to hold any patches that may be available on CrayPort.

```
smw# mkdir -p /var/adm/cray/release/patchsets
```

7. Download any SMW, CLE, and SMW HA patches to the patchset directory on the SMW, as described in the release notes (ERRATA and README files).

### 4.3.3 Mount Software Media and Prepare `install.cle.conf` during a Fresh Install

#### Prerequisites

The release software media have been collected and placed in the appropriate directories on the SMW.



## About this task

This procedure sets environment variables for the SMW, CLE, and SLES security updates media, mounts the SMW and CLE media, and prepares the `install.cle.conf` file.

## Procedure

### —— SET ENVIRONMENT VARIABLES ——

1. Set an environment variable for the SMW media.

- a. Confirm that this is the right SMW media.

```
smw# ls -l /root/isos/smw*iso
-rw-r--r-- 1 root root 427184128 July 10 21:42 smw-8.0.7128-201806242300.iso
```

- b. Set an environment variable for the SMW media.

```
smw# export SMW_RELEASE=/root/isos/smw-8.0.7128-201806242300.iso
smw# echo $SMW_RELEASE
```

2. Set an environment variable for the CLE media.

- a. Confirm that this is the right CLE media.

```
smw# ls -l /root/isos/cle*iso
-rw-r--r-- 1 root root 1146388480 July 10 20:38 cle-6.0.7128-201806242300.iso
```

- b. Set an environment variable for the CLE media.

```
smw# export CLE_RELEASE=/root/isos/cle-6.0.7128-201806242300.iso
smw# echo $CLE_RELEASE
```

3. Set an environment variable for the SLES 12 SP3 security updates media.

- a. Confirm that this is the right SLES 12 SP3 security updates media.

```
smw# ls -l /root/isos/sleupdate*iso
-rw-r--r-- 1 root root 3482064896 July 10 09:19 /root/isos/
sleupdate-12sp3+180209-201802091328.iso
```

- b. Set an environment variable for the SLES 12 SP3 security updates media.

```
smw# export SLE_UPDATE=/root/isos/sleupdate-12sp3+180209-201802091328.iso
smw# echo $SLE_UPDATE
```

### —— MOUNT THE SMW AND CLE MEDIA ——

4. Mount the SMW release media.

```
smw# mkdir -p /media/SMW
smw# mount -o loop,ro ${SMW_RELEASE} /media/SMW
```

5. Temporarily mount the CLE release media so that `install.cle.conf.example` will be accessible.

```
smw# mkdir -p /media/CLE
smw# mount -o loop,ro ${CLE_RELEASE} /media/CLE
```

---

———— PREPARE THE `install.cle.conf` FILE ————

6. Copy `install.cle.conf.example` to `install.cle.conf`.

The `install.cle.conf` file contains configuration that controls the installer's image building behavior. Copy `install.cle.conf.example` from the CLE installation media to `/var/adm/cray/install.cle.conf`, and then customize `install.cle.conf` for this system, if necessary.

```
smw# cp -p /media/CLE/products/cle/install.cle.conf.example \
/var/adm/cray/install.cle.conf
```

This file does not need to be changed for a fresh install. Later in the process it will be changed to prepare it for updates to CLE. However, if this site decides to enable image building during installation and wishes to enable image building in parallel, then make the following changes to `install.cle.conf`.

```
build_images: yes
...
build_images_processes: 16
```

Change the value of `build_images_processes` (16 in the example) to the number of images to be built concurrently for this system.

For more information about building images in parallel, see [About Parallel Image Creation](#) on page 40.

———— UNMOUNT CLE MEDIA ————

7. Unmount CLE media.

```
smw# umount /media/CLE
```

### 4.3.4 Determine the Persistent Device Name for a LUN

#### About this task

After initial partitioning of the boot RAID, always address the storage using its persistent `/dev/disk/by-id/` name. Do not use the short `/dev/sdxx` name, which cannot uniquely identify the disk between reboots.

Use either step 1 (for systems not using multipath) or step 2 (for systems using multipath) of this procedure to determine the persistent (by-id) device names for the following devices:

- Disk devices on the boot RAID that will be used for boot node persistent storage
- Disk devices on the boot RAID that will be used for SDB node persistent storage
- Disk devices on the boot RAID that will be used for SMW persistent storage

#### Procedure

1. (For systems not using multipath) Determine the persistent device name on a system not configured for multipath.
  - a. Use `lsscsi` to show the `/dev/sd*` device name associated with a LUN or volume group.

In the first column of the output, the LUN is the final number in the `[n:n:n:n]` value. In this example, LUN 15 is associated with `/dev/sdo`.

```
smw# lsscsi
[0:0:0:0]    disk    ATA      TOSHIBA MK1661GS ME0D  /dev/sda
[0:0:1:0]    disk    ATA      ST91000640NS    AA03  /dev/sdb
[0:0:2:0]    disk    ATA      TOSHIBA MK1661GS ME0D  /dev/sdc
.
.
.
[5:0:0:15]   disk    LSI      INF-01-00        0786  /dev/sdo
[5:0:0:16]   disk    LSI      INF-01-00        0786  /dev/sdp
[5:0:0:17]   disk    LSI      INF-01-00        0786  /dev/sdq
[5:0:0:18]   disk    LSI      INF-01-00        0786  /dev/sdr
```

- b. Use `ls -l` to map the `/dev/sd*` device name to the persistent device name.

To display the persistent device name for only one LUN, use `grep`. This example displays the persistent device name for `/dev/sdo` (that is, LUN 15). Substitute for `sdo` the device for which the persistent device name is being determined.

```
smw# ls -l /dev/disk/by-id | grep sdo
lrwxrwxrwx 1 root root 10 Sep  4 00:56
scsi-360080e500037667a000003a2519e3ff2 -> ../../sdo
lrwxrwxrwx 1 root root 10 Sep  4 00:56
wwn-0x60080e500037667a000003a2519e3ff2 -> ../../sdo
```

There are two results for LUN 15. The one with prefix "scsi" is the one to use, so the persistent device name for LUN 15 is `scsi-360080e500037667a000003a2519e3ff2`.

2. (For systems using multipath) Determine the persistent device name on a system configured for multipath.

On a system using multipath, when the multipath daemon is running, the persistent device name is the multipath path.

- a. Determine whether the multipath daemon is active (running).

```
smw# systemctl status -l multipathd
```

If the output of the `systemctl status` command looks like this, multipath is running.

```
• multipathd.service - Device-Mapper Multipath Device Controller
  Loaded: loaded (/usr/lib/systemd/system/multipathd.service; enabled;
  vendor preset: disabled)
  Active: active (running) since Thu 2017-09-07 19:06:04 CDT; 3 days ago
  Main PID: 2140 (multipathd)
  Status: "idle"
  Tasks: 10 (limit: 131072)
  CGroup: /system.slice/multipathd.service
          └─2140 /sbin/multipathd -d -s
```

If the output of the command looks like this, multipath is not running.

```
• multipathd.service - Device-Mapper Multipath Device Controller
  Loaded: loaded (/usr/lib/systemd/system/multipathd.service; disabled;
  vendor preset: disabled)
  Active: inactive (dead)
```

If the output of the command looks like this, the system is not using multipath.

```

• multipathd.service
  Loaded: not-found (Reason: No such file or directory)
  Active: inactive (dead)

```

- b. If multipathd is not running, start it now.

```
smw# systemctl start multipathd
```

- c. Use the `SMdevices` command to determine the boot RAID volume ID for each volume group.

This command is available on systems with the SANtricity Storage Manager software installed. Note that for each volume group, there are multiple paths that reference the same volume ID (shown in bold in the following examples).

For the boot node volume group:

```

smw# SMdevices | grep boot
/dev/sdj (/dev/sg10) [Storage Array RR_10000716_Boot, Volume boot0, LUN 1, Volume ID
<600a098000a9d1b9000000cb5805168d>, Alternate Path (Controller-B): Non owning controller -
Active/Non-optimized]
/dev/sdm (/dev/sg13) [Storage Array RR_10000716_Boot, Volume boot0, LUN 1, Volume ID
<600a098000a9d1b9000000cb5805168d>, Preferred Path (Controller-A): Owning controller - Active/
Optimized]
/dev/sdd (/dev/sg4) [Storage Array RR_10000716_Boot, Volume boot0, LUN 1, Volume ID
<600a098000a9d1b9000000cb5805168d>, Preferred Path (Controller-A): Owning controller - Active/
Optimized]
/dev/sdg (/dev/sg7) [Storage Array RR_10000716_Boot, Volume boot0, LUN 1, Volume ID
<600a098000a9d1b9000000cb5805168d>, Alternate Path (Controller-B): Non owning controller -
Active/Non-optimized]

```

For the SDB node volume group:

```

smw# SMdevices | grep sdb
/dev/sdk (/dev/sg11) [Storage Array RR_10000716_Boot, Volume sdb0, LUN 2, Volume ID
<600a098000a9d1b9000000cd5805169a>, Alternate Path (Controller-B): Non owning controller -
Active/Non-optimized]
/dev/sdn (/dev/sg14) [Storage Array RR_10000716_Boot, Volume sdb0, LUN 2, Volume ID
<600a098000a9d1b9000000cd5805169a>, Preferred Path (Controller-A): Owning controller - Active/
Optimized]
/dev/sde (/dev/sg5) [Storage Array RR_10000716_Boot, Volume sdb0, LUN 2, Volume ID
<600a098000a9d1b9000000cd5805169a>, Preferred Path (Controller-A): Owning controller - Active/
Optimized]
/dev/sdh (/dev/sg8) [Storage Array RR_10000716_Boot, Volume sdb0, LUN 2, Volume ID
<600a098000a9d1b9000000cd5805169a>, Alternate Path (Controller-B): Non owning controller -
Active/Non-optimized]

```

For the SMW node volume group:

```

smw# SMdevices | grep smw
/dev/sdl (/dev/sg12) [Storage Array RR_10000716_Boot, Volume smw0, LUN 0, Volume ID
<600a098000a9d1b9000000c858051681>, Preferred Path (Controller-A): Owning controller - Active/
Optimized]
/dev/sdc (/dev/sg3) [Storage Array RR_10000716_Boot, Volume smw0, LUN 0, Volume ID
<600a098000a9d1b9000000c858051681>, Preferred Path (Controller-A): Owning controller - Active/
Optimized]
/dev/sdf (/dev/sg6) [Storage Array RR_10000716_Boot, Volume smw0, LUN 0, Volume ID
<600a098000a9d1b9000000c858051681>, Alternate Path (Controller-B): Non owning controller -
Active/Non-optimized]
/dev/sdi (/dev/sg9) [Storage Array RR_10000716_Boot, Volume smw0, LUN 0, Volume ID
<600a098000a9d1b9000000c858051681>, Alternate Path (Controller-B): Non owning controller -
Active/Non-optimized]

```

- d. List the `/dev/disk/by-id/` (persistent) device names associated with the boot RAID volume IDs identified in the previous step, and then identify the correct one to use for each volume group.

The correct device name to use for multipath is the one that has "dm-uuid-mpath-3" prepended to the volume ID.

For example, the boot node volume ID is included in the following device names. The correct device name to use for multipath for the boot node volume group (`boot_node_vg`) is shown in bold.

```
smw# ls -l /dev/disk/by-id/ | grep 600a098000a9d1b9000000cb5805168d
lrwxrwxrwx 1 root root 10 Dec 22 15:08 dm-name-3600a098000a9d1b9000000cb5805168d -> ../../dm-2
lrwxrwxrwx 1 root root 10 Dec 22 15:08 dm-uuid-mpath-3600a098000a9d1b9000000cb5805168d -> ../../dm-2
lrwxrwxrwx 1 root root 10 Dec 22 15:08 scsi-3600a098000a9d1b9000000cb5805168d -> ../../dm-2
lrwxrwxrwx 1 root root 10 Dec 22 15:08 wwn-0x600a098000a9d1b9000000cb5805168d -> ../../dm-2
```

For this example, the device name `dm-uuid-mpath-3600a098000a9d1b9000000cb5805168d` would be entered as the value for `cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.devices` when configuring the boot node volume group (part of bootstrapping the SMW installation).

### 4.3.5 RAID Disk Space Requirements

The SMW, the boot node, and the SDB node all use space on the boot RAID. Here are the recommended sizes for the RAID LUNs, or LVM volume groups, based on the file systems for each. This information will be needed to bootstrap the SMW installation, which is next in the installation process. When configuring the size of volumes within a volume group, ensure that their combined size does not exceed the available space in that volume group.

### SMW File Systems

On the boot RAID, the LVM volume group for the SMW will have the file systems listed in this table in the Mount Point column. The third column shows the recommended LUN size for each file system assuming a standard 4.5 TB RAID. For sites with storage constraints or extra storage, the fourth and fifth columns show suggested LUN sizes.

**IMPORTANT:** The volume for the `/var/opt/cray/imps` file system on the SMW should be significantly larger than the volume for the `/var/opt/cray/imps` file system on the boot node. This is because that file system on the SMW contains boot images, config sets, and image roots, while that file system on the boot node contains only a subset of the image roots on the SMW. The boot node does an NFS mount of the SMW boot images, so no local space is needed for those.

Table 11. SMW RAID Requirements

Mount Point	FS Type	Recommended Size for 4.5 TB RAID	Minimum Size	Size for 9.0 TB RAID	Description
/home	XFS	200 GB	40 GB	200 GB	Home directories on SMW
/var/lib/mysql	Btrfs	10 GB	10 GB	10 GB	HSS database
/var/opt/cray/disk/1	XFS	1000 GB	400 GB	2000 GB	logs, debug, dumps
/var/opt/cray/imps	XFS	1000 GB	400 GB	1000 GB	IMPS data
/var/opt/cray/repos	Btrfs	200 GB	100 GB	200 GB	IMPS repos

### CLE File Systems

On the boot RAID, storage for the boot node and SDB node is defined in the CLE storage set. Within that storage set, storage for the boot node is in the boot node LVM volume group, and storage for the SDB node is in the SDB node LVM volume group. The file systems for those nodes are listed in the tables below in the Mount Point

column. The fourth column shows the recommended LUN size for each file system assuming a standard 4.5 TB RAID. For sites with storage constraints, the fifth column shows suggested LUN sizes.

Note that for partitioned systems, the requirements for LUN size apply to the boot node and SDB node in each partition.

**Expanding storage space.** The LUN sizes for the `/cray_home` and `/non_volatile` file systems may need to be adjusted depending on site usage of those file systems. For example, workload managers, DataWarp, and any node that needs permanent storage can store information in `/non_volatile`, so it may need to be larger than the suggested size. If size adjustment is not made at install time, it can be made later. See *XC™ Series System Administration Guide (S-2393)* for instructions on how to expand storage in a file system, volume, or volume group.

Table 12. Boot Node RAID Requirements

Owning Node	Mount Point	FS Type	Recommended Size for 4.5 TB RAID	Minimum Size	Description
boot	<code>/cray_home</code>	XFS	50 GB	50 GB	Home directories on CLE
boot	<code>/var/opt/cray/imps</code>	Btrfs	250 GB	250 GB	IMPS data for PE image roots and for netroot compute-large and login-large image roots
boot	<code>/non_volatile</code>	XFS	200 GB	50 GB	persistent data, including <code>/var</code> if necessary, for service nodes provided from boot node

Table 13. SDB Node RAID Requirements

Owning Node	Mount Point	FS Type	Recommended Size for 4.5 TB RAID	Minimum Size	Description
SDB	<code>/var/lib/mysql</code>	XFS	20 GB	10 GB	SDB database
SDB	<code>/alps_shared</code>	XFS	20 GB	10 GB	ALPS data
SDB	<code>/var/opt/cray/ncmd</code>	XFS	10 GB	10 GB	persistent data storage for ncmd

### 4.3.6 Bootstrap the SMW Installation

#### Prerequisites

The following information must be gathered before running the installer in bootstrap mode. To find the persistent devices names for these devices, see [Determine the Persistent Device Name for a LUN](#) on page 108. For typical file system sizes, see [RAID Disk Space Requirements](#) on page 111.

- Disk devices on the boot RAID that can be used for boot node persistent storage
- Disk devices on the boot RAID that can be used for SDB node persistent storage
- Disk devices on the boot RAID that can be used for SMW persistent storage
- Size of file systems to be created within LVM volumes within LVM volume groups

**NOTE:** Check that these file system sizes do not exceed the total size of the volume group containing them. Adjust file system sizes, if needed.

#### About this task

This procedure runs `SMWinstall` in bootstrap mode, which installs IMPS and Ansible on the SMW, along with some of the global configuration templates. The `SMWinstall` command also invokes the configurator to prepare the storage set configuration. The configurator initiates an interactive session to gather the necessary information, unless the storage configuration template is supplied as a command-line argument, in which case no interactive session is needed. This configuration can be updated later by running the configurator manually.

#### Procedure

1. Start the multipath daemon.

Start `multipathd` but do not enable it on the command line. The multipath service needs to be started so that it can display path information needed for some config set settings, but the multipath service must not be enabled at this point in the process. Because of a SLES bug involving multipath and swap, the multipath service must not be enabled before the first time the new snapshot is booted. After that snapshot is booted for the first time, the Ansible multipath play will enable and start `multipathd` and will create an `/etc/multipath.conf` file. This file will ensure that on subsequent restarts of multipath or reboots of the SMW, `multipathd` will do the right thing and ignore swap.

```
smw# systemctl start multipathd
```

2. Install in bootstrap mode.

- Method 1 (more common): Provide storage configuration information interactively.

```
smw# /media/SMW/SMWinstall --mode bootstrap
```

- Method 2 (less common): Provide storage configuration information using an existing storage configuration **template** (the `_config.yaml`, not the `_worksheet.yaml`). This method can be used only if the `cray_bootraid_config.yaml` file is already available—obtained from a previous installation of the same release (a reinstallation) or from a different, similarly configured SMW.

```
smw# /media/SMW/SMWinstall --mode bootstrap --storage-config \
/path/to/cray_bootraid_config.yaml
```

**Trouble?** If ERROR and WARNING messages appear shortly after running the installer with the `--storage-config` option, and they complain of template syntax and/or schema errors, first check to see if the right file was provided in the command line. It must be the template (a `_config.yaml` file, also known as the config file), NOT the worksheet (a `_worksheet.yaml` file).

If Method 1 used, continue to step 3 on page 114. If Method 2 used, skip to step 13 on page 119.

## ENABLE THE STORAGE SERVICE

3. Ensure that `cray_bootraid.enabled` is set to `true` to enable the storage service.

```
cray_bootraid.enabled
[<cr>=set 'true', <new value>, ?=help, @=less] $ <cr>
```

Configurator navigation tips:

- For context-sensitive command help, enter `?`.
- To add a single value, enter the data and press **Enter**.
- To add a list, enter `+`, enter each list item on its own line, press **Ctrl-d** when done entering list items, and then press **Enter** to set the list entries.
- To skip a setting, press the `>` key. Note that skipping an unconfigured setting leaves it unconfigured, which means the configurator will assign it the default value and will prompt for it again if invoked with the same command.
- To correct an error in a previous setting, press the `<` key to go back to the previous setting, correct it, then continue forward. Use `<` to back up several settings, if needed.

## CONFIGURE THE CLE DEFAULT STORAGE SET (`cledefault`) VOLUME GROUPS

The configurator now shows the settings for a `storage_set` entry named `cledefault`, which contains three `volume_groups` entries:

- `boot_node_vg`
- `sdb_node_vg`
- `compute_node_local`

The full name of settings within each volume group looks like

`cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.` followed by `<volume group name>.<field name>`. For brevity, the next steps show only `<volume group name>.<field name>` for each setting.

4. Configure the owner and devices of the boot node volume group (`boot_node_vg`).
  - a. Set the owner of the boot node volume group.
 

Ensure that `boot_node_vg.owner` is set to "boot" rather than a `cname`. For a partitioned system, include the partition name (e.g., "boot-p2" for partition p2).
  - b. Add entries for the physical volumes (disk devices) that are going to be part of the boot node LVM volume group.
 

Use persistent device names such as `/dev/disk/by-id/scsi-360080e50002f7160000014905640c0c4` for each physical volume. Do not use short names like `/dev/sdn`, which may vary from node to node (SMW, boot node, and SDB



node), and may vary from boot to boot of a particular node. To find the persistent device name, see [Determine the Persistent Device Name for a LUN](#) on page 108.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.devices
[<cr>=set 0 entries, +=add an entry, ?=help, @=less] $ +
```

When done adding devices to the list, remember to press **Enter** to set the list entries.

5. Configure the owner and devices of the SDB node volume group (`sdb_node_vg.owner`).

a. Set the owner of the SDB node volume group.

Ensure that `sdb_node_vg.owner` is set to "sdb" rather than a cname. For a partitioned system, include the partition name (e.g., "sdb-p2" for partition p2).

b. Add entries for the physical volumes (disk devices) that are going to be part of the SDB node LVM volume group.

This setting is a list. To add list data, enter **+** at the prompt for `sdb_node_vg.devices` to enter list entry mode. Add persistent device names such

as `/dev/disk/by-id/scsi-360080e50002f7160000014925640c108` for each physical volume. Do not use short names like `/dev/sdn`, which may vary from node to node (SMW, boot node, and SDB node), and may vary from boot to boot of a particular node.

Press **Enter** after each list entry, and when done adding entries, press **Ctrl-d** to exit list entry mode. Remember to press **Enter** again to set the list entries.

6. (Only for systems using compute nodes with SSDs) Change the compute node volume group (`compute_node_local`), as needed.

This is the third of three predefined volume groups in the `cledefault` storage set. It is needed only for systems using compute nodes with on-board SSDs. For all other systems, skip this step and go to step 7 on page 115.

a. Set the owner of the compute node volume group.

Set `compute_node_local.owner` to "compute."

b. Add entries for the physical volumes (disk devices) that are going to be part of the compute node LVM volume group.

This setting is a list. To add list data, enter **+** at the prompt for `compute_node_local.devices` to enter list entry mode. Add this entry to the `compute_node_local.devices` list:

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.devices:
[<cr>=set 0 entries, +=add an entry, ?=help, @=less] $ +
Add devices (Ctrl-d to exit) $ 'select: nvme0n1'
```

Press **Enter** after each list entry, and when done adding entries, press **Ctrl-d** to exit list entry mode. Remember to press **Enter** again to set the list entries.

c. When done with the last volume, press **Enter** to set the `compute_node_local` "volumes" entries, or add another volume, as needed for this site.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes
[<cr>=set 3 entries, +=add an entry, ?=help, @=less] $ <cr>
```

After completing this step, skip the next step.

7. (Only for systems NOT using compute nodes with SSDs) Set the compute node volume group (`compute_node_local`) owner setting (no other settings).

Perform this step only if this system does not use compute nodes with on-board SSDs. Set the owner field to null to ensure that this volume group is not used but is preserved in case this site decides to add compute nodes with SSDs to the system later.

Use the > key to skip the other `compute_node_local` settings when presented.

```
compute_node_local.owner: null
compute_node_local.devices: >
```

Set the `compute_node_local` "volumes" entries.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes
[<cr>=set 3 entries, +=add an entry, ?=help, @=less] $ <cr>
```

8. Change file system sizes for each volume in the boot node volume group and SDB node volume group in accordance with the recommended values in the Boot Node RAID Requirements table in [RAID Disk Space Requirements](#) on page 111.

- a. Select the boot node volume group volumes setting.

Enter **1c\*** to select the volumes setting of `boot_node_vg`, and then enter **\*** to see all volume entries.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups
[<cr>=set 3 entries, +=add an entry, ?=help, @=less] $ 1c*

cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.volumes
[<cr>=set 3 entries, +=add an entry, ?=help, @=less] $ *
```

- b. For each volume of the boot node volume group, change file system size in accordance with the recommended values in the Boot Node RAID Requirements table in [RAID Disk Space Requirements](#) on page 111.



**CAUTION:** Boot failure may occur if the file system sizes configured for the boot node volume group exceed the available space in that volume group.

At the following prompt, enter these values to get to the `fs_size` setting for each of the file systems.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.volumes
[<cr>=set 3 entries, +=add an entry, ?=help, @=less] $
```

To get to this setting	enter this at prompt
<code>boot_node_vg.volumes.home.fs_size</code> (corresponds to <code>/cray_home</code> )	<b>1c*</b>
<code>boot_node_vg.volumes.imps.fs_size</code> (corresponds to <code>/var/opt/cray/imps</code> )	<b>2c*</b>
<code>boot_node_vg.volumes.nvolatile.fs_size</code> (corresponds to <code>/non_volatile</code> )	<b>3c*</b>

Then at the prompt for that setting, enter a new file system size to change the value, if needed. Accept the current or newly entered value by pressing **Enter**.

- c. When done with the last volume, press **Enter** to set the `boot_node_vg` "volumes" entries.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.volumes
[<cr>=set 3 entries, +=add an entry, ?=help, @=less] $ <cr>
```

- d. Select the SDB node volume group volumes setting.

Enter **2c\*** to select the volumes setting of `sdb_node_vg`, and then enter **\*** to see all volume entries.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups
[<cr>=set 3 entries, +=add an entry, ?=help, @=less] $ 2c*

cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.sdb_node_vg.volumes
[<cr>=set 2 entries, +=add an entry, ?=help, @=less] $ *
```

- e. For each volume of the SDB node volume group, change file system size in accordance with the recommended values in the SDB Node RAID Requirements table in [RAID Disk Space Requirements](#) on page 111.

At the following prompt, enter these values to get to the `fs_size` setting for each of the file systems.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.sdb_node_vg.volumes
[<cr>=set 2 entries, +=add an entry, ?=help, @=less] $
```

To get to this setting	enter this at prompt
<code>sdb_node_vg.volumes.db.fs_size</code> (corresponds to <code>/var/lib/mysql</code> )	<b>1c*</b>
<code>sdb_node_vg.volumes.alps.fs_size</code> (corresponds to <code>/alps_shared</code> )	<b>2c*</b>
<code>sdb_node_vg.volumes.ncmd.fs_size</code> (corresponds to <code>/var/opt/cray/ncmd</code> )	<b>3c*</b>

Then at the prompt for that setting, enter a new file system size to change the value, if needed. Accept the current or newly entered value by pressing **Enter**.

- f. When done with the last volume, press **Enter** to set the `sdb_node_vg` "volumes" entries.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.sdb_node_vg.volumes
[<cr>=set 3 entries, +=add an entry, ?=help, @=less] $ <cr>
```

9. Set the `cledefault` "volume groups" entries.

Review the list of `cledefault` volume groups (enter **\*** to see the full list if not all volume groups are displayed), then at the prompt below, enter press **Enter** to set the entries.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups
[<cr>=set 3 entries, +=add an entry, ?=help, @=less] $ <cr>
```

#### CONFIGURE THE SMW DEFAULT STORAGE SET (`smwdefault`) VOLUME GROUPS

The configurator now shows the settings for a `storage_set` entry named `smwdefault`, within which is one `volume_groups` entry: `smw_node_vg`

The full name of settings within each volume group looks like

`cray_bootraid.settings.storage_sets.data.smwdefault.volume_groups.` followed by `<volume group name>.<field name>`. For brevity, the next steps show only the volume group name and field name of each setting.

10. Configure the SMW node volume group (`smw_node_vg`).

- a. Set the owner of the SMW node volume group to `smw`.

Ensure that `smw_node_vg.owner` is set to "smw" to maximize portability of the config set.

```
cray_bootraid.settings.storage_sets.data.smwdefault.volume_groups.smw_node_vg.owner
[<cr>=keep 'smw', <new value>, ?=help, @=less] $ <cr>
```

- b. Add entries for the physical volumes (disk devices) that are going to be part of the SMW node LVM volume group.

This setting is a list. To add list data, enter **+** at the prompt for `smw_node_vg.devices` to enter list entry mode. Add persistent device names such

as `/dev/disk/by-id/scsi-360080e50002f889c00000a0654e32232` for each physical volume. Do not use short names like `/dev/sdn`, which may vary from node to node (SMW, boot node, and SDB node), and may vary from boot to boot of a particular node.

Press **Enter** after each list entry, and when done adding entries, press **Ctrl-d** to exit list entry mode. Remember to press **Enter** again to set the list entries.

- c. For each volume of the SMW node volume group, change file system size in accordance with the recommended values in the SMW RAID Requirements table in [RAID Disk Space Requirements](#) on page 111.

The total of all file system sizes configured in the SMW volume group **MUST NOT EXCEED** the available space in that volume group.

At the following prompt, enter these values to get to the `fs_size` setting for each of the file systems.

```
cray_bootraid.settings.storage_sets.data.smwdefault.volume_groups.smw_node_vg.volumes
[<cr>=set 5 entries, +=add an entry, ?=help, @=less] $
```

To get to this setting	enter this at prompt
<code>smw_node_vg.volumes.home.fs_size</code> (corresponds to <code>/home</code> )	<b>1c*</b>
<code>smw_node_vg.volumes.db.fs_size</code> (corresponds to <code>/var/lib/mysql</code> )	<b>2c*</b>
<code>smw_node_vg.volumes.log.fs_size</code> (corresponds to <code>/var/opt/cray/disk/1</code> )	<b>3c*</b>
<code>smw_node_vg.volumes.imps.fs_size</code> (corresponds to <code>/var/opt/cray/imps</code> )	<b>4c*</b>
<code>smw_node_vg.volumes.repos.fs_size</code> (corresponds to <code>/var/opt/cray/repos</code> )	<b>5c*</b>

Then at the prompt for that setting, enter a new file system size to change the value, if needed. Accept the current or newly entered value by pressing **Enter**.

- d. When done with the last volume, press **Enter** to set those `smw_node_vg` "volumes" entries.

```
cray_bootraid.settings.storage_sets.data.smwdefault.volume_groups.smw_node_vg.volumes
[<cr>=set 5 entries, +=add an entry, ?=help, @=less] $ <cr>
```

## 11. Set the `smwdefault` "volume groups" entries.

Review the list of `smwdefault` volume groups, then at the prompt below, enter press **Enter** to set the entries.

```
cray_bootraid.settings.storage_sets.data.smwdefault.volume_groups
[<cr>=set 1 entries, +=add an entry, ?=help, @=less] $ <cr>
```

## 12. Set the boot RAID "storage sets" entries.

Review the storage sets. Press **Enter** (<cr>) to set the `cledefault` and `smwdefault` storage sets, unless this system has partitions. If configuring a partitioned system, enter + to add another CLE storage set. A separate storage set is needed for each partition.

```
cray_bootraid.settings.storage_sets
[<cr>=set 2 entries, +=add an entry, ?=help, @=less] $
```

**Trouble?** If `SMWinstall` fails during the installation, it is because `cfgset` failed, which was invoked by `SMWinstall` to gather configuration information. That failure may be due to missing information. Do not try running `SMWinstall --mode bootstrap` again.

Try one of these options instead:

Option	Description
<b>Run the configurator manually</b>	<ol style="list-style-type: none"> <li>1. Enable the <code>cfgset</code> command.</li> </ol> <pre>smw# . /opt/modules/default/etc/modules.sh smw# module use /opt/cray/ari/modulefiles smw# module load imps</pre> <ol style="list-style-type: none"> <li>2. Use <code>cfgset</code> to invoke the configurator in interactive mode to make any needed changes to the <code>cray_bootraid</code> configuration service in the global config set.</li> </ol> <pre>smw# cfgset update -m interactive -s cray_bootraid global</pre> <ol style="list-style-type: none"> <li>3. Run the installer in bootstrap mode again.</li> </ol> <pre>smw# /media/SMW/SMWinstall --mode bootstrap</pre>
<b>Run <code>SMWinstall</code> with the reconfigure option</b>	<ol style="list-style-type: none"> <li>1. Run <code>SMWinstall</code> in bootstrap mode with the reconfigure option, which invokes the configurator in interactive mode.</li> </ol> <pre>smw# /media/SMW/SMWinstall --mode bootstrap --reconfigure</pre>

## 13. Display `cray_bootraid` information.

```
smw# . /opt/modules/default/etc/modules.sh
smw# module use /opt/cray/ari/modulefiles
smw# module load imps
smw# cfgset search -s cray_bootraid -l basic global
smw# cfgset search -s cray_bootraid -l advanced global
```

## 14. (SMW HA only) Copy the storage configuration template.

If this is the primary/first SMW installed of an SMW HA pair, save the storage configuration template to another system not on this SMW for fast and consistent system bootstrapping when installing the secondary SMW.

```
smw# scp -p /var/opt/cray/imps/config/sets/global/config/cray_bootraid_config.yaml user@host:~/
```

Note that it is the **template** (the `_config.yaml` file), not the worksheet (the `_worksheet.yaml` file) that must be copied.

**15. Remove existing volume groups, as needed.**

If doing a fresh install onto a system, and there is a desire to reuse the storage in any existing LVM volume groups for SMW, boot node, and SDB node, then run these commands to remove the volume groups with storage to be reused.

- a. Use `cfgset search` to find the names of all of the volume groups defined in the storage configuration template.

```
smw# cfgset search -s cray_bootraid global |awk -F'.' '{print $7}' |sort -u
boot_node_vg
boot_test_vg
sdb_node_vg
sdb_test_vg
smw_node_vg
smw_test_vg
```

- b. Display the volume groups that exist.

```
smw# vgdisplay
```

Alternative (more concise):

```
smw# vgs
```

- c. Remove the volume groups with storage to be reused (in this example, the test volume groups).

```
smw# vgremove -f smw_test_vg
smw# vgremove -f boot_test_vg
smw# vgremove -f sdb_test_vg
```

The system is now ready for the provisioning of boot RAID LVM volumes.

### 4.3.7 Provision SMW Storage

#### About this task

The provision-storage mode of `SMWinstall` can be run at any time. It uses the boot RAID configuration template (`cray_bootraid_config.yaml`) to provision persistent storage on the boot RAID by creating LVM volume groups and LVM volumes. Improperly configured boot RAID LUN assignments may be identified using this procedure. This is a non-interactive procedure if `--mode bootstrap` was used to bootstrap the installation earlier in the process. Otherwise, it will gather the necessary site-specific configuration information interactively.

#### Procedure

1. Provision storage for the default SMW storage set.

```
smw# /media/SMW/SMWinstall --mode=provision-storage
```

If no errors reported, proceed to step 2 on page 121.

**Trouble?** If errors are reported, review the boot RAID configuration settings using one of these methods. Both methods run the installer in provision-storage mode again after reviewing the settings and making changes. Note that when the installer is run again, it will ask ALL storage configuration questions, and the defaults will be prefilled with existing data.

- Error recovery method 1: Modify using the configurator, then run installer again.

```
smw# cfgset update -s cray_bootraid -m interactive global
```

```
smw# /media/SMW/SMWinstall --mode=provision-storage
```

- Error recovery method 2: Modify manually, then run installer again.

```
smw# vi \
/var/opt/cray/imps/config/sets/global/config/cray_bootraid_config.yaml
```

```
smw# /media/SMW/SMWinstall --mode=provision-storage
```

## 2. View the new volumes.

```
smw# lvs
  LV          VG          Attr          LSize     Pool Origin Data%  Meta%
Move Log Cpy%Sync Convert
db           smw_node_vg -wi-a----- 10.00g
home         smw_node_vg -wi-a----- 200.00g
imps         smw_node_vg -wi-a----- 1000.00g
log          smw_node_vg -wi-a----- 700.00g
repos        smw_node_vg -wi-a----- 200.00g
tmp_lv       system      -wi-ao----- 45.00g
var_crash_lv system      -wi-ao----- 128.00g
var_lib_named_lv system     -wi-ao----- 48.00m
var_log_lv   system     -wi-ao----- 10.00g
var_spool_lv system     -wi-ao----- 2.00g
var_tmp_lv   system     -wi-ao----- 5.00g
```

Note that any I/O errors in the output may be normal depending on whether multipathing is configured or not.

When the provision-storage installer mode completes successfully, the system is ready for the installation of SMW and CLE software.

## 4.3.8 Run the Installer for an Initial Installation

### Prerequisites

ISOS for SLES 12 and CentOS have been downloaded and SMW storage has been successfully configured. If installing on a Dell R640, an additional software patch `CLE_6.0.UP07.PS04.iso` is required. The patch is available for download at CrayPort (<https://crayport.cray.com>). Place the R640 patch in `/root/isos` before running the installer.

### About this task

This procedure installs SMW and CLE software together to ensure that there is a matched set of software and configuration.

**NOTE:** Do NOT run the installer from the `/root/isos` directory. Instead, run it from a directory that is not included in any snapshot, such as `/var/adm/cray/release`.

## Procedure

### 1. Set a variable for the release snapshot name.

Setting a variable here enables better command substitution in later commands dealing with snapshots.

In the next step, the installer will create a snapshot with this name and install the release software into it. Then the SMW will boot from this snapshot.

```
smw# ls -l1st /root/isos
smw# export SNAPSHOT=SMW-8.0UP07_CLE-6.0UP07.${TODAY}
smw# echo $SNAPSHOT
```



**CAUTION:** (SMW HA only) Use this snapshot name for both SMWs of an SMW HA pair. On the second SMW, the same release software must be installed into a target snapshot with exactly the same name (including date, even if installed on a different day). If the snapshots are not identical in content and name, inconsistencies in the HSS database (MySQL) can result, which causes problems at failover.

### 2. Install SMW and CLE software and security updates together.

Installing SMW, SLE, and CLE media with a single command creates a unified "release" that is tagged as a snapshot on the SMW system. Running the `SMWinstall` program creates the "target" snapshot, which was named in step 1, and then installs into that target snapshot (note that in the absence of an existing target snapshot, the installer creates one from the current running snapshot by default). The installer assumes that all of the SLES 12 ISOs are in `/root/isos`.

**IMPORTANT:** The SLE media must be specified before the CLE media on the command line so that SUSE security updates are installed before the CLE software is installed.

**For Dell  
R630 or  
R815**

```
smw# cd /var/adm/cray/release
smw# /media/SMW/SMWinstall \
--plus-media=${SLE_UPDATE} --plus-media=${CLE_RELEASE} \
--target=${SNAPSHOT}
```

**For Dell  
R640**

For Dell R640 SMWs, ensure the `CLE_6.0.UP07.PS04.iso` patch is in `/root/isos` and follow the installation example below:

```
smw# cd /var/adm/cray/release
smw# /media/SMW/SMWinstall \
--plus-media=${SLE_UPDATE} --plus-media=${CLE_RELEASE} \
--plus-media=/root/isos/CLE_6.0.UP07.PS04.iso \
--target=${SNAPSHOT}
```

It can take from 10 to 25 minutes to run a combined installation of SMW, CLE, and security updates for the first time on the SMW. The output of `SMWinstall` provides several command hints, including these three:

**snaputil  
default**

The `snaputil default` command hint will be used to ensure that the SMW is booted from the correct (new) snapshot, which is essential to a successful reboot.

**snaputil  
chroot**

The `snaputil chroot` command hint is used in the software update process and may be used at other times to look around inside the snapshot.

**snaputil  
delete**

The `snaputil delete` command hint should be used to remove the newly created snapshot if the install command fails and needs to be run again. It can also be used at other times to remove snapshots that are no longer needed.



Logs will be in `/var/adm/cray/logs/install` for each invocation of `SMWinstall`.

### Trouble?

- If the installer command fails with the message "No such file or directory," `cd` to `/root` and then repeat this step.
- If the installer fails, use `snaputil delete` to remove the newly created snapshot and then repeat this step.

### 3. Check new snapshot software versions.

When `SMWinstall` completes, check the snapshot details for the expected SMW and CLE release versions (note that the actual output on this system may show different release versions than this example output).

```
smw# /boot/install-support/default/snaputil show ${SNAPSHOT}
active_maps      :
  p0:/var/opt/cray/imps/config/sets/global/nims/maps/cle6.0.7128
boot menu       : False
booted          : False
btrfs_object_id : 514
cle_version      : 6.0.7128
created         : 2018-07-10 12:16:36 -0600
default         : False
initrd          : initrd-4.4.92-6.18-default
kernel          : vmlinuz-4.4.92-6.18-default
kernels (avail) :
  vmlinuz-4.4.73-5-default
  vmlinuz-4.4.82-6.9-default
  vmlinuz-4.4.92-6.18-default
name            : SMW-8.0.7128-6.0.7128.20180710.save2.postboot
parent          : SMW-8.0.7128-6.0.7128.20180710
path            : /media/root-sv/snapshots/
SMW-8.0.7128-6.0.7128.20180710.save2.postboot
read-only       : False
smw_version     : 8.0.7128
smwha_version   : 12.0.7128
storage_set     : smwdefault
subvolumes      :
  /var/lib/mysql:SMW-8.0.7128-6.0.7128.20180710.save2.postboot
  /var/opt/cray/repos:SMW-8.0.7128-6.0.7128.20180710.save2.postboot
total size      : n/a
unshared size   : n/a
updated         : 2018-07-10 07:53:24.109109
```

The SMW is now ready to reboot, which starts with setting the default snapshot to boot from. Trying to boot the SMW without first setting the default snapshot will result in an unbootable SMW.

## 4.3.9 Set Default Snapshot and Boot the SMW

### Prerequisites

This procedure assumes that the snapshot variable has been set and the SMW and CLE software has been installed.

## About this task

When the `SMWinstall` command was invoked in the previous procedure, it provided several suggested `snaputil` commands. The one used in this procedure ensures that the snapshot target is set as the default snapshot for the next boot of the SMW.

## Procedure

1. Set the release snapshot as the default.

**IMPORTANT:** Do not skip this step. If the SMW is rebooted without first setting the default snapshot, the SMW becomes unbootable.

```
smw# /media/SMW/snaputil default ${SNAPSHOT}
```

2. Verify that the correct snapshot is the default.

```
smw# /media/SMW/snaputil list
```

3. Exit the typescript file prior to rebooting the SMW.

```
smw# exit
```

4. Reboot the SMW to switch to the new release.

```
smw# reboot
```

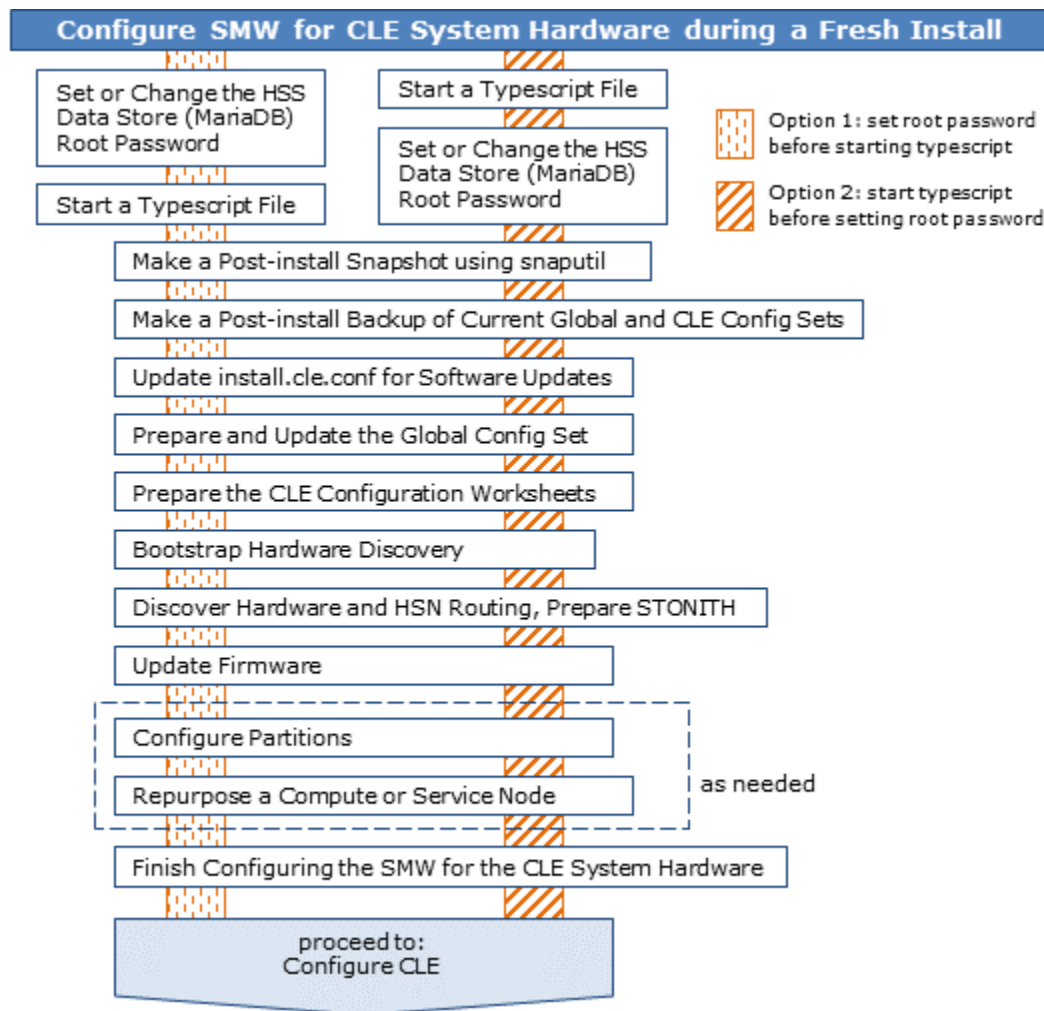
## 4.4 Configure SMW for CLE System Hardware during a Fresh Install

To create the global config set, initialize the power management database, discover hardware, and check the status of all SMW components, use the following procedures in the order shown in option 1 or option 2 of the following figure.

**option 1** Set the root password for the HSS database before starting a typescript file so that root password information is not captured in the typescript.

**option 2** Start the typescript file before doing setting the root password for the HSS database. If choosing this option, consider changing the permissions on the typescript file so that only root users can access it.

Figure 31. Visual Guide to Configuring the SMW for CLE System Hardware during a Fresh Install



Use [Installation Checklist 3: Configure SMW for CLE Hardware during a Fresh Install](#) on page 442 to track progress through this part of the fresh install process.

#### 4.4.1 Set or Change the HSS Data Store (MariaDB) Root Password

##### About this task

The method for setting or changing the HSS data store (database) root password has changed with the release of CLE 6.0. By default, MariaDB is installed with no password set up for the root account. Cray strongly recommends adding a password as part of the fresh install procedure.

**Old** The HSS database was implemented with MySQL. After initial installation, its root password was changed from the initial default empty string to a user-defined value by the `SMWconfig` script, which was run after `SMWinstall` and the initial discovery of the system.

**New** The HSS database is now implemented with MariaDB, a MySQL work-alike database with identically named commands. As before, the initial default root password is the empty string; however, the

SMWconfig script is no longer used to set it after installation. The administrator must use the following procedure to set the root password to a user-defined value.

After the MariaDB root password has been set, it must be placed in `/root/.my.cnf`, a file readable only by root that has the format shown in step 2. This file is the mechanism by which the installer and the `snaputil` command obtain the root password when they access MariaDB as root. If the file does not exist or it has no `password=` line, the system will attempt to access MariaDB using the default empty-string password, which will fail once the password has been changed.

- Create `/root/.my.cnf` the first time the root password is set to a user-defined value.
- Update `/root/.my.cnf` to match the MariaDB root password whenever it is changed.

**IMPORTANT:** For an SMW HA system, record the new MySQL root password. It will need to be changed on the second SMW later (by editing `/root/.my.cnf`). After the SMW HA cluster has been configured, the MySQL root password needs to be reset with `mysqladmin` on only one SMW, because the MySQL database is shared between both SMWs in the HA cluster.

## Procedure

1. Set or change the MariaDB root password.

```
smw# mysqladmin -uroot password -p
```

- a. Enter existing password.

At the "Enter password" prompt, do ONE of the following:

- If **setting** the root password for the first time (fresh install, migration, database reinitialization), the existing password is an empty string (the default initial password), so just press **Enter**.

```
Enter password: <cr>
```

- If **changing** the root password, enter the existing password.

```
Enter password: existing_password
```

- b. Enter and confirm the new password.

At these prompts, enter the new root password, and then enter it again.

```
New password:
Confirm new password:
```

2. Ensure that the root password in the `/root/.my.cnf` file matches the new root password.

If this file does not yet exist, create it and add the lines shown in the example, substituting the new password for the placeholder `<MariaDB-password>`.

```
smw# vi /root/.my.cnf
[client]
user=root
password=<MariaDB-password>
```

3. Ensure that only root can see or write to the `/root/.my.cnf` file.

```
smw# chmod 600 /root/.my.cnf
```

## 4.4.2 Start a Typescript File

### About this task

Sites can make as few or as many typescripts as they deem useful. Cray recommends starting a typescript file at these milestones:

- just before installing a new software release
- just before configuring the newly installed software

### Procedure

1. Log in as root to the SMW.
2. (First time only) Create a release directory for the typescript file.

```
smw# mkdir -p /var/adm/cray/release
```

3. Change to the release directory.

```
smw# cd /var/adm/cray/release
```

4. Set a variable equal to today's date.

```
smw# export TODAY=`date +%Y%m%d`  
smw# echo ${TODAY}
```

5. Start a typescript file.

```
smw# script -af ${TODAY}.suffix
```

For *suffix*, substitute a unique string to distinguish among typescript files, such as `install.1` or `update.2`.

6. Change prompt to include a timestamp.

```
smw# PS1="\[\e[1;31m\]\u@\h:\w \t # \[\e[0m\]\[\e[00m\]"
```

## 4.4.3 Make a Post-install Snapshot using snaputil

### About this task

This procedure uses `snaputil` to make an archival snapshot of the system after installing a new software release (fresh install or software update) and before configuring that software.

**Best Practice.** Make a snapshot and back up config sets at the same time to keep them in sync. Name the snapshot and config set backup using the same suffix and date/time stamp to help administrators identify which snapshot and config set backup pairs belong together.

For more information, see [About Snapshots and Config Set Backups](#) on page 21.

## Procedure

1. List the available snapshots on the system.

```
smw# snaputil list
```

2. Set the `SNAPSHOT` environment variable using the currently booted snapshot name.

```
smw# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')
smw# echo $SNAPSHOT
```

Setting a variable for the snapshot name enables better command substitution in later commands dealing with snapshots.

This is especially important for SMW HA systems because it makes it easier to use the exact same snapshot name for both SMWs. Using different snapshots results in HSS database (MySQL) inconsistencies, which causes problems at failover.

3. Create a new snapshot.

```
smw# snaputil create ${SNAPSHOT}.postinstall-${TODAY}
```

### 4.4.4 Make a Post-install Backup of Current Global and CLE Config Sets

#### About this task

This procedure uses the `cfgset` command to create a post-install backup of the global and CLE config sets after installing a new software release (fresh install or software update) and before configuring that software.

## Procedure

1. Back up the current global config set.

```
smw# cfgset create --clone global global-postinstall-${TODAY}
```

2. (Software update only) Back up the current CLE config set.

SKIP THIS STEP if this is a fresh install, because the CLE config set has not yet been created at this point in the process.

This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-postinstall-${TODAY}
```

### 4.4.5 Update `install.cle.conf` for Software Updates

#### Prerequisites

The installer will not be run again at this point in the installation and configuration process.

## About this task

The `/var/adm/cray/install.cle.conf` file contains configuration that controls the image building behavior of the installer. Changing this file now will make later updates of CLE software easier.

## Procedure

1. Edit the configuration file.

```
smw# vi /var/adm/cray/install.cle.conf
```

2. (For all systems) Ensure that `build_images` is set to `yes` to enable the SMW/CLE installer to build IMPS images as part of the install process. The remaining options determine what to do if `build_images` is set to `yes`.

```
build_images: yes
```

3. (For all systems) If parallel image building during installation is desired, set the value of `build_images_processes` to a number greater than 1 and less than the number of CPUs on the SMW.

```
build_images_processes: 8
```

For more information about building images in parallel, see [About Parallel Image Creation](#) on page 40.

4. (For partitioned systems only) Uncomment the `map_partition` line and specify the system partitions.

```
map_partition: ['p1', 'p2']
```

## 4.4.6 Prepare and Update the Global Config Set

### Prerequisites

This procedure assumes that the SMW and CLE software has been installed so that the global config set is present.

### About this task

The global config set must be updated with site-specific information about several services. This procedure describes how to add site configuration data to the configuration worksheets for each service in the global config set, update the config set with the edited configuration worksheets, and then run Ansible plays on the SMW to effect the changes there. The final steps check for external NTP servers and place the SMW time zone setting where cabinet and blade controllers can access it.

Notes on editing a configuration worksheet:

- Uncomment all settings that are marked `level=basic` and modify values as needed. All settings that remain commented are considered unconfigured.
- Settings that are already uncommented in the original worksheet are preconfigured to ensure proper configuration of the system; Cray recommends not modifying those preconfigured settings.

- Leave commented all settings that are marked level=advanced unless a default value needs to be modified. Leaving them commented (unconfigured) allows the configurator to safely update defaults that may change in later releases.
- To enter a value for a string that currently is set to ' ' (empty string), replace the quotes with the new value. For example, `ipv4_network: ' '` becomes `ipv4_network: 10.1.0.0`. In cases where the string value might be interpreted as a number, retain the single quotes. For example, a string setting with value '512' needs quotes.
- To enter one or more values for a list that is currently set to `[]` (empty list), remove the brackets and add each entry on a separate line, preceded by a hyphen and a space (`-` ). For example, a list with multiple entries would look like this:

```
cray_global_net.settings.networks.data.management.dns_servers:
- 172.31.84.40
- 172.30.84.40
```

- Do NOT change or remove the null value in lines like this that appear at the beginning of each multival entry (such as the network, host, and host interface entries in `cray_net_worksheet.yaml`). This line sets the key, or identifier, for that multival entry. In this example, `hsn` is the identifier for the HSN network entry.

```
cray_net.settings.networks.data.name.hsn: null
```

For more information about editing configuration worksheets and updating config sets, see *XC™ Series Configurator User Guide (S-2560)*.

**NOTE:** (SMW HA only) For SMW HA systems, the following procedures are done only on the first SMW because the config sets are shared between both SMWs in the HA cluster. In contrast, Ansible plays must be run on each SMW.

## Procedure

1. Save a copy of original global worksheets.

Copy the original configuration worksheets into a new directory to preserve them in case they are needed later for comparison.

```
smw# ls -l /var/opt/cray/imps/config/sets/global/worksheets

smw# cp -a /var/opt/cray/imps/config/sets/global/worksheets \
/var/opt/cray/imps/config/sets/global/worksheets.orig
```

2. Make a work area for global worksheets.

- a. Copy the global configuration worksheets to a new work area for editing.

The worksheets should not be edited in their original location for two reasons: (1) the configurator will not permit updating a config set from worksheets within that config set, and (2) edits would be overwritten when the config set is updated.

```
smw# mkdir -p /var/adm/cray/release
smw# cp -a /var/opt/cray/imps/config/sets/global/worksheets \
/var/adm/cray/release/global_worksheet_workarea
```

- b. Change to the work area directory to simplify the editing commands in the following steps.

```
smw# cd /var/adm/cray/release/global_worksheet_workarea
```



## UPDATE WORKSHEETS FOR GLOBAL SERVICES

3. Update `cray_firewall`.

- a. Edit `cray_firewall_worksheet.yaml`.

```
smw# vi cray_firewall_worksheet.yaml
```

- b. Uncomment `cray_firewall.enabled` and set it to `true`.

4. Update `cray_global_local_users`.

- a. Edit `cray_global_local_users_worksheet.yaml`.

```
smw# vi cray_global_local_users_worksheet.yaml
```

- b. Uncomment all lines of the pre-populated `craylogreaders` group.

Cray provides a pre-populated group called `craylogreaders` that has more restricted permissions than the `crayadm` group. It is intended for use when specifying group ownership for log files and directories. This step configures the `roles` field of that group.

Uncomment both lines that are commented, and leave the default value for the `roles` list, `smw`.

```
# ** 'groups' DATA **

cray_global_local_users.settings.groups.data.name.craylogreaders: null
cray_global_local_users.settings.groups.data.craylogreaders.gid: '14902'
cray_global_local_users.settings.groups.data.craylogreaders.description:
    Default Cray log readers group
cray_global_local_users.settings.groups.data.craylogreaders.deleted: false
#cray_global_local_users.settings.groups.data.craylogreaders.roles:
#- smw
```

- c. (Optional) Create a new group, if needed.

Cray recommends using the pre-populated `craylogreaders` group to specify ownership of log files and directories in `cray_logging`, which is configured in a later step in this procedure. However, if this site wishes to create and use a custom group, copy these lines and place them in the section for additional group definitions, as shown in the following example. Replace `sample_key_a` in all lines with the name (key) for this new group, then replace the default values with site-specific values, as needed.

The `gid` field cannot remain an empty string. Replace `' '` with some unused GID. To see what GIDs are already in use on this SMW, look in `/etc/groups`.

```
# NOTE: Place additional 'groups' setting entries here, if desired.

cray_global_local_users.settings.groups.data.name.sample_key_a: null
cray_global_local_users.settings.groups.data.sample_key_a.gid: ''
cray_global_local_users.settings.groups.data.sample_key_a.description: ''
cray_global_local_users.settings.groups.data.sample_key_a.deleted: false
cray_global_local_users.settings.groups.data.sample_key_a.roles: []
```

5. Update `cray_global_net`.

- a. Edit `cray_global_net_worksheet.yaml`.

```
smw# vi cray_global_net_worksheet.yaml
```

- b. Uncomment `cray_global_net.enabled` and ensure that it is set to `true`.
- c. Search in the file for 'networks' DATA, then uncomment all of the lines below it that begin with `cray_global_net.settings.networks` so that those settings will be applied and marked as configured. They define four networks: admin, SMW failover, HSS, and management.

**NOTE:** Do NOT uncomment the similar lines under this heading, because they are examples only and are not configured for these four networks.

```
# ** EXAMPLE 'networks' VALUE (with current defaults) **
```

- d. Enter SMW-specific or site-specific values for these management network fields.

```
cray_global_net.settings.networks.data.management.ipv4_network:
cray_global_net.settings.networks.data.management.ipv4_netmask:
cray_global_net.settings.networks.data.management.ipv4_gateway:
cray_global_net.settings.networks.data.management.dns_servers:
cray_global_net.settings.networks.data.management.dns_search:
cray_global_net.settings.networks.data.management.ntp_servers:
```

Add values for the `dns_servers` and `dns_search` fields for the management network only, not to any other network. The DNS information to use for these fields was entered during the SLES12 installation, so those values can be found in `/etc/resolv.conf`.

**NOTE:** If this site does not use DNS search but does use DNS domain in `/etc/resolv.conf`, then adding a single entry to the `dns_search` setting is functionally equivalent to setting the DNS domain.

- e. Set the management network external firewall to true.

```
cray_global_net.settings.networks.data.management.fw_external: true
```

- f. Search in the file for 'hosts' DATA, then uncomment all of the lines that begin with `cray_global_net.settings.hosts` so that those settings will be applied and marked as configured. They define a host called "primary\_smw" and two interfaces for it: one that connects to the customer management network ("customer\_ethernet") and one that connects to admin nodes ("admin\_interface"), such as the boot and SDB nodes.

- g. Enter SMW-specific or site-specific values for these items.

There are many more fields defining the "primary\_smw" host and its interfaces than are included in this example. These four fields are shown because they are the most likely to need site customization. Sites may wish to change the values of other fields as well.

See the notes on editing worksheets at the beginning of this procedure for information about changing empty string and empty list values.

```
cray_global_net.settings.hosts.data.primary_smw.aliases:
cray_global_net.settings.hosts.data.primary_smw.hostid:
cray_global_net.settings.hosts.data.primary_smw.hostname:
cray_global_net.settings.hosts.data.primary_smw.interfaces.customer_ethernet.ipv4_address:
```

Note that if the customer Ethernet IP address changes, the output from the `hostid` command will be different. After changing the following Ethernet field

```
cray_global_net.settings.hosts.data.primary_smw.interfaces.customer_ethernet.ipv4_address
```

ensure that this field (the SMW host ID) is set to the output of the `hostid` command.

```
cray_global_net.settings.hosts.data.primary_smw.hostid
```

- h. Set the `unmanaged_interface` field of the `customer_ethernet` and `admin_interface` interface settings to `true`.

This applies to both stand-alone SMWs and SMW HA systems. In the case of an SMW that is or will be configured for an SMW HA system, this prevents Ansible from managing `eth0` and `eth3` before the SMW HA cluster has been configured.

```
cray_global_net.settings.hosts.data.primary_smw.interfaces.customer_ethernet.unmanaged_interface: true
...
cray_global_net.settings.hosts.data.primary_smw.interfaces.admin_interface.unmanaged_interface: true
```

- i. (SMW HA only) Add another host definition for the second SMW of an SMW HA pair.

If this system will be configured for SMW HA, and the host IDs of the two SMWs in the HA pair are different, then a host entry for the second SMW must be added.

Copy the entire `primary_smw` host entry, including both interfaces, and paste it below the following line in the worksheet.

```
# NOTE: Place additional 'hosts' setting entries here, if desired.
```

Modify the new host entry so that it is correct for the second SMW. The new entry should be identical to the `primary_smw` host entry, except for the following changes:

1. Replace `primary_smw` with `second_smw` in every line. This is the key for this new entry, and it must be unique. The following example shows the first few settings with the new key.

```
# NOTE: Place additional 'hosts' setting entries here, if desired.

cray_global_net.settings.hosts.data.common_name.second_smw: null
cray_global_net.settings.hosts.data.second_smw.description: ''
cray_global_net.settings.hosts.data.second_smw.roles:
- smw
...
```

2. Change the host ID.

```
cray_global_net.settings.hosts.data.second_smw.hostid: second SMW host ID
```

3. Change the host name.

```
cray_global_net.settings.hosts.data.second_smw.hostname: second SMW host name
```

4. Change the IPv4 address for the customer network (the `eth0` interface).

```
cray_global_net.settings.hosts.data.second_smw.interfaces.customer_ethernet.ipv4_address: IP address
```

- j. (Optional) Configure a virtual LAN (VLAN) interface, as needed.

This example shows the configuration fields needed to configure a VLAN interface with common name set to `vlan0`. With the `vlan_id` set to `42` and the `etherdevice` set to `eth0`, the interface name will be set automatically to `eth0.42` (`vlan_etherdevice.vlan_id`) if the `name` field is left empty (recommended). If this site chooses to leave `vlan_id` empty instead (NOT recommended), the `name` field must be set to a non-empty string.

```
cray_global_net.settings.hosts.data.primary_smw.interfaces.common_name.vlan0: null
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan0.name: ''
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan0.vlan_id: 42
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan0.vlan_etherdevice: eth0
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan0.ipv4_address: some_IP_address
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan0.startmode: auto
```

- k. (Optional) Configure a bonded interface, as needed.

This example shows the configuration fields needed to configure a bonded interface with common name set to **bond0** and interface name set also to **bond0**. There is no field for bonding master because it is set automatically when the **bonding\_slaves** list has at least one member.

```
cray_global_net.settings.hosts.data.primary_smw.interfaces.common_name.bond0: null
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.name: bond0
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.bonding_slaves:
- eth0
- eth2
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.bonding_module_opts:
  miimon=100 mode=active-backup
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.ipv4_address: some_IP_address
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.startmode: onboot
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.bootproto: static
```

#### I. (Optional) Configure a bonded VLAN, as needed.

A "bonded VLAN" is a bonded interface with two ethernet NICs as slaves and two or more VLAN interfaces with the bonded interface as their etherdevice. The VLAN interfaces are typically on different subnets. These examples show the configuration fields needed to configure the necessary interfaces.

Example bonded interface. Note that there is no field for bonding master because it is set automatically when the **bonding\_slaves** list has at least one member. Also note that the **ipv4\_address** field is the default empty string because the address will be set on the VLAN.

```
cray_global_net.settings.hosts.data.primary_smw.interfaces.common_name.bond0: null
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.name: bond0
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.bonding_slaves:
- eth0
- eth2
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.bonding_module_opts:
  miimon=100 mode=active-backup
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.ipv4_address: some_IP_address
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.startmode: onboot
cray_global_net.settings.hosts.data.primary_smw.interfaces.bond0.bootproto: static
```

Example VLAN interfaces:

```
cray_global_net.settings.hosts.data.primary_smw.interfaces.common_name.vlan0: null
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan0.name: ''
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan0.vlan_id: 42
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan0.vlan_etherdevice: bond0
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan0.ipv4_address: some_IP_address
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan0.startmode: auto
```

```
cray_global_net.settings.hosts.data.primary_smw.interfaces.common_name.vlan1: null
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan1.name: ''
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan1.vlan_id: 43
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan1.vlan_etherdevice: bond0
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan1.ipv4_address: some_IP_address
cray_global_net.settings.hosts.data.primary_smw.interfaces.vlan1.startmode: auto
```

## 6. Update `cray_global_sysenv`.

The `cray_global_sysenv` config service enables sites to make any `sysctl`, `systemd`, or `limit` changes needed on the SMW. It provides the same functionality and works the same way as its counterpart in the CLE config set, `cray_sysenv`. The only difference between them is that `cray_sysenv` is used for CLE nodes and uses node groups to specify the scope of any change, while `cray_global_sysenv` is used for the SMW and uses the 'scope' field (always set to 'smw') instead of node groups.

**IMPORTANT:** Changes to `sysctl` settings take effect as soon as `cray-ansible` is run. However, changes to `systemd` or `limits` settings made after a system has booted take effect only at the next boot.

"DefaultTasksMax" and "UserTasksMax" limits on the CLE system and the SMW were increased in CLE 6.0.UP04. System administrators do not need to take any action.

- a. Edit `cray_global_sysenv_worksheet.yaml`.

```
smw# vi cray_global_sysenv_worksheet.yaml
```

- b. Uncomment `cray_global_sysenv.enabled`, if it is commented out, and ensure that it is set to `true`.

## 7. Update `cray_ipforward`.

- a. Edit `cray_ipforward_worksheet.yaml`.

```
smw# vi cray_ipforward_worksheet.yaml
```

- b. Uncomment `cray_ipforward.enabled`, if it is commented out, and ensure that it is set to `true`.

## 8. Update `cray_liveupdates`.

- a. Edit `cray_liveupdates_worksheet.yaml`.

```
smw# vi cray_liveupdates_worksheet.yaml
```

- b. Uncomment `cray_liveupdates.enabled` and ensure that it is set to `true`.

## 9. Update `cray_logging`.

The `cray_logging` configuration service configures the Lightweight Log Manager (LLM).

- a. Edit `cray_logging_worksheet.yaml`.

```
smw# vi cray_logging_worksheet.yaml
```

- b. Uncomment `cray_logging.enabled` and ensure that it is set to `true`.
- c. Uncomment `cray_logging.settings.global_options.data.raid`. If the boot RAID has a non-standard IP address, change the value of this setting.
- d. Configure the `site_loghost` setting.

Uncomment the following setting. All logs are stored locally on the SMW, but if this site wishes to also forward logs from the SMW to a site log host (a separate host that is reachable from the SMW), replace the empty string with the name of the site log host.

```
***** START Service Setting: site_loghost *****
...
#cray_logging.settings.site_loghost.data.name: ''
```

If this system has a site log host, and any of the following settings needs a value different than the default, uncomment that setting and change the value. Unchanged settings should remain commented.

```
***** START Service Setting: site_loghost *****
...
#cray_logging.settings.site_loghost.data.ip_protocol: tcp
#cray_logging.settings.site_loghost.data.ip_port: 514
#cray_logging.settings.site_loghost.data.syslog_format: rfc5424
```

- e. Use the current default permissions for log files and directories, which are not backward compatible, or change permissions to maintain backward compatibility.

The current default permissions for log files and directories are different than in releases prior to CLE 6.0.UP06, and they enable use of the `craylogreaders` group (defined in `cray_global_local_users`) for log ownership.



**WARNING:** The current default permissions for log files and directories are not backward compatible with releases older than CLE 6.0.UP06. If current permissions are used, this system will no longer be able to boot and run the older CLE 6.0 releases.

Sites must choose whether to go forward with the current default permissions and possibly use a non-default group for log ownership, or change the permissions to maintain backward compatibility.

- **If backward compatibility is NOT an issue**, use the current default permissions (no action needed), and use one of the following ownership groups for log files and directories:
  - `crayadm`, the default group
  - (recommended) `craylogreaders`, a pre-populated group that has more restricted permissions than the `crayadm` group
  - custom group defined in `cray_global_local_users` (see step 4 on page 131)

To use `crayadm` (the default) as the ownership group, do nothing. To use `craylogreaders` or a custom group instead, uncomment both of the following settings and set them to the desired group (this example sets them to `craylogreaders`):

```
cray_logging.settings.dirs.data.group: craylogreaders
cray_logging.settings.logs.data.group: craylogreaders
```

Note that the value of `group` in the `dirs` setting must always be the same as the value of `group` in the `logs` setting.

- **If backward compatibility is necessary**, keep `crayadm` as the ownership group for log files and directories, and change the permissions for log files and directories to values compatible with previous releases.

To keep `crayadm` (the default) as the ownership group, do nothing. Change the permissions for log files and directories to backward compatible values by uncommenting both of the following settings and setting them to the values shown in this example:

```
cray_logging.settings.dirs.data.mode: '0775'
cray_logging.settings.logs.data.mode: '0644'
```

Note that the ownership group specified for log files and directories, whether the default or some other group, will be set as a secondary group for the `crayadm` and `postgres` users.

## 10. Update `cray_multipath`.

Multipath does NOT need to be fully cabled to be used. The multipath driver can handle using one path or many.

- a. Edit `cray_multipath_worksheet.yaml`.

```
smw# vi cray_multipath_worksheet.yaml
```

- b. Choose one of the following options, depending on whether this site intends to use multipath.

**NOTE:** (SMW HA only) Cray recommends configuring multipath before configuring and enabling HA. If HA is configured and enabled first, then additional precautions must be taken when enabling multipath, as documented in *XC™ Series SMW HA Installation Guide*.

**Will multipath be used?**

If no, then uncomment `cray_multipath.enabled` and ensure that it is set to `false`. There is nothing else to configure in this step; proceed to step 11 on page 138.

If yes, then uncomment `cray_multipath.enabled` and set it to `true`. Continue with the following substeps.

## c. Enter the list of multipath nodes.

Uncomment `cray_multipath.settings.multipath.data.node_list`, remove the `[]` (denotes empty list), and add a list of nodes (by cname or host ID) in this system that have multipath devices and need to have multipath configured.

This example shows a list of three nodes: an SMW with host ID `1eac4e0c`, a boot node with cname `c0-0c0s0n1`, and an SDB node with cname `c0-0c0s1n1`.

```
cray_multipath.settings.multipath.data.node_list:
- 1eac4e0c
- c0-0c0s0n1
- c0-0c0s1n1
```

**Boot/SDB node failover.** If configuring boot and/or SDB node failover, add both the primary and backup (failover) nodes to this list. This example shows a list of five nodes: an SMW with host ID `1eac4e0c`, a primary boot node with cname `c0-0c0s0n1`, a backup boot node with cname `c0-2c0s4n1`, a primary SDB node with cname `c0-0c0s1n1`, and a backup SDB node with cname `c0-4c0s3n1`.

```
cray_multipath.settings.multipath.data.node_list:
- 1eac4e0c
- c0-0c0s0n1
- c0-2c0s4n1
- c0-0c0s1n1
- c0-4c0s3n1
```

## d. Configure enabled devices.

Cray has provided a number of enabled devices with pre-populated data under `# **`

`'enabled_devices' DATA **`. These storage devices are the devices that will be whitelisted, which means they will be listed as exceptions to the blacklist. The settings for these devices have default values provided by the device vendors and do not need to be changed. If this site intends to configure a multipath device that does not appear in this group of enabled devices, contact a Cray representative for help.

## e. (Optional) Configure aliases for the multipath devices.

This is the equivalent of adding aliases to the multipaths section of the `multipath.conf` file. If no aliases are specified, this setting will show as unconfigured when the config set is updated, but this is not a problem. It can remain unconfigured and will not cause the config set to be invalid.

In the worksheet, copy the two lines below `# ** EXAMPLE 'aliases' VALUE` (with current defaults) `**` and paste them below `# NOTE: Place additional 'aliases' setting entries here, if desired`.

```
# ** EXAMPLE 'aliases' VALUE (with current defaults) **
#   cray_multipath.settings.aliases.data.wwid.sample_key_a: null    <-- setting a multival key
#   cray_multipath.settings.aliases.data.sample_key_a.alias: ''
#
```

Uncomment the lines, replace `sample_key_a` with the World Wide Identifier (WWID) of the device to be aliased (`60080e50002e203c00002a085551b2c8` in this example) in all lines, and remove the `<-- setting a multival key` text at the end of the first line (note that the null value is required; do not



remove or change it). Finally, add the alias for this device (smw\_node\_pv1 in this example). Repeat this substep for each device, as needed.

```
# NOTE: Place additional 'aliases' setting entries here, if desired.
cray_multipath.settings.aliases.data.wwid.60080e50002e203c00002a085551b2c8: null
cray_multipath.settings.aliases.data.60080e50002e203c00002a085551b2c8.alias: smw_node_pv1
#***** END Service Setting: aliases *****
```

Note that as of CLE 6.0.UP06, cray\_multipath supports WWIDs that are 33 characters long, which is needed for DAL Lustre configuration.

## 11. Skip cray\_network\_boot\_packages.

The cray\_network\_boot\_packages configuration service is enabled by default and has no variables that need to be changed.

## 12. Update cray\_time.

- a. Edit cray\_time\_worksheet.yaml.

```
smw# vi cray_time_worksheet.yaml
```

- b. Uncomment cray\_time.enabled, if it is commented out, and ensure that it is set to true.
- c. Uncomment cray\_time.settings.service.data.timezone and change its value, as needed.

There are many possible values for time zone, such as I.E., US/Central, US/Eastern, and EMEA/BST.

### UPLOAD WORKSHEETS AND UPDATE/VALIDATE GLOBAL CONFIG SET



#### **CAUTION: Boot failure possible if using cfgset under certain conditions.**

The `cfgset create` and `cfgset update` commands always call pre- and post-configuration scripts. Some of these scripts require HSS daemons and other CLE services to be running. This can cause problems under these conditions:

- If `xtdiscover` is running, `cfgset` may hang or produce incorrect data that can result in system boot failure.
- If `xtbounce` is in progress or if the SMW is not connected to XC hardware, `cfgset` will fail.

In these circumstances, use the `--no-scripts` option with `cfgset create` or `cfgset update` to avoid running the scripts. Because using that option results in an invalid config set, remember to run `cfgset update` without the `--no-scripts` option afterwards, when circumstances permit, to ensure that all pre- and post-configuration scripts are run.

## 13. Upload modified worksheets into global config set.

Note that the full filepath must be specified in this `cfgset` command, and it must be enclosed in single quotes (to prevent the shell trying to expand the file glob).

```
smw# cfgset update -w \
'/var/adm/cray/release/global_worksheet_workarea/*_worksheet.yaml' global
```

## 14. Update the global config set.

Using the configurator in interactive mode to update the global config set is a good way to check whether all required settings and basic settings have been configured for services that are enabled. If they have, then all enabled services will show OK status in the Service Configuration List Menu. If configuration of a basic setting was missed, then the menu will show how many unconfigured settings there are for each service. Set or change any settings from this menu, as needed.



Note that some basic settings can be left unconfigured, such as aliases for multipath devices, because configuring them is optional.

```
smw# cfgset update -m interactive global
```

When the configurator session completes, it displays a message indicating the file name of the changelog file for this configuration session. The changelog is written to a file in the `/var/opt/cray/imps/config/sets/global/changelog` directory.

#### 15. Validate the global config set.

Because this validation step occurs after making changes to the config set and before Ansible plays are run to apply those changes, changes such as the addition of a new group (e.g., `craylogreaders`) may be flagged as not yet existing on the system. This will result in a warning message if the "missing" group or other setting has been defined correctly in the config set. This will result in an error message if it has not been defined correctly. If an error is reported, use the information in the message and return to the previous step to update the config set and correct the problem.

```
smw# cfgset validate global
```

APPLY CONFIGURATION CHANGES ON THE SMW

#### 16. Run Ansible plays on the SMW.

After the global config set has been updated, reapply any Ansible plays that consume global config set data.

**NOTE:** (SMW HA only) Both SMWs require this command. The procedure to install and configure the second SMW includes this command.

```
smw# /etc/init.d/cray-ansible start
```

Logs from running Ansible plays, such as `cray-ansible`, are stored on the SMW in `/var/opt/cray/log/ansible`.

#### 17. Restart PostgreSQL on the SMW.

The group membership of the `postgres` user was changed in CLE 6.0.UP06 to allow it to read `/var/opt/cray/log/xtdiag`. That change is applied late in the `cray-ansible` run, after the PostgreSQL daemon starts. This step ensures that PostgreSQL is aware of the new permissions.

```
smw# systemctl restart postgresql
```

CHECK TIME SETTINGS

#### 18. Check for external NTP servers.

Check that external NTP servers have been set as desired in the global config set.

**NOTE:** (SMW HA only) Both SMWs require this command. The procedure to install and configure the second SMW includes this command.

```
smw# grep server /etc/ntp.conf
server ntpserver1 minpoll 4 iburst
server ntpserver2 minpoll 4 iburst
```

#### 19. Put the SMW time zone setting where the cabinet and blade controllers can access it.

This SMW time zone setting will be applied to the cabinet and blade controllers when they are rebooted later in the process.

**NOTE:** (SMW HA only) Both SMWs require this command. The procedure to install and configure the second SMW includes this command.

```
smw# cp -p /etc/localtime /opt/tftpboot/localtime
```

## 4.4.7 Prepare the CLE Configuration Worksheets

### About this task

The Cray XC system stores configuration information used to boot and customize the CLE system in the p0 config set, or if the system is partitioned, in config set p1 for partition p1 and config set p2 for partition p2, and so forth. This procedure prepares the CLE configuration worksheets, which are later edited to include site-specific configuration data.

**NOTE:** (SMW HA only) For SMW HA systems, config set operations need to be performed on only one SMW because the config sets are shared between both SMWs in the SMW HA pair.

### Procedure

1. Obtain configuration worksheets for CLE from one of these sources.

- Find them in the CLE 6.0.UP07 release directory available on CrayPort and extract them to `/var/opt/cray/imps/config/sets/p0_example/worksheets`.
- Generate them by creating a CLE config set using prepare mode and the no-scripts option.

```
smw# cfgset create -m prepare -t cle --no-scripts p0_example
```

2. Save a copy of original worksheets.

Make a copy of the original CLE configuration worksheets directory to preserve the worksheets in case they are needed for comparison later.

```
smw# ls -l /var/opt/cray/imps/config/sets/p0_example/worksheets
```

```
smw# cp -a /var/opt/cray/imps/config/sets/p0_example/worksheets \
/var/opt/cray/imps/config/sets/p0_example/worksheets.orig
```

3. Copy the CLE worksheets to a work area.

Make a copy of the CLE configuration worksheets directory outside the config set to be used as a work area for editing. The worksheets should not be edited in their original location for two reasons: (1) the configurator will not permit updating a config set from worksheets within that config set, and (2) edits would be overwritten when the config set is updated.

**REMEMBER:** For partitioned systems, each partition generally has its own config set and associated configuration worksheets. Copy the CLE configuration worksheets to a separate work area for each partition.

```
smw# mkdir -p /var/adm/cray/release
```

```
smw# cp -a /var/opt/cray/imps/config/sets/p0_example/worksheets \
```

```
/var/adm/cray/release/p0_worksheet_workarea
```

These worksheets can be edited while the installation/configuration process continues with discovering hardware, updating firmware, and other hardware or HSS software activities.

- To edit the worksheets, see [Update CLE Configuration Worksheets](#) on page 150, but do not proceed to the task that creates the new CLE config set from the worksheets until hardware discovery and associated procedures are complete.
- To continue with hardware discovery, proceed to [Bootstrap Hardware Discovery](#) on page 141.

## 4.4.8 Bootstrap Hardware Discovery

### Prerequisites

This procedure assumes that the following information has been gathered. Enter this information in response to system prompts when performing this procedure.

Information needed	Default value
maximum X cabinet size (columns)	There is no default value. Find the X and Y cabinet sizes and the network topology class from <a href="#">Site-dependent Configuration Values</a> in <a href="#">Configuration Values</a> on page 50.
maximum Y cabinet size (rows)	No default value. See above.
network topology class	0 or 2 for Cray XC Series liquid-cooled systems, 0 for Cray XC Series air-cooled systems (XC30-AC, XC40-AC)
boot node name	c0-0c0s0n1
sdb node name	c0-0c0s1n1

### About this task

This procedure uses the `xtdiscover --bootstrap` command to collect some basic information that will be used to bootstrap the hardware discovery process. If boot node failover or SDB node failover will be enabled, then when `xtdiscover` asks for the boot node or the SDB node, instead of entering a single node, enter a pair of nodes with a comma between them, for example `c0-0c0s0n1, c0-2c0s4n1`. For more detailed information, see the `xtdiscover(8)` man page.

**NOTE:** (SMW HA only) Hardware discovery is done only on the first SMW. Do not repeat hardware discovery on the second SMW.

### Trouble?

- If the `xtdiscover --bootstrap` command is unable to power up the cabinets, try running `xtdiscover --testconfig` and then run `xtdiscover --bootstrap` again.
- If a step in this procedure fails because of a hardware issue, such as a cabinet failing to power up, resolve that issue and then go back to the last successful step in the procedure and continue from there. Do not skip steps or continue out of order.

## Procedure

1. Run `xtdiscover` in bootstrap mode.

```
smw# xtdiscover --bootstrap
```

The system prompts the user to enter the information gathered as a prerequisite to this procedure. Prior to powering on the cabinets, the system prompts the user to disable any blades that should not be powered on.

```
xtdiscover is about to power on the cabinets.
*** IF YOU NEED TO DISABLE COMPONENTS TO AVOID THEM
*** BEING POWERED ON, PLEASE DO SO NOW USING 'xtcli disable'

Please enter 'c' to continue, or 'a' or 'q' to abort [c]:
```

2. Disable any blades that should not be powered on.

If there are any blades or other components to be disabled, open a separate window and disable them (as `crayadm`) there. In this command, replace `cname` with the `cname` of the component to be disabled.

```
crayadm@smw> xtcli disable cname
```

3. Return to the `xtdiscover --bootstrap` window and enter `c` to continue the hardware discovery bootstrap.

```
Please enter 'c' to continue, or 'a' or 'q' to abort [c]: c
```

The `xtdiscover` command proceeds without further prompts.

**Trouble?** If the `xtdiscover` command fails with the message, The following cabinets were not detected by heartbeat, power cycle the cabinet controller and retry the `xtdiscover --bootstrap` command.

When the `xtdiscover --bootstrap` command has successfully completed, continue to the next step.

4. Power down the system.

```
smw# xtcli power down s0
Turning off power to cabinet and waiting for confirmation...
```

Component	Flags:	Result
-----	-----	-----
c0-0	noflags :	Success
c0-0c0s0	noflags :	Success
c0-0c0s1	noflags :	Success
c0-0c0s2	noflags :	Success
c0-0c0s3	noflags :	Success

5. Reboot the cabinet controllers (CC), then verify that all CCs are up.

- a. Reboot the cabinet controllers.

```
smw# xtccreboot -c all
xtccreboot: reboot sent to specified CCs
smw# sleep 180
```

- b. Are all cabinet controllers up now? Repeat this command until all of the cabinet controllers report in.

```
smw# xtalive -a llsysd -l 11 s0
The expected responses were received.
```

## 6. Power up the system.

```
smw# xtcli power up s0
Turning on power to cabinet and waiting for confirmation...
```

Component	Flags:	Result
-----	-----	-----
c0-0	noflags :	Success
c0-0c0s0	noflags :	Success
c0-0c0s1	noflags :	Success
c0-0c0s2	noflags :	Success
c0-0c0s3	noflags :	Success

Note that at this point the `xtcli status` output shows that all nodes are "off" because they have not yet been bounced.

The bootstrap process is now complete. The next task is to discover the Cray system hardware.

## 4.4.9 Discover Hardware and HSN Routing, Prepare STONITH

### Prerequisites

This procedure assumes that the `xtdiscover --bootstrap` command has been run successfully.

### About this task

**About Hardware Discovery.** This procedure uses `xtdiscover` to detect the Cray system hardware components on the system. The `xtdiscover` command confirms some basic information (entered earlier with `xtdiscover --bootstrap`) for the hardware discovery process, warns that changes will be made, and then confirms whether to abort or continue. Finally, this command creates entries in the system database to describe the hardware. To display the configuration, use the `xtcli` command after running `xtdiscover`. For more detailed information, see the `xtdiscover(8)` man page.

**About STONITH.** This procedure prepares STONITH, a Linux service that automatically powers down a node that has failed or is suspected of failure. If either boot node failover or SDB node failover will be used, then STONITH needs to be set on the primary blade and backup blade.

**IMPORTANT:** The primary boot node should not be on the same blade as the backup boot node or an SDB node. Likewise, the primary SDB node should not be on the same blade as the backup SDB node or a boot node. Four different blades should be used if there are two boot nodes and two SDB nodes.

**Trouble?** If a step in this procedure fails because of a hardware issue, such as a cabinet failing to power up, resolve that issue and then go back to the last successful step in the procedure and continue from there. Do not skip steps or continue out of order.

### Procedure

#### DISCOVER CRAY SYSTEM HARDWARE

1. Log on to the SMW as `root`, if not already logged in.

## 2. Run the `xtdiscover` command.

`xtdiscover` may pause with instructions to bounce the system.

```
smw# xtdiscover
***** xtdiscover started *****

...
```

In a separate window, please bounce the system now to continue discovery.

## 3. If `xtdiscover` pauses with instructions to bounce the system, open a separate window and, as `crayadm`, run the command to bounce the system.

Note that some HSS timeouts have been increased to accommodate the longer bounce times of ARM blades. As a result, this command could run for up to 30 minutes without producing additional output.

```
crayadm@smw> /opt/cray/hss/default/etc/xtdiscover-bounce-cmd
```

## 4. If it was necessary to bounce the system, then when the `xtbounce` command from the previous step has finished, return to the `xtdiscover` window and enter "c" to continue the hardware discovery.

```
After bounce completes, enter 'c' to complete discovery
or 'q' or 'a' to abort [c]: c
```

## 5. Commit the results of `xtdiscover` to the database.

When asked whether to commit the `xtdiscover` results to the database, enter **y**.

(optional) PREPARE STONITH FOR BOOT NODE AND SDB NODE FAILOVER

## 6. For sites using boot node failover, set STONITH for the blade containing the primary boot node and the blade containing the backup boot node.

Skip this step if there will be no boot node failover for this system.

In the example, the primary boot node is `c0-0c0s0n1`, so its blade is `c0-0c0s0`, and the backup boot node is `c0-2c0s4n1`, so its blade is `c0-2c0s4`.

```
smw# xtdaemonconfig c0-0c0s0 stonith=true
smw# xtdaemonconfig c0-2c0s4 stonith=true
```

## 7. For sites using SDB failover, set STONITH for the blade containing the primary SDB node and the blade containing the backup SDB node.

Skip this step if there will be no SDB node failover for this system.

In the example, the primary SDB node is `c0-0c0s1n1`, so its blade is `c0-0c0s1`, and the backup SDB node is `c0-4c0s3n1`, so its blade is `c0-4c0s3`.

```
smw# xtdaemonconfig c0-0c0s1 stonith=true
smw# xtdaemonconfig c0-4c0s3 stonith=true
```

## DISCOVER HSN ROUTING CONFIGURATION

## 8. Discover the routing configuration of the high-speed network (HSN).

After `xtdiscover` finishes, run the `rtr` command as `crayadm` to determine the exact configuration of the HSN.

```
smw# su - crayadm
crayadm@smw> PS1="\u@\h:\w \t> "
crayadm@smw> rtr --discover
```

The `rtr` command may produce the following message and prompt. Answer "y" to allow `rtr` to bounce the system in diagnostic mode.

```
rtr:WARNING: No HSN discover info found, Using defaults (100% bandwidth
assumed)
System was not bounced in diagnostic mode, should I re-bounce? y
```

## 4.4.10 Update Firmware

### Prerequisites

This procedure assumes that Cray hardware discovery has been completed successfully.

### About this task

This procedure first checks whether the firmware of cabinet and blade components (controllers) needs to be updated, then updates the firmware only if there are revision mismatches. For the current list of cabinet and blade controllers, see the `xtzap` man page.

### Procedure

**NOTE:** These commands are performed from the `crayadm` account, as indicated by the command prompts.

#### 1. Check firmware.

Check whether any firmware needs to be updated on the various controllers.

```
crayadm@smw> xtzap -r -v s0
```

If the firmware on any controllers is out of date, the output looks like this, and the firmware needs to be updated (reflashed).

Individual Revision Mismatches:

Type	ID	Expected	Installed
cc_bios	c0-0	0013	0012
bc_bios	c0-0c0s0	0013	0012
bc_bios	c0-0c0s1	0013	0012
bc_bios	c0-0c0s2	0013	0012
bc_bios	c0-0c0s3	0013	0012

#### 2. Update firmware, if any components are not current.



**CAUTION:** The `xtzap` command is normally intended for use by Cray Service personnel only. Improper use of this restricted command can cause serious damage to the computer system.

Run `xtzap -a` to update all components.

```
crayadm@smw> xtzap -a s0
```

Note that it is possible to update firmware in cabinets or blades only rather than the entire system. For more information, see *XC™ Series System Administration Guide (S-2393)*.

3. Run `xtbounce --linktune` if any components were not current.

Force `xtbounce` to do a `linktune` on the full system before checking firmware again.

```
crayadm@smw> xtbounce --linktune=all s0
```

4. Confirm that all components with out-of-date firmware have been updated.

Check firmware again after updating and linktuning those components.

```
crayadm@smw> xtzap -r -v s0
```

## 4.4.11 (Optional) Configure Partitions

### About this task

This procedure describes how to divide the CLE system into "logical machines" or partitions. By definition, `p0` is the entire system, and `p1` through `p31` are smaller partitions. Each partition must have its own set of boot, sdb, and other service nodes and compute nodes to boot the partition. See the `xtcli_part(8)` man page for more details.

**NOTE:** (SMW HA only) For a partitioned SMW HA system, only the first SMW requires this procedure, because the hardware configuration is stored in a shared MariaDB (formerly MySQL) database.

To add a partition, specify the boot node, SDB node, and the components that will be members of the partition. As an example, the following steps show how to add these two partitions to an unpartitioned system (`p0`).

```
partition: p1
boot node: c0-0c0s0n1
sdb node: c0-0c0s1n1
members:
c0-0c0s0,c0-0c0s1,c0-0c0s4,c0-0c0s5,c0-0c0s6,c0-0c0s7,c0-0c0s8,c0-0c0s9,c0-0c0s10
,c0-0c0s11,c0-0c0s12,c0-0c0s15

partition: p2
boot node: c0-0c0s2n1
sdb node: c0-0c0s3n1
members: c0-0c0s2,c0-0c0s3,c0-0c0s13,c0-0c0s14
```

### Procedure

1. Deactivate `p0`.

```
crayadm@smw> xtcli part_cfg deactivate p0
```



## 2. Add a partition.

Note that `-b` identifies the boot node, `-d` identifies the SDB node, and `-m` identifies all members of the partition.

```
crayadm@smw> xtcli part_cfg add p1 -i /raw0 -b c0-0c0s0n1 -d c0-0c0s1n1 \
-m c0-0c0s0,c0-0c0s1,c0-0c0s4,c0-0c0s5,c0-0c0s6,c0-0c0s7,\
c0-0c0s8,c0-0c0s9,c0-0c0s10,c0-0c0s11,c0-0c0s12,c0-0c0s15
```

## 3. Activate the new partition.

```
crayadm@smw> xtcli part_cfg activate p1
```

## 4. Add and activate a second partition.

```
crayadm@smw> xtcli part_cfg add p2 -i /raw0 -b c0-0c0s2n1 -d c0-0c0s3n1 \
-m c0-0c0s2,c0-0c0s3,c0-0c0s13,c0-0c0s14

crayadm@smw> xtcli part_cfg activate p2
```

## 4.4.12 Repurpose a Compute or Service Node

### Prerequisites

This procedure assumes that a fresh install is in progress and that the XC system is not yet booted. Do not use this procedure if these assumptions are not true—instead, use the "Repurpose a Compute or Service Node" procedure under "Modify an Installed System" in *XC™ Series System Administration Guide (CLE 6.0.UP07)* S-2393, which has additional steps necessary for a booted system.

### About this task

When a compute node is configured for a non-compute role, that node is called a repurposed compute node (RCN). Compute nodes can be repurposed to become service nodes for use as tier2 servers (recommended) or in other capacities. However, compute nodes should not be repurposed as service nodes for services that require external connectivity.

This procedure provides steps for repurposing a compute node to become a service node. It can also be used to repurpose a service node as a compute node, to return an RCN to its original compute role, for example. Note that nodes on a service blade must always have the service role.

In the example commands shown, two compute nodes (c0-0c0s2n0 and c0-0c0s2n1) are repurposed as service nodes.

### Procedure

Mark the compute node as 'service' in the HSS database.

```
crayadm@smw> xtcli mark_node service c0-0c0s2n0,c0-0c0s2n1
```

If repurposing a service node as compute, use the following command instead.

```
crayadm@smw> xtcli mark_node compute c0-0c0s2n0,c0-0c0s2n1
```

**Important.** Using the `xtcli mark_node` command to change the node type in the HSS database is considered a hardware change. To keep everything in synch (HSS, NIMS, and IMPS), it must be followed by changing the NIMS parameters of the node, updating the CLE and global config sets, and rebooting the node.

For a fresh install, changing the NIMS parameters, updating config sets, and rebooting are done later in the process, so they are omitted from this procedure.

## 4.4.13 Finish Configuring the SMW for the CLE System Hardware

### Prerequisites

Cray hardware has been discovered and component firmware has been updated (if needed).

### About this task

This procedure contains the final steps of configuring the SMW for the CLE system hardware. Note that a full system is referred to as "s0" here. The term "p0" could have been used, because in this context, the two terms are interchangeable. In contrast, commands that operate on config sets use only the term "p0" when referring to a full system. In the config set context, the terms are not interchangeable.

Note that all of the commands in this procedure are run as `crayadm`.

### Procedure

1. Check status on all components.

```
crayadm@smw> xtcli status s0
```

2. Check routing configuration of the system.

```
crayadm@smw> rtr -R s0
```

Note that the `rtr -R` command produces no output unless there is a routing problem.

3. Examine the hardware inventory and verify that all nodes are visible to the SMW.

```
crayadm@smw> xthwinv s0 > xthwinv.out
```

```
crayadm@smw> xthwinv -x s0 > xthwinv.xml
```

4. Check microcontroller information.

Execute the `xtmcinfo -u` command to retrieve microcontroller information from cabinet control processors and blade control processors. Ensure that all blade controllers have output and show similar uptime values.

```
crayadm@smw> xtmcinfo -u s0
```

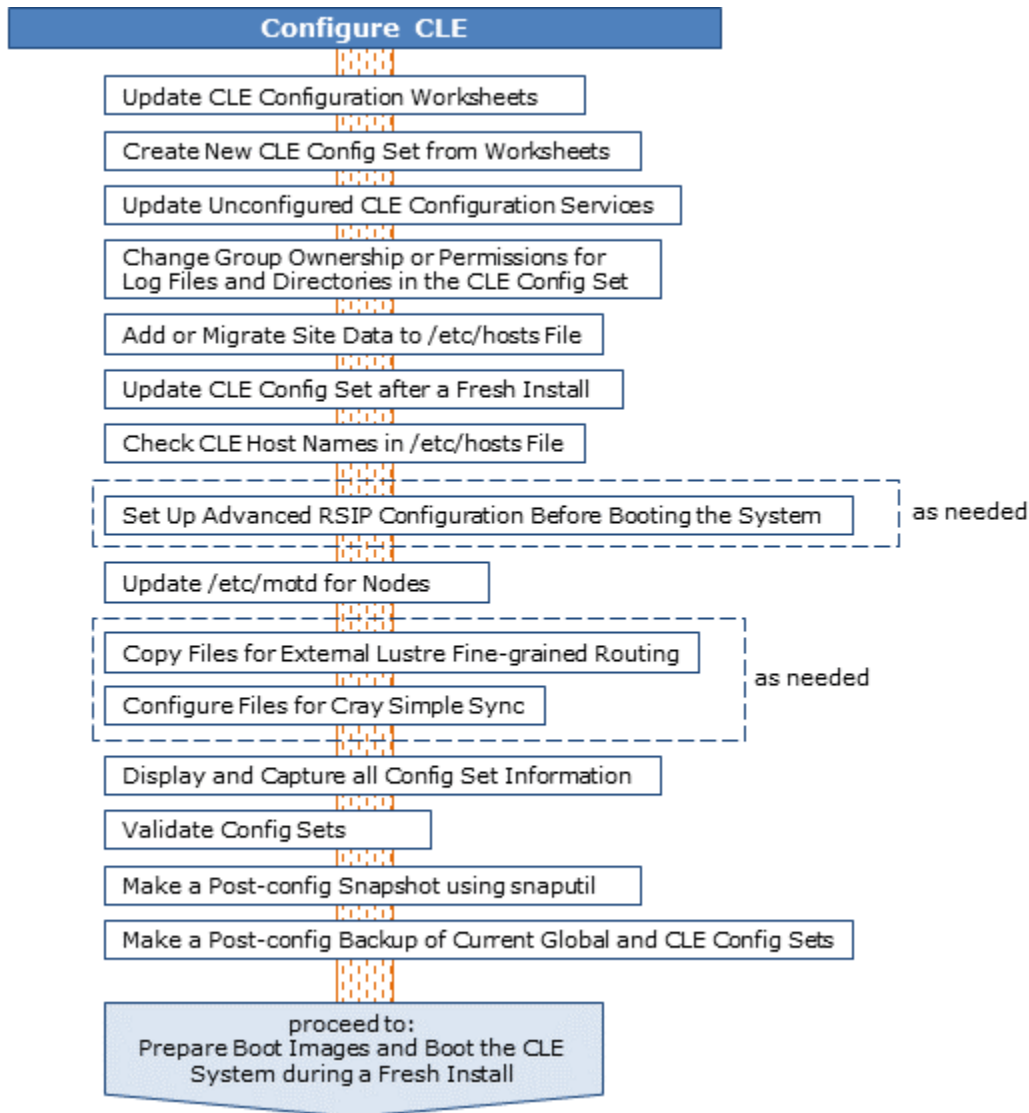
5. Exit from `crayadm` back to `root`.

```
crayadm@smw> exit  
smw#
```

## 4.5 Configure CLE

The Cray Linux Environment (CLE) config set stores configuration information used to boot and customize the CLE system. The CLE config set is usually named p0 for a full, unpartitioned system. For a partitioned system, a CLE config set must be created for each partition, with names p1, p2, and so forth. To create, update, validate, and back up a CLE config set, use the procedures shown in the visual guide.

Figure 32. Visual Guide to Configuring CLE



Use [Installation Checklist 4: Configure CLE](#) on page 443 to track progress through this part of the fresh install process.

**NOTE:** (SMW HA only) For SMW HA systems, the following procedures are done only on the first SMW because the config sets are shared between both SMWs in the HA cluster. In contrast, Ansible plays must be run on each SMW.

## 4.5.1 Update CLE Configuration Worksheets

### Prerequisites

This procedure assumes the following:

- SMW and CLE software has been installed.
- A set of CLE configuration worksheets reside in a work area ready to be edited with site-specific configuration information.

### About this task

The Cray XC system stores configuration information used to boot and customize the CLE system in the p0 config set, or if the system is partitioned, in config set p1 for partition p1 and config set p2 for partition p2, and so forth. Use the following procedures to edit selected CLE configuration worksheets and enter site-specific configuration data. Afterwards, all of the CLE configuration worksheets will be uploaded to the config set to create or update it.

Use [Installation Checklist 5: Update CLE Configuration Services](#) on page 444 to track progress updating the worksheets.

**REMEMBER:** For partitioned systems, each partition generally has its own config set and associated configuration worksheets. Assuming a work area directory was created for each partition, change to that directory and update worksheets there for each partition.

**NOTE:** (SMW HA only) For SMW HA systems, the following procedures are done only on the first SMW because the config sets are shared between both SMWs in the HA cluster. In contrast, Ansible plays must be run on each SMW.

### Procedure

1. Change to the work area directory to simplify the editing commands in the following procedures.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

2. Edit and update the configuration worksheets for selected CLE config services using the procedures that follow.

Edit only the worksheets associated with these procedures. The remaining worksheets should remain unchanged. Their associated configuration services will be updated in a different procedure after all worksheets are used to create a CLE config set.

**TIP:** Update `cray_node_groups_worksheet.yaml` and `cray_net_worksheet.yaml` first. Many configuration worksheets use node groups and network settings, and it will be much easier to update those worksheets if the necessary node groups and networks are already defined.

The procedures to update worksheets for the selected CLE config services are arranged alphabetically, except for the `cray_node_groups` and `cray_net` procedures, which have been placed before the others to encourage completing them first.

Notes on editing a configuration worksheet:

- Uncomment all settings that are marked `level=basic` and modify values as needed. All settings that remain commented are considered unconfigured.

- Settings that are already uncommented in the original worksheet are preconfigured to ensure proper configuration of the system; Cray recommends not modifying those preconfigured settings.
- Leave commented all settings that are marked level=advanced unless a default value needs to be modified. Leaving them commented (unconfigured) allows the configurator to safely update defaults that may change in later releases.
- To enter a value for a string that currently is set to ' ' (empty string), replace the quotes with the new value. For example, `ipv4_network: ' '` becomes `ipv4_network: 10.1.0.0`. In cases where the string value might be interpreted as a number, retain the single quotes. For example, a string setting with value '512' needs quotes.
- To enter one or more values for a list that is currently set to `[]` (empty list), remove the brackets and add each entry on a separate line, preceded by a hyphen and a space (`-` ). For example, a list with multiple entries would look like this:

```
cray_global_net.settings.networks.data.management.dns_servers:
- 172.31.84.40
- 172.30.84.40
```

- Do NOT change or remove the null value in lines like this that appear at the beginning of each multival entry (such as the network, host, and host interface entries in `cray_net_worksheet.yaml`). This line sets the key, or identifier, for that multival entry. In this example, `hsn` is the identifier for the HSN network entry.

```
cray_net.settings.networks.data.name.hsn: null
```

For more information about editing configuration worksheets and updating config sets, see *XC™ Series Configurator User Guide* (S-2560).

#### 4.5.1.1 Update `cray_node_groups` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

Node Groups are a mechanism for defining logical groupings of Cray system nodes to streamline node specifications for use in other Cray configuration services. The node groups defined are non-exclusive, that is, a node may belong to more than one node group. They are referenced in other configuration templates and are used in Ansible plays as well. For more information, see [About Node Groups](#) on page 23.

This procedure configures some basic settings in the Cray Node Groups service configuration worksheet to add site-specific data. For an explanation of the long variable names in configuration settings, see [About Variable Names in the Configurator and Configuration Worksheets](#) on page 20.

#### Notes about node groups:

- Architecture-specific platform keywords are used to create pre-populated node groups that contain all compute or service nodes with the x86-64 or AArch64 architecture.

- All platform keywords, such as `platform:compute`, `platform:service-ARM`, and `platform:compute-HW12`, include nodes that have been disabled.
- To identify disabled nodes, use the `platform:disabled` keyword.
- Groups of nodes can be excluded using a negation operator: `~` (the tilde symbol). All non-negated keywords are resolved first, then negated ones are removed (see example in the step for defining a custom node group).

## Procedure

1. Edit `cray_node_groups_worksheet.yaml`.

```
smw# vi cray_node_groups_worksheet.yaml
```

2. Uncomment `cray_node_groups.enabled` and ensure that it is set to `true`.
3. Customize pre-populated node groups.

These pre-populated node groups are provided by Cray, but sites must customize the `members` setting for many of the node groups. For example, the host ID of the SMW is `1eac199c` in the first node group, `smw_nodes`, but this must be replaced by the actual host ID for the SMW at this site. For more information about changing these settings, including the use of additional platform keywords for finer-grained groupings, see [About Node Groups](#) on page 23.

**Boot/SDB node failover.** If configuring boot node failover, add both the primary and the backup boot node cnames to the `boot_nodes.members` list. Similarly, if configuring SDB node failover, add both the primary and the backup SDB node cnames to the `sdb_nodes.members` list.

**Login nodes.** Note that beginning with CLE 6.0.UP06, Cray no longer supports a single node group for all login nodes. Instead, there are two architecture-specific login node groups: one for all login nodes with the x86-64 architecture and one for all login nodes with the AArch64 architecture. To specify all login nodes in the system, use both of those node groups.

**MOM nodes.** For MOM (machine-oriented miniserver) nodes that are not SDB nodes, Cray recommends booting with the login image so that they will have the same capabilities as a login node. In this case, add such MOM nodes to the appropriate architecture-specific login node group.

```
# ** 'groups' DATA **
cray_node_groups.settings.groups.data.group_name.compute_nodes: null
cray_node_groups.settings.groups.data.compute_nodes.description:
    Default node group which contains all of the compute nodes for
    the current partition.
cray_node_groups.settings.groups.data.compute_nodes.members:
- platform:compute

cray_node_groups.settings.groups.data.group_name.compute_nodes_aarch64: null
cray_node_groups.settings.groups.data.compute_nodes_aarch64.description:
    Default node group which contains all of the aarch64 compute nodes for
    the current partition.
cray_node_groups.settings.groups.data.compute_nodes_aarch64.members:
- platform:compute-ARM

cray_node_groups.settings.groups.data.group_name.compute_nodes_x86_64: null
cray_node_groups.settings.groups.data.compute_nodes_x86_64.description:
    Default node group which contains all of the x86_64 compute nodes for
    the current partition.
```

```

cray_node_groups.settings.groups.data.compute_nodes_x86_64.members:
- platform:compute-X86

cray_node_groups.settings.groups.data.group_name.service_nodes: null
cray_node_groups.settings.groups.data.service_nodes.description:
    Default node group which contains all of the service nodes for
    the current partition.
cray_node_groups.settings.groups.data.service_nodes.members:
- platform:service

cray_node_groups.settings.groups.data.group_name.service_nodes_aarch64: null
cray_node_groups.settings.groups.data.service_nodes_aarch64.description:
    Default node group which contains all of the aarch64 service nodes for
    the current partition.
cray_node_groups.settings.groups.data.service_nodes_aarch64.members:
- platform:service-ARM

cray_node_groups.settings.groups.data.group_name.service_nodes_x86_64: null
cray_node_groups.settings.groups.data.service_nodes_x86_64.description:
    Default node group which contains all of the x86_64 service nodes for
    the current partition.
cray_node_groups.settings.groups.data.service_nodes_x86_64.members:
- platform:service-X86

cray_node_groups.settings.groups.data.group_name.smw_nodes: null
cray_node_groups.settings.groups.data.smw_nodes.description:
    Default node group which contains the primary and failover (if
    applicable) SMW nodes.
cray_node_groups.settings.groups.data.smw_nodes.members:
- 1eac199c

cray_node_groups.settings.groups.data.group_name.boot_nodes: null
cray_node_groups.settings.groups.data.boot_nodes.description:
    Default node group which contains the primary and failover (if
    applicable) boot nodes associated with the current partition.
cray_node_groups.settings.groups.data.boot_nodes.members:
- c0-0c0s0n1

cray_node_groups.settings.groups.data.group_name.sdb_nodes: null
cray_node_groups.settings.groups.data.sdb_nodes.description:
    Default node group which contains the primary and failover (if
    applicable) SDB nodes associated with the current partition.
cray_node_groups.settings.groups.data.sdb_nodes.members:
- c0-0c0s1n1

cray_node_groups.settings.groups.data.group_name.login_nodes_x86_64: null
cray_node_groups.settings.groups.data.login_nodes_x86_64.description:
    Default node group which contains the x86_64 login nodes for
    the configured system.
cray_node_groups.settings.groups.data.login_nodes_x86_64.members:
- c0-0c0s2n2

cray_node_groups.settings.groups.data.group_name.login_nodes_aarch64: null
cray_node_groups.settings.groups.data.login_nodes_aarch64.description:
    Default node group which contains the aarch64 login nodes for
    the configured system.
cray_node_groups.settings.groups.data.login_nodes_aarch64.members:
- c0-0c0s3n2

cray_node_groups.settings.groups.data.group_name.elogin_nodes: null
cray_node_groups.settings.groups.data.elogin_nodes.description:

```

```

    Default node group which contains the elogin nodes for
    the configured system.
cray_node_groups.settings.groups.data.elogin_nodes.members:
- <ellogin_hostname>

cray_node_groups.settings.groups.data.group_name.all_nodes: null
cray_node_groups.settings.groups.data.all_nodes.description:
    Default node group which contains all of the nodes applicable to the
    current system. May also contain SMW nodes and external login nodes.
cray_node_groups.settings.groups.data.all_nodes.members:
- platform:compute
- platform:service

cray_node_groups.settings.groups.data.group_name.all_nodes_x86_64: null
cray_node_groups.settings.groups.data.all_nodes_x86_64.description:
    Default node group which contains all of the x86_64 nodes applicable to
    the current system. May also contain SMW nodes and external login nodes.
cray_node_groups.settings.groups.data.all_nodes_x86_64.members:
- platform:compute-X86
- platform:service-X86

cray_node_groups.settings.groups.data.group_name.all_nodes_aarch64: null
cray_node_groups.settings.groups.data.all_nodes_aarch64.description:
    Default node group which contains all of the aarch64 nodes applicable to
    the current system.
cray_node_groups.settings.groups.data.all_nodes_aarch64.members:
- platform:compute-ARM
- platform:service-ARM

cray_node_groups.settings.groups.data.group_name.tier2_nodes: null
cray_node_groups.settings.groups.data.tier2_nodes.description:
    Default node group which contains the tier2 nodes in the system.
    See the guidance in the cray_scalable_services service for a detailed
    description of tier2 nodes.
cray_node_groups.settings.groups.data.tier2_nodes.members:
- c0-0c0s8n0
- c0-0c0s15n0

```

To help with selecting nodes to be tier2 servers, here is a tier2 node FAQ:

**Q. How many tier2 nodes are needed?**

**A.** At least one server must be provided, and for resiliency, two nodes placed on different blades. The recommended ratio of tier2 nodes (servers) to tier3 nodes (clients) is 1 to 400.

**Q. Will adding more tier2 nodes help performance?**

**A.** Adding more tier2 nodes does not always yield additional performance and is subject to diminishing returns.

**Q. What kind of node can be used as a tier2 node?**

**A.** Use these:

- OPTIMAL: dedicated repurposed compute nodes (RCN)
- dedicated service nodes
- nodes with uniform light to moderate load
- nodes with relatively homogeneous single core speeds to reduce resource contention disparity during periods of partial availability

AVOID these (will result in sub-optimal performance):



- nodes with slower individual CPU cores, such as Intel® Xeon Phi™ "Knights Landing" (KNL) processors
- direct-attached Lustre (DAL) servers
- RSIP (realm-specific IP) servers
- login nodes

**Q. Can a tier2 node have more than one role?**

**A.** Small test and development systems (TDS) may use tier2 nodes that have additional roles, but generally, it is better for tier2 nodes to be dedicated.

**Q. Where should tier2 nodes be placed?**

**A.** Distribute nodes throughout the system (on different blades) for resiliency in the event of hardware failure.

#### 4. Define a custom node group, as needed.

Repeat this step for each additional node group.

Copy the three commented lines under **\*\* EXAMPLE 'groups' VALUE** (with current defaults) **\*\*** and paste them under **# NOTE: Place additional 'groups' setting entries here, if desired.**

```
** EXAMPLE 'groups' VALUE (with current defaults) **
#cray_node_groups.settings.groups.data.group_name.sample_key_a: null <--setting a multival key
#cray_node_groups.settings.groups.data.sample_key_a.description: ''
#cray_node_groups.settings.groups.data.sample_key_a.members: []
```

Uncomment the lines, replace `sample_key_a` with the identifier chosen for the node group in all lines, and remove the `<-- setting a multival key` text at the end of the first line (note that the `null` value is required; do not remove or change it). Finally, add values for the `description` (a string) and `members` (a list) fields. For the `members` field, add each list element on a separate line prefixed by a hyphen and space (`-`).

It may be convenient to define node groups that exclude certain nodes. The following examples place the negated keyword last in the list of group members; however the ordering of the list does not matter. All non-negated keywords are resolved first, then negated ones are removed.

```
# NOTE: Place additional 'groups' setting entries here, if desired.

cray_node_groups.settings.groups.data.group_name.enabled_service_compute_nodes: null
cray_node_groups.settings.groups.data.enabled_service_compute_nodes.description:
    All service and compute nodes, excluding disabled nodes.
cray_node_groups.settings.groups.data.enabled_service_compute_nodes.members:
- platform:compute
- platform:service
- ~platform_disabled

cray_node_groups.settings.groups.data.group_name.non_x86_64_compute_nodes: null
cray_node_groups.settings.groups.data.non_x86_64_compute_nodes.description:
    All compute nodes that do not have the x86-64 architecture.
cray_node_groups.settings.groups.data.non_x86_64_compute_nodes.members:
- platform:compute
- ~platform:compute-x86
```

The following example defines a node group called `lnet_nodes`, which could be the list of LNet router nodes to an external Lustre file system.

```
# NOTE: Place additional 'groups' setting entries here, if desired.
...
cray_node_groups.settings.groups.data.group_name.lnet_nodes: null
```

```
cray_node_groups.settings.groups.data.lnet_nodes.decription:
    Node group that contains all the LNet router nodes.
cray_node_groups.settings.groups.data.lnet_nodes.members:
- c0-0c2s1n1
- c0-2c2s1n2
#***** END Service Setting: groups *****
```

### Other custom node groups

Other useful custom node groups might be: `rsip_nodes` (for RSIP server nodes), `mom_nodes` (for MOM nodes with a workload manager), `dvs_nodes` (for a node DVS-projecting an external file system to internal nodes), `datawarp_nodes` (for the DataWarp SSD-endowed nodes), or `postproc_nodes` (for MAMU nodes in the former CLE 5.2 / SMW 7.2 `postproc` node\_class).

The following table lists all of the CLE configuration services that require node groups for one or more variables. In some cases, a custom node group may need to be defined.

cray_alps	cray_local_users	cray_persistent_data
cray_auth	cray_login	cray_rsip
cray_boot	cray_lustre_client	cray_scalable_services
cray_dvs	cray_lustre_server	cray_sdb
cray_dws	cray_nat_masq	cray_simple_services
cray_inet	cray_netroot_preload	cray_simple_shares

### A custom node group for use with Simple Sync

Node groups can be used in conjunction with Simple Sync to distribute files to members of a node group. Here is an example of a custom node group called `automount` that would have an associated `automount` directory in the Simple Sync directory structure on the SMW (in `/var/opt/cray/imps/config/sets/p0/files/simple_sync/nodegroups`), which could be used to distribute `automount` maps to the nodes in that node group (for more information, about the Simple Sync directory structure, see [About Simple Sync](#) on page 27 or see `/var/opt/cray/imps/config/sets/p0/files/simple_sync/README`).

```
# NOTE: Place additional 'group' setting entries here, if desired.
cray_node_groups.settings.groups.data.group_name.automount: null
cray_node_groups.settings.groups.data.automount.decription:
    Node group that contains all the service nodes which will get automount maps
    via Simple Sync.
cray_node_groups.settings.groups.data.automount.members:
- c0-0c1s4n2
#***** END Service Setting: groups *****
```

## 4.5.1.2 Update cray\_net Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

## About this task

The Cray Networking configuration service defines all network information for CLE nodes, which is necessary for a functional system. This procedure configures some basic settings in the `cray_net` configuration worksheet to add site-specific data.

**REMEMBER:** For partitioned systems, each partition generally has its own config set and associated configuration worksheets. Follow this procedure to make changes to the `cray_net_worksheet.yaml` for each partition. Some steps call out what settings should be different for different partitions.

There are two major sections to `cray_net`:

**Networks** Defines the networks to which the CLE nodes are connected.

All networks for CLE nodes must be defined here. The high speed network (HSN) will be connected to the `ipogif0` interface on each CLE node. The login network will be used by the internal login (or network gateway) nodes to an external-to-XC network. Additional networks can be added or described using unique network names, such as for an InfiniBand network or a 40GigEthernet network.

**Hosts** Defines the hosts that are on the previously defined networks. Within each host entry, entries can be defined for network interfaces for that host.

Host entries in `cray_net` are used to specify a host name alias in `/etc/hosts`. Every CLE node does not need to be listed here because the IP address, NID name, and cname entry in the `/etc/hosts` file will be generated based on the address range of the HSN.

For more information about how sites can modify the information in `/etc/hosts`, see [About the /etc/hosts File](#) on page 43.

**About assigning networks to network interfaces when defining hosts.** Each interface defined for a host must have a network assigned to it. There may be situations where a site wishes to assign two networks, each with a different `ipv4_gateway` value, to two interfaces on the same host. This can result in a routing inconsistency, because there is no way to configure which of those IP addresses should be used as the default gateway for that host. A solution involves defining an additional network to assign to one of those two interfaces. For details, see the tip at the beginning of the "DEFINE HOSTS AND THEIR NETWORK INTERFACES" section later in this procedure.

Notes on editing a configuration worksheet:

- Uncomment all settings that are marked `level=basic` and modify values as needed. All settings that remain commented are considered unconfigured.
- Settings that are already uncommented in the original worksheet are preconfigured to ensure proper configuration of the system; Cray recommends not modifying those preconfigured settings.
- Leave commented all settings that are marked `level=advanced` unless a default value needs to be modified. Leaving them commented (unconfigured) allows the configurator to safely update defaults that may change in later releases.
- To enter a value for a string that currently is set to `' '` (empty string), replace the quotes with the new value. For example, `ipv4_network: ' '` becomes `ipv4_network: 10.1.0.0`. In cases where the string value might be interpreted as a number, retain the single quotes. For example, a string setting with value `'512'` needs quotes.

- To enter one or more values for a list that is currently set to `[]` (empty list), remove the brackets and add each entry on a separate line, preceded by a hyphen and a space (`-` ). For example, a list with multiple entries would look like this:

```
cray_global_net.settings.networks.data.management.dns_servers:
- 172.31.84.40
- 172.30.84.40
```

- Do NOT change or remove the null value in lines like this that appear at the beginning of each multival entry (such as the network, host, and host interface entries in `cray_net_worksheet.yaml`. This line sets the key, or identifier, for that multival entry. In this example, `hsn` is the identifier for the HSN network entry.

```
cray_net.settings.networks.data.name.hsn: null
```

For more information about editing configuration worksheets and updating config sets, see *XC™ Series Configurator User Guide (S-2560)*.

## Procedure

1. Edit `cray_net_worksheet.yaml`.

```
smw# vi cray_net_worksheet.yaml
```

2. Uncomment `cray_net.enabled` and ensure that it is set to `true`.

```
----- DEFINE NETWORKS -----
```

### IMPORTANT:

- Add values for the `dns_servers` and `dns_search` fields for the login network only, not to any other network.
- DO NOT add a value for the `ntp_servers` setting for any network used for CLE nodes, because CLE nodes must source their time/NTP settings from the SMW rather than try to contact NTP servers on the login network.

3. Uncomment these two settings for the HSN (high speed network).

If this is a partitioned system, then enter different values for these settings. Partitions `p1` and `p2` will not have the same `ipv4` network, but will have similar `ipv4_netmask` (though different from the full machine).

```
# ** 'networks' DATA **
cray_net.settings.networks.data.hsn.ipv4_network: 10.128.0.0
cray_net.settings.networks.data.hsn.ipv4_netmask: 255.252.0.0
```

4. Configure a login network and add the information for the "Customer network" to which the login nodes connect.

Scroll down to the pre-populated network settings below the `# ** 'networks' DATA **` line and find the login network definition. Uncomment the commented lines and modify the values as needed for this site's internal systems. Note that in the first line, the `null` value is required; do not remove or change it.

**NOTE:** If this site does not use DNS search but does use DNS domain in `/etc/resolv.conf`, then adding a single entry to the `dns_search` setting is functionally equivalent to setting the DNS domain.

```
# ** 'networks' DATA **
...
cray_net.settings.networks.data.name.login: null
cray_net.settings.networks.data.login.description: Customer network
cray_net.settings.networks.data.login.ipv4_network: 172.30.48.0
cray_net.settings.networks.data.login.ipv4_netmask: 255.255.240.0
cray_net.settings.networks.data.login.ipv4_broadcast: ''
cray_net.settings.networks.data.login.ipv4_gateway: 172.30.48.1
cray_net.settings.networks.data.login.dns_servers:
- 172.30.84.40
- 172.31.84.40
- 172.28.84.40
cray_net.settings.networks.data.login.dns_search:
- us.cray.com
- americas.cray.com
- cray.com
cray_net.settings.networks.data.login.ntp_servers: []
cray_net.settings.networks.data.login.fw_external: false
```

**IMPORTANT:** If the login network should be treated as an external network for the firewall, then set `cray_net.settings.networks.data.login.fw_external` (the last line in the example) to `true`.

## 5. Configure additional networks, as needed for this system.

In the worksheet, copy the ten lines below `# ** EXAMPLE 'networks' VALUE` (with current defaults) `**` and paste one set for each network below the line `# NOTE: Place additional 'networks' setting entries here, if desired`.

```
# ** EXAMPLE 'networks' VALUE (with current defaults) **
# cray_net.settings.networks.data.name.sample_key_a: null <-- setting a multival key
# cray_net.settings.networks.data.sample_key_a.description: ''
# cray_net.settings.networks.data.sample_key_a.ipv4_network: ''
# cray_net.settings.networks.data.sample_key_a.ipv4_netmask: ''
# cray_net.settings.networks.data.sample_key_a.ipv4_broadcast: ''
# cray_net.settings.networks.data.sample_key_a.ipv4_gateway: ''
# cray_net.settings.networks.data.sample_key_a.dns_servers: []
# cray_net.settings.networks.data.sample_key_a.dns_search: []
# cray_net.settings.networks.data.sample_key_a.ntp_servers: []
# cray_net.settings.networks.data.sample_key_a.fw_external: false

# ** 'networks' FIELD SPECIFICATION -- MULTIVAL KEY FIELD **
```

Uncomment the lines, replace `sample_key_a` with an identifier for the network (`lnet` and `ethernet40gig` in the example below) in all lines, and remove the `<-- setting a multival key` text at the end of the first line (note that the null value is required; do not remove or change it). Finally, modify the values as needed for this site.

The following example shows two additional networks:

- Infiniband network (`lnet` in the example). Sites that have more than one InfiniBand network will need to add more networks like this first one.
- Network for nodes that have 40GigEthernet interfaces (`ethernet40gig` in the example). For such networks, the `fw_external` variable must be set to `true`.

```
# NOTE: Place additional 'networks' setting entries here, if desired.
cray_net.settings.networks.data.name.lnet: null
cray_net.settings.networks.data.lnet.description:
  The InfiniBand network for LNet router nodes to external Lustre server
cray_net.settings.networks.data.lnet.ipv4_network: 10.150.0.0
cray_net.settings.networks.data.lnet.ipv4_netmask: 255.255.0.0
cray_net.settings.networks.data.lnet.ipv4_broadcast: ''
```

```

cray_net.settings.networks.data.lnet.ipv4_gateway: ''
cray_net.settings.networks.data.lnet.dns_servers: []
cray_net.settings.networks.data.lnet.dns_search: []
cray_net.settings.networks.data.lnet.ntp_servers: []
cray_net.settings.networks.data.lnet.fw_external: false

cray_net.settings.networks.data.name.ethernet40gig: null
cray_net.settings.networks.data.ethernet40gig.description:
    Network for 40GigEthernet
cray_net.settings.networks.data.ethernet40gig.ipv4_network: 138.55.19.0
cray_net.settings.networks.data.ethernet40gig.ipv4_netmask: 255.255.255.0
cray_net.settings.networks.data.ethernet40gig.ipv4_broadcast: ''
cray_net.settings.networks.data.ethernet40gig.ipv4_gateway: ''
cray_net.settings.networks.data.ethernet40gig.dns_servers: []
cray_net.settings.networks.data.ethernet40gig.dns_search: []
cray_net.settings.networks.data.ethernet40gig.ntp_servers: []
cray_net.settings.networks.data.ethernet40gig.fw_external: true
#***** END Service Setting: networks *****

```

## ———— DEFINE HOSTS AND THEIR NETWORK INTERFACES ————

**TIP:** Each interface defined for a host must have a network assigned to it. If this site wishes to assign two networks, each with a different `ipv4_gateway` value, to two interfaces on the same host, do the following to avoid a routing inconsistency:

1. Decide which `ipv4_gateway` should be default for the host, and assign the network with that `ipv4_gateway` to one interface.
2. Define a new network identical to the network that has the other `ipv4_gateway` (the one that should not be default for the host).
3. On the new network, set `ipv4_gateway` to an empty string ( ' ' ).
4. Assign the new network to the other interface.

For example, suppose a site wants to assign different `ipv4_gateway` values to the `login` and `site` networks, and defines a host with one interface assigned to the `login` network and another assigned to the `site` network. The site wants the `ipv4_gateway` assigned to the `login` network to be the default for that host. So the site creates a new network named `site_2` that is identical to `site` except that `ipv4_gateway` is set to ' '. The site then assigns the `site_2` network to the interface that had been assigned to `site`.

### 6. Configure a host as the boot node.

Cray has defined a default `bootnode` host, which is located under the # \*\* 'hosts' DATA \*\* line. Every system has this host.

**IMPORTANT:** Never set `cray_net.settings.hosts.data.bootnode.aliases` to `boot` because that is a host name alias that belongs to the virtual IP address for the boot node in support of the boot node failover feature.

#### a. Configure the host ID of the boot node.

Uncomment `cray_net.settings.hosts.data.bootnode.hostid` and set it to the `cname` of the boot node.

#### b. Configure the host name of the boot node.

Uncomment `cray_net.settings.hosts.data.bootnode.hostname` and set it to the host name of the boot node.

Do not set the host name to `boot` because that name is reserved for the virtual IP address of the boot node, regardless of whether it is the full system or a partitioned system. Choose a name that includes the machine name and "boot" such as `boot-panda`, or if this is a partitioned system, then identify the boot node as `boot-p1`, `boot-p2`, and so forth.

- c. Configure the IP address for the primary Ethernet interface of the boot node.

Uncomment

`cray_net.settings.hosts.data.bootnode.interfaces.primary_ethernet.ipv4_addresses`, if commented out, and set it as follows. This is on the `admin` network to the SMW.

- 10.3.1.254 for a full system (p0).
- 10.3.1.254 for p1, 10.3.1.252 for p2, and so forth for partitioned systems.

- d. (Optional) Configure any secondary IP addresses for the primary Ethernet interface of the boot node.

Uncomment

`cray_net.settings.hosts.data.bootnode.interfaces.primary_ethernet.ipv4_secondary_addresses` and add any additional IP addresses for the node.

- e. Configure the IP address for the HSN boot alias interface of the boot node.

Uncomment

`cray_net.settings.hosts.data.bootnode.interfaces.hsn_boot_alias.ipv4_address`, if commented out, and set it as follows. This is on the HSN and is the "virtual IP address" for the virtual interface `ipogif0:1`.

- For a full system (p0): 10.131.255.254
- For a partitioned system: the highest address possible for a partition's HSN.

For example,

- If p1 HSN `ipv4_address=10.128.0.0` with `ipv4_netmask 255.255.0.0`, then set `hsn_boot_alias.ipv4_address` to 10.128.255.254 for p1.
- If p2 HSN `ipv4_address=10.129.0.0` with `ipv4_netmask 255.255.0.0`, then set `hsn_boot_alias.ipv4_address` to 10.129.255.254 for p2.

- f. (Optional) Configure any secondary IP addresses for the HSN boot alias interface of the boot node.

Uncomment

`cray_net.settings.hosts.data.bootnode.interfaces.hsn_boot_alias.ipv4_secondary_addresses` and add any additional IP addresses reserved for the node.

## 7. Configure a host as the SDB node.

Cray has defined a default `sdbnode` host, which is located under the `# ** 'hosts' DATA **` line. Every system has this host.

- a. Configure the host ID of the SDB node.

Uncomment `cray_net.settings.hosts.data.sdbnode.hostid` and set it to the cname of the SDB node.

- b. Configure the host name of the SDB node.

Uncomment `cray_net.settings.hosts.data.sdbnode.hostname` and set it to `sdb`.

- c. Configure the IP address for the primary Ethernet interface of the SDB node.

Uncomment

`cray_net.settings.hosts.data.sdbnode.interfaces.primary_ethernet.ipv4_address`, if commented out, and set it as follows. This is on the `admin` network to the SMW.

- 10.3.1.253 for a full system (p0).
- 10.3.1.253 for p1, 10.3.1.251 for p2, and so forth for partitioned systems.

- d. (Optional) Configure any secondary IP addresses for the primary Ethernet interface of the SDB node.

Uncomment

`cray_net.settings.hosts.data.sdbnode.interfaces.primary_ethernet.ipv4_secondary_addresses` and add any additional IP addresses reserved for the node.

- e. Configure the IP address for the HSN SDB alias interface of the SDB node.

Uncomment

`cray_net.settings.hosts.data.sdbnode.interfaces.hsn_sdb_alias.ipv4_address`, if commented out, and set it as follows. This is on the HSN and is the "virtual IP address" for the virtual interface `ipogif0:1`.

- For a full system (p0): 10.131.255.253
- For a partitioned system: the next highest address possible for a partition's HSN (the boot node gets the highest address possible).

For example,

- If p1 HSN `ipv4_address=10.128.0.0` with `ipv4_netmask 255.255.0.0`, then set `hsn_sdb_alias.ipv4_address` to 10.128.255.253 for p1.
- If p2 HSN `ipv4_address=10.129.0.0` with `ipv4_netmask 255.255.0.0`, then set `hsn_sdb_alias.ipv4_address` to 10.129.255.253 for p2.

- f. (Optional) Configure any secondary IP addresses for the HSN SDB alias interface of the SDB node.

Uncomment

`cray_net.settings.hosts.data.sdbnode.interfaces.hsn_sdb_alias.ipv4_secondary_addresses` and add any additional IP addresses reserved for the node.

## 8. Configure a host as the login node.

- a. Configure the aliases of the login node.

Uncomment `cray_net.settings.hosts.data.login_node.aliases` and set it to a list of aliases, as follows.

- If this site wishes the login node to have a host name alias of **login**:

```
cray_net.settings.hosts.data.login_node.aliases:
- login
```

- If this site has more than one login node, the first one could have aliases of **login** and **login1**, and the others would be set to **login2**, **login3**, and so forth.

```
cray_net.settings.hosts.data.login_node.aliases:
- login
- login1
```

- b. Configure the host ID of the login node.



Uncomment `cray_net.settings.hosts.data.login_node.hostid` and set it to the cname of the login node. If this system has more than one login node, set this variable to the first login node.

- c. Configure the host name of the login node.

Uncomment `cray_net.settings.hosts.data.login_node.hostname` and set it to the host name.

This could be the machine name, for systems that have only one login node. For example, on a machine known as `panda`, this would be `panda`. For systems with more than one login node, the host name could be `panda1` for the first one, `panda2` for the second one, and so forth.

- d. Configure the IP address for the login Ethernet interface of the login node.

Uncomment

`cray_net.settings.hosts.data.login_node.interfaces.login_ethernet.ipv4_addresses`, if commented out, and set it to the IP address of the login node's `eth0` interface on the `login` network.

- e. (Optional) Configure any secondary IP addresses for the login Ethernet interface of the login node.

Uncomment

`cray_net.settings.hosts.data.login_node.interfaces.login_ethernet.ipv4_secondary_addresses` and add any additional IP addresses reserved for the node.

## 9. Configure additional hosts, as needed.

If this system has additional service nodes that need to have host name or host name alias or network interface settings, then for each one add a host definition stanza like the following, placing it under `NOTE`: Place additional 'hosts' setting entries here, if desired. The first example shows the host configuration of a DVS node (`dvs_node`) with the host name set to `dvs1`, a host name alias of `dvs`, and one Ethernet interface connected to the `login` network. (The network field must be specified for any interface that has an assigned IP address, and that network must be defined in the `cray_net` config service.)

```
cray_net.settings.hosts.data.common_name.dvs_node: null
cray_net.settings.hosts.data.dvs_node.description: DVS node
cray_net.settings.hosts.data.dvs_node.aliases:
- dvs
cray_net.settings.hosts.data.dvs_node.hostid: c0-0c0s0n2
cray_net.settings.hosts.data.dvs_node.host_type: ''
cray_net.settings.hosts.data.dvs_node.hostname: dvs1
cray_net.settings.hosts.data.dvs_node.standby_node: false

cray_net.settings.hosts.data.dvs_node.interfaces.common_name.eth0: null
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.name: eth0
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.description:
  Ethernet connecting the DVS node to the customer network.
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.vlan_id: ''
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.vlan_etherdevice: ''
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.bonding_slaves: []
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.bonding_module_opts:
  mode=active-backup miimon=100
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.aliases: []
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.network: login
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.ipv4_address: 172.30.50.128
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.mac: ''
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.startmode: auto
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.bootproto: static
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.mtu: ''
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.extra_attributes: []
#cray_net.settings.hosts.data.dvs_node.interfaces.eth0.module: ''
#cray_net.settings.hosts.data.dvs_node.interfaces.eth0.params: ''
#cray_net.settings.hosts.data.dvs_node.interfaces.eth0.unmanaged_interface: false
```

The second example shows the host configuration for an LNet router node (`clfs_lnet_1`) that has two different InfiniBand interfaces (`ib0` and `ib2`) to connect to two different networks.

**NOTICE:** In this example, the interface parameter `mtu` for both interfaces is set to a numerical value within single quotes. The quotes are important. The configurator expects a string for this setting, and without the single quotes, it could interpret this value as a number and return an error. The values provided for other parameters of type string do not need single quotes because they would not be interpreted as anything other than strings.

```
cray_net.settings.hosts.data.common_name.clfs_lnet_1: null
cray_net.settings.hosts.data.clfs_lnet_1.description: CLFS router 1 node
cray_net.settings.hosts.data.clfs_lnet_1.aliases: []
cray_net.settings.hosts.data.clfs_lnet_1.hostid: c0-0c1s0n1
cray_net.settings.hosts.data.clfs_lnet_1.host_type: ''
cray_net.settings.hosts.data.clfs_lnet_1.hostname: lnet1
cray_net.settings.hosts.data.clfs_lnet_1.standby_node: false

cray_net.settings.hosts.data.clfs_lnet_1.interfaces.common_name.ib0: null
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.name: ib0
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.description:
    InfiniBand ib0 connecting the CLFS router 1 node to the lnet network.
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.vlan_id: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.vlan_etherdevice: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.bonding_slaves: []
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.bonding_module_opts:
    mode=active-backup miimon=100
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.aliases: []
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.network: lnet
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.ipv4_address: 10.150.10.65
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.mac: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.startmode: auto
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.bootproto: static
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.mtu: '65520'
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.extra_attributes:
- IPOIB_MODE='connected'
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.module: ''
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.params: ''
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.unmanaged_interface: false

cray_net.settings.hosts.data.clfs_lnet_1.interfaces.common_name.ib2: null
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.name: ib2
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.description:
    InfiniBand ib2 connecting the CLFS router 1 node to the lnet1 network.
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.vlan_id: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.vlan_etherdevice: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.bonding_slaves: []
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.bonding_module_opts:
    mode=active-backup miimon=100
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.aliases: []
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.network: lnet1
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.ipv4_address: 10.151.10.65
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.mac: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.startmode: auto
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.bootproto: static
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.mtu: '65520'
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.extra_attributes:
- IPOIB_MODE='connected'
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.module: ''
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.params: ''
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.unmanaged_interface: false
```

## 10. Configure a host as the second boot node for boot node failover.

**Boot/SDB node failover.** If configuring boot node failover, define a backup boot node host with the `standby_node` variable set to true.

**NOTE:** The host name for the primary and backup boot node should both be set to a name that includes the machine name and "boot" such as `boot-panda`. The aliases can be different so that the `/etc/hosts` entry for the cname has the host name alias.

```
cray_net.settings.hosts.data.common_name.backup_bootnode: null
cray_net.settings.hosts.data.backup_bootnode.description: backup Boot node for the system
```

```

cray_net.settings.hosts.data.backup_bootnode.aliases:
- cray-boot2
cray_net.settings.hosts.data.backup_bootnode.hostid: c0-2c0s4n1
cray_net.settings.hosts.data.backup_bootnode.host_type: admin
cray_net.settings.hosts.data.backup_bootnode.hostname: boot-panda
cray_net.settings.hosts.data.backup_bootnode.standby_node: true

cray_net.settings.hosts.data.backup_bootnode.interfaces.common_name.hsn_boot_alias: null
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.name: ipogif0:1
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.description:
    Well known address used for boot node services.
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.vlan_id: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.vlan_etherdevice: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.bonding_slaves: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.bonding_module_opts:
    mode=active-backup miimon=100
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.aliases: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.network: hsn
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.ipv4_address: 10.131.255.254
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.mac: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.startmode: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.bootproto: static
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.mtu: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.extra_attributes: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.module: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.params: ''
#cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.unmanaged_interface: false

cray_net.settings.hosts.data.backup_bootnode.interfaces.common_name.primary_ethernet: null
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.name: eth0
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.description:
    Ethernet connecting boot node to the SMW.
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.vlan_id: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.vlan_etherdevice: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.bonding_slaves: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.bonding_module_opts:
    mode=active-backup miimon=100
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.aliases: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.network: admin
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.ipv4_address: 10.3.1.254
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.mac: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.startmode: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.bootproto: static
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.mtu: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.extra_attributes: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.module: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.params: ''
#cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.unmanaged_interface: false

```

## 11. Configure a host as the second SDB node for SDB node failover.

**Boot/SDB node failover.** If configuring SDB node failover, define a backup SDB node host with the `standby_node` variable set to true.

**NOTE:** The host name for the primary and backup SDB node should both be set to `sdb`. The aliases can be different so that the `/etc/hosts` entry for the cname has the host name alias.

```

cray_net.settings.hosts.data.common_name.backup_sdbnode: null
cray_net.settings.hosts.data.backup_sdbnode.description: backup SDB node for the system
cray_net.settings.hosts.data.backup_sdbnode.aliases:
- cray-sdb2
cray_net.settings.hosts.data.backup_sdbnode.hostid: c0-4c0s3n1
cray_net.settings.hosts.data.backup_sdbnode.host_type: admin
cray_net.settings.hosts.data.backup_sdbnode.hostname: sdb
cray_net.settings.hosts.data.backup_sdbnode.standby_node: true

cray_net.settings.hosts.data.backup_sdbnode.interfaces.common_name.hsn_boot_alias: null
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.name: ipogif0:1
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.description:
    Well known address used for SDB node services.
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.vlan_id: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.vlan_etherdevice: ''

```

```

cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.bonding_slaves: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.bonding_module_opts:
    mode=active-backup miimon=100
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.aliases: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.network: hsn
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.ipv4_address: 10.131.255.253
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.mac: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.startmode: auto
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.bootproto: static
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.mtu: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.extra_attributes: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.module: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.params: ''
#cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.unmanaged_interface: false

cray_net.settings.hosts.data.backup_sdbnode.interfaces.common_name.primary_ethernet: null
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.name: eth0
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.description:
    Ethernet connecting SDB node to the SMW.
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.vlan_id: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.vlan_etherdevice: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.bonding_slaves: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.bonding_module_opts:
    mode=active-backup miimon=100
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.aliases: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.network: admin
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.ipv4_address: 10.3.1.253
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.mac: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.startmode: auto
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.bootproto: static
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.mtu: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.extra_attributes: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.module: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.params: ''
#cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.unmanaged_interface: false

```

## 12. (Optional) Configure a virtual LAN (VLAN) interface, as needed.

This example shows the configuration fields needed to configure a VLAN interface with common name set to **vlan0**. With the **vlan\_id** set to **42** and the **etherdevice** set to **eth0**, the interface name will be set automatically to **eth0.42** (**vlan\_etherdevice.vlan\_id**) if the name field is left empty (recommended). If this site chooses to leave **vlan\_id** empty instead (NOT recommended), the name field must be set to a non-empty string. The network field must be specified for any interface that has an assigned IP address, and that network must be defined in the **cray\_net** config service.

- required** Necessary for this type of interface. Uncomment this line and replace the default with an interface-specific value.
- default** Must be configured because it is level=basic, but the default value is sufficient for this interface. Uncomment this line.
- optional** Not necessary. Leave commented all fields that are level=advanced unless a default value needs to be modified. Leaving them commented (unconfigured) allows the configurator to safely update defaults that may change in later releases.
- ignore** Incompatible with this type of interface or this context. Leave unconfigured by leaving the line commented.

```

cray_net.settings.hosts.data.some_host.interfaces.common_name.vlan0: null # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.name: '' # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.description: '' # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.vlan_id: 42 # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.vlan_etherdevice: eth0 # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.bonding_slaves: [] # ignore
cray_net.settings.hosts.data.some_host.interfaces.vlan0.bonding_module_opts: # ignore
    mode=active-backup miimon=100
cray_net.settings.hosts.data.some_host.interfaces.vlan0.aliases: [] # default

```

```

cray_net.settings.hosts.data.some_host.interfaces.vlan0.network: some_network # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.ipv4_address: some_IP_addr # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.ipv4_secondary_addresses: [] # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.mac: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.vlan0.startmode: auto # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.bootproto: static # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.mtu: '' # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.extra_attributes: [] # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.module: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.vlan0.params: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.vlan0.unmanaged_interface: false # optional

```

### 13. (Optional) Configure a bonded interface, as needed.

This example shows the configuration fields needed to configure a bonded interface with common name set to **bond0** and interface name set also to **bond0**. There is no field for bonding master because it is set automatically when the `bonding_slaves` list has at least one member. The network field must be specified for any interface that has an assigned IP address, and that network must be defined in the `cray_net` config service. In the case of bonded interfaces, a network must be specified for the master bond interface but not for the slave bond interfaces.

- required** Necessary for this type of interface. Uncomment this line and replace the default with an interface-specific value.
- default** Must be configured because it is level=basic, but the default value is sufficient for this interface. Uncomment this line.
- optional** Not necessary. Leave commented all fields that are level=advanced unless a default value needs to be modified. Leaving them commented (unconfigured) allows the configurator to safely update defaults that may change in later releases.
- ignore** Incompatible with this type of interface or this context. Leave unconfigured by leaving the line commented.

```

cray_net.settings.hosts.data.some_host.interfaces.common_name.bond0: null # required
cray_net.settings.hosts.data.some_host.interfaces.bond0.name: bond0 # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.description: '' # required
cray_net.settings.hosts.data.some_host.interfaces.bond0.vlan_id: '' # ignore
cray_net.settings.hosts.data.some_host.interfaces.bond0.vlan_etherdevice: '' # ignore
cray_net.settings.hosts.data.some_host.interfaces.bond0.bonding_slaves: # required
- eth0
- eth2
cray_net.settings.hosts.data.some_host.interfaces.bond0.bonding_module_opts: # required
mode=active-backup miimon=100
cray_net.settings.hosts.data.some_host.interfaces.bond0.aliases: [] # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.network: some_network # required
cray_net.settings.hosts.data.some_host.interfaces.bond0.ipv4_address: some_IP_addr # required
cray_net.settings.hosts.data.some_host.interfaces.bond0.ipv4_secondary_addresses: [] # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.mac: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.bond0.startmode: onboot # required
cray_net.settings.hosts.data.some_host.interfaces.bond0.bootproto: static # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.mtu: '' # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.extra_attributes: [] # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.module: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.bond0.params: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.bond0.unmanaged_interface: false # optional

```

### 14. (Optional) Configure a bonded VLAN, as needed.

A "bonded VLAN" is a bonded interface with two ethernet NICs as slaves and two or more VLAN interfaces with the bonded interface as their etherdevice. The VLAN interfaces are typically on different subnets. These examples show the configuration fields needed to configure the necessary interfaces. The network field must be specified for any interface that has an assigned IP address, and that network must be defined in the `cray_net` config service. In the case of bonded interfaces, a network must be specified for the master bond interface but not for the slave bond interfaces.



- required** Necessary for this type of interface. Uncomment this line and replace the default with an interface-specific value.
- default** Must be configured because it is level=basic, but the default value is sufficient for this interface. Uncomment this line.
- optional** Not necessary. Leave commented all fields that are level=advanced unless a default value needs to be modified. Leaving them commented (unconfigured) allows the configurator to safely update defaults that may change in later releases.
- ignore** Incompatible with this type of interface or this context. Leave unconfigured by leaving the line commented.

Example bonded interface. Note that there is no field for bonding master because it is set automatically when the bonding\_slaves list has at least one member. Also note that the ipv4\_address field is the default empty string because the address will be set on the VLAN.

```
cray_net.settings.hosts.data.some_host.interfaces.common_name.bond0: null # required
cray_net.settings.hosts.data.some_host.interfaces.bond0.name: bond0 # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.description: '' # required
cray_net.settings.hosts.data.some_host.interfaces.bond0.vlan_id: '' # ignore
cray_net.settings.hosts.data.some_host.interfaces.bond0.vlan_etherdevice: '' # ignore
cray_net.settings.hosts.data.some_host.interfaces.bond0.bonding_slaves: # required
- eth0
- eth2
cray_net.settings.hosts.data.some_host.interfaces.bond0.bonding_module_opts: # required
    mode=active-backup miimon=100
cray_net.settings.hosts.data.some_host.interfaces.bond0.aliases: [] # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.network: some_network # required
cray_net.settings.hosts.data.some_host.interfaces.bond0.ipv4_address: '' # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.ipv4_secondary_addresses: [] # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.mac: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.bond0.startmode: onboot # required
cray_net.settings.hosts.data.some_host.interfaces.bond0.bootproto: static # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.mtu: '' # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.extra_attributes: [] # default
cray_net.settings.hosts.data.some_host.interfaces.bond0.module: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.bond0.params: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.bond0.unmanaged_interface: false # optional
```

Example VLAN interfaces:

```
cray_net.settings.hosts.data.some_host.interfaces.common_name.vlan0: null # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.name: '' # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.description: '' # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.vlan_id: 42 # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.vlan_etherdevice: bond0 # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.bonding_slaves: [] # ignore
cray_net.settings.hosts.data.some_host.interfaces.vlan0.bonding_module_opts: # ignore
    mode=active-backup miimon=100
cray_net.settings.hosts.data.some_host.interfaces.vlan0.aliases: [] # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.network: some_network # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.ipv4_address: some_IP_addr # required
cray_net.settings.hosts.data.some_host.interfaces.vlan0.ipv4_secondary_addresses: [] # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.mac: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.vlan0.startmode: auto # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.bootproto: static # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.mtu: '' # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.extra_attributes: [] # default
cray_net.settings.hosts.data.some_host.interfaces.vlan0.module: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.vlan0.params: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.vlan0.unmanaged_interface: false # optional

cray_net.settings.hosts.data.some_host.interfaces.common_name.vlan1: null # required
cray_net.settings.hosts.data.some_host.interfaces.vlan1.name: '' # default
cray_net.settings.hosts.data.some_host.interfaces.vlan1.description: '' # required
cray_net.settings.hosts.data.some_host.interfaces.vlan1.vlan_id: 43 # required
cray_net.settings.hosts.data.some_host.interfaces.vlan1.vlan_etherdevice: bond0 # required
cray_net.settings.hosts.data.some_host.interfaces.vlan1.bonding_slaves: [] # ignore
cray_net.settings.hosts.data.some_host.interfaces.vlan1.bonding_module_opts: # ignore
    mode=active-backup miimon=100
```

```

cray_net.settings.hosts.data.some_host.interfaces.vlan1.aliases: [] # default
cray_net.settings.hosts.data.some_host.interfaces.vlan1.network: some_network # required
cray_net.settings.hosts.data.some_host.interfaces.vlan1.ipv4_address: some_IP_addr # required
cray_net.settings.hosts.data.some_host.interfaces.vlan1.ipv4_secondary_addresses: [] # default
cray_net.settings.hosts.data.some_host.interfaces.vlan1.mac: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.vlan1.startmode: auto # default
cray_net.settings.hosts.data.some_host.interfaces.vlan1.bootproto: static # default
cray_net.settings.hosts.data.some_host.interfaces.vlan1.mtu: '' # default
cray_net.settings.hosts.data.some_host.interfaces.vlan1.extra_attributes: [] # default
cray_net.settings.hosts.data.some_host.interfaces.vlan1.module: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.vlan1.params: '' # optional
cray_net.settings.hosts.data.some_host.interfaces.vlan1.unmanaged_interface: false # optional

```

#### 15. Set the module and params settings for any hosts that are network nodes and use special network cards.

Sites that use special network cards (e.g., Mellanox ConnectX-3) must specify which kernel module is used by those cards. For each host that uses such a card, uncomment (if commented out) the following setting, then replace the empty string with the kernel module name.

This example specifies **mlx4\_en**, the module for Mellanox ConnectX-3 cards.

```

cray_net.settings.hosts.data.network_node.interfaces.eth0.module: mlx4_en

```

Any kernel module parameters that need to be set for that module can be specified by uncommenting the following setting, then replacing the empty string with "parameter=value" pairs (pairs separated by spaces). This is not common; no parameters need to be specified for the **mlx4\_en** module. Note that the = syntax may vary by kernel module; consult the documentation of the kernel module being used.

```

cray_net.settings.hosts.data.networknode.interfaces.eth0.params: param1=200 param2=30

```

### 4.5.1.3 Update cray\_alps Worksheet

#### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```

smw# cd /var/adm/cray/release/p0_worksheet_workarea

```

#### About this task

Cray ALPS (Application Level Placement Scheduler) is the Cray-supported mechanism for placing and launching applications on Cray system compute nodes. ALPS provides application placement, launch, and management functions and cooperates closely with third-party workload managers (WLM) for application scheduling across Cray systems. The third-party WLMs make policy and scheduling decisions, whereas ALPS provides a mechanism to place and launch the applications contained within batch jobs. ALPS also supports interactive application placement and launch.

This procedure enables the `cray_alps` service and configures some settings in the `cray_alps` configuration worksheet to add site-specific data. For an explanation of the long variable names in configuration settings, see [About Variable Names in the Configurator and Configuration Worksheets](#) on page 20.

#### Procedure

1. Edit `cray_alps_worksheet.yaml`.

```

smw# vi cray_alps_worksheet.yaml

```

2. Uncomment `cray_alps.enabled` and ensure that it is set to `true`.
3. Uncomment `cray_alps.settings.common.data.xhostname` and set it to the name of this Cray system.
4. Configure ALPS node groups.

If there are service nodes other than login nodes and the ALPS master node (the SDB node) that need to run ALPS commands, add them to a node group by editing `cray_node_groups_worksheet.yaml`. That node group should include the workload manager (WLM) server and MOM (machine-oriented miniserver) nodes.

Uncomment `cray_alps.settings.common.data.alps_node_groups`, remove the empty list (`[]`), and add that node group (and any other node groups, as needed) on a separate line prefixed by a hyphen and space (`-` ).

```
cray_alps.settings.common.data.alps_node_groups:
- NODE_GROUP_1
- NODE_GROUP_2
```

5. (Optional) If DRC (dynamic RDMA credentials) will be used in a large system, uncomment `cray_alps.settings.apshed.data.pDomainMax` and set it to 256.

If the maximum number of user protection domains is not increased from its default value of 10 to something like 256, DRC might exhaust all of the domains, which could cause problems for sites with larger, more complex systems.

6. Uncomment `cray_alps.settings.apsys.data.prologPath` and `cray_alps.settings.apsys.data.epilogPath`, even if they are assigned a null value.
7. (Optional) If Resource Utilization Reporting (RUR) will be used at this site, set the `prologPath` and `epilogPath` settings, which were uncommented in the previous step, to the following paths.

```
cray_alps.settings.apsys.data.prologPath: /opt/cray/rur/default/bin/rur_prologue.py
cray_alps.settings.apsys.data.epilogPath: /opt/cray/rur/default/bin/rur_epilogue.py
```

Also, ensure that the `cray_rur` config service is enabled (this is done later in the process).

#### 4.5.1.4 Update `cray_auth` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

The Cray Authentication configuration service provides a way to list the authentication domains that should govern how users of the system are identified and authenticated. Authentication domains include LDAP, NIS, and Active Directory.

This procedure configures some settings in the `cray_auth` configuration worksheet to add site-specific data. For an explanation of the long variable names in configuration settings, see [About Variable Names in the Configurator](#)



and [Configuration Worksheets](#) on page 20. For examples of modifying a config set for use with an authentication method other than the default LDAP setup, see [Modify a Config Set for Use with Advanced Authentication Configurations](#) on page 410.

## Procedure

1. Edit `cray_auth_worksheet.yaml`.

```
smw# vi cray_auth_worksheet.yaml
```

2. Uncomment `cray_auth.enabled` and set it to `true`.

3. Review the `nsswitch_sources` service setting of the worksheet.

```
#***** START Service Setting: nsswitch_sources *****
```

This service setting controls the settings in the `nsswitch.conf` file. Add, delete, or change these settings to modify the `nsswitch.conf` file, as needed.

4. Review the `common_ldap_options` service setting of the worksheet, especially if LDAP will NOT be used at this site.

```
#***** START Service Setting: common_ldap_options *****
```

This is an advanced level setting that has several pre-populated values for common LDAP configuration options.

- Sites NOT using LDAP for part or all of the authentication must change some settings in this section (for example, to use Kerberos or Active Directory).
- Sites using LDAP may need to add, change, or delete options in this section.

To add a `common_ldap_options` stanza to the worksheet, copy the two lines below `# ** EXAMPLE`

`'common_ldap_options' VALUE (with current defaults) **` and paste them below `# NOTE:`. Place additional `'common_ldap_options'` setting entries here, if desired.

```
# ** EXAMPLE 'common_ldap_options' VALUE (with current defaults) **
# cray_auth.settings.common_ldap_options.data.option.sample_key_a: null <-- setting a multival key
# cray_auth.settings.common_ldap_options.data.sample_key_a.value: ''

# ** 'common_ldap_options' FIELD SPECIFICATION -- MULTIVAL KEY FIELD **
```

Uncomment the lines, replace `sample_key_a` in all lines with the LDAP option to be specified, and remove the `<-- setting a multival key` text at the end of the first line (note that the `null` value is required; do not remove or change it). Finally, add the value of the option in the second line. Repeat this step for each option/value pair to be specified.

```
# NOTE: Place additional 'common_ldap_options' setting entries here, if desired.
cray_auth.settings.common_ldap_options.data.option.sample_key_a: null
cray_auth.settings.common_ldap_options.data.sample_key_a.value: ''
```

```
#***** END Service Setting: common_ldap_options *****
```

5. (If using NIS) Review the `common_nis_options` service setting of the worksheet and configure these settings if this site wishes to use NIS.

```
#***** START Service Setting: common_nis_options *****
```

This is an advanced level setting that has several pre-populated values for common NIS configuration options. Add, change, or delete options if this site has special authentication needs, such as when using Kerberos (not common).

In the worksheet, copy the two lines below `# ** EXAMPLE 'common_nis_options' VALUE` (with current defaults) `**` and paste them below `# NOTE: Place additional 'common_nis_options' setting entries here, if desired.`

```
# ** EXAMPLE 'common_nis_options' VALUE (with current defaults) **
# cray_auth.settings.common_nis_options.data.option.sample_key_a: null <-- setting a multival key
# cray_auth.settings.common_nis_options.data.sample_key_a.value: ''
```

Uncomment the lines, replace `sample_key_a` in all lines with the NIS option to be specified, and remove the `<-- setting a multival key` text at the end of the first line (note that the `null` value is required; do not remove or change it). Finally, add the value of the option in the second line. Repeat this step for each option/value pair to be specified.

```
# NOTE: Place additional 'common_nis_options' setting entries here, if desired.
cray_auth.settings.common_nis_options.data.option.sample_key_a: null
cray_auth.settings.common_nis_options.data.sample_key_a.value: ''

# ***** END Service Setting: common_nis_options *****
```

## 6. Review the domain service setting of the worksheet and configure settings, as needed.

```
# ***** START Service Setting: domain *****
```

### a. (If using LDAP) Configure LDAP domains to connect to LDAP servers

In the worksheet, copy the four lines below `# ** EXAMPLE 'domain' VALUE` (with current defaults) `**` and paste them below `# NOTE: Place additional 'domain' setting entries here, if desired.`

```
# ** EXAMPLE 'domain' VALUE (with current defaults) **
# cray_auth.settings.domain.data.reference.sample_key_a: null <-- setting a multival key
# cray_auth.settings.domain.data.sample_key_a.servers: []
# cray_auth.settings.domain.data.sample_key_a.schema: rfc2307
# cray_auth.settings.domain.data.sample_key_a.aux_settings: []
```

Uncomment the lines, replace `sample_key_a` in all lines with some unique authentication domain identifier, and remove the `<-- setting a multival key` text at the end of the first line (note that the `null` value is required; do not remove or change it). Finally, add values in accordance with site requirements. For settings that are lists, remove the empty brackets and add each list element on a separate line prefixed by a hyphen and space (`-` ).

```
# NOTE: Place additional 'domain' setting entries here, if desired.
cray_auth.settings.domain.data.reference.<ldap_domain_name>: null
cray_auth.settings.domain.data.<ldap_domain_name>.servers: []
cray_auth.settings.domain.data.<ldap_domain_name>.schema: rfc2307
cray_auth.settings.domain.data.<ldap_domain_name>.aux_settings: []
# ***** END Service Setting: domain *****
```

### b. Configure non-LDAP domains, as needed.

As in the previous substep, copy and paste the four-line stanza for a domain setting, but instead of uncommenting all four lines, leave commented the `servers` variable and the `schema` variable, which are specific to LDAP domains.

```
# NOTE: Place additional 'domain' setting entries here, if desired.
cray_auth.settings.domain.data.reference.<domain_name>: null
#cray_auth.settings.domain.data.<domain_name>.servers: []
#cray_auth.settings.domain.data.<domain_name>.schema: rfc2307
cray_auth.settings.domain.data.<domain_name>.aux_settings: []
# ***** END Service Setting: domain *****
```

7. (If using NIS) Review the `nis` service setting of the worksheet and configure these settings if this site wishes to use NIS.

```
#***** START Service Setting: nis *****
```

- a. Enable NIS (the `nis.data.enabled` setting).

Uncomment `cray_auth.settings.nis.data.enabled` and set it to `true`.

- b. Configure the domain name (the `nis.data.domainname` setting).

Uncomment `cray_auth.settings.nis.data.domainname` and set it to the domain name that was configured on the NIS server (must match).

- c. Configure the servers (the `nis.data.servers` setting).

Uncomment `cray_auth.settings.nis.data.servers: []`, remove the empty brackets, and add a list of NIS server host names or IP addresses.

```
cray_auth.settings.nis.data.servers:
- 172.32.3.4
- 172.32.4.55
```

8. Review the `access` service setting of the worksheet and configure these settings, as needed.

```
#***** START Service Setting: access *****
```

- a. Set the access policy (the `access.data.policy` setting).

Whether using NIS or LDAP, ensure that `cray_auth.settings.access.data.policy` is uncommented and set it to the list shown here. At a minimum, these values are recommended to ensure that root and crayadm are using the local passwd entries and not ones from the authentication service.

**NOTICE:** The initial `-` (hyphen and space) at the beginning of each list element is part of the YAML syntax. The access policy data, which begins with either a `-` or `+`, starts after that.

```
cray_auth.settings.access.data.policy:
- +:root:LOCAL
- +:crayadm:LOCAL
```

- b. Configure access to compute nodes (the `access.data.config_computes` setting).

Uncomment `cray_auth.settings.access.data.config_computes` and set in accordance with site requirements. For most systems, set this variable to `false`.

Set this variable to `true` for any of these conditions:

- This site wants to allow compute nodes to use network lookup services to identify users (setting `config_computes` to `true` does not mean that users will be allowed to log into compute nodes directly).
- This site is using Slurm and network authentication.
- This site is using cluster compatibility mode (CCM) with LDAP accounts (`ccmrun` will work regardless, but `ccmlogin` will work only if `config_computes` is set to `true`).

**IMPORTANT:** If `cray_auth.settings.access.data.config_computes` is set to `true`, ensure that:

- RSIP is configured to enable the compute nodes to contact the LDAP server.
- Network user lookup servers are equipped to handle the volume of requests made by the compute nodes.

- c. Configure node groups to recognize user IDs provided by off-node identification services, if needed (the `access.data.config_id_service_groups` setting).

If there are any non-login nodes that may need to identify users without allowing user access, such as DAL (direct-attached Lustre) MDS nodes or MOM nodes, add their cnames to a node group by editing `cray_node_groups_worksheet.yaml`, and then add that node group to the list of `config_id_service_groups`. Nodes within these groups should be provided with a network path to the relevant servers.

Uncomment `cray_auth.settings.access.data.config_id_service_groups`, remove the empty list (`[]`), and add that node group (and any other node groups, as needed) on a separate line prefixed by a hyphen and space (`-` ).

```
cray_auth.settings.access.data.config_id_service_groups:
- NODE_GROUP_1
- NODE_GROUP_2
```

9. Review the `section` service setting and the `options` embedded service setting (they go together) of the worksheet and configure these settings, as needed.

The "section" service setting refers to the section of the `sssd.conf` file, and "options" are the multiple entries within a "section" that an administrator can specify. This enables the administrator to override settings in any section of the `sssd.conf` file.

In the worksheet, copy the three lines below `# ** EXAMPLE 'section' VALUE` (with current defaults) `**` and paste them below `# NOTE: Place additional 'section' setting entries here, if desired`.

```
# ** EXAMPLE 'section' VALUE (with current defaults) **
# cray_auth.settings.section.data.section_name.sample_key_a: null <-- setting a multival key
# cray_auth.settings.section.data.sample_key_a.options.option_name.sample_key_b: null <-- setting a multival key
# cray_auth.settings.section.data.sample_key_a.options.sample_key_b.value: ''
```

Uncomment the lines, replace `sample_key_a` in all lines with a unique section identifier and remove the `<-- setting a multival key` text at the end of the first line (note that the `null` value is required; do not remove or change it). Replace `sample_key_b` in the second and third lines with a unique option identifier. Finally, add values in accordance with site requirements.

Here is an example that adds a `"debug_level = 7"` line to the otherwise unnamed/unused "[pam]" section in the `sssd.conf` file.

```
# NOTE: Place additional 'section' setting entries here, if desired.
cray_auth.settings.section.data.section_name.pam: null
cray_auth.settings.section.data.pam.options.option_name.debug_level: null
cray_auth.settings.section.data.pam.options.debug_level.value: 7
#***** END Service Setting: section *****
```

Here is the resulting section of the `sssd.conf` file.

```
[nss]
filter_users = root, crayadm

[pam] <-----
debug_level = 7 <-----

[domain/crayit]
```

### 4.5.1.5 Update `cray_boot` Worksheet

#### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

#### About this task

The Cray boot configuration service provides a way to specify which nodes will act as boot nodes on the high-speed network (HSN). This service must be enabled for the system to function properly. This procedure configures some basic settings in the `cray_boot` configuration worksheet.

#### Procedure

1. Edit `cray_boot_worksheet.yaml`.

```
smw# vi cray_boot_worksheet.yaml
```

2. Uncomment `cray_boot.enabled` and ensure that it is set to `true`.
3. Configure the boot groups setting.

This setting specifies a list of node groups whose members will act as boot nodes.

Uncomment `cray_boot.settings.node_groups.data.boot_groups` and the line immediately following it. By default, the `boot_nodes` node group is the first node group in the list of boot groups. To use other nodes as boot nodes on the HSN, add one or more node groups to this list.

**IMPORTANT:** Any node group added to `boot_groups` must first be defined in `cray_node_groups_worksheet.yaml`.

Because this is a list setting, each node group must be on a separate line prefixed by a hyphen and space (`-`).

```
cray_boot.settings.node_groups.data.boot_groups:
- boot_nodes
```

### 4.5.1.6 About Configuring Cray Dynamic RDMA Credentials (DRC)

Dynamic RDMA Credentials (DRC) is a new XC system service that enables shared network access between different user applications. DRC enables user applications to request managed network credentials, which can be shared with other users, groups, or jobs. Access to a credential is governed by the application and DRC to provide authorized and protected sharing of network access between applications. DRC extends the existing protection domain functionality provided by ALPS without exposing application data to unauthorized applications. DRC can also be used with other batch systems, such as Slurm, without any loss of functionality.

**Trouble?** Do not use DRC with VMDH (virtual memory domain handle). DRC does not use VMDH or limit its use; however, in a MAMU (multiple application multiple user) scenario, the use of VMDH by an application that is also using DRC could cause problems for other applications using VMDH on the same node, resulting in the failure of one or more of those processes.

Cray DRC has three groups of configuration settings. The default values provided for each setting are sufficient for most cases. However, there are a few level=required settings that must be configured with site-specific information, as noted:

- DRC client (DRCC) settings (no level=required settings)
- DRC server (DRCS) settings (one level=required setting: `server_cname`)
- DRC database settings (two level=required settings: `username` and `password`)

## DRC Client (DRCC) Settings

No DRCC settings are level=required.

<b>socket_location</b>	Location of the DRCC UNIX domain socket. This location should allow read-write access for any user, because libDRC must be able to write to the socket to make any necessary requests. Default value: <code>/tmp/drcc.sock</code>
<b>logging_directory</b>	Storage location for DRCC logs. This can be located anywhere convenient, as long as the directory is: <ul style="list-style-type: none"><li>• (required) writeable by root</li><li>• (recommended) persistent between reboots so that the log file can be retrieved in a node-down event</li></ul> Default value: <code>/tmp</code>
<b>logging_filename</b>	Name of the log file for DRCC. This name can be anything except a null value. Default value: <code>drcc.log</code>
<b>logging_level</b>	Verbosity of the DRCC logger. Possible values in order of increasing verbosity: <code>critical</code> , <code>error</code> , <code>warning</code> , <code>info</code> , and <code>debug</code> . Default value: <code>error</code>
<b>requests_log_level</b>	Verbosity of the python-requests logger. Possible values in order of increasing verbosity: <code>critical</code> , <code>error</code> , <code>warning</code> , <code>info</code> , and <code>debug</code> . Default value: <code>error</code>
<b>llm_log_enabled</b>	If enabled, DRCC will log messages to the lightweight log management (LLM) service. Default value: <code>true</code>
<b>llm_log_level</b>	Verbosity of DRCC log messages to the LLM service. Possible values in order of increasing verbosity: <code>critical</code> , <code>error</code> , <code>warning</code> , <code>info</code> , and <code>debug</code> . Default value: <code>error</code>

## DRC Server (DRCS) Settings

One of the DRCS settings is level=required: `server_cname`.

<b>server_cname (REQUIRED)</b>	The cname of the node where DRCS will reside (e.g., <code>c0-0c1s4n0</code> ). DRCS can reside on a login node or any unspecialized service node, but NOT on any boot or SDB nodes. Because this is a required field and no default is provided, a value must be entered.
------------------------------------	---

<b>logging_directory</b>	<p>Storage location for DRCS logs. This can be located anywhere convenient as long as the directory is:</p> <ul style="list-style-type: none"><li>• (required) writeable by root</li><li>• (recommended) persistent between reboots so that the log file can be retrieved in a node-down event</li></ul> <p>Default value: <code>/tmp</code></p>
<b>logging_filename</b>	<p>Name of the log file for DRCS. This name can be anything except a null value.</p> <p>Default value: <code>drcc.log</code></p>
<b>logging_level</b>	<p>Verbosity of the DRCS logger. Possible values in order of increasing verbosity: <code>critical</code>, <code>error</code>, <code>warning</code>, <code>info</code>, and <code>debug</code>.</p> <p>Default value: <code>error</code></p>
<b>port</b>	<p>TCP port on which the DRC server will listen to requests. Do not assign this port to any other TCP service.</p> <p>Default value: <code>4000</code></p>
<b>use_ssl</b>	<p>Determines whether the DRCS server uses SSL. This additional layer of security is not necessary but is recommended.</p> <p>Default value: <code>false</code></p>
<b>rpc_uri</b>	<p>Remote procedure call (RPC) URI used by both client and server to correctly address DRCS services.</p> <p>Default value: <code>json-rpc</code></p>
<b>werkzeug_log_level</b>	<p>Verbosity of the python-werkzeug logger. Possible values in order of increasing verbosity: <code>critical</code>, <code>error</code>, <code>warning</code>, <code>info</code>, and <code>debug</code>.</p> <p>Default value: <code>error</code></p>
<b>jsonrpc_log_level</b>	<p>Verbosity of the python-jsonrpc logger. Possible values in order of increasing verbosity: <code>critical</code>, <code>error</code>, <code>warning</code>, <code>info</code>, and <code>debug</code>.</p> <p>Default value: <code>error</code></p>
<b>requests_log_level</b>	<p>Verbosity of the python-requests logger. Possible values in order of increasing verbosity: <code>critical</code>, <code>error</code>, <code>warning</code>, <code>info</code>, and <code>debug</code>.</p> <p>Default value: <code>error</code></p>
<b>authorized_uids</b>	<p>List of UIDs that are allowed to interface directly with DRCS through DRCC, DRCCLI, and DRCJEDi (DRC job expiration director). If DRCC, DRCCLI, or DRCJEDi is run under a UID that is not in this list, any request made by that user will be rejected.</p> <p>Default value: <code>[ '0' ]</code></p>
<b>admin_uids</b>	<p>List of UIDs that are allowed to run DRCCLI. At present, this is limited to the values in the <code>authorized_uids</code> list.</p> <p>Default value: <code>[ '0' ]</code></p>
<b>cookie_provider</b> <b>(DEPRECATED)</b>	<p>Required in releases prior to CLE 6.0.UP04; no longer needed.</p>

<b>llm_log_enabled</b>	If enabled, DRCS will log messages to the lightweight log management (LLM) service. Default value: <code>true</code>
<b>llm_log_level</b>	Verbosity of DRCS log messages to the LLM service. Possible values in order of increasing verbosity: <code>critical</code> , <code>error</code> , <code>warning</code> , <code>info</code> , and <code>debug</code> . Default value: <code>error</code>
<b>database_directory (DEPRECATED)</b>	Used for persistent storage configuration in releases prior to CLE 6.0.UP04; no longer needed.
<b>database_filename (DEPRECATED)</b>	Used in releases prior to CLE 6.0.UP04; no longer needed.

## DRC Database Settings

Two database settings are level=required: username and password.

DRC now uses the MariaDB (mysql) database on the SDB node instead of using SQLite3. With SQLite3, sites were required to configure persistent storage for DRC by defining a DRC mount point in `cray_persistent_data` and by setting `database_directory` and `database_filename` (both now deprecated) in `cray_drc`. That is no longer necessary. To configure the DRC mysql database, only the following settings are needed. The username and password must be changed when configuring the `cray_drc` config service.

<b>username (REQUIRED)</b>	Name of the database user, which DRC uses to connect to the MariaDB database. Change this from the default value. Default value: <code>drc</code>
<b>password (REQUIRED)</b>	Password for the database user, which DRC uses to connect to the database. Change this from the default value. Default value: <code>drc</code>
<b>hostname</b>	Host name of the node running the DRC database instance. Default value: <code>sdb</code>
<b>name</b>	Name of the DRC database. Default value: <code>drc</code>

### 4.5.1.7 Update `cray_drc` Worksheet

## Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

## About this task

The Cray dynamic RDMA credentials (DRC) service configures dynamic RDMA (remote direct memory access) credentials, which are secure network credentials that can be shared between user applications to achieve



intercommunication between applications running in different job reservations. This procedure configures some basic settings in the `cray_drc` configuration worksheet to add site-specific data.

This service is disabled by default. For additional information about Cray DRC to help decide whether to enable it and know what configuration parameters are available, see [About Configuring Cray Dynamic RDMA Credentials \(DRC\)](#) on page 175. To enable and use this service, follow these steps.

**NOTICE:** Do not use DRC with VMDH (virtual memory domain handle).

## Procedure

1. Edit `cray_drc_worksheet.yaml`.

```
smw# vi cray_drc_worksheet.yaml
```

2. Uncomment `cray_drc.enabled`.

- To disable this service, set to false and skip the rest of the procedure.
- To enable this service, set to true and continue to the next step.

3. Uncomment `cray_drc.settings.server.data.server_cname` and set it to the cname of the service node that should be running the DRC server.

4. Configure the DRC database.

- a. Change the username. and password.

Uncomment the following two settings and change their values from the defaults. These settings are currently of type string, so do not enter an encrypted value for the password setting.

```
#cray_drc.settings.database.data.username: new_username
#cray_drc.settings.database.data.password: new_password
```

- b. (Optional) Uncomment `cray_drc.settings.database.data.name` and change its value if this site wishes to name the DRC database something other than `drc`.

5. Go back and uncomment the following settings, and set them in accordance with site preferences.

Cray recommends configuring these settings so that diagnostic information is available if needed. Using persistent storage for the logging directories is best; however that depends on available storage space.

```
cray_drc.settings.client.data.logging_directory
cray_drc.settings.client.data.logging_filename
cray_drc.settings.server.data.logging_directory
cray_drc.settings.server.data.logging_filename
```

If this system uses ALPS, Cray recommends increasing the maximum number of user protection domains when DRC is in use, especially for large systems. That parameter is set in the Cray ALPS service with the `pDomainMax` field. See [Update cray\\_alps Worksheet](#) on page 169.

### 4.5.1.8 Update `cray_dvs` Worksheet

#### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

#### About this task

Cray DVS (Data Virtualization Service) is a distributed network service that projects local file systems resident on I/O nodes or remote file servers to compute and service nodes within the Cray system. DVS provides a highly scalable mechanism to share file systems to a large number of client nodes using a fanout tree as configured in `cray_scalable_services`. This service must be enabled if Programming Environment (PE) software is to be used on compute and login nodes. It is also required if netroot image roots will be used on compute and login nodes (netroot is a mechanism that enables nodes booted with a minimal, local in-memory file system to execute within the context of a larger, full-featured root file system.).

This procedure enables the `cray_dvs` configuration service.

#### Procedure

1. Edit `cray_dvs_worksheet.yaml`.

```
smw# vi cray_dvs_worksheet.yaml
```

2. Uncomment `cray_dvs.enabled` and set it to `true`.

Enabling the DVS configuration service is sufficient for a fresh install. Sites that plan to use DVS to project an external file system or provide access to DataWarp will need to configure other `cray_dvs` settings. For more information, see *XC™ Series DVS Administration Guide (S-0005)*.

DVS uses the LNet (Lustre networking) networking layer, so ensure that `cray_lnet` is enabled as well. See [Update `cray\_lnet` Worksheet](#) on page 182.

### 4.5.1.9 Update `cray_image_binding` Worksheet

#### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

#### About this task

The Cray image binding service is a mechanism for mixing image content between booted IMPS (Image Management and Provisioning System) images and IMPS images that are projected onto a running system. This is a common scenario with the Cray Programming Environment (PE), which is installed into an IMPS image, pushed to the CLE boot node, then projected to compute nodes using DVS (Data Virtualization Service). The diagnostics (diag) image root is also pushed to the boot node and projected by DVS.

This procedure enables the `cray_image_binding` service but does not enable any bind mount profiles: those are enabled later in the process.

## Procedure

1. Edit `cray_image_binding_worksheet.yaml`.

```
smw# vi cray_image_binding_worksheet.yaml
```

2. Uncomment `cray_image_binding.enabled` and set it to `true`.
3. Uncomment the pre-populated PE and diags image binding profile settings.

Uncomment all of the following commented settings.

**IMPORTANT:** Do not enable any bind mount profiles now (the `enabled` fields should be set to `false`). Enabling them must wait until the associated image root has been pushed to the boot node. If the system has not been previously booted, failure to push an image root prior to enabling its bind mount profile may prevent the system from properly booting.

```
# ** 'profiles' DATA **

cray_image_binding.settings.profiles.data.profile_name.PE_x86_64: null
#cray_image_binding.settings.profiles.data.PE_x86_64.image:
#  pe_cle_6.0.up07_sles_12sp3_x86-64
cray_image_binding.settings.profiles.data.PE_x86_64.bind_directories: []
#cray_image_binding.settings.profiles.data.PE_x86_64.callbacks:
#- opt/cray/pe/bin/pe_postmount_callback.sh
#cray_image_binding.settings.profiles.data.PE_x86_64.cleanups:
#- opt/cray/pe/bin/pe_cleanup_callback.sh
#cray_image_binding.settings.profiles.data.PE_x86_64.enabled: false
#cray_image_binding.settings.profiles.data.PE_x86_64.client_groups:
#- login_nodes_x86_64
#- compute_nodes_x86_64

cray_image_binding.settings.profiles.data.profile_name.PE_aarch64: null
#cray_image_binding.settings.profiles.data.PE_aarch64.image:
#  pe_cle_6.0.up07_sles_12sp3_aarch64
cray_image_binding.settings.profiles.data.PE_aarch64.bind_directories: []
#cray_image_binding.settings.profiles.data.PE_aarch64.callbacks:
#- opt/cray/pe/bin/pe_postmount_callback.sh
#cray_image_binding.settings.profiles.data.PE_aarch64.cleanups:
#- opt/cray/pe/bin/pe_cleanup_callback.sh
#cray_image_binding.settings.profiles.data.PE_aarch64.enabled: false
#cray_image_binding.settings.profiles.data.PE_aarch64.client_groups:
#- login_nodes_aarch64
#- compute_nodes_aarch64

cray_image_binding.settings.profiles.data.profile_name.diags_x86_64: null
#cray_image_binding.settings.profiles.data.diags_x86_64.image:
#  diags_cle_6.0.up07_sles_12sp3_x86-64
cray_image_binding.settings.profiles.data.diags_x86_64.bind_directories:
- /opt/cray/diag
#cray_image_binding.settings.profiles.data.diags_x86_64.callbacks: []
#cray_image_binding.settings.profiles.data.diags_x86_64.cleanups: []
#cray_image_binding.settings.profiles.data.diags_x86_64.enabled: false
#cray_image_binding.settings.profiles.data.diags_x86_64.client_groups:
#- login_nodes_x86_64
```

```
#- compute_nodes_x86_64

cray_image_binding.settings.profiles.data.profile_name.diaqs_aarch64: null
#cray_image_binding.settings.profiles.data.diaqs_aarch64.image:
  diaqs_cle_6.0.up07_sles_12sp3_aarch64
cray_image_binding.settings.profiles.data.diaqs_aarch64.bind_directories:
- /opt/cray/diag
#cray_image_binding.settings.profiles.data.diaqs_aarch64.callbacks: []
#cray_image_binding.settings.profiles.data.diaqs_aarch64.cleanups: []
#cray_image_binding.settings.profiles.data.diaqs_aarch64.enabled: false
#cray_image_binding.settings.profiles.data.diaqs_aarch64.client_groups:
#- login_nodes_aarch64
#- compute_nodes_aarch64
```

At this point in the fresh install process, it is sufficient to uncomment those settings and leave the default values unchanged. Later procedures will return to those settings to customize the configuration of the PE and diaqs profiles for this system.

**IMPORTANT:** For a fresh install, ensure that all of these profiles are disabled, that is, the `enabled` field is set to `false`.

#### 4.5.1.10 Update `cray_lnet` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

LNet (Lustre networking) is the networking layer used by Lustre and DVS. The Cray LNet configuration service must be configured on any systems that use DVS to mount external file systems or have Lustre clients and/or servers.

This procedure configures some basic settings in the `cray_lnet` configuration worksheet to add site-specific data.

### Procedure

1. Edit `cray_lnet_worksheet.yaml`.

```
smw# vi cray_lnet_worksheet.yaml
```

2. Uncomment `cray_lnet.enabled` and do one of the following:
  - Set it to `true` if this system has external Lustre or DAL (direct-attached Lustre) or will use DVS to mount external file systems.
  - Set it to `false` otherwise.

For systems with external Lustre, continue to the next step. Otherwise, Cray LNet configuration is complete for a fresh install.

---

THE REMAINING STEPS ARE ONLY FOR SYSTEMS WITH EXTERNAL LUSTRE

See also the *XC™ Series Lustre® Administration Guide (S-2648)*.

### 3. Configure the following settings.

These settings are commonly configured with site-specific data when the system has external Lustre. Uncomment and set them as appropriate for this site.

```
cray_lnet.settings.ko2iblnd.data.peer_credits
cray_lnet.settings.ko2iblnd.data.concurrent_sends
cray_lnet.settings.local_lnet.data.lnet_name (set to something like gni, gni1, gni2, gni3)
cray_lnet.settings.local_lnet.data.ip_wildcard (change from default on a partitioned
system or any system that changes the HSN (high speed network) address range)
```

### 4. Configure the following settings if this system has an external Lustre file system with Intel Omni-Path Host Fabric Interface (HFI).

Uncomment these settings and replace the default values with the values shown. In the case of the `fmr_cache` setting, keep the default value of 1.

```
cray_lnet.settings.ko2iblnd.data.peer_credits_hiw: 64
cray_lnet.settings.ko2iblnd.data.map_on_demand: 32
cray_lnet.settings.ko2iblnd.data.fmr_pool_size: 2048
cray_lnet.settings.ko2iblnd.data.fmr_flush_trigger: 512
cray_lnet.settings.ko2iblnd.data.fmr_cache: 1
```

### 5. Configure the following group of settings if this system uses flat routes to an external Lustre file system. Repeat this step for each external Lustre file system.

Enter all external LNet that will be reached via flat routing. The information entered for each of these flat LNet will be used to set up ip2nets on the routers and routes to reach the external LNet through the routers on the clients.

In the worksheet, copy the six lines below # \*\* EXAMPLE 'flat\_routes' VALUE (with current defaults) \*\* and paste them below # NOTE: Place additional 'flat\_routes' setting entries here, if desired.

```
# ** EXAMPLE 'flat_routes' VALUE (with current defaults) **
# cray_lnet.settings.flat_routes.data.dest_lnet.sample_key_a: null <-- setting a multival key
# cray_lnet.settings.flat_routes.data.sample_key_a.dest_lnet_ip_wildcard: ''
# cray_lnet.settings.flat_routes.data.sample_key_a.router_groups: []
# cray_lnet.settings.flat_routes.data.sample_key_a.src_lnet: ''
# cray_lnet.settings.flat_routes.data.sample_key_a.ko2iblnd_peer_credits: 126
# cray_lnet.settings.flat_routes.data.sample_key_a.ko2iblnd_concurrent_sends: 63
#
# ** 'flat_routes' FIELD SPECIFICATION -- MULTIVAL KEY FIELD **
```

Uncomment the lines, replace `sample_key_a` with the name of the LNet on the external Lustre file system (`o2ib` in this example) in all lines, and remove the `<-- setting a multival key` text at the end of the first line (note that the `null` value is required; do not remove or change it). Finally, modify the values as appropriate for this site.

This example uses "o2ib" as the name for this destination LNet and has a `dest_lnet_ip_wildcard` of `10.149.*.*`.

**IMPORTANT:** The settings for `peer_credits` and `concurrent_sends` must match between the external Lustre server and `cray_lnet.settings.flat_routes.data.o2ib.ko2iblnd_peer_credits` and `cray_lnet.settings.flat_routes.data.o2ib.ko2iblnd_concurrent_sends`.

Note that the `cray_lnet.settings.flat_routes.data.o2ib.dest_lnet_ip_wildcard` setting is the IP address wildcard that matches the IP addresses of all router interfaces for this flat route. For example, for a flat route from CLE

clients to an external Lustre file system, the destination LNet might be 'o2ib', and the wildcard '10.149.\*.\*' would match the IB interfaces on the router nodes that are to be on the 'o2ib' LNet.

```
# NOTE: Place additional 'flat_routes' setting entries here, if desired.
cray_lnet.settings.flat_routes.data.dest_lnet.o2ib: null
cray_lnet.settings.flat_routes.data.o2ib.dest_lnet_ip_wildcard: 10.149.*.*
cray_lnet.settings.flat_routes.data.o2ib.router_groups:
- lnet_flat_routers
cray_lnet.settings.flat_routes.data.o2ib.src_lnet: gni2
cray_lnet.settings.flat_routes.data.o2ib.ko2iblnd_peer_credits: 63
cray_lnet.settings.flat_routes.data.o2ib.ko2iblnd_concurrent_sends: 63
# ***** END Service Setting: flat_routes *****
```

For the flat routes router\_groups setting, if there are no existing node groups that contain the router nodes for this site, create one or more node groups for this purpose (*lnet\_flat\_routers* in this example) using the procedure in [Update cray\\_node\\_groups Worksheet](#) on page 151 and reference the node group(s) in `cray_lnet.settings.flat_routes.data.o2ib.router_groups`.

6. Configure the following group of settings if this system uses fine-grained routing (FGR) to an external Lustre file system. Repeat this step for each external Lustre file system.

Enter all external LNetS that will be reached via FGR. The information entered for each of these FGR routes will be used to set up ip2nets on the routers and routes to reach the external LNetS through the routers on the clients.

In the worksheet, copy the six lines below # \*\* EXAMPLE 'fgr\_routes' VALUE (with current defaults) \*\* and paste them below # NOTE: Place additional 'fgr\_routes' setting entries here, if desired.

```
# ** EXAMPLE 'fgr_routes' VALUE (with current defaults) **
# cray_lnet.settings.fgr_routes.data.dest_name.sample_key_a: null <-- setting a multival key
# cray_lnet.settings.fgr_routes.data.sample_key_a.router_groups: []
# cray_lnet.settings.fgr_routes.data.sample_key_a.ip2nets file: ''
# cray_lnet.settings.fgr_routes.data.sample_key_a.routes_file: ''
# cray_lnet.settings.fgr_routes.data.sample_key_a.ko2iblnd_peer_credits: 126
# cray_lnet.settings.fgr_routes.data.sample_key_a.ko2iblnd_concurrent_sends: 63
#
# ** 'fgr_routes' FIELD SPECIFICATION -- MULTIVAL KEY FIELD **
```

Uncomment the lines, replace `sample_key_a` with the name of the external Lustre file system to which you are routing (*sonexion* in this example) in all lines, and remove the `<-- setting a multival key` text at the end of the first line (note that the null value is required; do not remove or change it). Finally, modify the values as appropriate for this site.

This example uses "sonexion" as the name for this destination LNet.

**IMPORTANT:** The settings for `peer_credits` and `concurrent_sends` must match between the external Lustre server and `cray_lnet.settings.fgr_routes.data.sonexion.ko2iblnd_peer_credits` and `cray_lnet.settings.fgr_routes.data.sonexion.ko2iblnd_concurrent_sends`.

```
# NOTE: Place additional 'fgr_routes' setting entries here, if desired.
cray_lnet.settings.fgr_routes.data.dest_name.sonexion: null
cray_lnet.settings.fgr_routes.data.sonexion.router_groups:
- lnet_fgr_routers
cray_lnet.settings.fgr_routes.data.sonexion.ip2nets file: 'ip2nets.conf'
cray_lnet.settings.fgr_routes.data.sonexion.routes_file: 'routes.conf'
cray_lnet.settings.fgr_routes.data.sonexion.ko2iblnd_peer_credits: 126
cray_lnet.settings.fgr_routes.data.sonexion.ko2iblnd_concurrent_sends: 63
# ***** END Service Setting: fgr_routes *****
```

To use fine grained routing, the two configuration files (`ip2nets.conf` and `routes.conf`) must be generated using an external tool, such as `clcv`, and then placed in `/var/opt/cray/imps/config/sets/p0/files/roles/lnet` (for p0 config set).

For the fine-grained routes `router_groups` setting, if there are no existing node groups that contain the router nodes for this site, create one or more node groups for this purpose (`lnet_fgr_routers` in this example) using the procedure in [Update cray\\_node\\_groups Worksheet](#) on page 151 and reference the node group(s) in `cray_lnet.settings.fgr_routes.data.sonexion.router_groups`.

There may be additional settings that should be set for sites with external Lustre servers. Seek advice from the site Lustre server administrator.

#### 4.5.1.11 Update cray\_local\_users Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

The Cray Local Users Configuration Service defines local user accounts such as `root` and `crayadm`. At a minimum, the `root` user account must be defined in order to log into the system.

Most sites use an external LDAP or NIS service for account management. The accounts listed in this configuration service are local accounts with entries in the `/etc/passwd` and `/etc/group` files on CLE nodes. Their home directories can be on the boot RAID file system `/cray_home`, such as for `crayadm`, or on an external file system. Most accounts using LDAP or NIS will have an external home directory mounted on a service node (or nodes) that are DVS-projected to the login and compute nodes.

This procedure configures some basic settings in the `cray_local_users` configuration worksheet to add site-specific data.

### Procedure

1. Edit `cray_local_users_worksheet.yaml`.

```
smw# vi cray_local_users_worksheet.yaml
```

2. Ensure that `cray_local_users.enabled` is uncommented and set to `true` (it should be by default).
3. If using local home directories (most sites mount an external home file system instead), configure the home directory location in two places in this worksheet.

Note that the directory specified for the two `cray_local_users` settings must match the value of the following setting in the `cray_bootraid` configuration service in the global config set, which is the home volume mount point of the boot node volume group:

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.volumes.home.fs_mount_point
```

- a. Uncomment this line. Replace `/cray_home` with the local home directory specified in the `cray_bootraid` setting, if different.

```
#cray_local_users.settings.directories.data.home: /cray_home
```

- b. Change the home directory for `crayadm` users.

Look for this line in the worksheet:

```
# ** 'users' DATA **
```

Underneath, there are pre-populated `users` settings for `crayadm` and `root`.

Change the value of the `crayadm` home directory. Replace `/cray_home` in the following line with the local home directory specified in the `cray_bootstrap` setting and the previous substep. This line is already uncommented.

```
cray_local_users.settings.users.data.crayadm.home: /cray_home/crayadm
```

4. Do nothing to the `crayadm` and `root` accounts' `crypt` settings (in the pre-populated `users` data section).

Do not set the `crayadm` and `root` accounts' `crypt` settings in the pre-populated `users` data section, which must be an encrypted string. Later in this process, all of the configuration worksheets will be imported into the new CLE config set, and the config set will be updated. During the update, the configurator will prompt for those `crypt` settings. Because they are encrypted, the configurator will ask for the password, ask a second time to verify that they match, and then put an encrypted form of that password into the config set. Attempting to place an encrypted string into this worksheet manually is prone to error and could result in accounts that cannot be accessed.

5. Ensure that the `root` domain groups are uncommented.

This parameter is also in the pre-populated `users` settings under this line in the worksheet:

```
# ** 'users' DATA **
```

```
cray_local_users.settings.users.data.root.domain_groups
- all_nodes
```

Make sure both lines are uncommented.

#### 4.5.1.12 Update `cray_login` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

The Login service provides information and listings for login nodes, which are used by users to access the Cray system. Also, the "nologin" feature is configured in this service. This procedure configures some basic settings in the Cray Login service configuration worksheet to add site-specific data.

### Procedure

1. Edit `cray_login_worksheet.yaml`.



```
smw# vi cray_login_worksheet.yaml
```

2. Uncomment `cray_login.enabled` and set it to `true`.
3. Enter the node group (or groups) of the login nodes on this system.

Cray has provided two pre-populated node groups called `login_nodes_x86_64` and `login_nodes_aarch64` to contain the login nodes (by cname) for this system. If those node groups have not yet been customized for this system, see [Update cray\\_node\\_groups Worksheet](#) on page 151.

Uncomment `cray_login.settings.login_nodes.data.member_groups` and the lines that follow it, which list the pre-populated node groups for login nodes.).

```
cray_login.settings.login_nodes.data.member_groups:
- login_nodes_x86_64
- login_nodes_aarch64
```

4. Change the eLogin groups setting.

This setting is necessary for the correct operation of SSH on eLogin nodes.

Uncomment these two lines:

```
#cray_login.settings.login_nodes.data.elogin_groups:
#- elogin_nodes
```

Cray has provided a pre-populated node group called `elogin_nodes` to contain the eLogin nodes for the system. If that node group has not yet been customized for this system, see [Update cray\\_node\\_groups Worksheet](#) on page 151.

- If this system does NOT have eLogin nodes, remove the second line and add an empty list (`[]`) at the end of the first line.

```
cray_login.settings.login_nodes.data.elogin_groups: []
```

- If this system has eLogin nodes, and the node group `elogin_nodes` has been or will be customized to specify ALL elogin nodes for this system, nothing else needs to be done.

```
cray_login.settings.login_nodes.data.elogin_groups:
- elogin_nodes
```

- If this system has eLogin nodes, and a custom node group with a different name has been defined that specifies ALL eLogin nodes (see caveat at end of step), substitute the name of that node group for `elogin_nodes` on the second line.

```
cray_login.settings.login_nodes.data.elogin_groups:
- my_elogin_nodes
```

- If this system has eLogin nodes, and this site uses one or more custom node groups in addition to the default node group `elogin_nodes` to specify ALL eLogin nodes for this system (see caveat at end of step), add the name of the custom node group(s) on separate lines (include the space and hyphen on each line).

```
cray_login.settings.login_nodes.data.elogin_groups:
- elogin_nodes
- my_elogin_nodes
```

**Caveat regarding custom node group(s) for eLogin nodes.** For systems with eLogin nodes, Cray recommends using the `elogin_nodes` node group instead of creating custom node groups for eLogin

nodes. Several other configuration services have settings with `elogin_nodes` as the default value, so if this site decides to use a custom node group for all eLogin nodes, the name of the custom node group must be substituted for `elogin_nodes` in those settings. To find which settings are affected:

```
smw# cfgset search -t elogin_nodes -l advanced -S all p0
smw# cfgset search -t elogin_nodes -l advanced -S all global
```

5. Uncomment `cray_login.settings.login_nodes.data.login_prohibited_after_boot` and do one of the following:

- Set it to `false` to have the `/etc/nologin` file removed automatically on each node in the list of login node groups (set in step 3) as it completes its boot.
- Set it to `true` to require a system administrator to remove `/etc/nologin` on each node by running a command like the following after all of the CLE nodes have been booted and the system is ready for users to log in. This command could be added to the boot automation file.

```
sdb# pcmd -r -n ALL_SERVICE "rm /etc/nologin"
```

#### 4.5.1.13 Update `cray_lustre_client` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

The Cray Lustre Client configuration service is used to configure Lustre clients on an XC system. This procedure configures some basic settings in the `cray_lustre_client` configuration worksheet to add site-specific data.

### Procedure

1. Edit `cray_lustre_client_worksheet.yaml`.

```
smw# vi cray_lustre_client_worksheet.yaml
```

2. Uncomment `cray_lustre_client.enabled` and do one of the following:

- Set it to `false` for systems that are NOT a Lustre client of either an external Lustre server or direct-attached Lustre (DAL). Skip the rest of the procedure.
- Set it to `true` for systems that are a Lustre client of either an external Lustre server or DAL. Proceed to the next step.

3. Configure a client mount for each Lustre file system that will be mounted.

Repeat this step for each client mount.

In the worksheet, copy the lines below `# ** EXAMPLE 'client_mounts' VALUE` (with current defaults) `**` and paste them below `# NOTE: Place additional 'client_mounts' setting entries here, if desired.`

```
# ** EXAMPLE 'client_mounts' VALUE (with current defaults) **
# cray_lustre_client.settings.client_mounts.data.fs_name.sample_key_a: null <-- setting a multival
key
# cray_lustre_client.settings.client_mounts.data.sample_key_a.lustre_fs_name: ''
# cray_lustre_client.settings.client_mounts.data.sample_key_a.mount_point: ''
# cray_lustre_client.settings.client_mounts.data.sample_key_a.mgs_lnet_nids: []
# cray_lustre_client.settings.client_mounts.data.sample_key_a.mount_options: rw,flock,lazystatfs
# cray_lustre_client.settings.client_mounts.data.sample_key_a.mount_at_boot: true
# cray_lustre_client.settings.client_mounts.data.sample_key_a.client_groups:
# - login_nodes_x86_64
# - login_nodes_aarch64
# - compute_nodes
# - elogin_nodes
#
```

Uncomment the lines, replace `sample_key_a` with a string that identifies that mount (`snx11023` in the example below), then remove the `<-- setting a multival key` text at the end of the first line in each set (note that the `null` value is required; do not remove or change it). Finally, modify the values as needed for this site.

### Example of mounting an external Lustre file system.

This example uses a Sonexion with a file system called `snx11023` that is mounted on `/lus/snx11023` by several node groups. All of them mount the file system as the node is booting.

```
# NOTE: Place additional 'client_mount' setting entries here, if desired.
cray_lustre_client.settings.client_mounts.data.fs_name.snx11023: null
cray_lustre_client.settings.client_mounts.data.snx11023.lustre_fs_name: snx11023
cray_lustre_client.settings.client_mounts.data.snx11023.mount_point: /lus/snx11023
cray_lustre_client.settings.client_mounts.data.snx11023.mgs_lnet_nids:
- 10.149.4.3@o2ib
- 10.149.4.4@o2ib
cray_lustre_client.settings.client_mounts.data.snx11023.mount_options: rw,flock,lazystatfs
cray_lustre_client.settings.client_mounts.data.snx11023.mount_at_boot: true
cray_lustre_client.settings.client_mounts.data.snx11023.client_groups:
- login_nodes_x86_64
- login_nodes_aarch64
- compute_nodes
- elogin_nodes

#***** END Service Setting: client_mounts *****
```

### Example of mounting an internal Lustre file system (DAL).

The following example shows values for two DAL client mounts: one for login nodes (first set of lines) and one for compute nodes (second set of lines).

The Lustre file system is started via `lustre_control` in the boot automation file after all service nodes have booted. The boot automation file then has a step to mount the Lustre file system on any service nodes that need to mount it. So those service nodes cannot have "mount\_at\_boot" set to true. However, the compute nodes are booted after the DAL file system has been started, so they can have "mount\_at\_boot" set to true. This example uses a DAL server with a file system called `dal` that is mounted on `/lus/dal` by several node groups. Only the compute nodes mount the file system as the node is booting (`mount_at_boot=true`).

```
# NOTE: Place additional 'client_mount' setting entries here, if desired.
cray_lustre_client.settings.client_mounts.data.fs_name.dal_login: null
cray_lustre_client.settings.client_mounts.data.dal_login.lustre_fs_name: dal
cray_lustre_client.settings.client_mounts.data.dal_login.mount_point: /lus/dal
cray_lustre_client.settings.client_mounts.data.dal_login.mgs_lnet_nids:
- 27@gni
- 29@gni
cray_lustre_client.settings.client_mounts.data.dal_login.mount_options: rw,flock,lazystatfs
cray_lustre_client.settings.client_mounts.data.dal_login.mount_at_boot: false
cray_lustre_client.settings.client_mounts.data.dal_login.client_groups:
- login_nodes_x86_64
- login_nodes_aarch64
- elogin_nodes

cray_lustre_client.settings.client_mounts.data.fs_name.dal_compute: null
cray_lustre_client.settings.client_mounts.data.dal_compute.lustre_fs_name: dal
```

```
cray_lustre_client.settings.client_mounts.data.dal_compute.mount_point: /lus/dal
cray_lustre_client.settings.client_mounts.data.dal_compute.mgs_lnet_nids:
- 27@gni
- 29@gni
cray_lustre_client.settings.client_mounts.data.dal_compute.mount_options: rw,flock,lazystatfs
cray_lustre_client.settings.client_mounts.data.dal_compute.mount_at_boot: true
cray_lustre_client.settings.client_mounts.data.dal_compute.client_groups:
- compute_nodes_x86_64
- compute_nodes_aarch64

#***** END Service Setting: client_mounts *****
```

4. Verify that the node groups used in previous steps have been accurately defined for this site.

To verify, edit `cray_node_groups_worksheet.yaml` and search for these node groups:

```
login_nodes_x86_64
login_nodes_aarch64
compute_nodes
compute_nodes_x86_64
compute_nodes_aarch64
ellogin_nodes
```

DAL servers also need to be configured. See [Update cray\\_lustre\\_server Worksheet](#) on page 190. Further configuration of DAL occurs later in the installation process.

#### 4.5.1.14 Update cray\_lustre\_server Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

The Cray Lustre server configuration service should be enabled and configured only if this system uses direct-attached Lustre (DAL). It enables configuration of Lustre-server-related kernel module parameters. This procedure configures some basic settings in the `cray_lustre_server` configuration worksheet to add site-specific data.

### Procedure

1. Edit `cray_lustre_server_worksheet.yaml`.

```
smw# vi cray_lustre_server_worksheet.yaml
```

2. Uncomment `cray_lustre_server.enabled` and do one of the following:
  - Set it to `false` for system that do not use DAL (direct-attached Lustre). Skip the remaining steps.
  - Set it to `true` for systems that do use DAL. Proceed to the next step.
3. (Only for systems with DAL) Enter the node group that contains the Lustre Management Server (MGS) node on this system.

To see which node group contains the MGS node (by cname) or to create such a node group for this system (*MGS\_NODE\_GROUP* in this example), edit `cray_node_groups_worksheet.yaml`.

Uncomment `cray_lustre_server.settings.lustre_servers.data.mgs_group`, remove the empty list (`[]`), and add that node group on a separate line prefixed by a hyphen and space (`-` ).

```
cray_lustre_server.settings.lustre_servers.data.mgs_group:
- MGS_NODE_GROUP
```

4. (Only for systems with DAL) Enter the node group(s) that contain the Lustre MetaData Server (MDS) nodes on this system.

To see which node group(s) contain the MDS nodes (by cname) or to create that node group(s) for this system (*MDS\_NODE\_GROUP\_1* and *MDS\_NODE\_GROUP\_2* in this example), edit `cray_node_groups_worksheet.yaml`.

Uncomment `cray_lustre_server.settings.lustre_servers.data.mds_groups`, remove the empty list (`[]`), and add the node group(s) on a separate line prefixed by a hyphen and space (`-` ).

```
cray_lustre_server.settings.lustre_servers.data.mds_groups:
- MDS_NODE_GROUP_1
- MDS_NODE_GROUP_2
```

5. (Only for systems with DAL) Enter the node group(s) that contain the Lustre Object Storage Server (OSS) nodes on this system.

To see which node group(s) contain the OSS nodes (by cname) or to create that node group(s) for this system (*OSS\_NODE\_GROUP\_1* and *OSS\_NODE\_GROUP\_2* in this example), edit `cray_node_groups_worksheet.yaml`.

Uncomment `cray_lustre_server.settings.lustre_servers.data.oss_groups`, remove the empty list (`[]`), and add the node group(s) on a separate line prefixed by a hyphen and space (`-` ).

```
cray_lustre_server.settings.lustre_servers.data.oss_groups:
- OSS_NODE_GROUP_1
- OSS_NODE_GROUP_2
```

6. (Only for systems with DAL) Set Lustre kernel module parameters, as needed.

This worksheet contains additional settings that tune the Lustre kernel modules. Seek advice from the site Lustre server administrator before changing them.

#### 4.5.1.15 Update `cray_multipath` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

The Cray multipath service provides a means to support redundant paths to a device for failover or performance reasons. Multipath does NOT need to be fully cabled to be used. The multipath driver can handle using one path or many.

**NOTE:** (SMW HA only) Cray recommends configuring multipath before configuring and enabling HA. If HA is configured and enabled first, then additional precautions must be taken when enabling multipath, as documented in *XC™ Series SMW HA Installation Guide*.

The multipath configuration service has a global template as well as a CLE template, and therefore the service can be configured to inherit settings from the global config set or use settings from the CLE config set(s), if there is a need to have it configured differently in different config sets. If multipath configuration is desired on the management node (SMW) as well as CLE nodes, Cray recommends enabling this service in the global config set and configuring it there for both SMW and CLE nodes. The multipath service in the CLE config sets would then inherit the global configuration data.

This procedure configures the inherit setting and possibly some other settings in the Cray multipath service configuration worksheet in a CLE config set.

## Procedure

1. Edit `cray_multipath_worksheet.yaml`.

```
smw# vi cray_multipath_worksheet.yaml
```

2. Uncomment `cray_multipath.inherit` and set it to one of the following values:

- Set it to `true` to manage multipath settings in the global config set instead of in the CLE config set. If this option is chosen, skip the rest of the steps.
- Set it to `false` to manage multipath settings in one or more CLE config sets instead of in the global config set. If this option is chosen, continue to the next step.

3. (If `inherit` set to `false`) Uncomment `cray_multipath.enabled`.

Set it to `true` if this site desires to use multipath, otherwise set it to `false`. If enabling this service, continue to the next step.

4. (If `enabled` set to `true`) Complete the configuration of multipath.

- a. Enter the list of multipath nodes.

Uncomment `cray_multipath.settings.multipath.data.node_list`, remove the `[]` (denotes empty list), and add a list of nodes (by cname or host ID) in this system that have multipath devices and need to have multipath configured.

This example shows a list of three nodes: an SMW with host ID `1eac4e0c`, a boot node with cname `c0-0c0s0n1`, and an SDB node with cname `c0-0c0s1n1`.

```
cray_multipath.settings.multipath.data.node_list:
- 1eac4e0c
- c0-0c0s0n1
- c0-0c0s1n1
```

**Boot/SDB node failover.** If configuring boot and/or SDB node failover, add both the primary and backup (failover) nodes to this list. This example shows a list of five nodes: an SMW with host ID `1eac4e0c`, a primary boot node with cname `c0-0c0s0n1`, a backup boot node with cname `c0-2c0s4n1`, a primary SDB node with cname `c0-0c0s1n1`, and a backup SDB node with cname `c0-4c0s3n1`.

```
cray_multipath.settings.multipath.data.node_list:
- 1eac4e0c
- c0-0c0s0n1
```

```
- c0-2c0s4n1
- c0-0c0s1n1
- c0-4c0s3n1
```

b. Configure enabled devices.

Cray has provided a number of enabled devices with pre-populated data under # \*\*

'enabled\_devices' DATA \*\*. These storage devices are the devices that will be whitelisted, which means they will be listed as exceptions to the blacklist. The settings for these devices have default values provided by the device vendors and do not need to be changed. If this site intends to configure a multipath device that does not appear in this group of enabled devices, contact a Cray representative for help.

c. (Optional) Configure aliases for the multipath devices.

This is the equivalent of adding aliases to the multipaths section of the `multipath.conf` file. If no aliases are specified, this setting will show as unconfigured when the config set is updated, but this is not a problem. It can remain unconfigured and will not cause the config set to be invalid.

In the worksheet, copy the two lines below # \*\* EXAMPLE 'aliases' VALUE (with current defaults) \*\* and paste them below # NOTE: Place additional 'aliases' setting entries here, if desired.

```
# ** EXAMPLE 'aliases' VALUE (with current defaults) **
#   cray_multipath.settings.aliases.data.wwid.sample_key_a: null    <-- setting a multival key
#   cray_multipath.settings.aliases.data.sample_key_a.alias: ''
#
```

Uncomment the lines, replace `sample_key_a` with the World Wide Identifier (WWID) of the device to be aliased (60080e50002e203c00002a085551b2c8 in this example) in all lines, and remove the <-- setting a multival key text at the end of the first line (note that the null value is required; do not remove or change it). Finally, add the alias for this device (`smw_node_pv1` in this example). Repeat this substep for each device, as needed.

```
# NOTE: Place additional 'aliases' setting entries here, if desired.
cray_multipath.settings.aliases.data.wwid.60080e50002e203c00002a085551b2c8: null
cray_multipath.settings.aliases.data.60080e50002e203c00002a085551b2c8.alias: smw_node_pv1
#***** END Service Setting: aliases *****
```

Note that as of CLE 6.0.UP06, `cray_multipath` supports WWIDs that are 33 characters long, which is needed for DAL Lustre configuration.

#### 4.5.1.16 About Configuring Netroot Preload

Netroot is a feature that enables nodes booted with a minimal, local in-memory file system to execute within the context of a larger, full-featured root file system. Netroot uses the Data Virtualization Service (DVS) to access the remote root content. While DVS has data- and attribute-caching features that minimize the impact of most remote references, files that are referenced frequently may still incur an undesirable performance penalty.

The "netroot preload" feature mitigates that performance penalty by copying specified remote files from the netroot to a node-local in-memory file system early in the node boot process. All future references to those files will be serviced by the local file system rather than requiring remote data and/or metadata DVS operations. This improves system and application performance. However, as a consequence, the amount of memory available on the node is reduced by the cumulative size of all files copied into its memory.

Netroot preload can be enabled, disabled, and customized using the configurator on the SMW or by editing the configuration worksheets on the SMW.

## Netroot Preload Configuration Settings

Netroot preload configuration consists of defining one or more "loads," or sets of data to be preloaded on specified nodes. The load setting has the following fields:

- label** A convenient, descriptive label for a particular load.
- targets** A list of node groups that reference the nodes that will be preloaded with files on their local file systems. Must provide at least one node group.
- content\_lists** Content lists are relative paths to files within the config set. These files contain file paths that are copied into the node-local memory by netroot preload. For example, content list `dist/compute-preload.cray` within config set `p0` has these contents:

```
smw# cat p0/dist/compute-preload.cray
/etc/ld.so.cache
/opt/cray/rca/*/bin/rca-helper
/lib64/libc-*.so
/lib64/ld-*.so
/opt/cray/rca/*/lib*/librca.so.*
[...]
```

Pattern matching is supported.

- size\_limit** The memory-consumption limit (in MB) set for this load, which limits how much can be copied to any node. As the files are copied via netroot, netroot preload checks the sizes and amount of data copied so far. When it reaches the size limit, it stops, and any remaining files are not copied. Setting this to zero (0) indicates no limit.

## Cray Provides Default Loads

Cray provides two default loads: the 'compute' load, which targets all compute nodes in the system, and the 'login' load, which targets all internal login nodes in the system.

The compute load has a single content list

specified: `/var/opt/cray/imps/config/sets/p0/dist/compute-preload.cray`. This file contains paths that are commonly referenced during the node boot and initialization process.

Similarly, the login load specifies this content list as the only entry in its `content_lists`

setting: `/var/opt/cray/imps/config/sets/p0/dist/login-preload.cray`.

**Do not disable or modify the Cray default loads.** Changes could increase the time it takes to boot and initialize nodes, and any changes will be lost when the config set is next updated. Cray recommends creating custom loads to optimize operations for this site.

## Sites can Create Custom Loads to Optimize for Specific Workloads

Sites may define their own loads as well. This enables sites to optimize for specific workloads. For targets, sites can use existing node groups or define their own (see [Update cray\\_node\\_groups Worksheet](#) on page 151).

To determine which file paths to add to load content lists, use the DVS request log, which is enabled by default. That log was used to create the Cray default content lists. The `/proc/fs/dvs/request_log` file contains a log of all DVS requests that take more than a certain number of seconds to complete (the default is 15 seconds). Look for file paths that are referenced often; these are good candidates for netroot preload.

Use the following commands (as root) to view, disable, enable, and clear the DVS request log.



Table 14. Commands to disable, enable, and clear the DVS request log

view	<code>cat /proc/fs/dvs/request_log</code>
disable	<code>echo 0 &gt; /proc/fs/dvs/request_log</code>
enable	<code>echo 1 &gt; /proc/fs/dvs/request_log</code>
clear	<code>echo 2 &gt; /proc/fs/dvs/request_log</code>

See "DVS Can Log Requests Sent to Servers" in *XC™ Series DVS Administration Guide (S-0005)* for additional information about this request log.

## The Netroot Preload Log File and a Note about Symlinks

Netroot preload creates a log file on affected nodes at `/var/opt/cray/log/netroot_preload.log`. This log file contains details on the files preloaded, which, if any, files were not found in the netroot, and the size of the files preloaded on the node. Any failures will also be logged to the console file on the SMW.

Note that any symlinks included in a load content list may not be copied from netroot to the node-local RAM file system (i.e., "promoted" in the log file), which might look confusing. For example, suppose a site content list contains `/etc/alternatives/unzip`, which is a symlink to `/usr/bin/unzip-plain`. While both the link and its target are present in netroot, neither of them appear in the node-local file system, despite the log saying `Promoted '/new_root/merge/etc/alternatives/unzip'`. This is expected and correct behavior. A site that is concerned about possible confusion for administrators can decide to exclude symlinks from content lists or simply list the target of the symlink in a content list to ensure that it is present in the node-local file system.

### 4.5.1.17 Update `cray_netroot_preload` Worksheet

#### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

#### About this task

Netroot Preload is a mechanism for populating a Cray system node's root file system early in the boot process to reduce load on the DVS (Data Virtualization Service) servers providing the data and thereby reduce boot times for netroot nodes. Netroot Preload also improves post-boot performance—how much improvement depends on the workloads. This service is needed if netroot is used, and does no harm if netroot is not used.

This procedure configures some settings in the `cray_netroot_preload` configuration worksheet to add site-specific "load" data. Cray provides two default load settings that define target nodes and files to be preloaded to them. Sites may define custom loads as well (optional). For more information, see [About Configuring Netroot Preload](#) on page 193.

#### Procedure

1. Edit `cray_netroot_preload_worksheet.yaml`.

```
smw# vi cray_netroot_preload_worksheet.yaml
```

2. Uncomment `cray_netroot_preload.enabled`. Keep it set to `true`, which is the default.

Continue to step 3 to define a custom load (optional).

3. Define a custom load.

In the worksheet, copy the four lines below `# ** EXAMPLE 'load' VALUE` (with current defaults) `**` and paste them below `# NOTE: Place additional 'load' setting entries here, if desired`.

```
# ** EXAMPLE 'load' VALUE (with current defaults) **
#  cray_netroot_preload.settings.load.data.label.sample_key_a: null    <-- setting a multival key
#  cray_netroot_preload.settings.load.data.sample_key_a.targets: []
#  cray_netroot_preload.settings.load.data.sample_key_a.content_lists: []
#  cray_netroot_preload.settings.load.data.sample_key_a.size_limit: 0
```

Uncomment the lines, replace `sample_key_a` with the label for this load (e.g., `my_load`) in all lines, and remove the `<-- setting a multival key` text at the end of the first line (note that the `null` value is required; do not remove or change it). Finally, add site-specific values. Add each list element on a separate line prefixed by a hyphen and space (`-` ).

```
# NOTE: Place additional 'load' setting entries here, if desired.
cray_netroot_preload.settings.load.data.label.my_load: null
cray_netroot_preload.settings.load.data.my_load.targets:
- site-defined_node_group
cray_netroot_preload.settings.load.data.my_load.content_lists:
- relative/path/to/oft-requested/files
cray_netroot_preload.settings.load.data.my_load.size_limit: 0
#***** END Service Setting: load *****
```

#### 4.5.1.18 Update `cray_opa` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

Intel® Omni-Path Architecture (OPA) includes host fabric interfaces, switches, cables, silicon, and management software. Cray uses the Intel Omni-Path Host Fabric Interface (HFI) to improve the LNet performance of Lustre file systems in Cray XC50 systems. An HFI driver is installed on eLogin nodes that mount an external Lustre file system with Omni-Path HFI and on LNet router nodes that provide routes to an external Lustre file system with Omni-Path HFI.

The `cray_opa` configuration service manages selected host fabric interface (`hfi1`) kernel module parameters. It uses Ansible to place the `hfi1` kernel parameters in `/etc/modprobe.d/cray-hfi1.conf` on all eLogin nodes and service nodes, but only the eLogin and service nodes that load the `hfi1` driver (because of the presence of HFI hardware) will be affected by the parameters.

This procedure enables or disables the `cray_opa` configuration service. If enabled, the parameters specified in this service will be placed on all login and service nodes. If disabled, the parameters will not be placed on those nodes.

## Procedure

1. Edit `cray_opa_worksheet.yaml`.

```
smw# vi cray_opa_worksheet.yaml
```

2. Enable or disable `cray_opa`, as needed.

Uncomment `cray_opa.enabled` and do one of the following:

- Ensure that it is set to true (default) if this system mounts an Omni-Path HFI Lustre file system.  
No other changes are necessary. Cray recommends leaving the remaining advanced-level settings commented and set to their default values. If they are left commented out, the configurator marks them as unconfigured, which enables it to update those values with any new defaults provided by Cray in later releases.
- Set it to false otherwise, to disable the service.

This is not essential. It is safe to leave the service enabled for systems without Omni-Path HFI hardware.

If this system uses Omni-Path HFI, ensure the following:

- An InfiniBand network and a host with an InfiniBand network interface have been defined, as described in [Update `cray\_net` Worksheet](#) on page 156 (see the examples in the "Configure additional networks" step and the "Configure additional hosts" step).
- The LNet configuration service has been configured, as described in [Update `cray\_inet` Worksheet](#) on page 182 (see the "Configure the following settings if this system has an external Lustre file system with Intel Omni-Path Host Fabric Interface (HFI)" step).

### 4.5.1.19 Update `cray_persistent_data` Worksheet

## Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

## About this task

The Cray Persistent Data service provides persistent storage to nodes, which can be configured on a per-node basis. This procedure configures some basic settings in the `cray_persistent_data` configuration worksheet to add site-specific data.

**NOTE:** `cray_persistent_data` must be enabled when using boot node failover or SDB node failover.

## Procedure

1. Edit `cray_persistent_data_worksheet.yaml`.

```
smw# vi cray_persistent_data_worksheet.yaml
```

2. Uncomment `cray_persistent_data.enabled` and set it to `true`.
3. Uncomment `cray_persistent_data.settings.directories.data.persistent_space_mount` and set it to match the mount point of a non-volatile volume in the CLE storage set (`cledefault`), which was specified previously.

The setting for that mount point is

`cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.volumes.nvolatile.fs_mount_point`, which is in the `cray_bootraid` service in the global config set. To find its value, use `cfgset search` and scan the list of matches for that setting.

```
smw# cfgset search --service cray_bootraid --level advanced \
--state all --term nvolatile global
```

4. Ensure that these `client_groups` settings are uncommented.

For each setting, uncomment both the variable and its value (the line that follows it, which is a list containing one node group). They should all be set to a list containing the node group `service_nodes`, except for the NFS mount: `nfs.client_groups` should be set to a list containing `boot_nodes` and `sdb_nodes`.

```
#cray_persistent_data.settings.mounts.data./var/opt/cray/alps.client_groups:
#- service_nodes

#cray_persistent_data.settings.mounts.data./var/opt/cray/aeld.client_groups:
#- service_nodes

#cray_persistent_data.settings.mounts.data./var/opt/cray/appterm.d.client_groups:
#- service_nodes

#cray_persistent_data.settings.mounts.data./var/lib/nfs.client_groups:
#- boot_nodes
#- sdb_nodes
```

5. If a workload manager (WLM) will be used, configure space for its spool area by defining a `cray_persistent_data` mount point.

Use these spool file paths as mount points for persistent storage, depending on the WLM used at this site. Note that for Moab/TORQUE, two mount points will need to be defined.

- Moab/TORQUE: `/var/spool/moab` and `/var/spool/torque`
- PBS: `/var/spool/PBS`
- Slurm: `/var/spool/slurm`

In the worksheet, copy the five lines below `# ** EXAMPLE 'mounts' VALUE` (with current defaults) `**` and paste them below `# NOTE: Place additional 'mounts' setting entries here, if desired.`

```
# ** EXAMPLE 'mounts' VALUE (with current defaults) **
# cray_persistent_data.settings.mounts.data.mount_point.sample_key_a: null <-- setting a multival key
# cray_persistent_data.settings.mounts.data.sample_key_a.alt_storage_path: ''
# cray_persistent_data.settings.mounts.data.sample_key_a.options: ''
```

```
# cray_persistent_data.settings.mounts.data.sample_key_a.ancestor_def_perms: '0771'
# cray_persistent_data.settings.mounts.data.sample_key_a.client_groups: []
```

Uncomment the lines, replace `sample_key_a` with one the correct spool file path in all lines, and remove the `<--` setting a multival key text at the end of the first line (note that the `null` value is required; do not remove or change it). For the `client_groups` setting (last line), remove the empty list (`[]`), and add a node group (one that contains the WLM server node) on a separate line prefixed by a hyphen and space (`-` ). To see which node group contains the node with this cname, or to create such a node group for this system (**`NODE_GROUP`** in this example), edit `cray_node_groups_worksheet.yaml`.

Leave all other settings at the default values.

This example shows the Slurm file path as the mount point (`sample_key_a`).

```
# NOTE: Place additional 'mounts' setting entries here, if desired.
cray_persistent_data.settings.mounts.data.mount_point: /var/spool/slurm: null
cray_persistent_data.settings.mounts.data./var/spool/slurm.alt_storage_path: ''
cray_persistent_data.settings.mounts.data./var/spool/slurm.options: ''
cray_persistent_data.settings.mounts.data./var/spool/slurm.ancestor_def_perms: '0771'
cray_persistent_data.settings.mounts.data./var/spool/slurm.client_groups:
- NODE_GROUP

# ***** END Service Setting: mounts *****
```

#### 4.5.1.20 Update `cray_rsisp` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

RSIP (realm-specific IP) helps to maintain packet integrity by allowing an RSIP host to borrow one or more IP addresses from a set of configured RSIP gateways. This procedure configures some simple settings in the Cray RSIP configuration service worksheet to add site-specific data, such as which nodes will be RSIP servers and which will be RSIP clients.

Systems with service nodes that will provide the RSIP service must have RSIP configured. Use this procedure to configure RSIP. The settings in the worksheet should suffice for simple RSIP configuration. For advanced RSIP configuration (e.g., RSIP failover, RSIP pools), only enable or disable `cray_rsisp`; RSIP configuration will be completed later in the process.

### Procedure

1. Edit `cray_rsisp_worksheet.yaml`.

```
smw# vi cray_rsisp_worksheet.yaml
```

2. Uncomment `cray_rsisp.enabled` and set it as follows.

- Option 1: System will NOT use RSIP.
  1. Set `cray_rsisp.enabled` to `false`.

2. Skip the rest of this procedure.
- Option 2: System requires advanced RSIP configuration (e.g., RSIP failover, RSIP pools).
  1. If RSIP will be configured **prior** to XC system boot, set `cray_rsip.enabled` to `true`.
  2. If RSIP will be configured **after** XC system boot, set `cray_rsip.enabled` to `false`. The service must be disabled because if RSIP is enabled but not fully configured, it will cause boot errors.
  3. Skip the rest of this procedure. RSIP configuration will be completed later in the installation/configuration process.
- Option 3: System does NOT require advanced RSIP configuration (the settings in the `cray_rsip` configuration worksheet suffice to configure RSIP for this system).
  1. Set `cray_rsip.enabled` to `true`.
  2. Proceed to the next step.

#### ———— SIMPLE RSIP CONFIGURATION (Option 3) ————

3. Create node groups for the RSIP servers and clients on this system, if they do not already exist.  
See [Update `cray\_node\_groups` Worksheet](#) on page 151 for instructions.
  - a. Create one or more node groups that contain the RSIP server nodes (by cname) for this system.
  - b. Create one or more node groups that contain the RSIP client nodes (by cname) for this system.

4. Enter the node group (or groups) of the service nodes that will be RSIP servers on this system.

Uncomment `cray_rsip.settings.service.data.server_groups`, remove the empty list (`[]`), and add the node group(s) on separate lines prefixed by a hyphen and space (`-` ). The example shows a single node group called `rsip_nodes`.

```
cray_rsip.settings.service.data.server_groups:
- rsip_nodes
```

5. Enter the node group (or groups) of the service nodes that will be RSIP clients on this system, such as a MOM node.

Uncomment `cray_rsip.settings.service.data.node_groups_as_client`, remove the empty list (`[]`), and add the node group(s) on separate lines prefixed by a hyphen and space (`-` ). The example shows a single node group called `rsip_servicenode_clients`.

```
cray_rsip.settings.service.data.node_groups_as_client:
- rsip_servicenode_clients
```

6. Configure other settings, as needed for this system.

#### 4.5.1.21 Update `cray_scalable_services` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

## About this task

Cray Scalable Services defines a tree of servers (nodes), starting with the server of authority (SoA), that are used in the scaling of the system. Configuration of Scalable Services is required for a functioning system. For more information, see [About Cray Scalable Services](#) on page 16. This procedure configures some basic settings in the `cray_scalable_services` configuration worksheet to add site-specific data.

## Procedure

1. Edit `cray_scalable_services_worksheet.yaml`.

```
smw# vi cray_scalable_services_worksheet.yaml
```

2. Uncomment `cray_scalable_services.enabled` and ensure that it is set to `true`.

3. Uncomment `cray_scalable_services.settings.scalable_service.data.server_of_authority` and ensure that it is set to `smw`.

4. Enter the node group (or node groups) of the nodes that will be tier1 servers on this system.

Ensure that these node groups include the `cname` of the boot node and any other nodes that have an Ethernet connection to the SMW. The SDB node should also have a connection to the SMW, so it can be a tier1 server.

**IMPORTANT:** If enabling boot node failover or SDB node failover, ensure that all boot nodes and all SDB nodes are in a tier1 node group and none of them are in a tier2 node group.

Uncomment `cray_scalable_services.settings.scalable_service.data.tier1_groups`, remove the empty list (`[]`), and add these predefined node groups on separate lines prefixed by a hyphen and space (`-` ).

```
cray_scalable_services.settings.scalable_service.data.tier1_groups:
- boot_nodes
- sdb_nodes
- OTHER_TIER1_NODE_GROUP
```

To verify that these node groups contain the tier1 server nodes (by `cname`) for this system, to add the correct tier1 nodes to them, or to add a new node group for tier1 servers, (`OTHER_TIER1_NODE_GROUP` in this example), edit `cray_node_groups_worksheet.yaml` (see [Update cray\\_node\\_groups Worksheet](#) on page 151).

5. Enter the node group (or node groups) of the nodes that will be tier2 servers on this system.

Uncomment `cray_scalable_services.settings.scalable_service.data.tier2_groups` and the line below it, which is a list of one predefined node group.

```
cray_scalable_services.settings.scalable_service.data.tier2_groups:
- tier2_nodes
```

To verify that the predefined tier2 node group contains the correct tier2 server nodes (by `cname`) for this system or to add the correct tier2 nodes to them, edit `cray_node_groups_worksheet.yaml`.

- Q. How many tier2 nodes are needed?** **A.** At least one server must be provided, and for resiliency, two nodes placed on different blades. The recommended ratio of tier2 nodes (servers) to tier3 nodes (clients) is 1 to 400.
- Q. Will adding more tier2 nodes help performance?** **A.** Adding more tier2 nodes does not always yield additional performance and is subject to diminishing returns.
- Q. What kind of node can be used as a tier2 node?** **A.** Use these:
- OPTIMAL: dedicated repurposed compute nodes (RCN)
  - dedicated service nodes
  - nodes with uniform light to moderate load
  - nodes with relatively homogeneous single core speeds to reduce resource contention disparity during periods of partial availability
- AVOID these (will result in sub-optimal performance):
- nodes with slower individual CPU cores, such as Intel® Xeon Phi™ "Knights Landing" (KNL) processors
  - direct-attached Lustre (DAL) servers
  - RSIP (realm-specific IP) servers
  - login nodes
- Q. Can a tier2 node have more than one role?** **A.** Small test and development systems (TDS) may use tier2 nodes that have additional roles, but generally, it is better for tier2 nodes to be dedicated.
- Q. Where should tier2 nodes be placed?** **A.** Distribute nodes throughout the system (on different blades) for resiliency in the event of hardware failure.

Check the guidance for tier2 nodes in this configuration worksheet for additional requirements or limitations.

## 6. Configure external tier1 node groups.

External nodes, such as eLogin nodes, need to use some of the same services as CLE nodes, but they are not included in the Scalable Services structure. Although external nodes can be considered "tier1" because they have a direct network connection to the SMW, they cannot simply be added to the `tier1_groups` setting because it applies to internal CLE nodes only.

To address this, the following setting identifies all external nodes (including eLogin nodes), and the Ansible play for Cray Scalable Services has been revised so that nodes in `external_tier1_groups` behave like the nodes in `tier1_groups` with regard to NTP, LLM, and LiveUpdates.

Uncomment the following two lines. Because this is a level basic setting, it must be configured, even if there are no external nodes in this system. If this system does have external nodes, and there are additional node groups defined for eLogin nodes or other external nodes, add those node groups on separate lines prefixed by a hyphen and space (- ).

```
#cray_scalable_services.settings.scalable_service.data.external_tier1_groups:
#- elogin_nodes
```



#### 4.5.1.22 Update `cray_sdb` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

The Cray Software Database (SDB) service configures the services and settings for the SDB node. This procedure configures some basic settings in the `cray_sdb` service configuration worksheet.

### Procedure

1. Edit `cray_sdb_worksheet.yaml`.

```
smw# vi cray_sdb_worksheet.yaml
```

2. Uncomment `cray_sdb.enabled` and ensure that it is set to `true`.

3. Configure the SDB node groups setting.

- a. Uncomment the SDB node groups setting.

Be sure to uncomment both lines.

```
#cray_sdb.settings.node_groups.data.sdb_groups:
#- sdb_nodes
```

- b. Verify that the `sdb_nodes` node group has been accurately defined for this site.

To verify, edit `cray_node_groups_worksheet.yaml` and search for `sdb_nodes`.

4. Configure the admin and root database passwords.

Uncomment the following two password settings and replace the default values with site-specific values.

These passwords will be stored in clear text in the config set. Note that the values of these passwords are excluded when the config set is distributed to eLogin nodes.

```
#cray_sdb.settings.database.data.db_admin_password: sys_mgmt
#cray_sdb.settings.database.data.db_current_root_password: ''
```

5. (Optional) Set the host for the daemon that syncs the HSS database.

Uncomment this setting to configure it. Cray recommends keeping the default value of `'sdb'`; however, if this site wishes `xtdbsyncd` to run on the boot node instead, change the value to `'boot'`.

```
#cray_sdb.settings.database.data.synchost: sdb
```

### 4.5.1.23 Update `cray_simple_services` Worksheet

## Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

## About this task

The Cray simple services configuration service enables administrators to control basic init(1) processing by enabling/disabling and starting/stopping selected system services on selected node groups. It provides no configuration of those services, only control over their state at boot or whenever the associated Ansible play is run.

This procedure enables the `cray_simple_services` config service and describes how to set up a 'services' entry, which involves identifying the affected nodes, the services to control, and the actions to take (enable/disable/start/stop) in the order given.

## Procedure

1. Edit `cray_simple_services_worksheet.yaml`.

```
smw# vi cray_simple_services_worksheet.yaml
```

2. Uncomment `cray_simple_services.enabled` and leave it set to `true`.

The `cray_simple_services` config service can be enabled even if it is not used to enable or disable system services at this time.

————— CONFIGURE A 'SERVICES' ENTRY (optional) —————

The remaining steps of this procedure show an example of enabling and starting the service named "service\_1" on all nodes in two node groups: `node_group_1` and `node_group_2`.

3. Prepare the lines of a `services` entry.

In the worksheet, copy the six lines below the following line:

```
# ** EXAMPLE 'services' VALUE (with current defaults) **
```

Paste them below the following line:

```
# NOTE: Place additional 'services' setting entries here, if desired.
```

```
# NOTE: Place additional 'services' setting entries here, if desired.
#cray_simple_services.settings.services.data.reference.sample_key_a: null <-- setting a multival key
#cray_simple_services.settings.services.data.sample_key_a.affected groups: []
#cray_simple_services.settings.services.data.sample_key_a.service_list.reference.sample_key_b: null <-- setting
a multival key
#cray_simple_services.settings.services.data.sample_key_a.service_list.sample_key_b.name: ''
#cray_simple_services.settings.services.data.sample_key_a.service_list.sample_key_b.enabled: false
#cray_simple_services.settings.services.data.sample_key_a.service_list.sample_key_b.action: ''
```

4. Provide a reference name (ID) for this `services` entry.

Uncomment the six lines and replace `sample_key_a` in all lines with an identifier for this set of affected nodes, services, and actions. Remove the `<--` setting a multival key text at the end of the first and third lines (note that the null value is required; do not remove or change it).

```
# NOTE: Place additional 'services' setting entries here, if desired.
cray_simple_services.settings.services.data.reference.my_services: null
cray_simple_services.settings.services.data.my_services.affected_groups: []
cray_simple_services.settings.services.data.my_services.service_list.reference.sample_key_b: null
cray_simple_services.settings.services.data.my_services.service_list.sample_key_b.name: ''
cray_simple_services.settings.services.data.my_services.service_list.sample_key_b.enabled: false
cray_simple_services.settings.services.data.my_services.service_list.sample_key_b.action: ''
```

## 5. Enter the list of affected node groups.

```
cray_simple_services.settings.services.data.my_services.affected_groups:
- node_group_1
- node_group_2
```

## 6. Configure a service\_list entry.

### a. Replace `sample_key_b` in the following lines with an identifier for this service.

```
cray_simple_services.settings.services.data.my_services.service_list.reference.svc_1: null
cray_simple_services.settings.services.data.my_services.service_list.svc_1.name: ''
cray_simple_services.settings.services.data.my_services.service_list.svc_1.enabled: false
cray_simple_services.settings.services.data.my_services.service_list.svc_1.action: ''
```

### b. Enter the name of the service to be controlled.

```
cray_simple_services.settings.services.data.my_services.service_list.svc_1.name: service_1
```

### c. State whether the service should be enabled during node boot. In this example, the service is enabled.

```
cray_simple_services.settings.services.data.my_services.service_list.svc_1.enabled: true
```

### d. (Optional) State whether the service should be started or stopped. This action is performed whenever `cray-ansible` is called (including immediately after node boot) and is separate from any `init(1)` processing associated with the previous setting.

```
cray_simple_services.settings.services.data.my_services.service_list.svc_1.action: start
```

To configure another `service_list` entry, copy the four-line `service_list` stanza, paste it below the first `service_list` stanza, and repeat step 6 on page 205.

Multiple `service_list` entries within the same `services` entry can control the same service. If there are conflicting `enabled` states for the same service, the last one parsed will prevail. If there are multiple `start/stop` actions for the same service, then each will be performed in order.

## 7. Configure another `services` entry, as needed.

If another group of nodes should have services controlled on them, return to the beginning of step 3 on page 204 to configure another `services` entry. The list will be parsed during boot, and the specified actions will be taken in the order they are recorded. If nonexistent services are specified, an error will occur and the nodes' boot will fail.

#### 4.5.1.24 Update `cray_simple_shares` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

The Cray Simple File-system Sharing service quickly shares files between compute nodes that are connected to the high speed network (HSN). This procedure configures some basic settings in the `cray_simple_shares` configuration worksheet.

### Procedure

1. Edit `cray_simple_shares_worksheet.yaml`.

```
smw# vi cray_simple_shares_worksheet.yaml
```

2. Uncomment `cray_simple_shares.enabled` and ensure that it is set to `true`.

3. Update the NFS mount settings.

- a. Ensure that the node groups settings are configured.

Search in the file for 'NFS' DATA, and below that line, find these `server_groups` and `client_groups` settings for several pre-populated NFS client mounts. If they are commented, uncomment them.

```
# ** 'NFS' DATA **

#cray_simple_shares.settings.NFS.data./alps_shared.server_groups:
#- sdb_nodes
#cray_simple_shares.settings.NFS.data./alps_shared.client_groups:
#- service_nodes
#cray_simple_shares.settings.NFS.data./alps_shared.client_exclude_groups:
#- boot_nodes
...
#cray_simple_shares.settings.NFS.data./cray_home.server_groups:
#- boot_nodes
#cray_simple_shares.settings.NFS.data./cray_home.client_groups:
#- service_nodes
...
#cray_simple_shares.settings.NFS.data./var/opt/cray/imps.server_groups:
#- boot_nodes
#cray_simple_shares.settings.NFS.data./var/opt/cray/imps.client_groups:
#- tier2_nodes
...
#cray_simple_shares.settings.NFS.data./non_volatile.server_groups:
#- boot_nodes
#cray_simple_shares.settings.NFS.data./non_volatile.client_groups:
#- service_nodes
```

- b. If the home directory was changed in other configuration worksheets (e.g., `cray_local_users_worksheet.yaml`), change it here also.

Under 'NFS' DATA, look for settings with `cray_home` or `home` as the path key. Ensure that they reflect the same home directory as used in `cray_local_users_worksheet.yaml`.

**Important!** The NFS path key MUST match the value of the following setting in `cray_bootraid_worksheet.yaml`.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.volumes.home.fs_mount_point
```

If there is a mismatch, the boot node volume group home is not created and the path is not mounted.

In the following lines, replace the key `cray_home` with the home directory used in `cray_local_users` and `cray_bootraid`.

```
cray_simple_shares.settings.NFS.data./cray_home.server_groups:
- boot_nodes
cray_simple_shares.settings.NFS.data./cray_home.fs_root: /cray_home
cray_simple_shares.settings.NFS.data./cray_home.fs_export_opt:
  'secure,rw,no_subtree_check,no_root_squash,no_acl'
cray_simple_shares.settings.NFS.data.path./cray_home: null
cray_simple_shares.settings.NFS.data./cray_home.client_groups:
- service_nodes
cray_simple_shares.settings.NFS.data./cray_home.unconditional_mount: false
```

#### 4. Update the DVS mount settings.

Search in the file for 'DVS' DATA, and below that line, find these settings for a pre-populated DVS client mount. If they are commented, uncomment them.

```
# ** 'DVS' DATA **
...
#cray_simple_shares.settings.DVS.data./var/opt/cray/imps.spath: /var/opt/cray/imps
#cray_simple_shares.settings.DVS.data./var/opt/cray/imps.client_groups:
#- all_nodes
```

**Disambiguation.** Notice that the path `/var/opt/cray/imps` appears twice in the first setting. The first instance is the path where clients will mount the file system. It is the 'key' (*mount\_point*) for this client mount, so it appears in all of the settings for this client mount. The second instance is the path to the file system on the server node that is to be projected. It is the default value provided for this pre-populated DVS client mount. That first setting is simply specifying that the file system will be projected from the same path on the server as it is mounted from the client.

#### 5. Verify that the node groups referenced in steps 3 and 4 have been accurately defined for this site.

To verify, edit `cray_node_groups_worksheet.yaml` and search for these node groups:

```
all_nodes
boot_nodes
sdb_nodes
service_nodes
tier2_nodes
```

```
smw# vi cray_node_groups_worksheet.yaml
```

### 4.5.1.25 Update `cray_ssh` Worksheet

#### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

## About this task

The SSH service allows the system to be accessed through a secure shell. This procedure enables the Cray SSH configuration service and enables/disables automatic SSH key management (optional).

## Procedure

1. Edit `cray_ssh_worksheet.yaml`.

```
smw# vi cray_ssh_worksheet.yaml
```

2. Uncomment `cray_ssh.enabled` and set it to `true`.

3. (Optional) Disable automatic SSH key generation.

If this site wishes to disable the automatic generation of SSH host and root user keys, uncomment the following line and set the value to `false`. Note that even with automatic key generation disabled, any SSH keys in the Simple Sync directory structure will still be synced by Simple Sync, unless the Simple Sync config service is disabled. For more information, see [About Secure Shell Configuration](#) on page 32.

```
#cray_ssh.settings.sshd.data.simple_ssh_keys: true
```

```
cray_ssh.settings.sshd.data.simple_ssh_keys: false
```

### 4.5.1.26 Update `cray_storage` Worksheet

## Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

## About this task

The Storage service defines which storage set the current partition or system may use for persistent storage. Storage sets are defined in the global config set. This procedure configures some basic settings in the `cray_storage` configuration worksheet.

## Procedure

1. Edit `cray_storage_worksheet.yaml`.

```
smw# vi cray_storage_worksheet.yaml
```

2. Uncomment `cray_storage.enabled` and set it to `true`.

3. Uncomment `cray_storage.settings.storage.data.active_storage_set` and set it to be the name of the CLE storage set in the `cray_bootraid` service, which is in the global config set.

Use this command to show all storage sets defined in the global config set.

```
smw# cfgset search -s cray_bootraid global |awk -F'.' '{print $5}' | sort -u
```

4. (For reinstall only) Uncomment `cray_storage.settings.storage.data.zero_volumes_on_create` and set it to true if this system is reinstalling to a CLE storage set that had been in use previously.

#### 4.5.1.27 Update `cray_sysconfig` Worksheet

### Prerequisites

A work area has been set up for editing CLE configuration worksheets, and the current directory has been set to that work area.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

### About this task

The Cray System Configuration service controls configuration of files in `/etc/sysconfig`. The `sysconfig` service can be used to specify particular configuration file settings and values.

This procedure enables the `cray_sysconfig` service and provides an example of changing a configuration file in `/etc/sysconfig`.

### Procedure

1. Edit `cray_sysconfig_worksheet.yaml`.

```
smw# vi cray_sysconfig_worksheet.yaml
```

2. Uncomment `cray_sysconfig.enabled` and set it to true.
3. Change configuration settings in a file in `/etc/sysconfig`, as needed.

Repeat this step for each file with settings to be changed.

In the worksheet, copy the six lines below

```
# ** EXAMPLE 'sysconfig_files' VALUE (with current defaults) **
```

and paste them below the line

```
# NOTE: Place additional 'sysconfig_files' setting entries here, if desired.
```

```
# ** EXAMPLE 'sysconfig_files' VALUE (with current defaults) **
#  cray_sysconfig.settings.sysconfig_files.data.name.sample_key_a: null    <-- setting a multival key
#  cray_sysconfig.settings.sysconfig_files.data.sample_key_a.file: ''
#  cray_sysconfig.settings.sysconfig_files.data.sample_key_a.scope:
#    - service
#    - compute
#  cray_sysconfig.settings.sysconfig_files.data.sample_key_a.key_values: []
```

Uncomment the lines, replace `sample_key_a` in all lines with an identifier for the file to be changed (sitekey in the example below), and remove the `<-- setting a multival key` text at the end of the first line

(note that the null value is required; do not remove or change it). Finally, modify the values as needed for this site.

There are two list settings in the `sysconfig_files` setting: `scope` and `key_values`. To enter a list, add each list item on a separate line prefixed by a hyphen and space (`-` ). If the list was initially set to `[]`, an empty list, remove the brackets before adding list items.

- For the `scope` list setting, enter a list of target node types (service, compute) and/or cnames (NOT node groups).
- For the `key_values` list setting, enter a list of key=value pairs.

The following example uses `sitekey` to identify the `/etc/sysconfig/filename` file and change the value of its `MYVAR` variable to `newsetting` for all service and compute nodes.

```
# NOTE: Place additional 'sysconfig_files' setting entries here, if desired.
cray_sysconfig.settings.sysconfig_files.data.name.sitekey: null
cray_sysconfig.settings.sysconfig_files.data.sitekey.file: filename
cray_sysconfig.settings.sysconfig_files.data.sitekey.scope:
- service
- compute
cray_sysconfig.settings.sysconfig_files.data.sitekey.key_values:
- MYVAR="newsetting"
```

## 4.5.2 Create New CLE Config Set from Worksheets

### Prerequisites

This procedure assumes that worksheets have been obtained, copied to a work area outside of `/var/opt/cray/imps/config/sets/CONFIG_SET_NAME/worksheets`, and modified to include site-specific configuration data.

### About this task

This procedure creates a new CLE config set from existing CLE configuration worksheets. Note that the worksheet path provided must be enclosed in single quotes because of the file glob used. There is no need to specify the config set type because the default is type CLE.

### Procedure

Create a CLE config set (p0 in example) using worksheets.

```
smw# cfgset create --worksheet-path \
'/var/adm/cray/release/p0_worksheet_workarea/*_worksheet.yaml' p0
```



**CAUTION:** Boot failure possible if using `cfgset` under certain conditions.

The `cfgset create` and `cfgset update` commands always call pre- and post-configuration scripts. Some of these scripts require HSS daemons and other CLE services to be running. This can cause problems under these conditions:

- If `xtdiscover` is running, `cfgset` may hang or produce incorrect data that can result in system boot failure.
- If `xtbounce` is in progress or if the SMW is not connected to XC hardware, `cfgset` will fail.



In these circumstances, use the `--no-scripts` option with `cfgset create` or `cfgset update` to avoid running the scripts. Because using that option results in an invalid config set, remember to run `cfgset update` without the `--no-scripts` option afterwards, when circumstances permit, to ensure that all pre- and post-configuration scripts are run.

### 4.5.3 Update Unconfigured CLE Configuration Services

#### Prerequisites

This procedure assumes that a CLE config set has been created from CLE configuration worksheets.

#### About this task

Only those CLE configuration worksheets named in the previous procedures were edited prior to creating the CLE config set. All of the worksheets were included in config set creation, but the unedited ones will result in associated configuration services that are still unconfigured.

This procedure updates those unconfigured config services using the `cfgset` CLI. Because those config services require only one or two settings to be changed for a fresh install (to enable/disable or set inheritance), this is quicker than editing their associated configuration worksheets. Moreover, the CLI command examples can be combined in a script to further increase efficiency.

**NOTE:** (SMW HA only) For SMW HA systems, the following procedures are done only on the first SMW because the config sets are shared between both SMWs in the HA cluster. In contrast, Ansible plays must be run on each SMW.

The following steps use the `cfgset modify` command, which does not produce output when successful. An optional `cfgset get` command is provided for each to verify the resulting setting value.

#### Procedure

1. Disable `cray_ccm`.

Cray cluster compatibility mode (CCM) enables users to run independent software vendor (ISV) applications without modification. Supported workload managers (WLM) include PBS, Moab/TORQUE, Slurm, and LSF. Enable and configure `cray_ccm` later, if needed, after a WLM is installed.

```
smw# cfgset modify --set false cray_ccm.enabled p0
smw# cfgset get cray_ccm.enabled p0
false
```

2. Disable `cray_cnat`.

The Cray Compute Node Administrative Tool (CNAT) is a mechanism for submitting and monitoring the execution of batch scripts; it requires a WLM to function. Enable and configure `cray_cnat` later, if needed, after a WLM is installed.

```
smw# cfgset modify --set false cray_cnat.enabled p0
smw# cfgset get cray_cnat.enabled p0
false
```

For more information about CNAT, see *XC™ Series System Administration Guide (CLE 6.0.UP07) S-2393*.

3. Disable `cray_cfgset_exclude`.

The `cray_cfgset_exclude` configuration service enables sites to specify which parts of the config set should be excluded when the config set is pushed from the SMW to the eLogin node. Sites that use eLogin will enable and configure `cray_cfgset_exclude` later, after the SMW/CLE installation process is complete.

```
smw# cfgset modify --set false cray_cfgset_exclude.enabled p0
smw# cfgset get cray_cfgset_exclude.enabled p0
false
```

#### 4. Disable `cray_docker_init`.

Docker provides containers that can wrap everything needed to run an application: code, libraries, runtime, and so forth. Use *XC™ Series Docker Administration Guide (S-2586)* to enable and configure Docker later, if needed, after the system has been booted.

```
smw# cfgset modify --set false cray_docker_init.enabled p0
smw# cfgset get cray_docker_init.enabled p0
false
```

#### 5. Disable `cray_dws`.

The Cray DataWarp Service (DWS) provides access to SSD (solid state device) storage for high bandwidth application I/O. Use *XC™ Series DataWarp™ Installation and Administration Guide (S-2564)* to enable and configure DWS later, if needed, after the system has been booted.

```
smw# cfgset modify --set false cray_dws.enabled p0
smw# cfgset get cray_dws.enabled p0
false
```

#### 6. Disable `cray_dw_wlm`.

The `cray_dw_wlm` service is an interface used by WLMs to interact with Cray DataWarp. Use *XC™ Series DataWarp™ Installation and Administration Guide (S-2564)* to enable and configure `cray_dw_wlm` later, after DataWarp has been installed.

```
smw# cfgset modify --set false cray_dw_wlm.enabled p0
smw# cfgset get cray_dw_wlm.enabled p0
false
```

#### 7. Enable or disable `cray_elogin_lnet`.

The Cray eLogin LNet (Lustre networking) configuration service is required for any system that has external login nodes that mount Lustre file systems.

If a Lustre client mount was configured (while updating `cray_lustre_client_worksheet.yaml`) that has in its `client_groups` field an eLogin node that is also a member of `cray_login.settings.login_nodes.elogin_groups`, then `cray_elogin_lnet` must be enabled.

- If required, enable `cray_elogin_lnet`.

```
smw# cfgset modify --set true cray_elogin_lnet.enabled p0
```

- If not required, disable `cray_elogin_lnet`. It can be enabled later, if needed, after eLogin is installed and configured.

```
smw# cfgset modify --set false cray_elogin_lnet.enabled p0
```

Verify the setting.

```
smw# cfgset get cray_elogin_lnet.enabled p0
```

## 8. Disable `cray_elogin_motd`.

The Cray eLogin MOTD configuration service generates the `/etc/motd` file for all eLogin nodes specified in the config set. Sites that use eLogin will enable and configure `cray_elogin_motd` later, after the SMW/CLE installation process is complete.

```
smw# cfgset modify --set false cray_elogin_motd.enabled p0
smw# cfgset get cray_elogin_motd.enabled p0
false
```

## 9. Disable `cray_elogin_networking`.

The Cray eLogin networking configuration service defines the number of eLogin nodes connected to the Cray system and their key network attributes. Sites that use eLogin will enable and configure `cray_elogin_networking` later, after the SMW/CLE installation process is complete.

```
smw# cfgset modify --set false cray_elogin_networking.enabled p0
smw# cfgset get cray_elogin_networking.enabled p0
false
```

## 10. Disable `cray_eproxy`.

The Cray eLogin eproxy configuration service wraps several XT, ALPS (Application Level Placement Scheduler), and WLM commands on eLogin nodes and executes them on the Cray login gateway. Sites that use eLogin will enable and configure `cray_eproxy` later, after the SMW/CLE installation process is complete.

```
smw# cfgset modify --set false cray_eproxy.enabled p0
smw# cfgset get cray_eproxy.enabled p0
false
```

## 11. Enable `cray_firewall` and set it to not inherit from the global config set.

The Cray firewall service is a mechanism for restricting packet traffic from various networks.

```
smw# cfgset modify --set false cray_firewall.inherit p0
smw# cfgset get cray_firewall.inherit p0
false
smw# cfgset modify --set true cray_firewall.enabled p0
smw# cfgset get cray_firewall.enabled p0
true
```

## 12. Set `cray_ipforward` to inherit from the global config set.

The Cray IP Forwarding service enables IP forwarding between service nodes and the SMW. Enable inheritance from the global config set.

```
smw# cfgset modify --set true cray_ipforward.inherit p0
smw# cfgset get cray_ipforward.inherit p0
true
```

## 13. Disable `cray_kdump`.

The `cray_kdump` configuration service configures the kernel dump tool on eLogin nodes. Sites that use eLogin will enable and configure `cray_kdump` later, after the SMW/CLE installation process is complete.

```
smw# cfgset modify --set false cray_kdump.enabled p0
smw# cfgset get cray_kdump.enabled p0
false
```

**14. Set `cray_liveupdates` to inherit from the global config set.**

The LiveUpdates service enables package manager (e.g., zypper, yum) actions (e.g., install, search, upgrade) on CLE nodes using repositories shared from the SMW to those nodes.

```
smw# cfgset modify --set true cray_liveupdates.inherit p0
smw# cfgset get cray_liveupdates.inherit p0
true
```

**15. Disable `cray_lmt`.**

The Cray Lustre Monitoring Tool (LMT) monitors Lustre servers using the Cerebro monitoring system. Enable and configure `cray_lmt` later if direct-attached Lustre (DAL) is configured and this site wishes to use LMT for DAL.

```
smw# cfgset modify --set false cray_lmt.enabled p0
smw# cfgset get cray_lmt.enabled p0
false
```

**16. Set `cray_logging` to inherit from the global config set.**

If this site prefers to configure logging differently for different CLE config sets instead, skip this step and edit `cray_logging_worksheet.yaml` in the work area set up for the CLE config set (for `p0`, that is `/var/adm/cray/release/p0_worksheet_workarea`).

```
smw# cfgset modify --set true cray_logging.inherit p0
smw# cfgset get cray_logging.inherit p0
true
```

**17. (Optional) Change the logging location of `atftpd` to `/var/opt/cray/log/smwmessages-<date>`.**

Cray uses the SUSE `atftpd` server. The `/etc/init.d/atftpd` file provided by SUSE is set up to log to `/var/log/atftpd/atftp.log`, and the logging location is not configurable in the associated `/etc/sysconfig/atftpd` file. This is different than most SMW daemons, which log to `/var/opt/cray/log/smwmessages-<date>`.

Perform this step if this site wants `atftpd` on this system to log to `/var/opt/cray/log/smwmessages-<date>`.

- a. Edit the `/etc/init.d/atftpd` file.

```
smw# sed --in-place=.bak 's/--logfile $ATFTP_LOG_FILE//' /etc/init.d/atftpd
```

- b. Restart the `atftpd` service.

```
smw# systemctl restart atftpd.service
```

**18. Enable or disable `cray_munge`.**

Cray MUNGE is an authentication service that creates and validates credentials. It is required by the DataWarp service (`cray_dws`), Slurm (a workload manager), ALPS (`cray_alps`), and Dynamic RDMA Credentials (`cray_drc`). If one or more of those configuration services are enabled, then enable `cray_munge`.

- If required, enable `cray_munge`.

```
smw# cfgset modify --set true cray_munge.enabled p0
```

- If not required, disable `cray_munge`. It can be enabled later if needed.

```
smw# cfgset modify --set false cray_munge.enabled p0
```

Verify setting.

```
smw# cfgset get cray_munge.enabled p0
```

#### 19. Disable `cray_nat_masq`.

The Cray NAT (network address translation) Masquerading configuration service defines a gateway node to serve as the default gateway for one or more client nodes, and it provides the option to configure `SuSEfirewall12` to permit network traffic from a client node to a network outside the Cray high-speed network (HSN).

This service will be enabled and configured later, if needed, after the system has been booted.

```
smw# cfgset modify --set false cray_nat_masq.enabled p0
smw# cfgset get cray_nat_masq.enabled p0
false
```

#### 20. Enable `cray_node_health`.

The Cray Node Health Checker (NHC) is automatically invoked by ALPS (Application Level Placement Scheduler) upon the termination of an application. NHC performs specified tests to determine if compute nodes allocated to the application are healthy enough to support running subsequent applications.

```
smw# cfgset modify --set true cray_node_health.enabled p0
smw# cfgset get cray_node_health.enabled p0
true
```

Cray recommends that sites install and configure CLE with default NHC plugins first, and then return to the `cray_node_health` config service after the first system boot to configure custom plugins, if needed, using the `custom_plugins` setting. More information is provided later in the process.

#### 21. Enable `cray_rur`.

RUR (Resource Utilization Reporting) is a scalable framework for collecting utilization data from nodes within a user application. It is also a collection of plugins that report an extensible list of statistics about the hardware and software resources consumed by the application.

```
smw# cfgset modify --set true cray_rur.enabled p0
smw# cfgset get cray_rur.enabled p0
true
```

Cray recommends that sites install and configure CLE with default RUR plugins first, and then return to the `cray_rur` config service after the first system boot to configure custom plugins, if needed, using the `data_plugins` or `output_plugins` settings. More information is provided later in the process.

#### 22. Enable `cray_service_node`.

The Cray Service Node configuration service configures the services and settings for service nodes.

```
smw# cfgset modify --set true cray_service_node.enabled p0
smw# cfgset get cray_service_node.enabled p0
true
```

#### 23. Disable `cray_shifter`.

Shifter is an HPC-focused implementation of Linux containers that enables a large-scale HPC system to efficiently and safely allow end-users to run a docker image. Use *XC™ Series Shifter Configuration Guide* (S-2572) to enable and configure Shifter later, if needed, after the system has been booted.

```
smw# cfgset modify --set false cray_shifter.enabled p0
smw# cfgset get cray_shifter.enabled p0
true
```

#### 24. Enable cray\_simple\_sync.

Simple Sync is a mechanism for automatically distributing files to targeted locations on the Cray XC system.

```
smw# cfgset modify --set true cray_simple_sync.enabled p0
smw# cfgset get cray_simple_sync.enabled p0
true
```

#### 25. Enable cray\_sysenv.

The Cray System Environment service enables sites to make any sysctl, systemd, or limit changes needed within the CLE system environment.

```
smw# cfgset modify --set true cray_sysenv.enabled p0
smw# cfgset get cray_sysenv.enabled p0
true
```

#### 26. Set cray\_time to inherit from the global config set.

The Cray Time service configures the time zone and several advanced features, such as the minimum poll interval for NTP messages.

```
smw# cfgset modify --set true cray_time.inherit p0
smw# cfgset get cray_time.inherit p0
true
```

#### 27. Enable cray\_user\_settings.

The Cray User Settings service sets the environment modules that should be loaded automatically when a user logs in to the SMW, login node, or service nodes. The SMW modules can be extended by adding to `/etc/opt/cray/modules/Base-opts.local`.

```
smw# cfgset modify --set true cray_user_settings.enabled p0
smw# cfgset get cray_user_settings.enabled p0
true
```

#### 28. Enable cray\_wlm\_detect and set the active WLM.

The Cray WLM (workload manager) Detect service is a C library and command used to identify the native WLM on the system. If this service is not configured, the default ALPS will be used.

##### a. Enable cray\_wlm\_detect.

```
smw# cfgset modify --set true cray_wlm_detect.enabled p0
smw# cfgset get cray_wlm_detect.enabled p0
true
```

##### b. Set the active WLM.

- For WLMs using BASIL, or to indicate no WLM, set the value to ALPS.

```
smw# cfgset modify --set ALPS \
cray_wlm_detect.settings.common.data.active_wlm p0
```

- For a native WLM, enter its name in uppercase (for example, enter SLURM for Slurm). Currently only ALPS and Slurm are supported.

```
smw# cfgset modify --set SLURM \
cray_wlm_detect.settings.common.data.active_wlm p0
```

Verify setting.

```
smw# cfgset get cray_wlm_detect.settings.common.data.active_wlm p0
```

### 29. Enable cray\_wlm\_trans.

The Cray WLM Trans service is a library that provides WLM-agnostic functions for common tasks such as setting node state and getting a list of jobs being run by a user. It is used primarily by NHC.

```
smw# cfgset modify --set true cray_wlm_trans.enabled p0
smw# cfgset get cray_wlm_trans.enabled p0
true
```

### 30. Enable cray\_zonesort.

The zonesort\_module kernel module sorts free memory on the node to improve the predictability of the MCDRAM (multi-channel dynamic random-access memory) cache performance. The Cray zone sort configuration service configures the loading of the zonesort\_module kernel module on compute nodes.

```
smw# cfgset modify --set true cray_zonesort.enabled p0
smw# cfgset get cray_zonesort.enabled p0
true
```

For partitioned systems, each partition generally has its own config set. Repeat this procedure for each CLE config set, substituting the correct CLE config set name for *p0* in the command examples provided.

## 4.5.4 Change Group Ownership or Permissions for Log Files and Directories in the CLE Config Set

### Prerequisites

The `cray_logging` configuration service in the CLE config set does not inherit from `cray_logging` in the global config set. If it does inherit from global, skip this procedure, because it was already done as part of "Prepare and Update the Global Config Set."

### About this task

Cray provides a pre-populated group called `craylogreaders` that has more restricted permissions than the `crayadm` group. It is defined in the `cray_global_local_users` configuration service, and it is intended for use when specifying group ownership for log files and directories in `cray_logging`. The current default permissions for log files and directories are different than in releases prior to CLE 6.0.UP06, and they enable use of the `craylogreaders` group for log ownership.



**WARNING:** The current default permissions for log files and directories are not backward compatible with releases older than CLE 6.0.UP06. If current permissions are used, this system will no longer be able to boot and run those older CLE 6.0 releases.

Sites must choose whether to go forward with the current default permissions and possibly use a non-default group for log ownership, or change the permissions to maintain backward compatibility and keep `crayadm` as owner of log files and directories.

- **If backward compatibility is NOT an issue**, use the current default permissions (no action needed), and use one of the following ownership groups for log files and directories:
  - `crayadm`, the default group
  - (recommended) `craylogreaders`
  - custom group defined in `cray_global_local_users` (this should have been done in [Prepare and Update the Global Config Set](#) on page 129)

To use `crayadm` (the default) as the ownership group, do nothing. To use `craylogreaders` or a custom group, go to step 1 on page 218.

- **If backward compatibility is necessary**, keep `crayadm` as the ownership group for log files and directories (no action needed), and change the permissions for log files and directories to values compatible with previous releases using step 2 on page 219.

Note that the ownership group specified for log files and directories, whether the default or some other group, will be set as a secondary group for the `crayadm` and `postgres` users.

## Procedure

### —— CHANGE GROUP OWNERSHIP (NOT BACKWARD COMPATIBLE) ——

1. Set the group ownership for log files and directories in the CLE config set.

If backward compatibility is necessary, skip this step.

To use `crayadm` (the default) as the ownership group, do nothing. To use `craylogreaders` or a custom group, use the following commands.

This example changes the value from `crayadm` to `craylogreaders` in CLE config set `p0`. Note that the value of `group` in the `dirs` setting must always be the same as the value of `group` in the `logs` setting.

```
smw# cfgset get cray_logging.settings.dirs.data.group p0
crayadm
smw# cfgset modify --set craylogreaders cray_logging.settings.dirs.data.group p0
smw# cfgset get cray_logging.settings.dirs.data.group p0
craylogreaders

smw# cfgset get cray_logging.settings.logs.data.group p0
crayadm
smw# cfgset modify --set craylogreaders cray_logging.settings.logs.data.group p0
smw# cfgset get cray_logging.settings.logs.data.group p0
craylogreaders
```

The CLE config set does not need to be updated at this time because it will be updated soon in another procedure.



When these changes are applied, they will affect all log directories and all newly created log files going forward, but they are not retroactive. The group ownership of previously created log files is not changed. If this site wishes to change ownership of older log files as well, those changes must be made manually.

Skip the rest of this procedure.

#### —— CHANGE PERMISSIONS TO MAINTAIN BACKWARD COMPATIBILITY ——

### 2. Change the permissions for log files and directories in the CLE config set.

If backward compatibility is necessary, keep the default value of `crayadm` as the ownership group for log files and directories, and change the permissions for log files and directories to values compatible with previous releases.

To keep `crayadm` (the default) as the ownership group, do nothing. Change the permissions for log files and directories to backward compatible values using the following commands.

This example changes the value of permissions/mode for directories and logs in CLE config set `p0`.

```
smw# cfgset get cray_logging.settings.dirs.data.mode p0
0771
smw# cfgset modify --set 0775 cray_logging.settings.dirs.data.mode p0
smw# cfgset get cray_logging.settings.dirs.data.mode p0
0775

smw# cfgset get cray_logging.settings.logs.data.mode p0
0660
smw# cfgset modify --set 0644 cray_logging.settings.logs.data.mode p0
smw# cfgset get cray_logging.settings.logs.data.mode p0
0644
```

## 4.5.5 Add or Migrate Site Data to `/etc/hosts` File

### About this task

Cray system management software uses Hardware Supervisory System (HSS) and config set data to automatically generate the `hosts` file for installation on internal CLE nodes and the `hosts.external` file for installation on eLogin nodes. The `hosts` file is installed onto internal CLE nodes as `/etc/hosts`, and the `hosts.external` file is installed onto eLogin nodes as `/etc/hosts`.

Sites often need to edit the `hosts` and `hosts.external` files to add their own data, but because these files are generated each time the CLE config set is updated, those additions could be overwritten. To address this issue, those files are now composed of four components: the auto-generated hosts data and three additional files that can be modified by sites.

- `hosts.generic`: contains generic entries from Cray.
- head file
  - `hosts.head`: contains site entries that will be placed above the auto-generated hosts data in the `hosts` file that will be installed on internal CLE nodes.
  - `hosts_ext.head`: contains site entries that will be placed above the auto-generated hosts data in the `hosts` file that will be installed on eLogin nodes.
- Auto-generated hosts data: contains entries generated from HSS and configuration data.

Note that site entries in this component of the `/etc/hosts` file (comments and other changes) are no longer preserved. To add extra labels to a host entry line, add them to the configured aliases list of that host definition in the `cray_net` configuration service.

- tail file
  - `hosts.tail`: contains site entries that will be placed below the auto-generated hosts data in the `hosts` file that will be installed on internal CLE nodes.
  - `hosts_ext.tail`: contains site entries that will be placed below the auto-generated hosts data in the `hosts` file that will be installed on eLogin nodes.

These component files are located in the config set on the SMW in the following directory:

```
/var/opt/cray/imps/config/sets/<CONFIG_SET>/files/roles/common/etc/
```

Cray recommends using the head and tail files to supply entries for nodes that cannot be defined in either the `cray_global_net` or `cray_net` configuration services. Do not add an entry to a head/tail file that may conflict with entries generated automatically from `cray_global_net` and `cray_net` configuration data.

## Procedure

1. (Optional) Edit the `hosts.generic` to customize for this site, if desired.
2. Migrate site-specific data and comments to `hosts.head` (`hosts_ext.head`) and/or `hosts.tail` (`hosts_ext.tail`).

Site host entries and other data previously entered directly into `hosts` (`hosts.external`) or added to `/etc/hosts` files in the Simple Sync directory must be moved to `hosts.head` (`hosts_ext.head`) or `hosts.tail` (`hosts_ext.tail`).

Create the needed head or tail files and enter data into them. For more information and examples, see [About the /etc/hosts File](#) on page 43. If the head and/or tail files are not present when the CLE config set is updated, the missing file(s) will be automatically generated and will be empty.

3. Add new site-specific data and comments to the head and/or tail files, as needed.

### 4.5.6 Update CLE Config Set after a Fresh Install

#### Prerequisites

This procedure assumes that one or more CLE config sets have been created.

#### About this task

This procedure uses the configurator in auto mode to check for any required or basic settings that were not configured earlier in the process. The configurator will prompt for values for those settings.

The `crayadm` and root passwords from the `cray_local_users` service were not configured earlier using worksheets because they must be encrypted, and it is difficult to enter encrypted values in a worksheet. Therefore, the configurator will prompt for those values now.

In addition, the configurator may prompt for the value of the `flat_routes` setting or the `fgr_routes` setting or both (from the `cray_lnet` service), depending on which one is not being used for external Lustre servers or whether direct-attached Lustre (DAL) is used.

Configurator navigation tips:

- For context-sensitive command help, enter `?`.
- To add a single value, enter the data and press **Enter**.
- To add a list, enter `+`, enter each list item on its own line, press **Ctrl-d** when done entering list items, and then press **Enter** to set the list entries.
- To skip a setting, press the `>` key. Note that skipping an unconfigured setting leaves it unconfigured, which means the configurator will assign it the default value and will prompt for it again if invoked with the same command.
- To correct an error in a previous setting, press the `<` key to go back to the previous setting, correct it, then continue forward. Use `<` to back up several settings, if needed.

## Procedure

Invoke `cfgset` to update the CLE config set (p0 in the example).

```
smw# cfgset update p0
```

Enter values for any settings presented by the configurator. The following steps provide instructions for specific settings.

- Set the `crayadm` password when prompted for

```
cray_local_users.settings.users.data.crayadm.crypt.
```

This `cray_local_users` setting is for a CLE/Linux account. It is of type "protected," so it must be entered twice (the second time for confirmation), and it will not be displayed while being entered. The configurator will encrypt it before storing it in the config set.

Enter `+`, then enter the password (NOT its encrypted form) and press **Enter**. Re-enter the password and press **Enter** again.

This example shows the value `crayadm_password` entered at the prompt, but actually, the configurator will not display what is entered.

```
cray_local_users.settings.users.data.crayadm.crypt
[+=modify, ?=help, @=less] $ +
Modify crypt (Ctrl-d to cancel, <cr> to set) $ crayadm_password
Re-enter value for crypt (Ctrl-d to cancel, <cr> to set) $ crayadm_password
```

- Set the root password when prompted for

```
cray_local_users.settings.users.data.root.crypt.
```

This is another `cray_local_users` setting for a CLE/Linux account, also of type "protected."

```
cray_local_users.settings.users.data.root.crypt
[+=modify, ?=help, @=less] $ +
Modify crypt (Ctrl-d to cancel, <cr> to set) $ root_password
Re-enter value for crypt (Ctrl-d to cancel, <cr> to set) $ root_password
```

- c. Set the "users" entries when done setting the crayadm and root passwords.

```
cray_local_users.settings.users
[<cr>=set N entries, ?=help, @=less] $ <cr>
```

**Not prompted for these passwords?** If the configurator did not prompt for one or both of these settings, wait until `cfgset` finishes, then run `cfgset` in interactive mode (example shows command for config set `p0`), and select and set these settings from the `cray_local_users` service.

```
smw# cfgset update -m interactive -s cray_local_users p0
```

For more information about using the configurator, see *XC™ Series Configurator User Guide* (S-2560).

If no more settings are presented, it means that all required and basic settings have been set.

When the configurator is done, it displays a message indicating the file name of the changelog file for this configuration session. The changelog is written to a file in the `/var/opt/cray/imps/config/sets/p0/changelog` directory (for a CLE config set named `p0`).

## 4.5.7 Check CLE Host Names in `/etc/hosts` File

### Prerequisites

The CLE config set has been created and updated.

### About this task

This procedure confirms that the post-configuration callback scripts, which were run when the CLE config set was updated, added the correct host name entries to the `/etc/hosts` file.

### Procedure

1. Confirm that host name entries exist in the CLE `/etc/hosts` file for `boot`, `sdb`, `login`, `lnet`, `rsip`, `dvs`, and any other hosts defined on this system.

```
smw# egrep "boot|sdb|login|lnet|rsip|dvs" \
/var/opt/cray/imps/config/sets/p0/files/roles/common/etc/hosts
```

**Trouble?** If any expected host names are missing, proceed to step 2.

2. If any expected host names are missing, do one of the following:

#### Option

#### Description

#### Option 1: Update `cray_net`

Use `cfgset` to update the `cray_net` configuration service in this config set (`p0` in the example) and add any missing host name, host name alias, or network interface information.

```
smw# cfgset update -m interactive -s cray_net p0
```

#### Option 2: Add to the `/etc/hosts` file

Return to [Add or Migrate Site Data to `/etc/hosts` File](#) on page 219 to add host names and IP addresses to the head file (`hosts.head` or `hosts_ext.head`) or the tail file (`hosts.tail` or `hosts_ext.tail`) in this config set on the SMW.

Option	Description
	Be sure to update the config set afterwards, so that the <code>/etc/hosts</code> file is regenerated with the new data.

**IMPORTANT:**

- Adding content to configuration files by editing them on nodes is ephemeral.
- Adding content to configuration files by using `cfgset` to update the config set on the SMW (as in Option 1) or by editing them within the config set on the SMW (as in Option 2) is permanent.

Changes made to a config set on the SMW are shared with CLE nodes through config set caching. For more information, see [About Config Set Caching](#) on page 22.

## 4.5.8 Set Up Advanced RSIP Configuration Before Booting the System

### Prerequisites

- Advanced RSIP (realm-specific IP) configuration is needed on this system, such as RSIP failover and RSIP pools.
- The `cray_rsip` configuration service has been enabled.
- The required settings in `cray_net` have been set.
- The `/etc/hosts` file has been populated with correct information.
- The CLE system is not yet booted.

Skip this procedure if this system does NOT need advanced RSIP configuration.

### About this task

RSIP (realm-specific IP) is a method of IP address sharing. This procedure changes a setting in the `cray_rsip` configuration service and uses `xtrsipcfg_v2` to generate the RSIP client and server configuration files that are needed for more advanced RSIP configuration: `krsip.conf` (client) and `rsipd.conf` (server). These configuration files can be generated before the CLE system is booted by using the `--not_booted` (or `-n`) flag with `xtrsipcfg_v2`, which enables it to run offline and use the information in `cray_net` and `/etc/hosts`.

Advanced RSIP configuration can be done later on a booted system. If this site wishes to do this later rather than during a fresh install, finish the installation process, then use the "Set Up Advanced RSIP Configuration on a Booted System" procedure, which is under "Modify an Installed System" in *XC™ Series System Administration Guide (CLE 6.0.UP07) S-2393*.

### Procedure

1. Modify the `use_xtrsipcfg` setting in the `cray_rsip` config service in the CLE config set (p0 in the example).

```
smw# cfgset modify --set true \
cray_rsip.settings.service.data.use_xtrsipcfg p0
```

2. Update the CLE config set that was just modified (p0 in the example) and generate worksheets.

The command in the previous step modified the config set without running pre- and post-configuration scripts, so that config set was marked invalid. This step uses `cfgset update` in prepare mode (no user interaction) to ensure that all configuration scripts are run.

```
smw# cfgset update -m prepare p0
```

3. Copy the new worksheets to the work area set up earlier in the process.

```
smw# cp -a /var/opt/cray/imps/config/sets/p0/worksheets/* \
/var/adm/cray/release/p0_worksheet_workarea/
```

The `cray_net` worksheet will be updated later in this procedure.

### ————— RUN THE XTRSIPCFG\_V2 SCRIPT —————

4. Run `xtrsipcfg_v2` as root.

This command generates the needed configuration files and places them in `/var/opt/cray/imps/config/sets/p0/files/roles/rsip/`, where `p0` is the current CLE config set.

```
smw# /opt/cray/xtrsipcfg/*/bin/xtrsipcfg_v2 -b --not_booted
```

Use the following substeps to enter information when prompted by `xtrsipcfg_v2`.

- a. Enter the name of the config set to update.

```
config_set_name: p0
Gathering Node Information
```

- b. Indicate whether isolated service nodes should be set up as RSIP clients.

- If uncertain or it is known that they should not be set up as RSIP clients, enter **n** (default).

```
Should the isolated service nodes be setup as RSIP clients? [y/N]: n
```

- If they should be set up as RSIP clients, enter **y**.

```
Should the isolated service nodes be setup as RSIP clients? [y/N]: y
```

A list of the isolated service nodes displays. Enter a space-delimited list of `cnames` to configure them as RSA clients. All other isolated nodes are configured as RSAP clients.

Note that the list of isolated service nodes could be missing an isolated node that is targeted as an RSIP client. The SDB node for instance, may not show up because it has an Ethernet interface. Do not attempt to use non-isolated nodes as RSIP clients. However, exceptions can be made if the node simply has an interface that is connected only internally. The next prompt asks if there were any missing nodes to be added as RSIP clients (RSAP or RSA). It is possible to add the previously described nodes as clients in response to these prompts, but be very careful to only add nodes that do not have external network connectivity.

- c. Specify an alternate location for the list of compute nodes, if desired.

```
By default a file containing all compute_names is created in /tmp/
rsip_compute_names.txt
Refer to this file for the next steps if necessary.
Enter Alternate filepath for compute_names file or hit return:
```

- d. Specify compute nodes to be used as RSA clients.

By default all compute nodes are configured as RSAP clients. For this example, do not enter any nodes for RSA client configuration.

Enter a space delimited list of COMPUTE Node cnames to be RSA CLIENTS.  
 \*\* Unlisted nodes will be configured as RSAP CLIENTS \*\*

Compute RSA Clients:

The script displays connectivity information to use in response to subsequent questions from the script.

```
Service node network connectivity:
(login)      c1-0c0s0n2: eth0 : 192.0.2.88 255.255.240.0
              eth1 : DOWN -
              c1-0c1s3n1: eth0 : DOWN -
              eth1 : DOWN -
              eth2 : DOWN -
              eth3 : DOWN -
              c1-0c1s3n2: eth0 : 192.0.2.253 255.255.0.0
              eth1 : DOWN -
              eth2 : 192.0.2.17 255.255.255.0
              eth3 : DOWN -
              c1-0c1s4n1: eth0 : DOWN -
              eth1 : DOWN -
              eth2 : DOWN -
              eth3 : DOWN -
              c1-0c1s4n2: eth0 : 192.0.2.43 255.255.240.0
              eth1 : DOWN -
              eth2 : DOWN -
              eth3 : DOWN -
```

- e. Enter the cnames of service nodes to use as RSIP servers.

*Specify only a dedicated RSIP node.* Clients are automatically assigned to servers by the script. Specify one or more service nodes from the connectivity information just provided by the script.

Auto Config Servers: **c1-0c1s4n2**

- f. Enter an address pool for each server.

When the cnames of the specified servers are displayed, for each server, provide a pool of IP addresses to use. The script accepts a space-delimited list or a range of IP addresses. Not specifying a pool of available IP addresses causes the server to instead use its primary external interface.

In this example, use multiple IP addresses for the node c1-0c1s4n2. The specified addresses should be on the same subnet. Accept the default for IPs reserved for RSA because no RSA clients or servers are configured in this example.

Provide an Address Pool as a combination of IPs and IP ranges in a space delimited list.

ex: 192.0.2.0-192.0.2.24 192.0.2.30 192.0.2.34-192.0.2.40

Leave this field empty if using RSAP-IP with the server's primary external interface

c1-0c1s4n2: 192.0.2.43 192.0.2.44

How many IPs should be reserved for RSA? [Default 0]:

- g. Specify RSIP servers to manually assign clients to them.

This example does not include any manually assigned clients to any servers, so press **Enter**.

```
Enter a space delimited list of node cnames for nodes that will be RSIP
SERVERS.
For example: c0-0c0s7n0 c0-0c0s7n3 c0-0c1s1n3
* RSIP Servers may be manually or automatically configured with clients
You will list below the servers which you will manually assign clients to,
followed by the servers which you want clients automatically assigned to.
* RSIP Servers MUST have external network connectivity.

Manual Config Servers:
```

- h. Enter an RSIP port range and a system port range for each server.

For this example, accept the defaults by pressing **Enter**.

```
Provide RSIP port range and System port range for the following servers (Non
overlapping ranges).
These ranges will be used only for RSAP IP.
Defaults RSIP: 1-60000 Default System: 60001-65535. Hit enter to use defaults
c1-0c1s4n2
    RSIP:
    System:
Should all subsequent servers utilize these settings? (Y/n):
```

The script displays the locations of the configuration files that it has created.

```
Created krsip config file at /var/opt/cray/imps/config/sets/myconfigset-p0/files/roles/rsip/etc/krsip.yaml
Created RSIPD.<nid>.conf files in /var/opt/cray/imps/config/sets/myconfigset-p0/files/roles/rsip/etc/opt/cray/
rsipd/
Created rsipd.yaml at /var/opt/cray/imps/config/sets/myconfigset-p0/files/roles/rsip/etc/opt/cray/rsipd/rsipd.yaml
```

5. Verify that `xtrsrcfg_v2` successfully created the configuration files.

- a. Verify that the `krsip.conf` (client) and `rsipd.conf` (server) configuration files are present and in the correct locations.

The following example shows the locations for config set p0. If using a different CLE config set, substitute the correct name in the file path.

```
smw# ls -l /var/opt/cray/imps/config/sets/p0/files/roles/rsip/etc
smw# ls -l /var/opt/cray/imps/config/sets/p0/files/roles/rsip/etc/opt/cray/rsipd
```

- b. Verify the contents of the `rsipd.conf` file for the example in the previous step.

```
smw# cd /var/opt/cray/imps/config/sets/p0
smw# cd files/roles/rsip/etc/opt/cray/rsipd
rsipd.c1-0c1s4n2.conf rsipd.yaml
smw# grep "^pool " rsipd.c1-0c1s4n2.conf
pool 192.0.2.90
pool 192.0.2.91
```

————— DEFINE RSIP HOST IN CRAY\_NET CONFIG SERVICE —————

6. Edit `cray_net_worksheet.yaml`.

```
smw# vi /var/adm/cray/release/p0_worksheet_workarea/cray_net_worksheet.yaml
```

7. Update the RSIP host definition with values (bold) shown in the following listing.



If an RSIP host is not defined in the config set, add a host definition stanza for an RSIP server like the following one, placing it under NOTE: Place additional 'host' setting entries here, if desired. A sample host definition that includes default host settings is included in the worksheet under `cray_net.settings.hosts.data.common_name.sample_key_a: null`.

```
cray_net.settings.hosts.data.common_name.rsip_node: null
cray_net.settings.hosts.data.rsip_node.description: RSIP node
cray_net.settings.hosts.data.rsip_node.aliases:
- rsip
cray_net.settings.hosts.data.rsip_node.hostid: c1-0c1s4n2
cray_net.settings.hosts.data.rsip_node.host_type: ''
cray_net.settings.hosts.data.rsip_node.hostname: rsip1
cray_net.settings.hosts.data.rsip_node.standby_node: false

cray_net.settings.hosts.data.rsip_node.interfaces.common_name.eth0: null
cray_net.settings.hosts.data.rsip_node.interfaces.eth0.name: eth0
cray_net.settings.hosts.data.rsip_node.interfaces.eth0.description:
    Ethernet connecting the RSIP node to the customer network.
cray_net.settings.hosts.data.rsip_node.interfaces.eth0.aliases: []
cray_net.settings.hosts.data.rsip_node.interfaces.eth0.network: login
cray_net.settings.hosts.data.rsip_node.interfaces.eth0.ipv4_address: 192.0.2.43
cray_net.settings.hosts.data.rsip_node.interfaces.eth0.mac: ''
cray_net.settings.hosts.data.rsip_node.interfaces.eth0.startmode: auto
cray_net.settings.hosts.data.rsip_node.interfaces.eth0.bootproto: static
cray_net.settings.hosts.data.rsip_node.interfaces.eth0.mtu: ''
cray_net.settings.hosts.data.rsip_node.interfaces.eth0.extra_attributes:
- IPADDR1='192.0.2.90/20'
- IPADDR2='192.0.2.91/20'
#cray_net.settings.hosts.data.rsip_node.interfaces.eth0.module: ''
#cray_net.settings.hosts.data.rsip_node.interfaces.eth0.params: ''
#cray_net.settings.hosts.data.rsip_node.interfaces.eth0.unmanaged_interface: false
```

8. Import the completed `cray_net` worksheet to the CLE config set (p0 in the example).

```
smw# cfgset update --worksheet-path \
/var/adm/cray/release/p0_worksheet_workarea/cray_net_worksheet.yaml p0
```

## 4.5.9 Update /etc/motd for Nodes

### About this task

The standard `/etc/motd` on CLE nodes has this information.

```
Identity of node
Compute or service node
Boot image
Size of boot image
CLE release and build
Core and memory info
```

To append a custom message to the standard message of the day for all nodes, edit the `/etc/motd` file as shown in the example, which uses the config set common role to distribute the `/etc/motd` file to all nodes.

### Procedure

1. Create the `files/roles/common/etc` path below the config set directory.

```
smw# cd /var/opt/cray/imps/config/sets/p0
smw# mkdir -p files/roles/common/etc
```

2. Edit the message of the day to append the custom message.

```
smw# vi files/roles/common/etc/motd
```

## 4.5.10 Copy Files for External Lustre Fine-grained Routing

### Prerequisites

This procedure is only for systems that use an external Lustre file system. It assumes the following:

- Fine-grained routing (FGR) files have been generated by `clcvt`
- Cray LNet configuration service (`cray_lnet`) has been configured with FGR

### About this task

This procedure places the `ip2nets.conf` and `routes.conf` files in the CLE config set for the LNet routers.

### Procedure

1. Create an `lnet` directory under `roles` in the CLE config set directory structure.

This example uses a config set named `p0`. Substitute the correct config set name for this site.

```
smw# mkdir -p /var/opt/cray/imps/config/sets/p0/files/roles/lnet
```

2. Confirm the file names of the fine-grained routing files.

It is possible that these two files were created with names other than `ip2nets.conf` and `routes.conf`. Check these two settings in the `cray_lnet` configuration service to see what file names are used (example settings are for config set `p0` and a file system with key `sonexion`).

```
smw# cfgset search -l advanced -s cray_lnet -t fgr_routes p0
# 2 matches for 'fgr_routes' from cray_lnet_config.yaml
#-----
cray_lnet.settings.fgr_routes.data.sonexion.ip2nets_file: ip2nets.conf
cray_lnet.settings.fgr_routes.data.sonexion.routes_file: routes.conf
```

3. Copy the `ip2nets.conf` and `routes.conf` files to the `lnet` directory.

```
smw# cd directory_containing_ip2nets.conf_and_routes.conf
```

```
smw# cp -p ip2nets.conf routes.conf /var/opt/cray/imps/config/sets/p0/files/roles/lnet
```

## 4.5.11 Configure Files for Cray Simple Sync Service

### About this task

Cray Simple Sync provides a generic mechanism to automatically distribute files to targeted locations on a Cray XC system. This mechanism can be used to override or change default system behavior through the contents of

the distributed files. When enabled, the Simple Sync service is executed on all internal CLE nodes and eLogin nodes at boot time and whenever the administrator executes `/etc/init.d/cray-ansible start` on a CLE node or eLogin node. When Simple Sync is executed, files placed in the following directory structure are copied to the root file system (/) on the target nodes.

### About the Simple Sync Directory Structure

The Simple Sync directory structure has this root:

```
smw:/var/opt/cray/imps/config/sets/<config_set>/files/simple_sync/
```

Below that root are the directories listed on the left. Files placed in those directories are copied to their associated target nodes.

Files placed here	are copied to
<code>./common/files/</code>	all internal and eLogin nodes
<code>./platform/[compute service]/files/</code>	all CLE compute nodes or all service nodes (not applicable to eLogin nodes)
<code>./hardwareid/&lt;hardwareid&gt;/files/</code>	nodes with matching hardware ID, which is the cname of a CLE node or the output of the <code>hostid</code> command (e.g., <code>1eac0b0c</code> ) on other nodes (not applicable to eLogin nodes)
<code>./hostname/&lt;hostname&gt;/files/</code>	nodes with matching host name (use this for eLogin nodes ONLY)
<code>./nodegroups/&lt;node_group_name&gt;/files/</code>	nodes in the matching node group

**NOTE:** The directory structure for a particular hardware ID or host name (everything below `./hardwareid/` and `./hostname/`) must be created manually as needed. This is unnecessary for node groups because their associated directories are created automatically by post-configuration callback scripts when the config set is created or updated using `cfgset`.

Anything (directory structure and files) placed below `./files/` in the Simple Sync directory structure on the SMW is replicated on the target node starting at root (/). For example, if the `myapplication.conf` file is placed in this path on the SMW

```
/var/opt/cray/imps/config/sets/p0/files/simple_sync/common/files/etc/myapplication.conf
```

then Simple Sync will place `myapplication.conf` here on all nodes:

```
/etc/myapplication.conf
```

Note that the ownership and permissions of files in the config set are preserved in the copies made to nodes.

For more information and use cases, see [About Simple Sync](#) on page 27.

## 4.5.12 Display and Capture all Config Set Information

### About this task

This procedure displays all of the configuration settings in a config set and captures them in a typescript file of this software update. It is not required, but it may aid in troubleshooting. Note that the `cfgset search` command

does not search guidance text in the configuration templates and worksheets, so that information will not be included in the output.

## Procedure

1. Display all configuration settings in the CLE config set (p0 in example), and capture them in a typescript file.

```
smw# cfgset search -l advanced --format full p0 | tee \
/var/adm/cray/release/p0.${TODAY}.fresh_install.advanced.conf.full
```

2. Display all configuration settings in the global config set, and capture them in a typescript file.

```
smw# cfgset search -l advanced --format full global | tee \
/var/adm/cray/release/global.${TODAY}.fresh_install.advanced.conf.full
```

### 4.5.13 Validate Config Sets

#### About this task

It is important to validate any config set that has been modified, because there is currently no mechanism to prevent the system from trying to use an invalid config set. Validation is useful for determining if the config set is minimally viable for use with the system it is intended to configure.

**IMPORTANT:** Validation ensures that a config set passes all rules stored on the system. A validated config set does not necessarily equate to a config set with configuration data that will result in a properly configured system.

When validating a config set, the configurator checks the following:

- Config set has the proper directory structure and permissions.
- All configuration templates have correct YAML syntax.
- All configuration templates adhere to the configurator schema.
- All fields of type `lookup` reference values and settings that exist in the available configuration services.
- All level `required` fields in enabled services are configured (i.e., their state is `set`).
- Pre-configuration and post-configuration callback scripts ran successfully during the latest config set update.
- `cfgset validate` has run all validation rules installed on the system.

For more information on how `lookup` fields work, see the "Advanced: Lookup" section in "Configurator Data Types and How to Set Them," which is in *XC™ Series Configurator User Guide* (S-2560). For more information about validation rules, see "Validate a Config Set and List Validation Rules," also in that publication.

## Procedure

1. Validate the global config set.

```
smw# cfgset validate global
```

2. Validate the CLE config set.

This example uses CLE config set p0. Substitute the correct config set name for this site.

```
smw# cfgset validate p0
```

## 4.5.14 Make a Post-config Snapshot using snaputil

### About this task

This procedure uses `snaputil` to make an archival snapshot of the system after configuring SMW/CLE software and before booting the CLE system.

**Best Practice.** Make a snapshot and back up config sets at the same time to keep them in sync. Name the snapshot and config set backup using the same suffix and date/time stamp to help administrators identify which snapshot and config set backup pairs belong together.

For more information, see [About Snapshots and Config Set Backups](#) on page 21.

### Procedure

1. List the available snapshots on the system.

```
smw# snaputil list
```

2. Set the `SNAPSHOT` environment variable using the currently booted snapshot name.

```
smw# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')
smw# echo $SNAPSHOT
```

Setting a variable for the snapshot name enables better command substitution in later commands dealing with snapshots.

This is especially important for SMW HA systems because it makes it easier to use the exact same snapshot name for both SMWs. Using different snapshots results in HSS database (MySQL) inconsistencies, which causes problems at failover.

3. Create a new snapshot.

```
smw# snaputil create ${SNAPSHOT}.postconfig-${TODAY}
```

## 4.5.15 Make a Post-config Backup of Current Global and CLE Config Sets

### About this task

This procedure uses the `cfgset` command to create a post-install backup of the global and CLE config sets after configuring SMW/CLE software and before booting the CLE system.

### Procedure

1. Back up the current global config set.

```
smw# cfgset create --clone global global-postconfig-${TODAY}
```

## 2. Back up the current CLE config set.

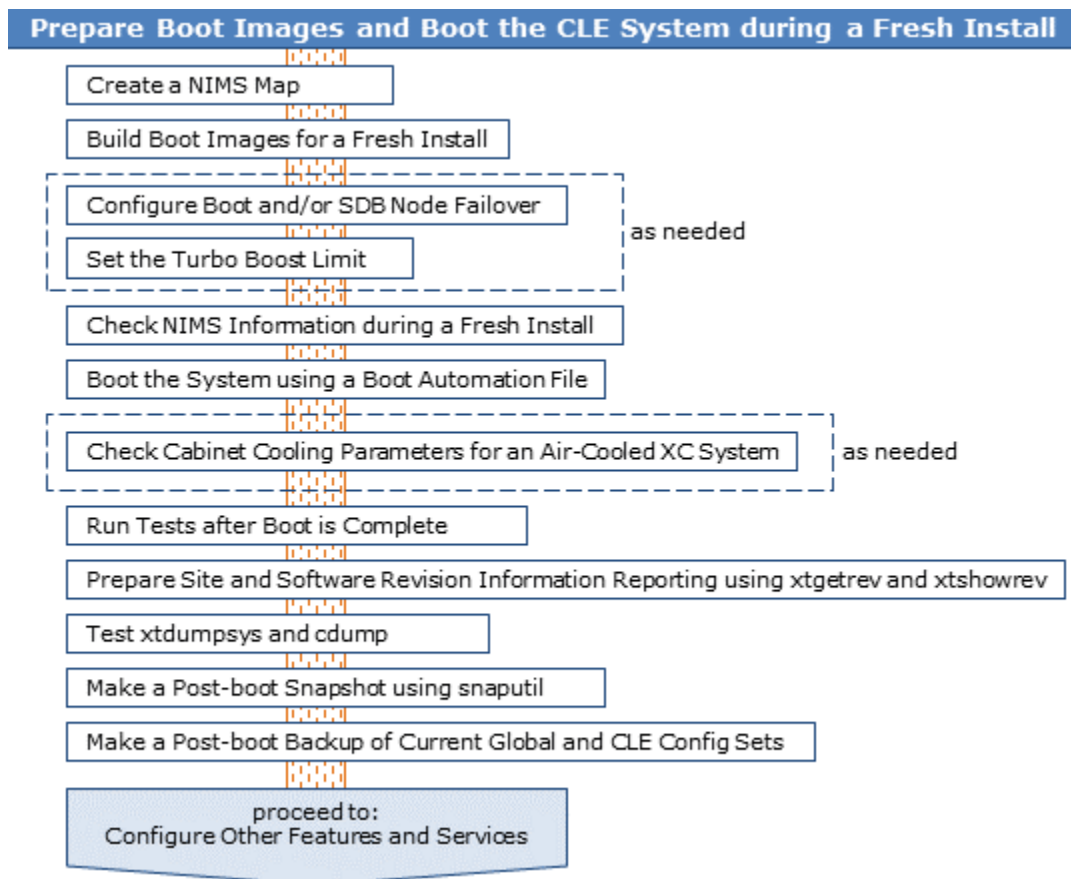
This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-postconfig-$(TODAY)
```

## 4.6 Prepare Boot Images and Boot the CLE System during a Fresh Install

Most system configuration is now complete. To prepare boot images and make sure those images are mapped to nodes correctly so that the system can boot, use the following procedures and reference topics in the order listed. The first topic helps sites decide where to place the root file system for this system, because some of these procedures depend on that decision.

Figure 33. Visual Guide to Preparing Boot Images and Booting the CLE System during a Fresh Install



Use [Installation Checklist 6: Prepare Boot Images and Boot the CLE System during a Fresh Install](#) on page 445 to track progress through this part of the fresh install process.

## 4.6.1 Create a NIMS Map

### Prerequisites

This procedure assumes that hardware is available and all previous procedures to install the operating system, discover hardware, and set up the config sets have been completed.

### About this task

For a fresh installation, a new NIMS (Node Image Mapping Service) map needs to be created. This procedure creates a NIMS map and designates it as the active map.

### Procedure

1. Create a NIMS map and set it as active.

It is typical to name a NIMS map for the CLE config set (p0 in this example) with which it is associated.

```
smw# cmap create p0
```

**Wrong name?** If the name specified for the NIMS map just created is not the desired name (it happens), use the same command to create another NIMS map with the desired name, then delete the first NIMS map (smw# `cmap delete p0`). See [Rename a NIMS Map](#) on page 409 for more information.

2. Set the new NIMS map as active.

```
smw# cmap setactive p0
```

If this command is invoked in the current snapshot, it sets the active NIMS map for the currently running system. If this command is invoked in a different, non-running snapshot, it sets the active NIMS map for that snapshot. When the SMW is rebooted into that snapshot, then the active map for that snapshot will be the active NIMS map for the running system.

## 4.6.2 Build Boot Images for a Fresh Install

### Prerequisites

This procedure assumes some knowledge of image groups: where to find them, how they are defined and managed, and how to customize them for a site. See [About Image Groups and How to Customize Them](#) on page 38 for that information.

### About this task

This procedure uses the `imgbuilder` command to build boot images. The `imgbuilder` command uses information in the `cray_image_groups` configuration file (`/var/opt/cray/imps/config/sets/global/config/cray_image_groups.yaml`) to know which images to build, how to build them, what to call the built images, and which NIMS (Node Image Mapping Service) groups to map those images to.

When invoked, the `imgbuilder` command builds all of the image specifications from one of the image groups defined in the `cray_image_groups` configuration file, beginning with the first image specification and working down the list of specifications within that group. Frequently used `imgbuilder` options include:

- bootstrap-nims** To successfully map an image to a node, `imgbuilder` also needs to know which NIMS group that node belongs to, which means the node must have its NIMS group (i.e., its "group" field) populated. But for an initial fresh install, that field may not be populated yet. To ensure that the required node information gets populated prior to building boot images, use the `--bootstrap-nims` option. With this option, `imgbuilder` looks at the "group" field of each node, and if it is empty, `imgbuilder` adds "service" or "compute" depending on the type of that node, as reported by HSS (Hardware Supervisory System).
- image-group** To specify which image group to build, use the `--image-group` option. If that option is not used, `imgbuilder` will build the group called `default`.
- map** When `imgbuilder` is invoked with the `--map` option, it maps the image in each image specification to the associated NIMS group (the `nims_group` field).
- processes=NUM** To build images in parallel, use the `--processes=NUM` option, replacing `NUM` with the number of processes to be made available for `imgbuilder` tasks. The default value is 1, and it is capped at the number of CPUs, as reported by `lscpu`.
- dry-run** To see what IMPS and NIMS commands `imgbuilder` would run, without actually running them, use the `--dry-run` option.

`imgbuilder` logs are found at `var/adm/cray/logs/imgbuilder`. For more information, see the `imgbuilder` man page or type `imgbuilder -h`.

## Procedure

1. Bootstrap NIMS (Node Image Mapping Service) using `imgbuilder` with the bootstrap option.

```
smw# imgbuilder --bootstrap-nims
```

All nodes have now been assigned to a default NIMS group, based on the node type reported by HSS.

2. Install SMW and CLE patches.

Check for any available CLE 6.0 and SMW 8.0 patches in `/var/adm/cray/release/patchsets`. This directory was created and patches (if any) downloaded to it earlier in the process. If there are any SMW HA patches (for example, `SMWHA_12.0.UPxx.PSxx`), do not install them at this time. They will be installed after SMW HA software is installed, following instructions in *XC™ Series SMW HA Installation Guide (S-0044)*.

The first substep prevents the patch scripts from creating images and mapping them to NIMS. Image creation and NIMS mapping are done at the end of this procedure instead, after the login and DAL nodes have been assigned and any changes to the `default` image group have been made.

**NOTE:** (SMW HA only) Make a note of all SMW and CLE patch sets that will be applied on the first SMW (SMW HA patch sets are not applied at this time). The second SMW must have exactly the same patch sets.

- a. Temporarily suppress building and mapping images.

```
smw# export PATCHSET_BUILD_IMAGES=false
smw# echo $PATCHSET_BUILD_IMAGES

smw# export PATCHSET_NIMS_TIMING=deferred
smw# echo $PATCHSET_NIMS_TIMING
```



- b. Follow all of the instructions in the patch `.readme` files.

These instructions will include running the `.load` script and the `.install` script for each patch, and there may be additional steps for some patches, such as running `xtzap` again to update firmware from an SMW patch.

Note that a "script" file might not be a runnable script. If necessary, copy and paste the commands into the command line and run them manually.

### ———— REASSIGN SOME NODES TO DIFFERENT NIMS GROUPS ————

3. Assign the boot and SDB nodes to the `admin` NIMS group.

For sites with boot node failover and/or SDB node failover, assign the `admin` NIMS group to both the primary and backup (failover) nodes. All nodes in the `admin` NIMS group will be assigned the admin boot image for booting. For information about the admin image, see [About the Admin Image](#) on page 36.

**NOTE:** If a custom recipe will be created and used for the SDB node(s) instead of the admin recipe (for example, to add content for a workload manager), assign the SDB node(s) to a different NIMS group, where the name of the NIMS group may have the same name as the custom recipe.

This example uses `c0-0c0s0n1` and `c0-0c0s1n1` as the admin (boot and SDB) nodes. Substitute the correct cnames for this system.

Remove from the `service` NIMS group and add to the `admin` NIMS group:

```
smw# cnode update -G service -g admin c0-0c0s0n1 c0-0c0s1n1
```

4. Assign login nodes to the `login` NIMS group.

Nodes in the `login` NIMS group will be assigned the login boot image for booting. To assign more than one node, use a space-separated list of nodes. This example uses `c0-0c1s4n0` and `c0-0c0s3n2` as the login nodes. Substitute the correct cnames for this system.

Remove from the `service` NIMS group and add to the `login` NIMS group:

```
smw# cnode update -G service -g login c0-0c1s4n0 c0-0c0s3n2
```

5. For systems using MOM nodes, assign them to the `login` NIMS group.

This will assign a MOM node the login boot image for booting. For MOM nodes that are not SDB nodes, Cray recommends booting with the login image so that the MOM node will have the same capabilities as a login node. This example uses `c0-1c1s3n0` as the MOM node. Substitute the correct cname for this system.

Remove from the `service` NIMS group and add to the `login` NIMS group:

```
smw# cnode update -G service -g login c0-1c1s3n0
```

If a custom recipe will be created and used for the MOM node(s) instead of the login recipe, assign the MOM node(s) to a different NIMS group, where the name of the NIMS group may have the same name as the custom recipe. Cray recommends adding the login recipe as a subrecipe of the custom MOM recipe.

6. For systems using direct-attached Lustre (DAL), assign DAL service nodes to the `dal` NIMS group.

Nodes in the `dal` NIMS group will be assigned the DAL boot image for booting. To assign more than one node, use a space-separated list of nodes. This example uses `c0-0c0s2n1` and `c0-0c0s2n2`. Substitute the correct cnames for this system.

Remove from the `service` NIMS group and add to the `dal` NIMS group:

```
smw# cnode update -G service -g dal c0-0c0s2n1 c0-0c0s2n2
```

7. For systems that have both x86-64 and AArch64 nodes, add the AArch64 nodes to an AArch64-specific NIMS group.

Nodes in an AArch64 NIMS group will be assigned the appropriate AArch64 boot image for booting.

- a. Remove all AArch64 compute nodes from the `compute` NIMS group and add them to the `compute_aarch64` NIMS group.

```
smw# cnode update -G compute -g compute_aarch64 \
--filter arch=aarch64,type=compute
```

- b. Remove any AArch64 login nodes from their current NIMS group and add them to the `login_aarch64` NIMS group.

If this system will have AArch64 login nodes, then those nodes must be repurposed AArch64 compute nodes. Use one of the following commands, depending on whether the nodes were repurposed earlier in the process using the procedure [Repurpose a Compute or Service Node](#) on page 147. The examples use nodes `c0-0c1s1n1` and `c0-0c2s1n2`. Substitute the correct cnames of the AArch64 login nodes for this system.

- If the nodes were repurposed earlier in the process, then remove them from the `service` NIMS group and add them to the `login_aarch64` NIMS group.

```
smw# cnode update -G service -g login_aarch64 c0-0c1s1n1 c0-0c2s1n2
```

- If the nodes were NOT repurposed earlier in the process, do that now. After the nodes have been repurposed, remove them from the `compute_aarch64` NIMS group and add them to the `login_aarch64` NIMS group.

```
smw# su - crayadm
crayadm@smw> xtcli mark_node service c0-0c1s1n1 c0-0c2s1n2
crayadm@smw> exit
smw#
```

```
smw# cnode update -G compute_aarch64 -g login_aarch64 c0-0c1s1n1 c0-0c2s1n2
```

8. Confirm that the intended nodes were added to the NIMS groups.

```
smw# cnode list --fields name,type,arch --filter group=admin
```

```
smw# cnode list --fields name,type,arch --filter group=login
```

```
smw# cnode list --fields name,type,arch --filter group=dal
```

```
smw# cnode list --fields name,type,arch --filter group=compute_aarch64
```

```
smw# cnode list --fields name,type,arch --filter group=login_aarch64
```

————— PREPARE CRAY IMAGE GROUPS AND CUSTOM RECIPES —————

9. Customize Cray image groups.

Customize the `cray_image_groups` configuration file, as needed, by editing `/var/opt/cray/imps/config/sets/global/config/cray_image_groups.yaml`. Move image specifications (stanzas) into the `default` image group for anything to be built by default, and modify or create other image groups as appropriate for this site (for more information, see [About Image Groups and How to Customize Them](#) on page 38).

- a. Ensure that the `admin` stanza is in the `default` image group.

If not already there, add it to the end of the `default` image group.

```
cray_image_groups:
  default:
    ...
    - recipe: "admin_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "admin{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-created{date}"
      export_format: "cpio"
      nims_group: "admin"
```

- b. For systems using DAL, ensure that this DAL stanza is in the `default` image group, or customize and use the `tmpfs-w-dal` image group, which already has it.

```
cray_image_groups:
  default:
    ...
    - recipe: "dal_cle_{cle_release_lowercase}_centos_6.5_ari"
      dest: "dal{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_centos_6.5-created{date}"
      export_format: "cpio"
      nims_group: "dal"
    ...
```

- c. For systems that have both x86-64 and AArch64 nodes, ensure that AArch64 stanzas are in the `default` image group.

If not already there, add the following stanzas to the end of the `default` image group, as needed.

```
cray_image_groups:
  default:
    ...
    - recipe: "compute_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "compute{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-aarch64-created{date}"
      arch: "aarch64"
      export_format: "cpio"
      nims_group: "compute_aarch64"
    - recipe: "login_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "login{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-aarch64-created{date}"
      arch: "aarch64"
      export_format: "cpio"
      nims_group: "login_aarch64"
```

- d. For systems using netroot for either compute or login nodes, those images will be created in a different procedure later in the fresh install process.

For more information about netroot, see [Where to Place the Root File System: tmpfs versus netroot](#) on page 36.

- e. Ensure that any site custom recipes are added as stanzas to the `default` image group or a site-specific image group so that they will get built.

If a custom recipe will be used for MOM nodes, add a MOM-specific stanza.

Note that beginning with CLE 6.0.UP06, an `export_format` field has been added to stanzas that will be used to export images, and an `arch` field has been added to stanzas for recipes that can be used to build images for different architectures.

---

## BUILD AND MAP IMAGES

---

There are two approaches to building images and mapping them to NIMS groups. Choose only ONE of them:

- combined** Uses `imgbuilder` with the `--map` option to both build and map images.  
To choose the "combined" approach, use step [10](#) on page 238.
- separate** Uses `imgbuilder` to build the images, then uses `cnode` to map them manually.  
To choose the "separate" approach, use step [11](#) on page 238.

Note that in either case, `imgbuilder` output will include instructions to push any `initrd` netroot image to the boot node. Disregard those instructions if netroot is not yet configured. Not sure? Look for "initrd-compute-large" or "initrd-login-large" in image names when checking NIMS information later in the fresh install process.

#### 10. ("Combined") Build images and map them to NIMS groups.

Create a set of images as defined in `/var/opt/cray/imps/config/sets/global/config/cray_image_groups.yaml` and map them to the specified NIMS groups using the `--map` option.

Building images can take a significant amount of time depending on the number and type of recipes in the image group. To reduce that time, images can be built in parallel. Build images using one of the following commands.

- To build all images in the `default` image group in parallel (this is typically quicker):

```
smw# imgbuilder --map --processes=NUM
```

Replace `NUM` with the number of images to be built concurrently (capped at the number of CPUs as reported by `lscpu`).

- To build all images in the `default` image group serially:

```
smw# imgbuilder --map
```

For more information about building images in parallel, see [About Parallel Image Creation](#) on page 40.

**Trouble?** If `imgbuilder` warns about a NIMS group not existing in the current NIMS map, check `cray_image_groups.yaml` for any `nims_group` entries that are not used in this NIMS map. Either remove those entries (because they will build images that are not used) or add the missing NIMS groups.

When that command is finished, combined image building and NIMS mapping will be complete. Skip the next step, which builds images and maps them separately.

#### 11. ("Separate") Build images and then manually map them to NIMS groups.

As an alternative to mapping the images using the `--map` option, that mapping can be done manually.

##### a. Build the images.

- To build all images in the `default` image group in parallel (this is typically quicker):

```
smw# imgbuilder --processes=16
```

For 16, substitute a different number of processes to make available in the pool, as needed.

- To build all images in the `default` image group serially:

```
smw# imgbuilder
```

- b. Map the images to specified NIMS groups.

Replace the cpio file names in these examples with the file names from the `imgbuilder` output in step a.

If any other boot images have been created for special nodes, assign them with similar `cnode update` commands filtered for the NIMS groups to which those special nodes have been assigned.

To map the images to specified NIMS groups:

```
smw# cd /var/opt/cray/imps/boot_images
smw# ls -ltr

smw# cnode update -i compute_img.cpio --filter group=compute
smw# cnode update -i service_img.cpio --filter group=service
smw# cnode update -i login_img.cpio --filter group=login
smw# cnode update -i dal_img.cpio --filter group=dal
```

### 4.6.3 Configure Boot and/or SDB Node Failover

#### Prerequisites

SKIP THIS PROCEDURE if boot and/or SDB node failover will not be configured for this system.

Boot/SDB node failover requirements:

- Both the primary node and the backup node must have a Fibre Channel or SAS connection to the boot RAID.
- Both the primary node and the backup node must have an Ethernet connection to the network shared with the SMW in order to PXE boot and transfer data as a tier1 node.
- The primary and backup nodes must not be on the same blade.
- The boot and SDB nodes must not be on the same blade.



**CAUTION:** The system will fail if a blade containing both the boot node and the SDB node fails, because Cray does not support concurrent failover of boot and SDB nodes. Therefore, the boot and SDB nodes and their backups (for boot/SDB node failover) must be on different blades.

This procedure assumes that the CLE system is not booted.

#### About this task

If a secondary (backup) boot node is configured, boot node failover will occur automatically if the primary boot node fails. Similarly, if a secondary (backup) service database (SDB) node is configured, SDB node failover will occur automatically if the primary SDB node fails. This procedure configures the system for boot and/or SDB node failover.

For the examples in this procedure, the following cnames are used. In commands, substitute the actual cnames for this system.

- boot node: primary c0-0c0s0n1, backup c0-2c0s4n1
- SDB node: primary c0-0c0s1n1, backup c0-4c0s3n1

Also, the examples use config set p0. Substitute the appropriate CLE config set for this system.

## Procedure

### 1. Configure `cray_multipath` for boot/SDB node failover.

- a. Determine whether `cray_multipath` is enabled.

```
smw# cfgset get cray_multipath.enabled global
smw# cfgset get cray_multipath.enabled p0
```

If neither setting is set to true, skip the rest of this step.

- b. Determine whether `cray_multipath` inherits from the global config set.

```
smw# cfgset get cray_multipath.inherit p0
```

- c. Determine which nodes are included in `cray_multipath.settings.multipath.data.node_list`.

The multipath node list must contain both the primary and backup boot nodes if configuring boot node failover, and it must contain both the primary and backup SDB nodes if configuring SDB node failover.

- If `cray_multipath` inherits from the global config set:

```
smw# cfgset get cray_multipath.settings.multipath.data.node_list global
```

- If `cray_multipath` does NOT inherit from the global config set:

```
smw# cfgset get cray_multipath.settings.multipath.data.node_list p0
```

If the multipath node list already contains the correct nodes, skip the rest of this step.

- d. Add nodes to `cray_multipath.settings.multipath.data.node_list`.

Use the `--add` option to add as many cnames as needed.

- If `cray_multipath` inherits from the global config set:

```
smw# cfgset modify --add c0-0c0s0n1 --add c0-2c0s4n1 \
--add c0-0c0s1n1 --add c0-4c0s3n1 \
cray_multipath.settings.multipath.data.node_list global
```

- If `cray_multipath` does NOT inherit from the global config set:

```
smw# cfgset modify --add c0-0c0s0n1 --add c0-2c0s4n1 \
--add c0-0c0s1n1 --add c0-4c0s3n1 \
cray_multipath.settings.multipath.data.node_list p0
```

### 2. Configure `cray_node_groups` for boot node failover, as needed.

- a. Check the members of the `boot_nodes` node group.

The `boot_nodes` node group must contain both the primary and backup boot nodes.

```
smw# cfgset get cray_node_groups.settings.groups.data.boot_nodes.members p0
```

- b. If the backup boot node is not in the members list, add it now.

```
smw# cfgset modify --add c0-2c0s4n1 \
cray_node_groups.settings.groups.data.boot_nodes.members p0
```

### 3. Configure `cray_node_groups` for SDB node failover, as needed.

- a. Check the members of the `sdb_nodes` node group.

The `sdb_nodes` node group must contain both the primary and backup SDB nodes.

```
smw# cfgset get cray_node_groups.settings.groups.data.sdb_nodes.members p0
```

- b. If the backup SDB node is not in the members list, add it now.

```
smw# cfgset modify --add c0-4c0s3n1 \
cray_node_groups.settings.groups.data.sdb_nodes.members p0
```

#### 4. Configure `cray_persistent_data` for boot/SDB node failover.

- a. Check the members of the NFS clients group.

The NFS clients group must contain both the `boot_nodes` and `sdb_nodes` node groups.

```
smw# cfgset get \
cray_persistent_data.settings.mounts.data./var/lib/nfs.client_groups p0
```

- b. If either node group is missing, add it now.

```
smw# cfgset modify --add boot_nodes --add sdb_nodes \
cray_persistent_data.settings.mounts.data./var/lib/nfs.client_groups p0
```

#### 5. Configure `cray_scalable_services` for boot/SDB node failover.

- a. Check the list of tier1 groups.

Tier1 groups must contain both the `boot_nodes` and `sdb_nodes` node groups.

```
smw# cfgset get \
cray_scalable_services.settings.scalable_service.data.tier1_groups p0
```

- b. If either node group is missing, add it now.

```
smw# cfgset modify --add boot_nodes --add sdb_nodes \
cray_scalable_services.settings.scalable_service.data.tier1_groups p0
```

#### 6. Configure `cray_net` for boot/SDB node failover, as needed.

If this was done while editing `cray_net_worksheet.yaml` earlier in the fresh install process, skip this step.

- a. Generate a new worksheet for `cray_net`.

```
smw# cfgset update -m prepare p0
```

- b. Copy the new worksheets to the work area set up earlier in the process.

```
smw# cp -a /var/opt/cray/imps/config/sets/p0/worksheets/* \
/var/adm/cray/release/p0_worksheet_workarea/*
```

- c. Edit the new `cray_net` worksheet.

```
smw# vi /var/adm/cray/release/p0_worksheet_workarea/cray_net_worksheet.yaml
```

- d. If configuring boot node failover, configure a host for the backup boot node.

**NOTE:** The host name for the primary and backup boot node should both be set to a name that includes the machine name and "boot" such as `boot-panda`. The aliases can be different so that the `/etc/hosts` entry for the cname has the host name alias.



The backup boot node host should have `standby_node` set to `true`.

```
cray_net.settings.hosts.data.common_name.backup_bootnode: null
cray_net.settings.hosts.data.backup_bootnode.description: backup Boot node for the system
cray_net.settings.hosts.data.backup_bootnode.aliases:
- cray-boot2
cray_net.settings.hosts.data.backup_bootnode.hostid: c0-2c0s4n1
cray_net.settings.hosts.data.backup_bootnode.host_type: admin
cray_net.settings.hosts.data.backup_bootnode.hostname: boot-panda
cray_net.settings.hosts.data.backup_bootnode.standby_node: true

cray_net.settings.hosts.data.backup_bootnode.interfaces.common_name.hsn_boot_alias: null
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.name: ipogif0:1
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.description:
  Well known address used for boot node services.
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.vlan_id: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.vlan_etherdevice: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.bonding_slaves: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.bonding_module_opts:
  mode=active-backup miimon=100
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.aliases: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.network: hsn
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.ipv4_address: 10.131.255.254
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.mac: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.startmode: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.bootproto: static
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.mtu: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.extra_attributes: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.module: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.params: ''
#cray_net.settings.hosts.data.backup_bootnode.interfaces.hsn_boot_alias.unmanaged_interface: false

cray_net.settings.hosts.data.backup_bootnode.interfaces.common_name.primary_ethernet: null
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.name: eth0
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.description:
  Ethernet connecting boot node to the SMW.
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.vlan_id: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.vlan_etherdevice: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.bonding_slaves: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.bonding_module_opts:
  mode=active-backup miimon=100
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.aliases: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.network: admin
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.ipv4_address: 10.3.1.254
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.mac: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.startmode: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.bootproto: static
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.mtu: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.extra_attributes: []
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.module: ''
cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.params: ''
#cray_net.settings.hosts.data.backup_bootnode.interfaces.primary_ethernet.unmanaged_interface: false
```

- e. If configuring SDB node failover, configure a host for the backup SDB node.

The host name for the primary and backup SDB node should both be set to `sdb`. The aliases can be different so that the `/etc/hosts` entry for the crame has the host name aliases.

The backup SDB node host should have `standby_node` set to `true`.

```
cray_net.settings.hosts.data.common_name.backup_sdbnode: null
cray_net.settings.hosts.data.backup_sdbnode.description: backup SDB node for the system
cray_net.settings.hosts.data.backup_sdbnode.aliases:
- cray-sdb2
cray_net.settings.hosts.data.backup_sdbnode.hostid: c0-4c0s3n1
cray_net.settings.hosts.data.backup_sdbnode.host_type: admin
cray_net.settings.hosts.data.backup_sdbnode.hostname: sdb
cray_net.settings.hosts.data.backup_sdbnode.standby_node: true

cray_net.settings.hosts.data.backup_sdbnode.interfaces.common_name.hsn_boot_alias: null
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.name: ipogif0:1
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.description:
  Well known address used for SDB node services.
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.vlan_id: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.vlan_etherdevice: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.bonding_slaves: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.bonding_module_opts:
  mode=active-backup miimon=100
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.aliases: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.network: hsn
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.ipv4_address: 10.131.255.253
```



```

cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.mac: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.startmode: auto
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.bootproto: static
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.mtu: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.extra_attributes: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.module: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.params: ''
#cray_net.settings.hosts.data.backup_sdbnode.interfaces.hsn_boot_alias.unmanaged_interface: false

cray_net.settings.hosts.data.backup_sdbnode.interfaces.common_name.primary_ethernet: null
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.name: eth0
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.description:
    Ethernet connecting SDB node to the SMW.
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.vlan_id: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.vlan_etherdevice: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.bonding_slaves: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.bonding_module_opts:
    mode=active-backup miimon=100
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.aliases: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.network: admin
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.ipv4_address: 10.3.1.253
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.ipv4_secondary_addresses: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.mac: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.startmode: auto
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.bootproto: static
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.mtu: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.extra_attributes: []
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.module: ''
cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.params: ''
#cray_net.settings.hosts.data.backup_sdbnode.interfaces.primary_ethernet.unmanaged_interface: false

```

- f. Import the completed `cray_net` worksheet to the CLE config set (p0 in the example).

```

smw# cfgset update --worksheet-path \
/var/adm/cray/release/p0_worksheet_workarea/cray_net_worksheet.yaml p0

```

7. Update and validate the CLE and global config sets.

Among other things, this will regenerate the CLE `/etc/hosts` file so that it contains the appropriate backup node settings.

```

smw# cfgset update p0
smw# cfgset validate p0

smw# cfgset update global
smw# cfgset validate global

```

8. Confirm that host name entries exist in the CLE `/etc/hosts` file for all boot and SDB nodes.

```

smw# egrep "boot|sdb" \
/var/opt/cray/imps/config/sets/p0/files/roles/common/etc/hosts

```

9. Set the primary and backup nodes.

On a booted CLE system, it would be necessary to shut down and halt the primary and backup nodes before using the following commands. However, because this is a fresh install and the CLE system has not yet been booted, those preliminary steps are not needed.

```

smw# su - crayadm

```

- For boot node failover, use the `-b` option:

```

crayadm@smw> xtcli part_cfg update p0 -b c0-0c0s0n1,c0-2c0s4n1

```

- For SDB node failover, use the `-d` option:

```

crayadm@smw> xtcli part_cfg update p0 -d c0-0c0s1n1,c0-4c0s3n1

```

```
crayadm@smw> exit
smw#
```

**10.** Ensure that the backup node has the correct NIMS group and boot image.

- a. Check which NIMS group and boot image are being used for the primary and backup nodes.

For boot node failover:

```
smw# cnode list c0-0c0s0n1
smw# cnode list c0-2c0s4n1
```

For SDB node failover:

```
smw# cnode list c0-0c0s1n1
smw# cnode list c0-4c0s3n1
```

If the backup node has the same NIMS group and boot image assigned as the primary node, skip the remaining steps.

- b. Remove the old NIMS group from the backup node.

For boot node failover:

```
smw# cnode update -G oldNIMSgroup c0-2c0s4n1
```

For SDB node failover:

```
smw# cnode update -G oldNIMSgroup c0-4c0s3n1
```

- c. Assign the primary node's NIMS group and boot image to the backup node and confirm the change.

For boot node failover:

```
smw# cnode update -g primaryNIMSgroup \  
-i /path/to/primary/bootimage c0-2c0s4n1  
smw# cnode list c0-2c0s4n1
```

For SDB node failover:

```
smw# cnode update -g primaryNIMSgroup \  
-i /path/to/primary/bootimage c0-4c0s3n1  
smw# cnode list c0-4c0s3n1
```

Boot and/or SDB node failover has been configured. Changes will take effect when the CLE system is booted later in the fresh install process.

To ensure that the STONITH settings persist across boots, several lines must be added to the site boot automation file. Similarly, to ensure that the `xtbootsys` shutdown process handles backup nodes properly, a change must be made to the shutdown automation file for this site. Instructions for making those changes are included in a procedure later in the fresh install process.

#### 4.6.4 Set the Turbo Boost Limit

Turbo boost limiting is supported on the Intel® Xeon® processor Scalable family. Turbo boost limiting is NOT supported on Intel® Xeon Phi™ "Knights Landing" (KNL) or on Intel® Xeon® "Sandy Bridge" processors.

Because Intel processors have a high degree of variability in the amount of turbo boost each processor can supply, limiting the amount of turbo boost can reduce performance variability and reduce power consumption. Turbo boost can be limited by setting the `turbo_boost_limit` kernel parameter to one of these valid values:

Value	Result
0	Disable turbo boost.
100	Limits turbo boost to 100 MHz.
200	Limits turbo boost to 200 MHz.
300	Limits turbo boost to 300 MHz.
400	Limits turbo boost to 400 MHz.
999 (default)	No limit is applied.

The limit applies only when a high number of cores are active. On an N-core processor, the limit is in effect when the active core count is N, N-1, N-2, or N-3. For example, on a 12-core processor, the limit is in effect when 12, 11, 10, or 9 cores are active.

## Set or Change the Turbo Boost Limit Parameter

To make a persistent change, use `cnode` (as `crayadm` or `root`) to change the parameter. This change will take effect later when the nodes are rebooted. Note that the following commands target all nodes or all compute nodes. To specify individual nodes, add their `cnames` at the end of the command line.

1. To list the current kernel parameters:

```
smw# cnode list
```

2. To change the `turbo_boost_limit` kernel parameter for all compute nodes, substitute one of the values listed above for `value` in this command:

```
smw# cnode update --filter group=compute \
--add-parameter turbo_boost_limit=value
```

3. To remove the change, if needed, use one of these commands:

```
smw# cnode update --filter group=compute \
--remove-parameter turbo_boost_limit
```

## 4.6.5 Check NIMS Information during a Fresh Install

### About this task

This procedure lists NIMS (Node Image Mapping Service) information: which maps are active on the SMW and what NIMS information is stored for each node.

### Procedure

1. Check active NIMS maps.

```
smw# cmap list
```

## 2. Check the default config set of the active NIMS map.

```
smw# cmap list --fields default_config_set map_name
```

If this is not the desired default config set, use [Set Default Config Set for a NIMS Map](#) on page 408 to change it. If selected nodes need to use a different config set, see [Set Config Set for a Node](#) on page 409.

### ————— CHECK NIMS INFORMATION —————

When checking NIMS information, things to look for include:

- Are nodes in the correct NIMS groups?
- Does each node have the correct boot image?
- Does each node have the correct config set assigned?

## 3. Check NIMS information for all nodes.

```
smw# cnode list
```

## 4. Check NIMS information for each NIMS group.

```
smw# cnode list --filter group=admin
smw# cnode list --filter group=service
smw# cnode list --filter group=login
smw# cnode list --filter group=compute
```

Check any additional NIMS groups that may have been created for netroot compute and login nodes (typically created only when netroot is used on a subset of compute and login nodes instead of all of them, so that the NIMS compute and login groups cannot be used for that subset of nodes).

```
smw# cnode list --filter group=compute_netroot
smw# cnode list --filter group=login_netroot
```

Check any additional NIMS groups that may have been created for DataWarp with Fusion IO SSDs.

```
smw# cnode list --filter group=fio-service
```

Check any additional NIMS groups that may have been created with WLM (workload manager) or other site names.

```
smw# cnode list --filter group=wlm-admin
smw# cnode list --filter group=wlm-service
smw# cnode list --filter group=wlm-login
```

## 4.6.6 Boot the System using a Boot Automation File

### Prerequisites

This procedure assumes that configuration and image preparation are complete and the system is now ready to boot.

## About this task

This procedure describes how to customize a boot automation file and use it to boot the XC system with `xtbootsys`. For more information about boot automation files, see [About Boot Automation Files](#) on page 34.

## Procedure

### ———— CREATE SITE BOOT AND SHUTDOWN AUTOMATION FILES ————

1. Create a site boot automation file.

Copy the Cray generic boot automation file and rename it. Add site customizations, as needed. For sites booting tmpfs images (instead of netroot) with no SDB node failover, no changes may be necessary. Sites that choose to boot netroot images will make those changes later in the process after this first boot with tmpfs.

Replace `<hostname>` with the host name of the system that will use this automation file.

```
smw# cp -p /opt/cray/hss/default/etc/auto.generic \
/opt/cray/hss/default/etc/auto.<hostname>.start
```

2. Create a site automation file for shutting down the system.

Copy the Cray shutdown automation file and rename it. Add site customizations, as needed. For example, customization may be needed to cleanly shut down queues for the workload manager (WLM) on MOM or SDB nodes. The specific commands will vary based on the WLM.

Replace `<hostname>` with the host name of the system that will use this automation file.

```
smw# cp -p /opt/cray/hss/default/etc/auto.xtshutdown \
/opt/cray/hss/default/etc/auto.<hostname>.stop
```

### ———— CUSTOMIZE THE SITE AUTOMATION FILES ————

Sites can create a separate boot automation file for each of the following customizations, as needed.

3. If the SDB boot image is too large for a PXE boot (often the case if a WLM is installed in that image), change `auto.<hostname>.start` to enable booting the SDB node(s) via HSN rather than PXE.

See [About Boot Automation Files](#) on page 34 and [About the Admin Image](#) on page 36 for more information.

4. If this system uses DAL (direct-attached Lustre), change `auto.<hostname>.start` to enable service nodes to boot before compute nodes.

For systems with DAL, the following boot order is necessary:

1. Boot + SDB (if SDB image small enough to PXE boot)
2. SDB (if SDB image too large to PXE boot)
3. Service
4. Compute

5. If boot or SDB node failover is used, add boot node or SDB node failover to `auto.<hostname>.start`.

If boot node failover or SDB node failover will be used, add settings to the boot automation file to ensure that STONITH is enabled on the blades that contain the primary and backup nodes. The STONITH setting does

not survive a power cycle or any other action that causes the `bcsysd` daemon to restart. Adding these lines to the boot automation file maintains the STONITH setting.

Note that it is the blade that must be specified in the `xtdaemonconfig` command, not the node. For example, to specify the blade that contains `c0-0c0s0n1` as the primary boot node, use `c0-0c0s0`.

- For boot node failover, add these lines **before** the line for booting the boot node (substitute the correct blade cnames for this system):

```
# Set STONITH for primary boot node
lappend actions {crms_exec "xtdaemonconfig c0-0c0s0 stonith=true"}
# Set STONITH for the backup boot node
lappend actions {crms_exec "xtdaemonconfig c0-2c0s4 stonith=true"}
```

- For SDB node failover, add these lines **before** the line for booting the SDB node (substitute the correct blade cnames for this system):

```
# Set STONITH for primary SDB node
lappend actions {crms_exec "xtdaemonconfig c0-0c0s1 stonith=true"}
# Set STONITH for the backup SDB node
lappend actions {crms_exec "xtdaemonconfig c0-4c0s3 stonith=true"}
```

6. If boot or SDB node failover is used, enable the `xtfailover_halt` command in the `auto.<hostname>.stop` file.

Uncomment the `lappend actions` line in `auto.<hostname>.stop`. The `xtfailover_halt` command ensures that the `xtbootsys` shutdown process sends a STOP NMI to the failover nodes.

```
# Enable the following line if boot or sdb failover is enabled:
lappend actions { crms_exec \
"/opt/cray/hss/default/bin/xtfailover_halt --partition $data(partition,given) --shutdown" }
```

7. If `cray_login.settings.login_nodes.data.login_prohibited_after_boot` is set to `true`, then to allow user access later, remove the `/etc/nologin` file using one of the following methods:

- Remove it manually.

This can be done only after all of the CLE nodes have been booted and the system is ready for users to log in. To choose this option, wait until step 9 on page 249.

- Set up the boot automation file to remove it.

Edit the site boot automation file.

```
smw# vi /opt/cray/hss/default/etc/auto.<hostname>.start
```

Add the following lines, placing them after the lines that boot all of the compute nodes and after any other special commands that prepare the system for user access.

```
# Remove /etc/nologin from all service nodes as the last step in the system boot.
lappend actions { crms_exec_on_bootnode "sdb" "pcmd -r -n ALL_SERVICE 'rm /etc/nologin'" }
```

## BOOT THE SYSTEM

8. Run `xtbootsys` with `auto.<hostname>.start`.

```
smw# su - crayadm
crayadm@smw> xtbootsys -a auto.<hostname>.start
```

**Trouble?** If there are any problems booting CLE, see the *XC™ Series Boot Troubleshooting Guide (S-2565)* for techniques to determine what might be causing the problem.

9. Remove the `/etc/nologin` file manually after the system boots, as needed.

If `cray_login.settings.login_nodes.data.login_prohibited_after_boot` is set to `true`, and the `/etc/nologin` file was NOT removed by means of an entry in the boot automation file in step 7 on page 248, remove it manually after all of the CLE nodes have been booted and the system is ready for users to log in.

```
sdb# pcmd -r -n ALL_SERVICE "rm /etc/nologin"
```

#### Build image roots on the SMW during system boot to save time.

Image building can be done at any time on the SMW without negative impact to the running CLE system. To save time, the following installation tasks can be started on the SMW while the CLE nodes are booting.

- Build netroot images on the SMW. See [Configure Netroot Images](#) on page 262.
- Build the PE image root on the SMW. See [Install Cray Programming Environment \(PE\) Software on x86-64](#).
- Build any WLM or other custom image roots on the SMW.

## 4.6.7 Check Cabinet Cooling Parameters for an Air-Cooled XC System

### Prerequisites

- This is an XC Series air-cooled (XC-AC) system running SMW 8.x / CLE 6.x software.
- Patches have been installed.
- The system is booted.

### About this task

Cray provides the `xtaccheckcool` tool to enable sites to check and/or set the cooling parameters of an XC-AC system. Use this tool in the following circumstances:

- after a fresh install
- after a software update
- as part of customizing a preinstalled system
- after adding hardware to a system
- periodically, to verify that parameters are set correctly for an operational system

### Procedure

1. Display current and recommended cabinet cooling parameters.

Substitute the elevation (in feet) of this site.

```
smw# xtaccheckcool -e elevation_in_feet
```

2. Write the recommended cabinet cooling parameters to all cabinet controllers, as needed.

This command uses the `-w` option to write the recommended cooling parameters for the specified elevation to every cabinet in the system. Substitute the elevation (in feet) of this site.

```
smw# xtaccheckcool -w -e elevation_in_feet
```

To target a specific cabinet, use the `-t cname` option, where *cname* is the cname of the cabinet.

For more information and example output, see the `xtaccheckcool(8)` man page.

## 4.6.8 Run Tests after Boot is Complete

### Prerequisites

This procedure assumes the following:

- The system has completed booting.
- The compute nodes are "interactive" (i.e., not under workload manager control).
- ALPS is available.

If ALPS is not available and Slurm is used as the workload manager (WLM), then the compute nodes can be either "interactive" or "batch" and `srun` (the Slurm command equivalent to `aprun`) should be used instead of the `aprun` commands in the steps that follow.

### About this task

Log in to the login node as `crayadm`. This can be done from the SMW to the boot node to the login node, or directly from another computer to the login node without passing through the SMW and boot node. Then perform these rudimentary functionality checks.

### Procedure

1. Run `apstat` to get the number of nodes to use for the following commands.

```
crayadm@login> NUMNODES=$((apstat -v | grep XT | awk '{print $3}'))
crayadm@login> echo NUMNODES is $NUMNODES
```

2. Verify that all nodes run (from `/tmp`).

```
crayadm@login> cd /tmp
crayadm@login> aprun -b -n $NUMNODES -N 1 /bin/cat /proc/sys/kernel/hostname
```

3. Verify that the home directory is working by running a job.

```
crayadm@login> cd ~
crayadm@login> aprun -b -n $NUMNODES -N 1 /bin/cat /proc/sys/kernel/hostname
```

4. Verify that the Lustre directory is working by running a job.



```
crayadm@login> cd /lustre_file_system
crayadm@login> aprun -b -n $NUMNODES -N 1 /bin/cat /proc/sys/kernel/hostname
```

#### CHECK CURRENT STATE OF COMPUTE NODE SSDs

The next step is intended only for XC systems that have compute nodes with SSDs, for example, systems with DataWarp SSDs or Intel® Xeon Phi™ "Knights Landing" processors.

5. Run `xtcheckssd` to ensure that SMW databases have the current state of compute node SSDs.

```
root@login# pcmd -r -n ALL_COMPUTE "/opt/cray/ssd/bin/xtcheckssd"
```

## 4.6.9 Prepare Site and Software Revision Information Reporting using `xtgetrev` and `xtshowrev`

### Prerequisites

To run `xtgetrev`, the boot node must be booted and accessible.

### About this task

System administrators use the `xtgetrev` and `xtshowrev` commands to gather and display machine, software revision, Field Notice (FN), and patch set information. The `xtgetrev` command collects information from the administrator and from the SMW and boot node. The `xtshowrev` command displays that information, even when CLE is not running. These tools are useful for gathering information to send to Cray after installing a software upgrade, FN, or patch set and for help with troubleshooting.

This procedure describes how to use these two tools on a Cray XC Series system. These steps (except for running `xtshowrev`) must be executed as root.

**ATTENTION:** Any information that is submitted to `site_install_data@cray.com` will be used only within Cray, Inc. and will not be made public. The `xtshowrev` command does not submit any information to Cray automatically.

### Procedure

1. Load the module to enable use of the tools.

```
smw# module load xtshowrev
```

2. Run `xtgetrev` to create and populate the initial files.

Only root can run this command. The first time `xtgetrev` is executed, when there are no files populated, the tool will prompt for site information. If the boot node does not have passwordless ssh, then the tool will prompt for the password.

This example uses `CRAY/INTERNAL` as the site name and `9999` as the serial number of the machine. Substitute the actual values for this site.

```
smw# xtgetrev
xtgetrev: No site information has been defined.
```

```
Site name: CRAY/INTERNAL
Serial Number: 9999
System Name [panda1]:
System Type [XC40]:
```

<snip>

**Trouble?** If `xtgetrev` does not allow entry of those values, it may be because the initial configuration files have been created already. In that case, manually edit `/etc/opt/cray/release/pkginfo/site_config` and modify 'site name:' and 'serial number:' values.

```
smw# vi /etc/opt/cray/release/pkginfo/site_config
```

3. Run `xtshowrev` to see the formatted information.

Note the prompt, which indicates that any user can run this command.

```
user@smw> xtshowrev
Site:                CRAY/INTERNAL
S/N:                 9999
System Type:         XC40
Install Date:        2016-06-01
```

<snip>

```
user@smw>
```

## 4.6.10 Test `xtdumpsys` and `cdump`

### Prerequisites

This procedure assumes that the system has been booted.

### About this task

This procedure tests the `xtdumpsys` and `cdump` tools. The example output is for illustrative purposes only. Actual output may differ for the current release.

### Procedure

1. Start an `xtdumpsys` typescript.

Start a new window. Start a typescript session for `xtdumpsys` in that new window.

```
smw# su - crayadm
crayadm@smw> export TODAY=`date +%Y%m%d`
crayadm@smw> . /etc/opt/cray/release/cle-release
crayadm@smw> mkdir -p /home/crayadm/dump/${TODAY}_${BUILD}
crayadm@smw> cd /home/crayadm/dump/${TODAY}_${BUILD}
crayadm@smw> script -af hss.xtdumpsys
```

2. Start `xtdumpsys`.

Start the dump, but do not press **Ctrl-d** until step 5 on page 253. When `xtdumpsys` asks for a dump reason, it will have created the dump directory.

```
crayadm@smw> xtdumpsys
INFO: Beginning dump
INFO: Gathering system partition information
INFO: Gathering system hardware information
INFO: No session specified, defaulting to current.
INFO: Moving temporary log files to the dump directory.
INFO:
#####
INFO: # Your dump is available in /var/opt/cray/dump/p0-YYYYMMDDtHHMMSS-
NNNNNNNNNN #
INFO:
#####
Enter reason for dump:
(an EOF terminates input, usually CTRL-D)
```

### 3. Start a `cdump` typescript in a different window.

Start another window. Start a typescript session for `cdump` in that window.

```
smw# su - crayadm
cdump crayadm@smw> export TODAY=`date +%Y%m%d`
cdump crayadm@smw> ./etc/opt/cray/release/cle-release
cdump crayadm@smw> cd /home/crayadm/dump/${TODAY}_${BUILD}/
cdump crayadm@smw> script -af hss.cdump
```

### 4. Dump a node with `cdump`.

Change to the directory created in the `xtdumpsys` window (after `INFO: # Your dump is available in`), then use `cdump` to dump a compute node that successfully booted.

```
cdump crayadm@smw> cd /var/opt/cray/dump/p0-YYYYMMDDtHHMMSS-NNNNNNNNNN
cdump crayadm@smw> mkdir cumps; cd cumps
```

This example uses the `c0-0c0s3n0` node.

```
cdump crayadm@smw> cdump -AmD -r xt-hsn@boot c0-0c0s3n0
Wed Mar 1 09:08:08 CDT 2017 start cdump
...
makedumpfile Completed.
- done
Wed Mar 1 09:08:08 CDT 2017 cdump: # of nodes 1
  success 1
  failed 0
  skipped 0
cdump crayadm@smw> exit
```

For a partitioned system, use the host name to specify which boot node.

### 5. Continue `xtdumpsys`: enter a reason.

After `cdump` completes, return to the `xtdumpsys` window and enter a reason.

```
xtdumpsys window> testdump
```

Then enter an end-of-file (**Ctrl-d**) to end the dump reason.

```

xtdumpsys window> <Ctrl-d>
testdump
INFO: Dump reason:
...
INFO:
#####
INFO: # Your dump is available in /var/opt/cray/dump/
p0-20170301t081927-1304240904 #
INFO:
#####
INFO: No post-processing plugin found at '/etc/opt/cray/dumpsys/
postprocessing.py'
INFO: Example plugins can be found at '/opt/cray/dumpsys/
1.2.5-1.0000.35873.20.1/bin/plugins/examples/postprocessing.py.*'
INFO: Cleaning up

xtdumpsys crayadm@smw> exit

```

#### 6. Remove dump directory, if desired.

If there are no errors, it is probably safe to delete the dump directory.

```

xtdumpsys crayadm@smw> rm -rf /var/opt/cray/dump/pX-YYYYMMDDtHHMMSS-NNNNNNNNNN
crayadm@smw> exit

```

### 4.6.11 Make a Post-boot Snapshot using snaputil

#### About this task

This procedure uses `snaputil` to make an archival snapshot of the system after booting the CLE system.

**Best Practice.** Make a snapshot and back up config sets at the same time to keep them in sync. Name the snapshot and config set backup using the same suffix and date/time stamp to help administrators identify which snapshot and config set backup pairs belong together.

For more information, see [About Snapshots and Config Set Backups](#) on page 21.

#### Procedure

##### 1. List the available snapshots on the system.

```
smw# snaputil list
```

##### 2. Set the `SNAPSHOT` environment variable using the currently booted snapshot name.

```

smw# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')
smw# echo $SNAPSHOT

```

Setting a variable for the snapshot name enables better command substitution in later commands dealing with snapshots.

This is especially important for SMW HA systems because it makes it easier to use the exact same snapshot name for both SMWs. Using different snapshots results in HSS database (MySQL) inconsistencies, which causes problems at failover.

3. Change the `zypper` repo type.

This step is recommended, but not required, for both a standalone SMW and the first SMW of an SMW HA pair.

```
smw# sed -i 's/type=rpm-md/type=plaindir/' /etc/zypp/repos.d/*.repo
smw# zypper refresh
```

4. Create a new snapshot.

```
smw# snaputil create ${SNAPSHOT}.postboot-${TODAY}
```

## 4.6.12 Make a Post-boot Backup of Current Global and CLE Config Sets

### About this task

This procedure uses the `cfgset` command to create a post-boot backup of the global and CLE config sets.

### Procedure

1. Back up the current global config set.

```
smw# cfgset create --clone global global-postboot-${TODAY}
```

2. Back up the current CLE config set.

This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-postboot-${TODAY}
```

## 4.7 Configure Other Features and Services

At this stage in the fresh install process, the basic SMW and CLE software has been installed, configured, and booted. To complete the configuration of a functional system, use the applicable procedures and references provided in this section.

Use [Installation Checklist 7: Configure Other Features and Services](#) on page 445 to track progress through this part of the fresh install process.

**NOTE:** The CLE system cannot be in use during the power management configuration procedure. If this is not a fresh install, CLE must be shut down before performing that procedure. All of the other procedures require CLE to be running.

REQUIRED	<a href="#">Configure Power Management Disk</a> on page 256  (SMW HA only) Skip this procedure if doing a fresh install of the first SMW in an SMW HA system, and the Cray SMWHA software will be installed immediately afterwards, because power management for the SMW HA system will be configured later in the SMW HA fresh install process.
----------	--

REQUIRED (if using diags)	<a href="#">Push Diag Image to Boot Node and Update the Diags Bind Mount Profile</a> on page 260
REQUIRED (if using netroot)	<a href="#">Configure Netroot</a> on page 262
REQUIRED (if using SEC)	<a href="#">Configure the SEC and check_xt Monitoring and Notification Utilities</a> on page 268
REQUIRED (if using DAL)	<a href="#">Configure Direct-attached Lustre (DAL)</a> on page 269
optional (if using DAL)	<a href="#">LMT Configuration for DAL</a> on page 277 (Lustre Monitoring Tool for direct-attached Lustre)
REQUIRED (if using Docker)	Enable and configure Docker, if needed for this system. Use <i>XC™ Series Docker Administration Guide (S-2586)</i> .
REQUIRED (if using Shifter)	Enable and configure Shifter, if needed for this system. Use <i>XC™ Series Shifter Configuration Guide (S-2572)</i> .
recommended	<a href="#">Reduce Impact of Btrfs Periodic Maintenance on SMW Performance</a> on page 282
optional	<a href="#">Configure Cray NAT Masquerading Service</a> on page 283
optional	<a href="#">Prevent Unintentional Re-creation of Mail Configuration Files</a> on page 291
optional	Configure custom Node Health Checker (NHC) plugins, if needed for this system. See "Configure Node Health Checker Tests" under "Modify an Installed System" in <i>XC™ Series System Administration Guide (CLE 6.0.UP07) S-2393</i> .
optional	Configure custom Resource Utilization Reporting (RUR) data plugins or output plugins, if needed for this system.  see "Resource Utilization Reporting" under "Monitor the System" in <i>XC™ Series System Administration Guide (CLE 6.0.UP07) S-2393</i> .
optional	Set up advanced RSIP configuration, if needed for this system and not already configured prior to system boot.  See "Set Up Advanced RSIP Configuration on a Booted System" under "Modify an Installed System" in <i>XC™ Series System Administration Guide (CLE 6.0.UP07) S-2393</i> .
informational	<a href="#">About System Environmental Data Collections (SEDC)</a> on page 292

## 4.7.1 Configure Power Management Disk

### Prerequisites

This is a required step in bringing up a Cray XC system with releases later than CLE 6.0 UP01 / SMW 8.0 UP01. The PostgreSQL database on the SMW is needed even if a site will be using a remote (off-SMW) database node to store power and SEDC data.

This procedure assumes that a disk drive is available for use as a dedicated drive for the PMDB. The drive should be physically located within the SMW at slot 4. The partition path for the PMDB disk is different for each SMW hardware type.

- On a Dell PowerEdge™ R815 Rack Server, the device for PMDISK is:

```
/dev/disk/by-path/pci-0000:05:00.0-sas-phy3-0x4433221103000000-lun-0
```

- On a Dell PowerEdge™ R640 Rack Server, the device for PMDISK is:

```
/dev/disk/by-path/pci-0000:18:00.0-scsi-0:2:1:0
```

- On a Dell PowerEdge™ R630 Rack Server, the device for PMDISK is:

```
/dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:4:0
```

The system cannot be in use during this procedure. If this is not a fresh install, CLE must be shut down.

### About this task

Once the disk is configured, Power Management monitors, profiles, and limits power usage administrators. These steps are performed as `root`.

### Procedure

1. Verify that the PMDISK is inserted into the SMW by entering the correct device name. This example, and the ones that follow, are for a Dell R815.

```
smw# fdisk -l /dev/disk/by-path/pci-0000:05:00.0-sas-phy3-0x4433221103000000-lun-0
Disk /dev/disk/by-path/pci-0000:05:00.0-sas-0x4433221103000000-lun-0: 931.5 GiB,
1000204886016 bytes, 1953525168 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x10692081
```

2. Create a new primary partition for the PMDISK, and write it to the partition table. If there are any existing partitions on this disk, manually delete them first.

```
smw# fdisk /dev/disk/by-path/pci-0000:05:00.0-sas-phy3-0x4433221103000000-lun-0
Welcome to fdisk (util-linux 2.25).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
Command (m for help): n
Partition type
  p   primary (0 primary, 0 extended, 4 free)
```

```

    e    extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-1953525167, default 2048): [press return]
Last sector, +sectors or +size{K,M,G,T,P} (2048-1953525167, default 1953525167): [press return]

Created a new partition 1 of type 'Linux' and of size 931.5 GiB.

Command (m for help): w
The partition table has been altered!
Calling ioctl() to re-read partition table.
Syncing disks.

```

3. Verify that the partition has been created. The partition path for the PMDB disk is different for each SMW hardware type:

- Dell R815:  
/dev/disk/by-path/pci-0000:05:00.0-sas-phy3-0x4433221103000000-lun-0-part1
- Dell R640: /dev/disk/by-path/pci-0000:18:00.0-scsi-0:2:1:0-part1
- Dell R630: /dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:4:0-part1

```

smw# fdisk -l \
/dev/disk/by-path/pci-0000:05:00.0-sas-phy3-0x4433221103000000-lun-0

Disk /dev/disk/by-path/pci-0000:05:00.0-sas-phy3-0x4433221103000000-lun-0: 931.5 GiB, 1000204886016
bytes, 1953525168 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x96c1b0f0

Device                               Boot Start      End
Sectors  Size Id Type
/dev/disk/by-path/pci-0000:05:00.0-sas-phy3-0x4433221103000000-lun-0-part1      2048 1953525167
1953523120 931.5G 83 Linux

```

4. Create an ext4 file system on the PMDISK partition.

```

smw# mkfs.ext4 \
/dev/disk/by-path/pci-0000:05:00.0-sas-phy3-0x4433221103000000-lun-0-part1

Creating filesystem with 244190390 4k blocks and 61054976 inodes
Filesystem UUID: 6d791409-e327-4620-a80c-2933271b3eec
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000, 7962624, 11239424, 20480000, 23887872, 71663616, 78675968,
    102400000, 214990848

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

```

5. Stop the RSMS services.

```

smw# systemctl stop rsms
smw# systemctl status rsms
rsms.service - hss daemon control
    Loaded: loaded (/usr/lib/systemd/system/rsms.service; enabled)

```



```

Active: inactive (dead) since Wed 2015-11-04 15:42:04 CST; 19s ago
Process: 5471 ExecStop=/opt/cray/hss/default/bin/hssctl stop (code=exited, status=0/SUCCESS)
Process: 30305 ExecStart=/opt/cray/hss/default/bin/hssctl start (code=exited, status=0/SUCCESS)

Nov 03 16:01:43 smw hssctl[30305]: Starting daemons: erd erdh state_mana...md
Nov 04 15:42:04 smw hssctl[5471]: Stopping daemons: sec_cmd boot_cmds ca...rd
Hint: Some lines were ellipsized, use -l to show in full.

```

- Verify that the RSMS services are stopped. While the RSMS services are stopped, the system may continue to run applications, however the high-speed network will be throttled until RSMS is restarted.

```

smw# rsms status
PID                DAEMON                STATE                UPTIME
                  erd                  stopped
                  erdh                  stopped
                  state_manager         stopped
                  nid_mgr               stopped
                  bootmanager           stopped
                  sedc_manager           stopped
                  xtpmd                 stopped
                  erfsd                 stopped
                  xtremoted              stopped
                  xtpowerd               stopped
                  nimsd                  stopped
                  xtsnmpd                stopped
                  xtdiagd                stopped

```

- Run the xtmvpmdb script.

```

smw# xtmvpmdb \
/dev/disk/by-path/pci-0000:05:00.0-sas-phy3-0x4433221103000000-lun-0-part1 ext4
- Checking userid
- Checking destination directory name
- Checking destination directory existence

Move database to: /dev/disk/by-path/pci-0000:05:00.0-sas-0x4433221103000000-lun-0-part1
[y/n] [y]: y
- Checking current PM database directory existence
- Checking for booted system
- Checking for rsms daemons
- Creating directory /media/temp_pgsql_data
Dir: /media/temp_pgsql_data created
- Checking status of PM database process
Checking for PostgreSQL 9.3.8: ..running
postgresql.service - LSB: Start the PostgreSQL master daemon
   Loaded: loaded (/etc/init.d/postgresql)
   Active: active (exited) since Mon 2017-02-20 15:38:45 CST; 24h ago
 Process: 16633 ExecReload=/etc/init.d/postgresql reload (code=exited, status=0/SUCCESS)
 Process: 16255 ExecStart=/etc/init.d/postgresql start (code=exited, status=0/SUCCESS)

- Stopping PM database
- Copy contents of /var/lib/pgsql to /media/temp_pgsql_data
- This may take a few minutes to complete.
- Rename previous DB directory from: /var/lib/pgsql to: /var/lib/pgsql.
11-04-2015t15:43:04
- Unmount device from temporary mount point: /media/temp_pgsql_data
- Unmount btrfs subvolume: /var/lib/pgsql
- Mount device at permanent mount point: /var/lib/pgsql
- Add mount point to /etc/fstab
- Start PM database

- Transfer of PM database complete.

```

## 8. Change ownership and permissions of /var/lib/pgsql.

Postgres requires permissions to make necessary changes to the folder.

```
smw# chown -R postgres:postgres /var/lib/pgsql
smw# chmod -R 0700 /var/lib/pgsql
```

## 9. Restart the RSMS services and verify that the daemons are starting.

```
smw# systemctl start rsms
smw# systemctl status rsms
● rsms.service - hss daemon control
Loaded: loaded (/usr/lib/systemd/system/rsms.service; enabled)
Active: active (exited) since Tue 2017-02-21 15:44:24 CST; 9s ago
Process: 5471 ExecStop=/opt/cray/hss/default/bin/hssctl stop (code=exited, status=0/SUCCESS)
Process: 9227 ExecStart=/opt/cray/hss/default/bin/hssctl start (code=exited, status=0/SUCCESS)

Jun 04 15:44:24 smw hssctl[9227]: Starting daemons: erd erdh state_manag...md
Hint: Some lines were ellipsized, use -l to show in full.
```

## 4.7.2 Push Diag Image to Boot Node and Update the Diags Bind Mount Profile

### Prerequisites

This procedure assumes that the system has been booted after a fresh install.

### About this task

The online diagnostics image provides some useful tools that are made available on CLE nodes through the Cray Image Binding service using the profile for the diag image root. This procedure describes how to push the diag image root to the boot node. It also enables that service and configures it to reference the correct diag image and enable the diag profile.

### Procedure

#### 1. Check for existing diags image roots.

```
smw# image list | grep diag
diags_cle_6.0.up07_sles_12sp3_x86-64
diags_cle_6.0.up07_sles_12sp3_aarch64
```

#### 2. Push the diag image root(s) to the boot node.

```
smw# image sqpush -d boot diags_cle_6.0.up07_sles_12sp3_x86-64
smw# image sqpush -d boot diags_cle_6.0.up07_sles_12sp3_aarch64
```

**Trouble?** If passwordless `ssh` has not been prepared between `root@smw` and `root@boot`, then the system will prompt for the password for `root@boot` twice.

————— CONFIGURE THE DIAGS BIND MOUNT PROFILE —————

The following steps use the `cfgset modify` command, which does not produce output when successful. An optional `cfgset get` command is provided for each to verify the resulting setting value. Using the `cfgset modify` command causes the config set to be marked as invalid because no pre- and post-configuration scripts are run; therefore the config set must be updated and validated afterwards.

3. Ensure that the `cray_image_binding` configuration service is enabled.

```
smw# cfgset modify --set true cray_image_binding.enabled p0
smw# cfgset get cray_image_binding.enabled p0
true
```

4. Change the value of the diags profile `image` field.

Determine the name of the image root used in the `diags_x86_64` and `diags_aarch64` profiles in `cray_image_binding`.

```
smw# cfgset get cray_image_binding.settings.profiles.data.diags_x86_64.image p0
smw# cfgset get cray_image_binding.settings.profiles.data.diags_aarch64.image p0
```

The image name in the diags profile `image` field MUST match the name of the diags image root that was pushed to the boot node (if performing a fresh install) or will be pushed to the boot node (if performing a software update). If the diags profile `image` field is set to the wrong image, change it. Substitute the correct image name in the following example command(s).

```
smw# cfgset modify --set diag_x86-64_image_name \
cray_image_binding.settings.profiles.data.diags_x86_64.image p0
smw# cfgset get cray_image_binding.settings.profiles.data.diags_x86_64.image p0
smw# cfgset modify --set diag_aarch64_image_name \
cray_image_binding.settings.profiles.data.diags_aarch64.image p0
smw# cfgset get cray_image_binding.settings.profiles.data.diags_aarch64.image p0
```

5. Enable the diags profile(s).

```
smw# cfgset modify --set true \
cray_image_binding.settings.profiles.data.diags_x86_64.enabled p0
smw# cfgset get cray_image_binding.settings.profiles.data.diags_x86_64.enabled p0
true
smw# cfgset modify --set true \
cray_image_binding.settings.profiles.data.diags_aarch64.enabled p0
smw# cfgset get cray_image_binding.settings.profiles.data.diags_aarch64.enabled p0
true
```

6. Update the CLE config set.

The previous steps modified the config set without running pre- and post-configuration scripts, so that config set was marked invalid. This step uses `cfgset update` in prepare mode (no user interaction) to ensure that all configuration scripts are run.

```
smw# cfgset update -m prepare p0
```

7. Validate the CLE config set.

```
smw# cfgset validate p0
```

---

REBOOT THE SYSTEM

8. To use diags on the system, reboot the system.

### 4.7.3 Configure Netroot

This part of the installation and configuration process is optional unless this site has decided to use netroot. See [Where to Place the Root File System: tmpfs versus netroot](#) on page 36 for more information.

Netroot needs a matched pair of images for compute nodes and login nodes.

- |                      |   |
|----------------------|---|
| <b>compute nodes</b> | <ul style="list-style-type: none"><li>• <code>initrd-compute-large</code>: the NIMS boot image is set to this image</li><li>• <code>compute-large</code>: the NIMS kernel parameter "netroot" is set to this image, and this image is pushed to the boot node</li></ul> |
| <b>login nodes</b>   | <ul style="list-style-type: none"><li>• <code>initrd-login-large</code>: the NIMS boot image is set to this image</li><li>• <code>login-large</code>: the NIMS kernel parameter "netroot" is set to this image, and this image is pushed to the boot node</li></ul>     |

The following procedures describe how to configure the netroot images, push netroot images to the boot node, and reboot nodes with netroot.

#### 4.7.3.1 Configure Netroot Images

### Prerequisites

This procedure assumes the following:

- Basic configuration is complete and the system has been booted.
- No netroot images have been built yet. If that is not the case, and the netroot specifications shown below are already in the "default" image group of `cray_image_groups.yaml`, then skip this procedure.

### About this task

This procedure assigns netroot-specific NIMS groups (if needed), ensures that the `default` image group has netroot image specifications, and then builds the netroot images. Going forward, with the netroot image specifications included in the `default` image group, both compute and login netroot images will be built every time `imgbuilder` is run.

### Procedure

1. Prepare netroot-specific NIMS groups.
  - If this site plans to use netroot for ALL compute and login nodes, skip this step and proceed to step 3 on page 264. Netroot-specific NIMS groups are not needed because the NIMS login groups will be used for all login nodes and the NIMS compute group will be used for all compute nodes.
  - If this site plans to use netroot on only a subset of compute and login nodes instead of all of them, then continue with this step.

## a. Create and assign netroot-specific NIMS groups.

In the following example, the new NIMS groups are called `login_netroot` and `compute_netroot`, and each subset of nodes (`SUBSET_LOGIN_NODES` and `SUBSET_COMPUTE_NODES`) is a space-separated list of nodes.

```
smw# cnode update -G login -g login_netroot SUBSET_LOGIN_NODES
smw# cnode update -G compute -g compute_netroot SUBSET_COMPUTE_NODES
```

For x86\_64 architecture nodes:

```
smw# cnode update -G login -g login_netroot SUBSET_x86_64_LOGIN_NODES
smw# cnode update -G compute -g compute_netroot SUBSET_x86_64_COMPUTE_NODES
```

For aarch64 architecture nodes:

```
smw# cnode update -G login -g login_netroot_aarch64 \
SUBSET_aarch64_LOGIN_NODES
smw# cnode update -G compute -g compute_netroot_aarch64 \
SUBSET_aarch64_COMPUTE_NODES
```

## b. Confirm that the intended nodes were added to the NIMS netroot groups.

```
smw# cnode list --filter group=login_netroot

smw# cnode list --filter group=compute_netroot

smw# cnode list --filter group=login_netroot
smw# cnode list --filter group=login_netroot_aarch64

smw# cnode list --filter group=compute_netroot
smw# cnode list --filter group=compute_netroot_aarch64
```

2. Ensure that netroot-specific image specifications for use by a subset of compute and login nodes are in the default image group in `cray_image_groups.yaml`.

- If this site plans to use netroot for ALL compute and login nodes, skip this step and proceed to step 3 on page 264.
- If this site plans to use netroot on only a subset of compute and login nodes instead of all of them, then continue with this step.

Add these netroot image specifications to the default image group, if they are not already there. The safest way to do this is to find these image specifications under the `netroot` image group in the same file, then copy and paste them from there to the default image group and substitute the correct NIMS group names, as shown in this example.

```
cray_image_groups:
  default:
    ...
    - recipe: "initrd-compute-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "initrd-compute-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-x86_64-
created{date}"
      arch: "x86_64"
      export_format: "cpio"
      nims_group: "compute_netroot"
    - recipe: "initrd-compute-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "initrd-compute-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-aarch64-
created{date}"
      arch: "aarch64"
      export_format: "cpio"
      nims_group: "compute_netroot_aarch64"
    - recipe: "initrd-login-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "initrd-login-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-x86_64-
created{date}"
```

```

arch: "x86_64"
export_format: "cpio"
nims_group: "login_netroot"
- recipe: "initrd-login-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
  dest: "initrd-login-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-aarch64-
created{date}"
  arch: "aarch64"
  export_format: "cpio"
  nims_group: "login_netroot_aarch64"
...

```

Each of these netroot image recipes builds two image roots and only one boot image (the `.cpio` file). For example, the first builds an `initrd-compute-large` image root, a `compute-large` image root, and an `initrd-compute-large` boot image.

3. Replace the default image group with the `netroot` image group so that all compute and login nodes will use netroot.
  - If this site plans to use netroot on only a subset of compute and login nodes instead of all of them, then skip this step and proceed to step 4 on page 265.
  - If this site plans to use netroot for ALL compute and login nodes, then continue with this step.
- a. Change the name of the default image group from `default` to `original-default`.

```

cray_image_groups:
  original-default:
    ...
  elogin:
    ...
  dal:
    ...
  netroot:
    - recipe: "initrd-compute-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "initrd-compute-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-
x86_64-created{date}"
      arch: "x86_64"
      export_format: "cpio"
      nims_group: "compute"
    - recipe: "initrd-compute-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "initrd-compute-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-
aarch64-created{date}"
      arch: "aarch64"
      export_format: "cpio"
      nims_group: "compute_aarch64"
    - recipe: "initrd-login-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "initrd-login-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-x86_64-
created{date}"
      arch: "x86_64"
      export_format: "cpio"
      nims_group: "login"
    - recipe: "initrd-login-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "initrd-login-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-aarch64-
created{date}"
      arch: "aarch64"
      export_format: "cpio"
      nims_group: "login_aarch64"

```

- b. Change the name of the `netroot` image group from `netroot` to `default` and move it to the top of the file.

```

cray_image_groups:
  default:
    - recipe: "initrd-compute-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "initrd-compute-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-
x86_64-created{date}"
      arch: "x86_64"
      export_format: "cpio"
      nims_group: "compute"
    - recipe: "initrd-compute-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "initrd-compute-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-
aarch64-created{date}"
      arch: "aarch64"
      export_format: "cpio"
      nims_group: "compute_aarch64"
    - recipe: "initrd-login-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "initrd-login-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-x86_64-
created{date}"

```

```

    arch: "x86_64"
    export_format: "cpio"
    nims_group: "login"
  - recipe: "initrd-login-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
    dest: "initrd-login-large{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-aarch64-
created{date}"
    arch: "aarch64"
    export_format: "cpio"
    nims_group: "login_aarch64"
  ...
original-default:
  ...

```

Each of these netroot image recipes builds two image roots and only one boot image (the `.cpio` file). For example, the first builds an `initrd-compute-large` image root, a `compute-large` image root, and an `initrd-compute-large` boot image.

- c. Add other image specifications to new `default` image group, as needed.

If the new default image group (formerly the `netroot` image group) does not have all the needed image specifications for this system, add them now.

#### 4. Create new images using the default image group.

- To build all images in the `default` image group in parallel (this is typically quicker):

```
smw# imgbuilder --map --processes=NUM
```

Replace `NUM` with the number of images to be built concurrently (capped at the number of CPUs as reported by `lscpu`).

- To build all images in the `default` image group serially:

```
smw# imgbuilder --map
```

For more information about building images in parallel, see [About Parallel Image Creation](#) on page 40.

**Trouble?** If `imgbuilder` warns about a NIMS group not existing in the current NIMS map, check `cray_image_groups.yaml` for any `nims_group` entries that are not used in this NIMS map. Either remove those entries (because they will build images that are not used) or add the missing NIMS groups.

At the end of the output from `imgbuilder`, there will be a command hint for how to push the resulting netroot images to the boot node. Note the `image sqpush` command with the specific image name that needs to be pushed to the boot node.

```

IMPORTANT: The netroot image for initrd-compute-large_cle_6.0.up07-build6.0.7128_sles_12sp3-x86_64-
created20180710 must be staged onto the boot node:
smw:# image sqpush -d boot compute-large_cle_6.0.up07-build6.0.7128_sles_12sp3-x86_64-created20180710

IMPORTANT: The netroot image for initrd-login-large_cle_6.0.up07-build6.0.7128_sles_12sp3-
created20180710 must be staged onto the boot node:
smw:# image sqpush -d boot login-large_cle_6.0.up07-build6.0.7128_sles_12sp3-created20180710

```

### 4.7.3.2 Push Netroot Images to Boot Node

#### Prerequisites

This procedure assumes the following:

- The boot node is booted.
- Netroot images have been built using `imgbuilder`, and the output of that command provided the specific image name that needs to be pushed to the boot node.

## Procedure

1. Check for existing netroot image roots for both compute-large and login-large.

```
smw# image list | grep "^compute-large"
compute-large_cle_6.0.up07-build6.0.7128_sles_12sp3-created20180710

smw# image list | grep "^login-large"
login-large_cle_6.0.up07-build6.0.7128_sles_12sp3-created20180710
```

2. (Optional) If both netroot image names have a common string, set an environment variable for it.

Using the output generated by one of the image list commands in step 1, set an environment variable to represent the common string appearing in both image names. This example assumes that the common string is everything that follows "-large." If this is not the case (for example, if the date-time stamp is different), creating an environment variable may not be worthwhile.

```
smw# export BASEIMAGE=cle_6.0.up07-build6.0.7128_sles_12sp3-created20180710
```

3. Push the netroot images to the boot node.

Note that the following examples use `image sqpush`. If these image roots will be modified and pushed more than once to test them, consider using `image push` instead. For more information, see [About Image Pushes: push versus sqpush](#) on page 42.

These commands may take 10 minutes or more to complete.

If no environment variable was defined use the following commands:

```
smw# image sqpush -d boot \
compute-large_cle_6.0.up07-build6.0.7128_sles_12sp3-created20180710

smw# image sqpush -d boot \
login-large_cle_6.0.up07-build6.0.7128_sles_12sp3-created20180710
```

If an environment variable was defined that applies to both image names, use the following commands instead:

```
smw# image sqpush -d boot compute-large_$BASEIMAGE
smw# image sqpush -d boot login-large_$BASEIMAGE
```

**Trouble?** If passwordless `ssh` has not been prepared between `root@smw` and `root@boot`, then the system will prompt for the password for `root@boot` twice.

4. Push custom netroot image roots to boot node.

If any custom image roots were created with netroot content or something that will be used by a profile in `cray_image_binding`, push that image root to the boot node. Installed workload managers (WLM) will have such custom images if login netroot is included in the WLM recipe for the login nodes. For example, if a WLM ('wlm') is installed, there will be `wlm-login` and `wlm-admin` or `wlm-service` image roots created. However, only if `wlm-login-large` was created as a netroot image root will it need to be pushed to the boot node.

- a. Check for existing custom netroot image roots.

This example shows checking for 'wlm' image roots. Substitute the name for the particular WLM used in this system.

```
smw# image list | grep wlm
wlm-login-large_cle_6.0.up07-build6.0.7128_sles_12sp3-created20180710
```



- b. Push custom netroot image roots to the boot node, if any were found.

This example uses the output of the previous substep as the image name. Substitute the image name(s) displayed in the output of the `image list` command for this system. If this image name contains the same base string as the images from step 1, and an environment variable was defined, that can be substituted for everything after 'wlm-login-large' in this push command.

```
smw# image sqpush -d boot \
wlm-login-large_cle_6.0.up07-build6.0.7128_sles_12sp3-created20180710
```

After the image roots have been pushed to the boot node, the nodes that will use the netroot images can be warm booted, or the entire system can be rebooted.

### 4.7.3.3 Reboot Nodes with Netroot

#### Prerequisites

This procedure assumes that netroot images have been pushed out to the boot node.

#### About this task

This procedure reboots nodes with the new netroot images, either by shutting down the entire system and rebooting it or by warm booting only the nodes that need the new netroot images.

When a node is booted using a netroot image, during the early stages of the boot, cray-ansible runs only Ansible plays of type `netroot_setup`, and it logs to these three files in `/var/opt/cray/log/ansible`.

```
ansible-init-netroot_setup (has Ansible play output)
file-changelog-init-netroot_setup (shows each file changed by an Ansible play)
file-changelog-init-netroot_setup.yaml (YAML version of the previous log file)
```

#### Procedure

##### ————— REBOOT NETROOT NODES —————

There are two options for rebooting the system with netroot images. Choose only ONE of them:

- Option 1**      Reboot all nodes using step 1 on page 267.
- Option 2**      Warm boot only netroot nodes using step 2 on page 268.

1. (Option 1) Reboot entire system with new netroot images.

In these example commands, replace `auto.hostname.stop` and `auto.hostname.start` with the boot automation files used for this system.

```
smw# su - crayadm
crayadm@smw> xtbootsys -s last -a auto.hostname.stop
crayadm@smw> xtbootsys -a auto.hostname.start
```

If this site does not have an `auto.hostname.stop`, use the following command to shut down the system.

```
crayadm@smw> xtbootsys -s last -a auto.xtshutdown
```

If Option 1 chosen, skip the next step.

## 2. (Option 2) Warm boot only nodes needing new netroot images.

```
smw# su - crayadm
```

### a. Warm boot all login nodes.

Provide the same list of cnames for the login nodes to both `xtnmi` and `xtbootsys --reboot`.

```
crayadm@smw> export LOGINNODES=c0-0c0s7n3,c0-0c0s8n0
crayadm@smw> echo -e "Login nodes are:\n$LOGINNODES"

crayadm@smw> xtnmi $LOGINNODES
crayadm@smw> sleep 60
crayadm@smw> xtbootsys --reboot -r \
"warmboot for login netroot" $LOGINNODES
```

### b. Warm boot all compute nodes.

This example uses cabinet c0-0.

```
crayadm@smw> export COMPUTENODES=$(xtcli status p0 | \
egrep -v "empty|service|disabled" | grep c0-0 | \
awk '{ FS=":"; print $1 }' | tr ':' ' ' | \
awk '{ printf "%s,", $1 }' | sed s'/.$//')
crayadm@smw> echo -e "Compute nodes are:\n$COMPUTENODES"

crayadm@smw> xtcli status $COMPUTENODES
crayadm@smw> xtcli shutdown $COMPUTENODES
crayadm@smw> xtcli status $COMPUTENODES
crayadm@smw> xtnmi $COMPUTENODES
crayadm@smw> sleep 60
crayadm@smw> xtcli status $COMPUTENODES

crayadm@smw> xtbootsys --reboot -r \
"warmboot for compute netroot" $COMPUTENODES
crayadm@smw> xtcli status $COMPUTENODES
```

## 4.7.4 Configure the SEC and check\_xt Monitoring and Notification Utilities

### About SEC

The Simple Event Correlator (SEC) is an SMW utility that parses every line being appended to system log files, watching for specific strings that represent the occurrence of significant system events. When a specified string is detected, SEC sends notification that this has happened, either by email, IRC, writing to a file, or some user-configurable combination of all three.

SEC is enabled by default, and by default is configured to generate email notifications to `crayadm`. The types of notifications generated and the recipients to whom notifications are sent are defined in the SEC configuration file, `/etc/opt/cray/cray_sec_actions_config`.

The System Management Workstation (SMW) release includes `sec-2.7.6` and an SEC support package, `cray-sec-8.0.0`. The SEC support package contains control scripts to manage the starting and stopping of SEC around a Cray mainframe boot session, in addition to other utilities and a rule set designed for Cray systems.

## About `check_xt`

The SEC package includes the `check_xt` utility, which reports on state changes in the Cray system (such as system boots or compute nodes going down), and logs data about the state of nodes and jobs. `check_xt` is designed to work in conjunction with SEC to send email and text alerts about critical system issues.

Note that unlike SEC, `check_xt` is NOT enabled by default. It must be configured and called using a crontab entry.

## Configure SEC and `check_xt`

For procedures to configure SEC and `check_xt`, see *XC™ Series SEC and `check_xt` Software Configuration Guide* (S-2542) for release CLE 6.0.UP07.

## 4.7.5 Configure Direct-attached Lustre (DAL)

### Prerequisites

- Service nodes to support direct-attached Lustre® (DAL) have been identified with `xtdiscover` as management server (MGS), metadata server (MDS), or object storage server (OSS) nodes.
- Configuration worksheets for Cray Linux Environment (CLE) have been created and updated for DAL:
  - The `cray_lnet` worksheet is updated and the `cray_lnet.enabled` setting is uncommented and set to `true`. See [Update `cray\_lnet` Worksheet](#) on page 182.
  - The `cray_lustre_client` worksheet is updated and the `cray_lustre_client.enabled` setting is uncommented and set to `true`. See [Update `cray\_lustre\_client` Worksheet](#) on page 188.
  - `cray_lustre_server` worksheet is updated and the `cray_lustre_server.enabled` setting is uncommented and set to `true`. See [Update `cray\_lustre\_server` Worksheet](#) on page 190

**NOTE:** There are additional settings which tune the Lustre kernel modules.

- If using the Lustre Monitoring Tool (LMT), a MySQL database, storage space for that database, Cerebro, and the LMT GUI must be configured on the MGS node. The `cray_lmt` worksheet includes settings for configuring LMT.
- All DAL service nodes are assigned to the `dal` NIMS group so that they are assigned the DAL boot image for booting.
- The `imgbuilder` configuration for DAL has the DAL stanza added to the to the default image group.

### About this task

This procedure configures direct-attached Lustre (DAL) nodes that provide a Lustre file system.

### Procedure

#### Determine the Persistent Device Names for DAL

1. Determine persistent device names for DAL.

The persistent device names for the metadata target (MDT), management target (MGT), and object storage target (OST) devices on DAL nodes must be added to the Lustre `fs_defs` file.



**CAUTION:** Use persistent device names in the Lustre file system definition. Non-persistent device names (for example, `/dev/sdc`) can change when the system reboots. If non-persistent names are specified in the `fs_name.fs_defs` file, then Lustre may try to mount the wrong devices and fail to start when the system reboots.

For more information about Lustre control utilities, see the `lustre_control(8)` and `lustre.fs_defs(5)` man pages.

- a. Find the volume IDs of the MDT, MGT, and OST devices.

For example, the following command uses `SMdevices` to find the volume ID of the MDT device called `dalmdt0`. For `mdt`, substitute the device name (or part of the name) used for the MDT device in this system.

```
smw# SMdevices | grep mdt
/dev/sdx (/dev/sg24) [Storage Array percival-bootraid, Volume dalmdt0, LUN 5, Volume ID
<60080e50002e39a600001cde55769e28>, Alternate Path (Controller-B): Non owning controller -
Active/Non-optimized]
/dev/sdan (/dev/sg40) [Storage Array percival-bootraid, Volume dalmdt0, LUN 5, Volume ID
<60080e50002e39a600001cde55769e28>, Alternate Path (Controller-B): Non owning controller -
Active/Non-optimized]
/dev/sdbd (/dev/sg56) [Storage Array percival-bootraid, Volume dalmdt0, LUN 5, Volume ID
<60080e50002e39a600001cde55769e28>, Preferred Path (Controller-A): Owning controller - Active/
Optimized]
/dev/sdh (/dev/sg8) [Storage Array percival-bootraid, Volume dalmdt0, LUN 5, Volume ID
<60080e50002e39a600001cde55769e28>, Preferred Path (Controller-A): Owning controller - Active/
Optimized]
```

In the example, the volume ID of the MDT device is `60080e50002e39a600001cde55769e28`.

Repeat the `SMdevices` command for the MGT and OST devices.

- b. Use the volume IDs of the MDT, MGT, and OST devices to find the persistent device names of those devices.

The following command continues the example for the MDT device called `dalmdt0`, which has volume ID `60080e50002e39a600001cde55769e28`. Substitute the actual volume ID found in the previous substep.

```
smw# ls -l /dev/disk/by-id/ | grep 60080e50002e39a600001cde55769e28
lrwxrwxrwx 1 root root 11 Jul  3 09:51 dm-uuid-mpath-360080e50002e39a600001cde55769e28 -> ../../dm-11
lrwxrwxrwx 1 root root  9 Jul  3 09:51 scsi-360080e50002e39a600001cde55769e28 -> ../../sdx
lrwxrwxrwx 1 root root 11 Jul  3 09:51 wwn-0x60080e50002e39a600001cde55769e28 -> ../../dm-11
```

The example output shows three results. Which one to use?

- If multipath has been set up, use the name with the "dm-uuid-mpath" prefix: `dm-uuid-mpath-360080e50002e39a600001cde55769e28`.
- Otherwise, use the name with the "scsi" prefix: `scsi-360080e50002e39a600001cde55769e28`.

So for this example and assuming multipath, the persistent device name path that would be added to the Lustre `fs_defs` file for the MDT device

is `/dev/disk/by-id/dm-uuid-mpath-360080e50002e39a600001cde55769e28` (this is done in step 9 on page 272).

Repeat the `ls -l /dev/disk/by-id/` command for the MGT and OST devices.

- c. Verify that the MDT, MGT, and OST devices can be reached from the SMW.

The following command continues the example for the MDT device. Substitute the actual persistent device name path found in the previous substep.

```
smw# ls -l /dev/disk/by-id/dm-uuid-mpath-360080e50002e39a600001cde55769e28
lrwxrwxrwx 1 root root 11 Jul  3 09:51 /dev/disk/by-id/dm-uuid-
mpath-360080e50002e39a600001cde55769e28 -> ../../dm-11
```

The example output shows that the SMW can reach the MDT device using that path.

Repeat the `ls -l /dev/disk/by-id/` command for the MGT and OST devices.

- d. Log in to each DAL service node to verify that the associated device(s) can be reached from that node.

The MDT device(s) must be reachable from the MDT node(s), the MGT device must be reachable from the MGT node, and the OST device(s) must be reachable from the OST node(s). The following command continues the example for the MDT node and device. Substitute the actual node name and persistent device name path found in an earlier substep.

```
smw# ssh boot
boot# ssh dal-mdt
```

```
dal-mdt# ls -l /dev/disk/by-id/dm-uuid-mpath-360080e50002e39a600001cde55769e28
```

### Create and Install the Lustre fs\_defcs File

2. Prepare the Lustre `fs_defcs` file on the system management workstation (SMW).

This file is used by `lustre_control` to format, reformat, start, and stop the file system. When creating the Lustre `fs_defcs` file in this example, use `/dev/disk/by-id/scsi-0x60080e500036ae3e000002e6524a8369` for LUN 17. Refer to the *XC™ Series Lustre® Administration Guide* (S-2648) for detailed information about how to create an `fs_defcs` file for a Lustre file system.

3. Create a variable called `FS_NAME` to be the name of the file system using 8 characters or less ("dal" in this example). The file name of the `fs_defcs` file should be similar to the file system it defines.

```
smw# export FS_NAME=dal
smw# echo $FS_NAME
dal
```

4. Copy the `example.fs_defcs` file to the one named after the DAL file system.

```
smw# cp -p /opt/cray-xt-lustre-utils/default/etc/example.fs_defcs \
/home/crayadm/${FS_NAME}.fs_defcs
```

5. Determine the LNet name by which the external Lustre server is accessed for this system (will be something like `gni` or `gni1`).

This information will be needed in several steps that follow. This example shows that the local LNet name in CLE config set `p0` was set to `gni4`.

```
smw# cfgset search -t gni -l advanced -s cray_lnet p0
# 2 matches for 'gni' from cray_lnet_config.yaml
#-----
cray_lnet.settings.local_lnet.data.lnet_name: gni4
cray_lnet.settings.flat_routes.data.o2ib.src_lnet: gni4
```

6. Edit the `$FS_NAME.fs_defcs` file.

This example for the `p0` config set calls the file system `dal`, has the local LNet name set to `gni4`, and has the MGT on `nid00027`, the MDT on `nid00027` and `nid00029`, a first OST on `nid00028`, and a second OST on `nid00031`. Substitute site-specific values in this system's `fs_defcs` file.

```
smw# vi /home/crayadm/${FS_NAME}.fs_defs
```

## 7. Set the file system name.

Locate `fs_name:` `example` and change `example` to the name defined by `$FS_NAME` (`dal` in this example).

```
fs_name: dal
```

## 8. Map the Lustre server hosts to LNet NIDs.

The name of an LNet NID, such as `27@gni4`, is a combination of the NID number of the associated node and the local LNet name, which was determined in step 5 on page 271.

```
# Lustre server hosts to LNET NIDs mapping.
# Multiple lines are additive.
# Use multiple lines with the same nodes if you have more than one nid for each
# node.
# Nodes and nids can be specified using range expressions. See the
# lustre.fs_defs man page for more information on range expressions.
# Each line should have a one-to-one mapping between the nodes and nids.
nid_map: nodes=nid000[27-29,31] nids=[27-29,31]@gni4
```

## 9. Add the node IDs and persistent device names to the `fs_defs` file to identify which nodes and devices are being used for MGT, MDT, and OSTs.

Update the `fs_defs` file with these settings, substituting appropriate site-specific values.

```
## MGT
## Management Target
mgt: node=nid00027
      dev=/dev/disk/by-id/scsi-360001ff020021101061ad79111170000

## MDT
## MetaData Target(s)
mdt: node=nid00027
      dev=/dev/disk/by-id/scsi-360001ff020021101061ad79111170100
      index=0
mdt: node=nid00029
      dev=/dev/disk/by-id/scsi-360001ff020021101061ad79111170200
      index=1

## OST
## Object Storage Target(s)
ost: node=nid00028
      dev=/dev/disk/by-id/scsi-360001ff020021101061ad79111170300
      index=0
ost: node=nid00031
      dev=/dev/disk/by-id/scsi-360001ff020021101061ad7a811170400
      index=1
```

There are other settings in the `fs_defs` file that can be changed, but their values are probably acceptable for most systems.

## 10. Install the `fs_defs` file into the appropriate CLE config set (p0 in the example).

```
smw# lustre_control install -c p0 /home/crayadm/${FS_NAME}.fs_defs
```

The `lustre_control install` command copies the `fs_defs` file into a directory in the config set, makes a `lustre_control` readable version of it with the suffix `.config.data`, and updates the list of installed file systems.

11. Verify that the `fs_defs` file is installed in the config set by listing the files in the `lustre/.lctrl/` directory of the config set.

```
smw# ls /var/opt/cray/imps/config/sets/p0/lustre/.lctrl/
dal.config.data      dal.filesys.data      dal.service.data
dal.failover.data    dal.fs_defs.20160421.1461256838  installed_filesystems
```

### Modify the Config Set to Load the `lustre-utils` Module

12. Modify `cray_user_settings.settings.default_modules.data.service` to add `lustre-utils`.

- a. Update the `cray_user_settings` service in config set `p0`.

```
smw# cfgset update -s cray_user_settings -m interactive -l advanced p0
```

- b. Select the default modules `service` setting (a list of autoloaded modules for non-login service nodes) to configure it.

Enter **2** and press **Enter** to select `service`, then enter **c** and press **Enter** to configure it.

```
Cray User Settings Menu [default: save & exit - Q] $ 2
...
Cray User Settings Menu [default: configure - C] $ c
```

- c. Add the `lustre-utils` module to the list.

Enter **+** to add an entry, then enter "`lustre-utils`" and press **Enter**. Press **Ctrl-d** to finish adding entries, then press **Enter** to set the entries for this setting.

```
cray_user_settings.settings.default_modules.data.service
[<cr>=set 7 entries, +=add an entry, ?=help, @=less] $ +
Add service (Ctrl-d to exit) $ lustre-utils
Add service (Ctrl-d to exit) $ <Ctrl-d>
...
cray_user_settings.settings.default_modules.data.service
[<cr>=set 8 entries, +=add an entry, ?=help, @=less] $ <cr>
```

- d. Save the changes and exit the configurator.

```
Cray User Settings Menu [default: save & exit - Q] $ Q
```

13. Validate the config set.

- Entire system:

```
smw# cfgset validate p0
```

- Partitioned system:

```
smw# cfgset validate p1
smw# cfgset validate p2
```

### Boot the System and Reformat the DAL File System

The DAL file system must be formatted using `lustre_control` from the boot node after initial set up, and before automating the start up and mounting of the DAL file system.

**14. Boot the system.**

- If CLE is not booted, proceed to step 15 on page 274.
- If CLE is booted, proceed to step 18 on page 274.

**15. If CLE is not booted:**

```
crayadm@smw> xtbootsys -a auto.hostname.start
```

**16. Reformat the DAL file system after a full system boot.**

```
smw# ssh boot
boot# export FS_NAME=dal
boot# lustre_control reformat -f $FS_NAME
```

**17. Proceed to step 21 on page 274**

**18. If CLE is booted, run `cray-ansible`, then reboot only the DAL nodes.**

Restarting `/etc/init.d/cray-ansible` refreshes the config set cache on the boot node. This example specifies a comma-separated list of `cnames` (for example `c0-0c0s0n0`) for all DAL nodes (MGS, MDS, and OSS) to create a `$DALNODES` variable.

```
boot# /etc/init.d/cray-ansible start
```

Note that the following commands are run as `crayadm`, not `root`.

```
crayadm@smw> export DALNODES=mgsnode,mdsnode,ossnode1,ossnode2
crayadm@smw> xtbounce -s $DALNODES
crayadm@smw> xtcli boot DEFAULT $DALNODES
```

**19. Confirm that the DAL nodes have complete their reboot.**

```
crayadm@smw> xtcli status s0 -E
```

Wait until a `READY` state is listed, then continue to the next step to reformat the DAL file system.

**20. Reformat the DAL file system after a reboot of only the DAL nodes.**

```
smw# ssh boot
boot# module load lustre-utils
boot# export FS_NAME=dal
boot# lustre_control reformat -f $FS_NAME
Continue? (y|n|q) y
```

### Start and Mount the DAL File System

**21. Start the DAL file system using `lustre_control` on the boot node.**

```
boot# lustre_control start -p -f $FS_NAME
```

**22. Verify that the Lustre targets are mounted on each DAL node.**



```
boot# lustre_control status -f $FS_NAME
```

**23. Test mount the DAL file system on a login node.**

```
boot# ssh login
login# export FS_NAME=dal
login# mkdir -p /lus/$FS_NAME

login# mount -t lustre 27@gni4:$FS_NAME /lus/$FS_NAME
```

In the mount command, substitute the site-specific value for `27@gni4`, which is a combination of the NID number of the MGT node and the local LNet name. The MGT node NID number was defined in the `fs_defs` file in step 9 on page 272, and the LNet name was determined in step 5 on page 271.

**Add DAL file system to `cray_lustre_client` Configuration**

**24. Add the DAL file system to `cray_lustre_client` configuration so that Lustre clients can mount the file system from the Lustre server.**

Note that the `cray_lustre_client` service must be enabled in addition to setting information like the settings below (substitute appropriate site-specific values).

```
smw# cfgset update -s cray_lustre_client -l advanced -m interactive p0
```

In the `client_mounts` setting, add two new client mount entries for the DAL file system. One will be for the compute nodes, which can mount the file system at boot time. The other will be for the login node(s). These cannot currently mount the file system at boot time since they are booted before the DAL file system is started. Follow the guidance for the `client_mounts` settings.

Substitute site-specific values for the list of `mgs_lnet_nids` (`27@gni4` and `29@gni4` in the example), which are a combination of the NID number of the MGS node (and failover MGS, if applicable) and the local LNet name. The MGS node NID number was defined in the `fs_defs` file in step 9 on page 272, and the LNet name was determined in step 5 on page 271. Set `mount_at_boot` to `false` for the login node entry and set it to `true` for the compute node entry.

```
cray_lustre_client.settings.client_mounts.data.fs_name.dal_login: null
cray_lustre_client.settings.client_mounts.data.dal_login.lustre_fs_name: dal
cray_lustre_client.settings.client_mounts.data.dal_login.mount_point: /lus/dal
cray_lustre_client.settings.client_mounts.data.dal_login.mgs_lnet_nids:
- 27@gni4
- 29@gni4
cray_lustre_client.settings.client_mounts.data.dal_login.mount_options:
  rw, flock, lazystatfs
cray_lustre_client.settings.client_mounts.data.dal_login.mount_at_boot: false
cray_lustre_client.settings.client_mounts.data.dal_login.client_groups:
- login_nodes_x86_64
- login_nodes_aarch64

cray_lustre_client.settings.client_mounts.data.fs_name.dal_compute: null
cray_lustre_client.settings.client_mounts.data.dal_compute.lustre_fs_name: dal
cray_lustre_client.settings.client_mounts.data.dal_compute.mount_point: /lus/dal
cray_lustre_client.settings.client_mounts.data.dal_compute.mgs_lnet_nids:
- 27@gni4
- 29@gni4
cray_lustre_client.settings.client_mounts.data.dal_compute.mount_options:
  rw, flock, lazystatfs
cray_lustre_client.settings.client_mounts.data.dal_compute.mount_at_boot: true
```

```
cray_lustre_client.settings.client_mounts.data.dal_compute.client_groups:
- compute_nodes
```

### Add DAL File System to cray\_lustre\_server Configuration

25. Add the DAL file system node groups to the cray\_lustre\_server service.

```
smw# cfgset update -s cray_lustre_server -l advanced -m interactive p0
```

```
cray_lustre_server.settings.lustre_servers.data.mgs_group: MGS_NODE_GROUP
cray_lustre_server.settings.lustre_servers.data.mds_groups:
- MDS_NODE_GROUP_1
- MDS_NODE_GROUP_2
cray_lustre_server.settings.lustre_servers.data.oss_groups:
- OSS_NODE_GROUP_1
- OSS_NODE_GROUP_2
```

### Configure LMT to Monitor DAL

26. (Optional) If using LMT to enable monitoring of DAL, see [LMT Configuration for DAL](#) on page 277.

### Enable Realm-Specific Internet Protocol (RSIP) on DAL Nodes

27. Enable RSIP on DAL nodes so they can communicate with an external LDAP or NIS server.

DAL nodes do not have external network connections, but require access to LDAP or NIS servers external to the system for uid/gid information associated with the Lustre file system.

- a. Add DAL node groups to the list of  
cray\_rsip.settings.service.data.node\_groups\_as\_client.

```
smw# cfgset update -s cray_rsip -l advanced -m interactive p0
```

Add the DAL MDS node group(s).

```
cray_rsip.settings.service.data.node_groups_as_client:
- MDS_NODE_GROUP_1
- MDS_NODE_GROUP_2
```

28. Validate the config set.

- Entire system:

```
smw# cfgset validate p0
```

- Partitioned system:

```
smw# cfgset validate p1
smw# cfgset validate p2
```

### Update the Boot Automation File for DAL

29. Edit the site boot automation file (`/opt/cray/hss/default/etc/auto.hostname.start`) so that the DAL file system is started during the CLE boot.

Because the config set modifications made in an earlier step set it up so that login and eLogin nodes do not attempt to mount DAL at boot time, but the compute nodes do, add these DAL lines to the site boot automation file **after** the boot of the service nodes but **before** the boot of the compute nodes.

Multiple comma-separated mount clients can be listed on the same line. Substitute the correct login host names or node IDs (NID) for this system.

```
#Boot all the service nodes
lappend actions {crms_boot_all_serv}

# start Lustre server on DAL nodes & mount Lustre filesystem on login nodes
lappend actions { crms_exec_on_bootnode "root" "lustre_control start -f dal" }
lappend actions { crms_exec_on_bootnode "root" "lustre_control mount_clients -f dal -w
login,login-aarch" }

#Boot specific compute nodes
#lappend actions [list crms_boot_loadfile DEFAULT compute "c0-0c0s7n0 c0-0c0s7n1" linux]

#Boot compute nodes
lappend actions {crms_boot_all_comp}
```

This uses a `pdsh` style list of nodes as an argument for the `mount_clients` command. For example, `lustre_control` will interpret `login[1-8]` as nodes `login1` through `login8`. Replace `dal` in the command with the name of the DAL file system for this site.

With `client_mounts.data.dal_compute.mount_at_boot` set to `true` in the `cray_lustre_clients` service, the compute nodes automatically mount the DAL file system when they boot. This also ensures that they mount the DAL file system even when rebooted individually, outside the control of the auto boot file.

## 4.7.6 LMT Configuration for DAL

The Lustre® monitoring tool (LMT) for direct-attached Lustre (DAL) on Cray Linux environment (CLE 6.0) requires some manual configuration during the software installation process.

- |   |  |
|---|--|
| <b>Configure Storage for the LMT Database</b> | At least 40GB of storage space must be made available to the MGS node. See <a href="#">LMT Disk Usage</a> on page 281.   |
| <b>Configure the LMT MySQL Database</b>       | The IMPS configuration does not set up this database, so this must be configured manually for CLE 6.0 UP01 and later releases. See <a href="#">Configure LMT MySQL Database for DAL</a> on page 277. |
| <b>Configure the LMT GUI (Optional)</b>       | See <a href="#">Configure the LMT GUI</a> on page 279.   |

Use the configurator to configure the LMT for DAL on CLE 6.0. Guidance is provided for each LMT configuration setting in the `cfgset` utility.

The `cray_lmt` configurator template configures LMT settings for specific nodes when they are booted. The default system configuration value for the LMT service is disabled (`false`). Log in to the SMW as `root` and use the `cfgset` command to modify the `cray_lmt` configuration settings to configure LMT.

```
smw# cfgset update -s cray_lmt -m interactive CONFIG_SET
```

### 4.7.6.1 Configure LMT MySQL Database for DAL

#### Prerequisites

A MySQL server instance must be configured on the management server (MGS) node. All commands described below should be executed on the MGS for the direct-attached Lustre (DAL) file system.

## About this task

A MySQL server instance on the management server (MGS) node stores real-time and historical Lustre monitoring tool (LMT) data. The configurator does not handle the initial setup of the LMT MySQL users and database. It must, therefore, be done manually. All commands described below should be executed on the MGS for the DAL file system.

## Procedure

1. Log on to the MGS as `root`.

(Where `nidMGS` is the node ID (NID) of the MGS node.)

```
boot# ssh nidMGS
```

2. Start the MySQL server daemon (if not already running).

```
mgs# /sbin/service mysqld start
```

3. Run the `mysql_secure_installation` script to improve MySQL server instance security.

This sets the password for the `root` MySQL user, disallows remote `root` access to the database, removes anonymous users, removes the test database, and reloads privileges. If this is the first time configuring LMT, create a symlink before running `mysql_secure_installation` to ensure that MySQL uses the correct socket.

- a. Create a symbolic link.

```
mgs# ln -s /var/run/mysql/mysql.sock /var/lib/mysql/mysql.sock
```

- b. Run `mysql_secure_installation` utility.

```
mgs# mysql_secure_installation
```

- c. Respond to script prompts.

Prompts and recommended responses generated by the script.

```
Enter current password for root (enter for none): <Enter>

Set root password? [Y/n] Y
New password: Enter a secure password
Re-enter new password: Enter the secure password again

Remove anonymous users? [Y/n] Y

Disallow root login remotely? [Y/n] Y

Remove test database and access to it? [Y/n] Y

Reload privilege tables now? [Y/n] Y
```

4. Ensure `root` only access to the LMT user configuration file, `/usr/share/lmt/mkusers.sql`.

```
mgs# chmod 600 /usr/share/lmt/mkusers.sql
```

5. Edit the LMT user configuration file `/usr/share/lmt/mkusers.sql`.

This file is not used at run time by LMT or MySQL processes. This script creates the MySQL users on the persistent storage configured for the MySQL databases. After it is run through MySQL, it is no longer needed.

This file contains MySQL statements that create users named `lwatchclient` and `lwatchadmin`. It gives them privileges only on databases that start with `filesystem_`. Cray recommends making the following changes to `mkusers.sql`.

**Edit the GRANT Statement** Edit the GRANT statements to grant privileges on only `filesystem_`*fsname*.`*` where *fsname* is the name of the file system. This will only grant permissions on the database for the file system being monitored.

**Edit the Password** Edit the password for `lwatchadmin` by changing `mypass` to the desired password. Also add a password for the `lwatchclient` user.

```
CREATE USER 'lwatchclient'@'localhost' IDENTIFIED BY 'foo';
GRANT SELECT ON filesystem_scratch.* TO 'lwatchclient'@'localhost';

CREATE USER 'lwatchadmin'@'localhost' IDENTIFIED BY 'bar';
GRANT SELECT,INSERT,DELETE ON filesystem_scratch.* TO 'lwatchadmin'@'localhost';
GRANT CREATE,DROP ON filesystem_scratch.* TO 'lwatchadmin'@'localhost';

FLUSH PRIVILEGES;
```

6. Save the changes and execute the following command. (This prompts for the MySQL `root` user password, which was set when `mysql_secure_installation` was executed.)

```
mgs# mysql -u root -p < /usr/share/lmt/mkusers.sql
```

7. Create the database for the file system to be monitored.

(Where *fsname* is the name of the DAL file system.)

```
mgs# lmtinit -a fsname
```

LMT data will be inserted into the LMT MySQL database the next time the Cerebro service is restarted on the MGS.

8. Restart Cerebro.

```
mgs# service cerebrod restart
```

9. Verify that LMT is adding data to the MySQL database.

- a. Initiate the LMT shell.

```
mgs# lmtsh -f fsname
```

- b. List tables.

```
fsname> t
```

- c. List tables again after several seconds to verify that `Row Count` is increasing.

#### 4.7.6.2 Configure the LMT GUI

##### About this task

The Lustre monitoring tool (LMT) graphical user interface (GUI) package is installed on login nodes. It contains a GUI called `lwatch` and a command-line tool for viewing live data called `lstat`. The configuration file `~/lmtrc` must be set up prior to using either tool.

## Procedure

1. Login to the MGS node as `root`.
2. Edit the sample configuration file `/usr/share/doc/packages/lmt-gui/sample.lmtrc` to reflect the site specific LMT configuration—where `db_name` is set to the name of the MySQL database used by LMT, that is, `filesystem_fsname`.

```
# LMT Configuration File - place in $HOME/.lmtrc

filesys.1.name=<insert_fsname_here>
filesys.1.mountname=<insert_/path/to/mountpoint_here>
filesys.1.dbhost=<insert_db_host_ip_here>
filesys.1.dbport=<insert_db_port_here>
filesys.1.dbuser=<insert_db_client_username_here>
# Leave dbauth blank if the given client has no password
filesys.1.dbauth=<insert_db_client_password_here>
filesys.1.dbname=<insert_db_name_here>
```

3. Save the updated `.lmtrc` as `~/lmtrc`.

Here is an example for configuring access to the LMT database for the file system named `scratch_1`, which was set up so that the user `lwatchclient` has no password. In this example, access is being configured on the LMT server node, so the database is local. Thus, the `db_host` is `localhost`.

```
filesys.1.name=scratch_1
filesys.1.mountname=/lus/scratch_1
filesys.1.dbhost=localhost
filesys.1.dbport=3306
filesys.1.dbuser=lwatchclient
filesys.1.dbauth=
filesys.1.dbname=filesystem_scratch_1
```

After setting up `~/lmtrc`, `lwatch` and `lstat` can be run on this node. To run the GUI from a remote node, the MySQL database must be configured to allow remote access for the read-only user, `lwatchclient`. See [Configure LMT MySQL for Remote Access](#) on page 280.

### 4.7.6.3 Configure LMT MySQL for Remote Access

In order to run the Lustre monitoring tool (LMT) graphical user interface (GUI) on a separate node from the LMT server, the MySQL server instance (running on the LMT server) must be configured to enable remote access for the LMT read-only user, `lwatchclient`. These MySQL statements can be added to `/usr/share/lmt/mkusers.sql` prior to executing the statements in that file. They can also be executed directly. In these examples, `FSNAME` is the name of the file system being monitored.

```
CREATE USER 'lwatchclient'@'%' IDENTIFIED BY 'foo';
GRANT SELECT ON filesystem_FSNAME.* TO 'lwatchclient'@'%';
```

To execute these statements directly, log on to the DAL MGS node, open a `mysql` shell as the root MySQL user, and run the statements as follows.

1. Connect to the database as `root`.

```
mgs# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
```

2. Create `lwatchclient` user.

```
mysql> CREATE USER 'lwatchclient'@'%';
Query OK, 0 rows affected (0.00 sec)
...
```

### 3. Grant privileges to lwatchclient user.

```
mysql> GRANT SELECT ON filesystem_FSNAME.* TO 'lwatchclient'@'%';
Query OK, 0 rows affected (0.00 sec)
```

This enables the user named `lwatchclient` to connect from any hostname.

To allow connections from a certain IP address, replace the `'%'` with an IP address in single quotes.

```
CREATE USER 'lwatchclient'@'10.11.255.252' IDENTIFIED BY 'foo';
GRANT SELECT ON filesystem_FSNAME.* TO 'lwatchclient'@'10.11.255.252';
```

#### 4.7.6.4 LMT Disk Usage

LMT requires at least 40GB persistent storage attached to the LMT server (i.e., the MGS) to store historical data. If the storage becomes full, data can be deleted from the database using MySQL delete statements.

### MySQL Tables

Five tables store general file system statistics. These tables are populated by `lmt_agg.cron` script.

*Table 15. General File System Tables*

Table Name	On-Disk Growth Rate
FILESYSTEM_AGGREGATE_HOUR	0.8 KB/hour
FILESYSTEM_AGGREGATE_DAY	0.8 KB/day
FILESYSTEM_AGGREGATE_WEEK	0.8 KB/week
FILESYSTEM_AGGREGATE_MONTH	0.8 KB/month
FILESYSTEM_AGGREGATE_YEAR	0.8 KB/year

*Table 16. MDS Aggregate Tables and Growth Rates*

Table Name	Approximate On-Disk Growth Rate
MDS_AGGREGATE_HOUR	0.5 KB/hour/MDS
MDS_AGGREGATE_DAY	0.5 KB/day/MDS
MDS_AGGREGATE_WEEK	0.5 KB/week/MDS
MDS_AGGREGATE_MONTH	0.5 KB/month/MDS
MDS_AGGREGATE_YEAR	0.5 KB/year/MDS

*Table 17. OST Aggregate Tables and Growth Rates*

Table Name	On-Disk Growth Rate
OST_AGGREGATE_HOUR	0.7 KB/hour/OST
OST_AGGREGATE_DAY	0.7 KB/day/OST

Table Name	On-Disk Growth Rate
OST_AGGREGATE_WEEK	0.7 KB/week/OST
OST_AGGREGATE_MONTH	0.7 KB/month/OST
OST_AGGREGATE_YEAR	0.7 KB/year/OST

## Calculate Expected Disk Usage for a File System

Use this formula to calculate the approximate rate of disk space usage for a file system. Disregard the AGGREGATE tables as they grow so much more slowly than the raw data tables.

```
(56 KB/hour/filesystem) * (# of filesystems) + (1000 KB/hour/MDS) * (# of MDSs)
+ (44 KB/hour/OSS) * (# of OSSs) + (70 KB/hour/OST) * (# of OSTs) = Total KB/hour
```

## Calculate the Disk Usage for a File System for 1 Year

In this example, LMT is monitoring one file system with one MDS, four object storage servers (OSS), and eight object storage targets (OST). The amount of disk space used by the LMT database to is expected to grow at this hourly rate.

```
56 KB/hour/filesystem * 1 filesystem + 1000 KB/hour/MDS * 1 MDS
+ 44 KB/hour/OSS * 4 OSSs + 70 KB/hour/OST * 8 OSTs = 1792 KB/hour
```

Which translates to this yearly rate.

```
1792 KB/hour * 24 hours/day * 365 days/year * 1 MB/1024KB
* 1 GB/1024MB = 15 GB / year
```

## 4.7.7 Reduce Impact of Btrfs Periodic Maintenance on SMW Performance

### About this task

Btrfs (B-tree file system) runs periodic maintenance. The weekly and monthly maintenance scripts, which include balance, trim, and scrub actions, can consume large amounts of compute resource. This can impact a site's ability to use the SMW for normal operations, including using SSH to log into nodes. This procedure describes how to reduce the impact to SMW performance by controlling when these scripts are run.

Note that Cray does not manage the `/etc/sysconfig/btrfsmaintenance` file, so a site administrator wishing to perform Btrfs maintenance commands should manage that file based on the site's best practices for Btrfs maintenance.

### Procedure

1. Create a file `/etc/cron.d/cray_btrfs.cron`.

The new cron file needs to be in `/etc/cron.d` because the Btrfs RPM installs links to maintenance scripts into the `/etc/cron.{weekly,monthly}` directories.

```
smw# vi /etc/cron.d/cray_btrfs.cron
```

Add these lines to the new file. Adjust as needed for this site.



```
# Control when btrfs maintenance scripts run by deleting the corresponding
# 'lastrun' files at a predetermined time. Caveat, this affects all of the
# scripts in the corresponding cron directories (/etc/cron.{weekly,monthly})

# Run weekly on Saturday at 2 AM as root
0 2 * * 6 root rm -f /var/spool/cron/lastrun/cron.weekly
# Run monthly on the first Sunday of the month at 2 AM as root
0 2 * * 0 root [ $(date +%d) -le 07 ] && rm -f /var/spool/cron/lastrun/cron.monthly
```

2. Set ownership of the new cron file to root,root with permissions 644.

```
smw# chown root:root /etc/cron.d/cray_btrfs.cron
smw# chmod 644 /etc/cron.d/cray_btrfs.cron
```

## 4.7.8 Configure Cray NAT Masquerading Service

### Prerequisites

The Cray XC system is booted.

### About this task

The Cray NAT (network address translation) Masquerading configuration service defines a gateway node to serve as the default gateway for one or more client nodes, and provides the option to configure `SuSEfirewall12` to permit network traffic from a client node to a network outside the Cray high-speed network (HSN). To use this service, the Cray firewall configuration service (`cray_firewall`) must be enabled.

This service can be used for a number of use cases. For example, it can be used to enable an AArch64 (ARM) compute node repurposed as a login node to use a default gateway to safely access networks outside the HSN. This may be needed for SSH, license servers, PE debuggers, X11 forwarding, and so forth. And it can be used to enable users to access an AArch64 login node without having to perform two "hops" to get there.

This procedure begins by ensuring that the firewall is enabled and then defining the node groups needed to specify gateway node(s) and client nodes and enabling this service. For this service to function, at least one gateway node group and one client node group are required. The rest of the procedure consists of use cases that illustrate what can be done with the `cray_nat_masq` service. Each use case builds on the previous one.

Besides settings to define the gateway node and its clients, the `cray_nat_masq` config service includes three settings for configuring firewall rules in `SuSEfirewall12` (options `FW_MASQ_DEV`, `FW_MASQ_NETS`, `FW_FORWARD`, and `FW_FORWARD_MASQ`). These settings may be left empty or configured appropriately to permit network traffic from the client node to a network outside the HSN. `FW_FORWARD_MASQ` could be used to redirect and forward a port addressed from an x86\_64 login node to an AArch64 login node. For example, a user could access the AArch64 login node with the following command:

```
user@hostname> ssh -p PORT_NUM x86_64_login_node
```

If a secondary IPv4 address is assigned to the x86-64 login node, that IP address could be forwarded to the AArch64 login node instead of using an alternate port, as shown in the following command:

```
user@hostname> ssh x86_64_login_node_secondary_ipv4
```

Because the `cray_nat_masq` config service provides no default values for the port setting and the firewall list settings, it is up to system administrators to enable this forwarding if they are willing to allow users access to the AArch64 login node via this routing.

## Procedure

----- ENSURE FIREWALL IS ENABLED -----

### 1. Ensure that the firewall is enabled.

- a. Determine whether `cray_firewall` is enabled in the global config set.

```
smw# cfgset get cray_firewall.enabled global
```

- b. If `cray_firewall` in the global config set is NOT enabled, enable it now.

Enable `cray_firewall` in global.

```
smw# cfgset modify --set true cray_firewall.enabled global
smw# cfgset get cray_firewall.enabled global
true
```

Update and validate the global config set.

```
smw# cfgset update global
```

```
smw# cfgset validate global
```

Apply the changes on the SMW.

```
smw# /etc/init.d/cray-ansible start
```

- c. Determine whether `cray_firewall` in the CLE config set (`p0` in example) inherits from the global config set.

```
smw# cfgset get cray_firewall.inherit p0
```

If `cray_firewall` in the CLE config set inherits from global, no change is needed.

- d. If `cray_firewall` in the CLE config set does NOT inherit from global, then ensure `cray_firewall` in the CLE config set is enabled.

```
smw# cfgset modify --set true cray_firewall.enabled p0
smw# cfgset get cray_firewall.enabled p0
true
```

----- DEFINE NODE GROUPS USED IN `cray_nat_masq` -----

### 2. Define a node group that contains a single gateway node.

Create a custom node group containing only the node that will be used as a gateway. The gateway node must NOT be an RSIP server.

Add an entry for the new node group (`gateway_node1` in the example).

```
smw# cfgset modify --add gateway_node1 cray_node_groups.settings.groups.data p0
smw# cfgset get cray_node_groups.settings.groups p0
gateway_node1
```

Add a member node to the new node group (c0-0c1s3n2 in the example). Substitute the correct cname for this system.

```
smw# cfgset modify --add c0-0c1s3n2 \
cray_node_groups.settings.groups.data.gateway_node1.members p0

smw# cfgset get cray_node_groups.settings.groups.data.gateway_node1.members p0
c0-0c1s3n2
```

Repeat this step for each additional gateway node needed for this system. There should be a separate node group for each gateway node.

### 3. Define a node group that contains the client nodes of a gateway node.

If the nodes that will be clients of a gateway node do not already belong to an existing node group, such as `login_nodes_aarch64`, then repeat the previous step to define a new node group for client nodes.

Note that a group of client nodes can be assigned to only one gateway.

----- ENABLE cray\_nat\_masq -----

### 4. Ensure that `cray_nat_masq` is enabled.

```
smw# cfgset modify --set true cray_nat_masq.enabled p0
```

The remaining steps in this section are use cases. Each use case builds on the previous one, and each shows how the various `cray_nat_masq` setting values will impact the default gateway's `SuSEfirewall12` configuration. The examples illustrating each use case include a "ranger" node (x86\_64) and a "ranger-aarch" node (AArch64) with the following routing:

```
ranger# ip r
default via 172.30.48.1 dev eth0
10.3.1.1 via 10.131.255.254 dev ipogif0
10.128.0.0/14 dev ipogif0 proto kernel scope link src 10.128.0.46
172.30.48.0/20 dev eth0 proto kernel scope link src 172.30.49.170
```

```
ranger_aarch# ip r
10.3.1.1 via 10.131.255.254 dev ipogif0
10.128.0.0/14 dev ipogif0 proto kernel scope link src 10.128.0.53
```

----- USE CASES -----

Each use case listed below builds on the previous one, and each shows how the `cray_nat_masq` setting values will impact the default gateway's `SuSEfirewall12` configuration. The examples illustrating each use case include a "ranger" node (x86\_64) and a "ranger-aarch" node (AArch64) with the following routing:

```
ranger# ip r
default via 172.30.48.1 dev eth0
10.3.1.1 via 10.131.255.254 dev ipogif0
10.128.0.0/14 dev ipogif0 proto kernel scope link src 10.128.0.46
172.30.48.0/20 dev eth0 proto kernel scope link src 172.30.49.170
```

```
ranger_aarch# ip r
10.3.1.1 via 10.131.255.254 dev ipogif0
10.128.0.0/14 dev ipogif0 proto kernel scope link src 10.128.0.53
```

```
-----
-----
```

#### ----- USE CASE 1: SET A DEFAULT GATEWAY -----

### 5. Set a default gateway.

An administrator wants to set a default gateway for one or more AArch64 login nodes.

Add an entry for a new gateway (*ARM* in the example).

```
smw# cfgset modify --add ARM cray_nat_masq.settings.gateways.data p0
smw# cfgset get cray_nat_masq.settings.gateways p0
```

Add a node group that contains the gateway node (*gateway\_node1* in the example).

```
smw# cfgset modify --add gateway_node1 \
cray_nat_masq.settings.gateways.data.ARM.gateway_node_group p0

smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.gateway_node_group p0
gateway_node1
```

Add a node group that contains the client node(s).

```
smw# cfgset modify --add login_nodes_aarch64 \
cray_nat_masq.settings.gateways.data.ARM.client_node_group p0

smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.client_node_group p0
login_nodes_aarch64
```

When the system is booted, the AArch64 login node at 10.128.0.53 will show the new default gateway:

```
ranger-aarch# ip r
default via 10.128.0.46 dev ipogif0
10.3.1.1 via 10.131.255.254 dev ipogif0
10.128.0.0/14 dev ipogif0 proto kernel scope link src 10.128.0.53
```

With the above configuration, the following will be set in `/etc/sysconfig/SuSEfirewall12` on the gateway node:

```
FW_ROUTE=true
FW_MASQUERADE=true
```

#### ----- USE CASE 2: ALLOW AARCH64 LOGIN NODE TO CONTACT LICENSE SERVER -----

### 6. Allow the AArch64 login node to contact a license server.

An administrator wants to allow the AArch64 login node to contact a license server on 172.1.2.3.

To allow a license server to contact the AArch64 login node, an administrator needs to further configure the settings of use case 1 by setting `masquerade_interface` and `fw_forward`.

```
smw# cfgset modify --set eth0 \
cray_nat_masq.settings.gateways.data.ARM.masquerade_interface p0

smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.masquerade_interface p0
eth0
```

```
smw# cfgset modify --add 10.128.0.53,172.1.2.3 \
cray_nat_masq.settings.gateways.data.ARM.fw_forward p0
```

```
smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.fw_forward p0
10.128.0.53,172.1.2.3
```

With the above configuration, the following will be set in `/etc/sysconfig/SuSEfirewall12` on the gateway node:

```
FW_ROUTE=true
FW_MASQUERADE=true
FW_MASQ_DEV=eth0
FW_MASQ_NETS=10.128.0.53
FW_FORWARD=10.128.0.53,172.1.2.3
```

#### ----- USE CASE 3: FORWARD NETWORK TRAFFIC TO AARCH64 LOGIN NODE -----

### 7. Forward network traffic to the AArch64 login node.

An administrator wants to allow the AArch64 login node to send and receive network traffic from the network 172.2.0.0/14.

To allow certain network traffic to be forwarded from the AArch64 login node, an administrator would further configure the settings of use case 2 by adding a value to `fw_forward`.

```
smw# cfgset modify --add 10.128.0.53,172.2.0.0/14 \
cray_nat_masq.settings.gateways.data.ARM.fw_forward p0

smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.fw_forward p0
10.128.0.53,172.1.2.3
10.128.0.53,172.2.0.0/14
```

With the above configuration, the following will be set in `/etc/sysconfig/SuSEfirewall12` on the gateway node:

```
FW_ROUTE=true
FW_MASQUERADE=true
FW_MASQ_DEV=eth0
FW_MASQ_NETS=10.128.0.53
FW_FORWARD=10.128.0.53,172.1.2.3 10.128.0.53,172.2.0.0/14
```

#### ----- USE CASE 4: ALLOW SSH ACCESS TO AARCH64 LOGIN NODE ----- ----- FROM SPECIFIC NETWORK AND PORT -----

### 8. Allow SSH access to AArch64 login node from a specific network and port.

An administrator would like to allow end users to log in to the AArch64 login node from the network 172.1.0.0/14 if they `ssh` to ranger (10.128.0.46) on port 207.

To allow SSH traffic to connect to the AArch64 login node from a particular network on a specific port, an administrator would further configure the settings of use case 3 by adding a value to `fw_forward_masq`.

```
smw# cfgset modify --add 172.1.0.0/14,10.128.0.53,tcp,207,22 \
cray_nat_masq.settings.gateways.data.ARM.fw_forward_masq p0

smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.fw_forward_masq p0
172.1.0.0/14,10.128.0.53,tcp,207,22
```

With the above configuration, the following will be set in `/etc/sysconfig/SuSEfirewall12` on the gateway node:

```
FW_ROUTE=true
FW_MASQUERADE=true
```

```
FW_MASQ_DEV=eth0
FW_MASQ_NETS=10.128.0.53
FW_FORWARD=10.128.0.53,172.1.2.3 10.128.0.53,172.2.0.0/14
FW_FORWARD_MASQ=172.1.0.0/14,10.128.0.53,tcp,207,22
```

----- USE CASE 5: ADD ANOTHER AARCH64 LOGIN NODE -----

## 9. Add another AArch64 login node.

An administrator wants to apply use cases 1–3 to an additional AArch64 login node.

First add the second AArch64 login node to the node group `login_nodes_aarch64`, then add new firewall rules for the second login node.

```
smw# cfgset modify --add 10.128.0.54,172.1.2.3 --add 10.128.0.54,172.2.0.0/14 \
cray_nat_masq.settings.gateways.data.ARM.fw_forward p0
```

```
smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.fw_forward p0
10.128.0.53,172.1.2.3
10.128.0.53,172.2.0.0/14
10.128.0.54,172.1.2.3
10.128.0.54,172.2.0.0/14
```

With the above configuration, the following will be set in `/etc/sysconfig/SuSEfirewall12` on the gateway node:

```
FW_ROUTE=true
FW_MASQUERADE=true
FW_MASQ_DEV=eth0
FW_MASQ_NETS=10.128.0.53 10.128.0.54
FW_FORWARD=10.128.0.53,172.1.2.3 10.128.0.53,172.2.0.0/14 10.128.0.54,172.1.2.3
10.128.0.54,172.2.0.0/14
FW_FORWARD_MASQ=172.1.0.0/14,10.128.0.53,tcp,207,22
```

----- USE CASE 6: ALLOW SSH ACCESS TO SECOND AARCH64 LOGIN NODE -----  
----- ON ALTERNATE PORT -----

## 10. Allow SSH access to a second AArch64 login node on an alternate port.

An administrator wants to allow end users to log in to the second AArch64 login node from network 127.2.0.0/14 if they `ssh` to port 207.

```
smw# cfgset modify --add 127.2.0.0/14,10.128.0.54,tcp,207,22 \
cray_nat_masq.settings.gateways.data.ARM.fw_forward_masq p0

smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.fw_forward_masq p0
172.1.0.0/14,10.128.0.53,tcp,207,22
127.2.0.0/14,10.128.0.54,tcp,207,22
```

With the above configuration, the following will be set in `/etc/sysconfig/SuSEfirewall12` on the gateway node:

```
FW_ROUTE=true
FW_MASQUERADE=true
FW_MASQ_DEV=eth0
FW_MASQ_NETS=10.128.0.53 10.128.0.54
FW_FORWARD=10.128.0.53,172.1.2.3 10.128.0.53,172.2.0.0/14 10.128.0.54,172.1.2.3
10.128.0.54,172.2.0.0/14
FW_FORWARD_MASQ=172.1.0.0/14,10.128.0.53,tcp,207,22
127.2.0.0/14,10.128.0.54,tcp,207,22
```

----- USE CASE 7: ALLOW SSH ACCESS TO SECOND AARCH64 LOGIN NODE -----  
 ----- WITHOUT USING ALTERNATE PORT -----

## 11. Allow SSH access to a second AArch64 login node without using an alternate port.

An administrator wants to allow end users to log in to the AArch64 login node but not have to use an alternate port number. This requires that a secondary IPv4 address be assigned to the gateway node.

In this example, the new IPv4 address 172.30.50.182 will resolve to ranger-aarch. That IPv4 address will be used in the `cray_net` configuration service for the login node network's `ipv4_secondary_addresses`.

- a. Add a secondary IPv4 address to the login node network.

```
smw# cfgset modify --add 172.30.50.182 \
cray_net.settings.hosts.data.login_node.interfaces.login_ethernet.ipv4_secondary_addresses p0

smw# cfgset get
cray_net.settings.hosts.data.login_node.interfaces.login_ethernet.ipv4_secondary_addresses p0
172.30.50.182
```

- b. Configure the `fw_forward_masq` setting to send any traffic to 172.30.50.182 on port 22 to the AArch64 node at 10.128.0.53.

```
smw# cfgset modify --clear \
cray_nat_masq.settings.gateways.data.ARM.fw_forward_masq p0

smw# cfgset modify --add 0/0,10.128.0.53,tcp,22,22,172.30.50.182 \
cray_nat_masq.settings.gateways.data.ARM.fw_forward_masq p0

smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.fw_forward_masq p0
0/0,10.128.0.53,tcp,22,22,172.30.50.182
```

With the above configuration, the following will be set in `/etc/sysconfig/SuSEfirewall12` on the gateway node:

```
FW_ROUTE=true
FW_MASQUERADE=true
FW_MASQ_DEV=eth0
FW_MASQ_NETS=10.128.0.53 10.128.0.54
FW_FORWARD=10.128.0.53,172.1.2.3 10.128.0.53,172.2.0.0/14
FW_FORWARD_MASQ=0/0,10.128.0.53,tcp,22,22,172.30.50.182
```

----- USE CASE 8: FORWARD ALL TRAFFIC FROM AARCH64 LOGIN NODE -----

## 12. USE CASE 8: Forward all traffic from the AArch64 login node.

An administrator wants to allow the AArch64 login node to send and receive network traffic from anywhere, so the network 0/0 may be used.

```
smw# cfgset modify --clear \
cray_nat_masq.settings.gateways.data.ARM.fw_forward p0

smw# cfgset modify --clear \
cray_nat_masq.settings.gateways.data.ARM.fw_forward_masq p0

smw# cfgset modify --add 10.128.0.53,0/0 \
cray_nat_masq.settings.gateways.data.ARM.fw_forward p0

smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.fw_forward p0
10.128.0.53,0/0

smw# cfgset get cray_nat_masq.settings.gateways.data.ARM.fw_forward_masq p0
```

With the above configuration, the following will be set in `/etc/sysconfig/SuSEfirewall12` on the gateway node:

```
FW_ROUTE=true
FW_MASQUERADE=true
FW_MASQ_DEV=eth0
FW_MASQ_NETS=10.128.0.53 10.128.0.54
FW_FORWARD=10.128.0.53,0/0
FW_FORWARD_MASQ=0/0,10.128.0.53,tcp,22,22,172.30.50.182
```

----- CONFIGURE SOME SETTINGS IN `cray_net` -----

If an AArch64 login node was configured as a gateway node in previous steps, some additional configuration is needed in `cray_net`.

### 13. Add a host entry in `cray_net` for the AArch64 login node.

If a default gateway was defined for an AArch64 login node in previous steps, that login node must be defined as a host in `cray_net`.

- a. List all hosts defined in the CLE config set (p0 in example).

```
smw# cfgset get cray_net.settings.hosts.data p0
```

- b. If there is no host for the AArch64 login node, add an entry for that node (`login_aarch64` in the example).

```
smw# cfgset modify --add login_aarch64 \
cray_net.settings.hosts.data p0
```

- c. Add a description.

```
smw# cfgset modify --set 'AArch64 login node' \
cray_net.settings.hosts.data.login_aarch64.description p0
```

- d. Set the host ID.

This example uses the cname of the AArch64 login node defined as a gateway node in previous steps.

```
smw# cfgset modify --set c0-0c1s3n2 \
cray_net.settings.hosts.data.login_aarch64.hostid p0
```

- e. Set the host name.

```
smw# cfgset modify --set hostname_aarch64 \
cray_net.settings.hosts.data.login_aarch64.hostname p0
```

- f. Add one or more aliases to the list of aliases for this host.

```
smw# cfgset modify --add login-arm --add login-aarch --add login-aarch64 \
cray_net.settings.hosts.data.login_aarch64.aliases p0
```

No interfaces are required for this basic host definition.

### 14. Add DNS name servers on the HSN network.

- a. List all DNS servers currently defined for HSN in the CLE config set (p0 in example).



```
smw# cfgset get cray_net.settings.networks.data.hsn.dns_servers p0
```

- b. Add a DNS server name, as needed.

```
smw# cfgset modify --add dns_server_ip \
cray_net.settings.networks.data.hsn.dns_servers p0
```

#### 15. Add to the DNS search list on the HSN network.

- a. List all DNS search terms currently defined for HSN in the CLE config set (p0 in example).

```
smw# cfgset get cray_net.settings.networks.data.hsn.dns_search p0
```

- b. Add a DNS search term, as needed.

```
smw# cfgset modify --add dns_search_url \
cray_net.settings.networks.data.hsn.dns_search p0
```

----- UPDATE AND VALIDATE CLE CONFIG SET -----

#### 16. Update the CLE config set.

```
smw# cfgset update p0
```

#### 17. Validate the CLE config set.

```
smw# cfgset validate p0
```

----- SHUT DOWN AND REBOOT CLE SYSTEM -----

#### 18. Shut down the CLE system.

```
smw# su - crayadm
```

If this site has an `auto.hostname.stop` file, use it to shut down the system.

```
crayadm@smw> xtbootsys -s last -a auto.hostname.stop
```

Otherwise, use the following command to shut down the system.

```
crayadm@smw> xtbootsys -s last -a auto.xtshutdown
```

#### 19. Boot the CLE system.

Replace `auto.hostname.start` with the boot automation file used for this system.

```
crayadm@smw> xtbootsys -a auto.hostname.start
```

**Trouble?** If there are any problems booting CLE, see the *XC™ Series Boot Troubleshooting Guide (S-2565)* for techniques to determine what might be causing the problem.

### 4.7.9 Prevent Unintentional Re-creation of Mail Configuration Files

This procedure is optional. It applies to systems where postfix or sendmail are configured on the SMW.

To prevent the `master.cf` and `main.cf` postfix configuration files from being re-created during software updates or fixes, edit the `/etc/sysconfig/mail` file on the SMW and ensure that the `MAIL_CREATE_CONFIG` setting is set to `"no"`.

```
smw# vi /etc/sysconfig/mail

MAIL_CREATE_CONFIG="no"
```

#### 4.7.10 About System Environmental Data Collections (SEDC)

The System Environment Data Collections (SEDC) tool collects and reports environmental data on all Cray systems in real time. Data includes information from sensors located on significant hardware components at the cabinet and blade level, such as power supplies, processors, memory, and fans. SEDC refers to these sensors as scan IDs. Examples of collected data include cabinet and blade/node temperatures, voltage, current, power, cooling system air pressure, humidity, and statuses. At the node level, data is collected only from the nodes that are powered on.

Effective with the SMW 8.0.UP05 release, SEDC is enabled automatically when the SMW is rebooted after software installation.

For information about how SEDC data is stored in the power management database (PMDb) and querying the PMDb for that data, see *XC™ Series Power Management and SEDC Administration Guide (CLE 6.0.UP07) S-0043*.

## 4.8 Install Additional Software

This is the final stage in the fresh install process. To add features to a Cray XC™ Series system, use the procedures listed. Each feature is optional, but the procedure associated with a feature is required for any feature added to this system.

Use [Installation Checklist 8: Install Additional Software](#) on page 447 to track progress through this final part of the fresh install process.

optional	<a href="#">Install the Dell Systems Management Tools and Documentation DVD</a> on page 292
optional	<a href="#">Install and Configure DataWarp</a> on page 293
optional	<a href="#">Install Cray Programming Environment (PE) Software on x86-64</a>
optional	<a href="#">Install and Configure a Workload Manager (WLM)</a> on page 306
optional	<a href="#">Install and Configure eLogin</a> on page 306

**ATTENTION:** (SMW HA only) If this is an install of the first SMW of an SMW HA system, after completing the additional installation procedures needed for this system, return to the flowchart at the beginning of Chapter 2 "Install and Configure an SMW HA System" in *XC™ Series SMW HA Installation Guide (SLEHA12.SP3.UP07) S-0044* for guidance on how to continue the SMW HA installation process.

## 4.8.1 Install the Dell Systems Management Tools and Documentation DVD

### About this task

This procedure installs the OpenManage Server Administrator (OMSA) software from the Dell Systems Management Tools and Documentation DVD, which is shipped with the SMW. This software enables advanced control over the Integrated Dell Remote Access Controller (iDRAC) and provides features such as Automatic Recovery (automatic system boot after a power event).

Visit the Dell OpenManage Linux Repository to view the Dell OpenManage Server Administrator documentation: <http://linux.dell.com/wiki/index.php/Repository/OMSA>

### Procedure

1. Obtain the Dell System Management Tools and Documentation DVD.
2. Log on to the SMW as `root`.
3. Mount the DVD.

```
smw# mount /dev/cdrom /media/cdrom
```

4. Go to the location of the installation scripts.

```
smw# cd /media/cdrom/SYSMGMT/srvadmin/linux/supportscripts
```

5. Execute the script to install the software.

```
smw# sh srvadmin-install.sh --express
```

6. Start the Server Administrator services.

```
smw# sh srvadmin-services.sh start
```

7. Double-click the icon named **Launch Server Administrator** on the SMW screen.
8. Enter the SMW user name `root`.
9. Enter the SMW `root` account password.

The system can now be managed for Properties, Shutdown, Logs, Alert Management, and Session Management.

## 4.8.2 Install and Configure DataWarp

Cray DataWarp provides an intermediate layer of high bandwidth, file-based storage to applications running on compute nodes. It is comprised of commercial SSD hardware and software, Linux community software, and Cray system hardware and software. DataWarp storage is located on server nodes connected to the Cray system's Aries high speed network (HSN). I/O operations to this storage complete faster than I/O to the attached parallel file system (PFS), allowing the application to resume computation more quickly and resulting in improved application performance. DataWarp storage is transparently available to applications via standard POSIX I/O operations and can be configured in multiple ways for different purposes. DataWarp capacity and bandwidth are dynamically allocated to jobs on request and can be scaled up by adding DataWarp server nodes to the system.

For installation and configuration procedures, see *XC™ Series DataWarp™ Installation and Administration Guide* (S-2564) for this release.

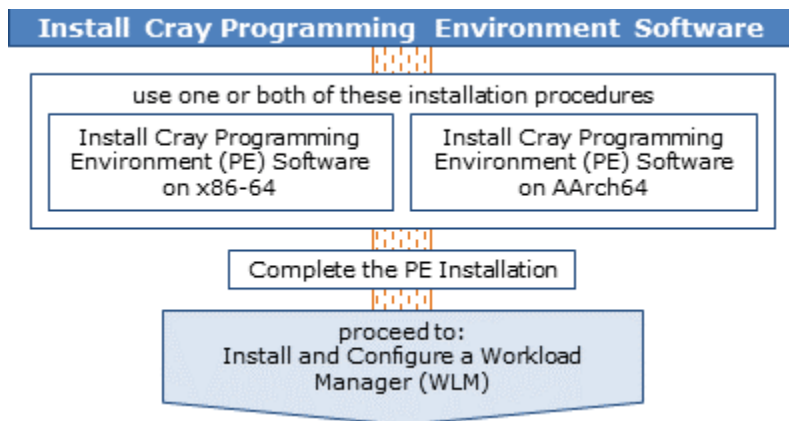
### 4.8.3 Install Cray Programming Environment (PE) Software

Beginning with CLE 6.0.UP07, Cray supports the use of Cray Programming Environment (PE) on AArch64 nodes as well as x86-64 nodes. The approach to installing PE on an AArch64 node is significantly different than the approach that has been used to install PE on x86-64 nodes, because it leverages the Cray Management System (CMS) tool chain—installer, recipes, and `image` CLI commands—instead of using a special installer and `rsync` of ISO content. Because of this, there are two different procedures for installing Cray Programming Environment (PE) software: one for each approach/architecture. The following table summarizes the different approaches.

Table 18. Comparison of PE Installation on x86-64 Nodes versus AArch64 Nodes

Feature	x86-64 Nodes	AArch64 Nodes
software installation per site-specific requirements	<code>install-product.yaml</code> , special PE installer, and dynamic system inspection	standard installer with multiple ISOs Separate ISOs have instructions to set up different recipes and repositories.
package removal	custom script for the PE use case	<code>image prune</code> command that works with installed recipes
incremental updates	special PE installer script that adds packages to an existing CMS image built from a base PE recipe	<code>image install</code> command that works with Cray-provided recipes

Figure 34. Visual Guide to Installing Cray Programming Environment Software



#### 4.8.3.1 Install Cray Programming Environment (PE) Software on x86-64

##### Prerequisites

The boot node is up and available (this procedure includes pushing an image to the boot node).

## About this task

The Cray Developers Toolkit (CDT) for Cray XC Series systems is a package that consists of the basic libraries and components needed to develop and compile code on Cray systems, including the GNU Fortran, C, and C++ compilers. For customers who have purchased the Cray Compiling Environment (CCE) and/or Cray Performance and Measurement Analysis Tools (CPMAT) and are entitled to use them, CCE and/or CPMAT will be included in CDT, but not CDT-NCC. All other compilers are sold, installed, and licensed separately.

This procedure installs and configures the Cray Programming Environment (PE) software to make its content available on Cray XC Series x86-64 compute nodes. A typical PE installation takes 30-45 minutes.

## Procedure

### 1. Create the PE image root.

A PE image can be reused for the monthly releases of PE software, but a fresh image must be used for each new CLE release.

- a. Set an environment variable for the PE image name.

```
smw# export PE_X86_IMAGE=pe_cle_6.0.up07_sles_12sp3_x86-64
smw# echo $PE_X86_IMAGE
```

If this site wishes to use a different name for the PE image when setting the `$PE_X86_IMAGE` environment variable, update the name in the PE profile of the `cray_image_binding` service for the CLE configuration set to match (a later step in this procedure).

- b. Check for an existing PE image.

```
smw# image list | egrep "^[ ]*$PE_X86_IMAGE"
```

- c. Get the name of the PE image recipe on the system.

```
smw# recipe list | grep ^pe
pe_base_cle_6.0.up07_sles_12sp3
```

- d. Create the PE image (`$PE_X86_IMAGE`) using that recipe name.

```
smw# image create -r pe_base_cle_6.0.up07_sles_12sp3 $PE_X86_IMAGE
```

### 2. Install the compiler license RPMs.

If using PE 17.06 or later, skip this step and proceed to step 3 on page 295. If using PE 17.05 or earlier, the Cray Compiling Environment (CCE), Intel, and PGI compilers; and the Cray Performance Measurement and Analysis Tools (CPMAT) all require licenses. These licenses must be installed at this point before installing any of the PE software. For instructions, see the following:

**CCE** *Cray Compiling Environment Release Overview and Installation Guide*, available at <http://pubs.cray.com>

**CPMAT** *Cray Performance Measurement and Analysis Tools Installation Guide*, available at <http://pubs.cray.com>

**Intel compilers** <http://software.intel.com/en-us/articles/intel-software-technical-documentation>

**PGI compilers** <http://www.pgroup.com>

### 3. Copy the most recent PE ISOs to the SMW and mount the ISOs.

Starting with the CDT 16.06 release, the full CDT release is now provided on multiple DVDs rather than on a single one. One DVD will be provided for each of the following files:

- CDT-base-<version>.iso
  - CDT-PrgEnv-cray-<version>.iso (not provided for CDT-NCC)
  - CDT-PrgEnv-intel-<version>.iso
  - CDT-PrgEnv-pgi-<version>.iso
- a. Remove the following directory in case it exists from a previous installation, where `ISO_MOUNT_DIR` is the variable in the `.yaml` configuration file that points to the directory where the contents of the ISO are being copied. In the following instructions, `$ISO_MOUNT_DIR` refers to the directory specified in the `ISO_MOUNT_DIR` field in `install-product.yaml`.

```
# rm -f -r $ISO_MOUNT_DIR
```

- b. Perform the following steps for each ISO file downloaded to combine the contents into a single installation directory.

The possible ISO files and their respective required/optional status are:

- `product-base-version.iso` (REQUIRED)
- `product-PrgEnv-cray-version.iso` (OPTIONAL and not provided for CDT-NCC)
- `product-PrgEnv-intel-version.iso` (OPTIONAL)
- `product-PrgEnv-pgi-version.iso` (OPTIONAL)

If `install-product.yaml` sets `INSTALL_CCE_LIBRARIES : YES` then `product-PrgEnv-cray-version.iso` should be mounted and rsynced.

(NOTE: Above `.iso` is not provided in CDT-NCC packages.)

If `install-product.yaml` sets `INSTALL_INTEL_LIBRARIES : YES` then `product-PrgEnv-intel-version.iso` should be mounted and rsynced.

If `install-product.yaml` sets `INSTALL_PGI_LIBRARIES : YES` then `product-PrgEnv-pgi-version.iso` should be mounted and rsynced.

1. Mount the base ISO listed above.

```
# mount -r -o loop product-<xxx>-version.iso /mnt
```

2. Use the `rsync` command to copy the ISO file content to `ISO_MOUNT_DIR`, the directory where the contents of the ISO are being copied:

```
# rsync -a -v /mnt/ $ISO_MOUNT_DIR/
```

3. Unmount the ISO.

```
# umount /mnt
```

4. Repeat these steps for each optional ISO to be installed. Again, the base ISO is required, but the remaining ISO files (PrgEnv-cray, PrgEnv-intel, PrgEnv-pgi) are optional (where PrgEnv-cray is not provided for CDT-NCC).

4. Install the `craype-installer` RPM from the PE ISO on the SMW.

```
smw# rpm -ivh \
/var/adm/cray/release/pe/mount_iso/installer/craype-installer-*.x86_64.rpm
```

## 5. Configure the installer configuration file.

- a. Copy the install configuration file from the `craype-installer` installation directory.

```
smw# cp -p /opt/cray/craype-installer/default/conf/install-cdt.yaml .
```

- b. Create logs directory that will be used by the installer.

```
smw# mkdir ./logs
```

- c. Update the configuration file, `install-cdt.yaml`.

When `install-cdt.yaml` is opened, there are comment blocks before every keyword listed below describing the valid values for each.

1. For `IMAGE_DIRECTORIES` specify the directory (or directories) for the installer to install into. This parameter must have data on the next line. The data must have four space characters and then a dash character and then a space character and the path to the directory.

For example, this line

```
IMAGE_DIRECTORIES : NONE
```

would be changed to look like this:

```
IMAGE_DIRECTORIES :
- /var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64
```

2. Specify **YES** in each of the `INSTALL_*_LIBRARIES` for the compiler specific PE libraries to be installed. The Pathscale compiler is no longer supported by PE.
3. If the system includes an `ACCELERATOR`, change **NONE** to a comma separated list of one or more of the supported accelerators - **FERMI** or **KEPLER**. See the comments in `install-cdt.yaml` for examples and more information.
4. If the system has more than one type of processor installed, then specify the lowest common denominator for the processor for `CRAY_CPU_TARGET`.

Because this file supports older releases as well, some of the items are not applicable for this release. Those that are applicable for this release are shown in bold.

```
smw# vi install-cdt.yaml
```

```
---
HAS_MAMU_NODES : NO
ACCELERATORS : NONE
NETWORK_TYPE : NONE
CRAY_CPU_TARGET : sandybridge
BOOTNODE_HOSTNAME : NONE
BOOTNODE_ROOT_DIRS :
- /rr/current
ESMS_HOSTNAME : NONE
ESMS_IMAGE_DIRS :
- /cm/images/<your image name>
UNMANAGED_ESLOGINS : NONE
IMAGE_DIRECTORIES :
- /var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64
LOGS_DIR : ./logs
ISO_MOUNT_DIR : ./mount_iso
INSTALL_CCE_LIBRARIES : YES
INSTALL_GNU_LIBRARIES : YES
```

```

INSTALL_INTEL_LIBRARIES      : YES
INSTALL_PATHSCALE_LIBRARIES : NO
INSTALL_PGI_LIBRARIES       : YES

```

## 6. Install PE software from the most recent PE installation media and installer.

### a. Run the PE installer.

This step can take about 30 minutes.

```

smw# cd /var/adm/cray/release/pe
smw# module load craype-installer
smw# craype-installer.pl --install --install-yaml-path ./install-cdt.yaml

```

When the installation completes, output such as the following will be displayed, summarizing the installed packages.

```

1) atp-1.7.5-0_3605.x86_64      (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
2) cray-cddb-1.0.3-0_3575.x86_64 (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
3) cray-dwarf-14.2.0-0.x86_64   (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
<snip>
71) perftools-clients-6.2.2-1.x86_64 (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)

```

### b. Set the default versions for PE (if the install succeeds) by running `set_default` scripts.

```

smw# craype-installer.pl --set-default --install-yaml-path ./install-cdt.yaml

```

Note that at the monthly PE update, this step would not be done until after the image is pushed to the boot node and tested.

### c. Unmount the ISO.

```

smw# umount ./mount_iso

```

### d. Clean up the PE ISO and PE RPMs.

These can be removed because they are large and use up disk space but are no longer needed.

```

smw# rm *.iso *.rpm *.tar.gz

```

Modify `/etc/*rc.local` files before pushing the x86-64 PE image root to the boot node.

`/etc/*rc.local` files are generated as a part of Enable PE (step 8). Site configuration files are provided to tailor these scripts to a site's particular needs. In the `/var/opt/cray/imps/image_roots/$PEIMAGE` directory:

- `/etc/opt/cray/pe/admin-pe/site-config` is provided to define syntax agnostic commands such as loading modules.
- `/etc/opt/cray/pe/admin-pe/site-config.[csh|sh]` is provided for commands requiring shell specific syntax.

Changes to these files will persist across updates to the `pe-setup` package that initially installs them.

## 7. Push the x86-64 PE image root to the boot node.

Note that the example uses `image sqpush` instead of `image push`. For more information, see [About Image Pushes: push versus sqpush](#) on page 42.

This step can take about 10 minutes.

```

smw# image sqpush -d boot $PE_X86_IMAGE

```



## 8. Configure and enable the x86-64 PE bind mount profile.

For a fresh install, configure and enable the `PE_x86_64` bind mount profile in the Cray Image Binding configuration service.

- a. Ensure that the `cray_image_binding` configuration service is enabled.

```
smw# cfgset modify --set true cray_image_binding.enabled p0
smw# cfgset get cray_image_binding.enabled p0
true
```

- b. Change the value of the `PE_x86_64` profile image field to match the name of the image that was used to set `$PE_X86_IMAGE`.

```
smw# cfgset modify --set pe_cle_6.0.up07_sles_12sp3_x86-64 \
cray_image_binding.settings.profiles.data.PE_x86_64.image p0

smw# cfgset get cray_image_binding.settings.profiles.data.PE_x86_64.image p0
pe_cle_6.0.up07_sles_12sp3_x86-64
```

- c. Ensure that the `PE_x86_64` profile callbacks field is set.

**NOTE:** A callback entry must be a relative path, which does not start with a forward slash (/) character.

Clear the list of callback entries.

```
smw# cfgset modify --clear \
cray_image_binding.settings.profiles.data.PE_x86_64.callbacks p0
```

- For the CDT 17.05 release and earlier releases, add the following callback entry:

```
smw# cfgset modify --add opt/cray/pe/bin/pe_postmount_callback.sh \
cray_image_binding.settings.profiles.data.PE_x86_64.callbacks p0
```

- For the CDT 17.06 release and later releases, add a different callback entry:

```
smw# cfgset modify --add opt/cray/pe/bin/pe_setup_callback.sh \
cray_image_binding.settings.profiles.data.PE_x86_64.callbacks p0
```

Confirm that the correct callback entry has been added.

```
smw# cfgset get cray_image_binding.settings.profiles.data.PE_x86_64.callbacks p0
```

- d. Ensure that the `PE_x86_64` profile cleanups field is set.

**NOTE:** A cleanup entry must be a relative path, which does not start with a forward slash (/) character.

Clear the list of cleanup entries.

```
smw# cfgset modify --clear \
cray_image_binding.settings.profiles.data.PE_x86_64.cleanups p0
```

- For the CDT 17.05 release and earlier releases, leave the list of cleanup entries empty.
- For the CDT 17.06 release and later releases, add the following cleanup entry:

```
smw# cfgset modify --add opt/cray/pe/bin/pe_cleanup_callback.sh \
cray_image_binding.settings.profiles.data.PE_x86_64.cleanups p0
```

Confirm that the cleanup entry is correct for this release.

```
smw# cfgset get cray_image_binding.settings.profiles.data.PE_x86_64.cleanups p0
```

- e. Enable the `PE_x86_64` profile.

Has the x86-64 PE image root been pushed to the boot node? If not, first do step 7 on page 298, and then return to this step.

```
smw# cfgset modify --set true \
cray_image_binding.settings.profiles.data.PE_x86_64.enabled p0

smw# cfgset get cray_image_binding.settings.profiles.data.PE_x86_64.enabled p0
true
```

### 4.8.3.2 Install Cray Programming Environment (PE) Software on AArch64

#### About this task

This procedure describes the installation of the Cray Programming Environment (PE) for AArch64 using the CMS installer.

#### Procedure

1. Copy all ISOs for installing PE AArch64 to the SMW.

PE ISOs include the following:

- `CMS-CDT-AArch64-<version>.iso`
- `CMS-CDT-AArch64-PrgEnv-allinea-<version>.iso` (optional)
- `CMS-CDT-AArch64-PrgEnv-allineasup-<version>.iso` (optional)

2. Set a variable for the snapshot name.

- a. Set the `SNAPSHOT` environment variable using the currently booted snapshot name.

```
smw# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')
smw# echo $SNAPSHOT
```

Setting a variable for the snapshot name enables better command substitution in later commands dealing with snapshots.

3. Install a PE ISO.

This step is used for installing recipes and RPMs.

- a. Mount the PE ISO at `/mnt` by doing one of the following options:

- Mount the `CMS-CDT-AArch64-<version>.iso` PE ISO.

```
smw# mount -ro,loop CMS-CDT-AArch64-<version>.iso /mnt
```

- Mount the `CMS-CDT-AArch64-PrgEnv-allinea-<version>.iso` PE ISO. (optional)

```
smw# mount -ro,loop CMS-CDT-AArch64-PrgEnv-allinea-<version>.iso /mnt
```

- Mount the `CMS-CDT-AArch64-PrgEnv-allineasup-<version>.iso` PE ISO. (optional)

```
smw# mount -ro,loop CMS-CDT-AArch64-PrgEnv-allineasup-<version>.iso /mnt
```

- b. Run the installer from the mounted PE ISO.

The installation will add new repositories and new recipes to the system.

```
smw# cd /mnt
smw# ./install.py --target $SNAPSHOT
```

- c. Unmount the PE ISO after the installation completes successfully.

```
smw# cd
smw# umount /mnt
```

Repeat this step for each PE ISO.

#### 4. Set the environment variables.

- a. Set an environment variable for the PE image name.

```
smw# export PE_AARCH_IMAGE=pe_cle_6.0.up07_sles_12sp3_aarch64
smw# echo $PE_AARCH_IMAGE
```

To use a different name for the PE image when setting the `$PE_AARCH_IMAGE` environment variable, update the name in the PE profile of the `cray_image_binding` service for the CLE configuration set to match (a later step in this procedure).

- b. Set an environment variable to prevent the creation of autosave backups.

```
smw# export NOSAVE='--no-save'
smw# echo $NOSAVE
```

Cray recommends removing autosave backups during the install procedure. This will make the install process much faster, and doesn't pose any major risks. If backups are desired, remove the `$NOSAVE` environment variable from the commands in the following steps.

#### 5. Create the PE Image (for Initial Installs only).

This step is only to be used for the initial creation of the PE image, or to create a new PE image. This step is used for installing RPMs into the image.

- a. Find the recipe to be installed by using the `recipe list` command to list all the recipes.

```
smw# recipe list | grep cdt-aarch64
```

- b. Use the `image create` command to create an image with the recipe found in the previous step.

Substitute the correct recipe version in the following command.

```
smw# image create -r cdt-aarch64-<version> $PE_AARCH_IMAGE
```

- c. If installing the ARM Allinea Studio compiler, perform one of the following procedures based on who the compiler and license was obtained from:

- If the ARM Allinea Studio compiler and license were obtained from Cray:

When installing `CMS-CDT-AARCH64-PrgEnv-allinea-<version>.iso`, ensure that `CMS-CDT-AARCH64-<version>.iso` is already installed, and then use the following command:

```
smw# image install -r cdt-aarch64-prgenv-allinea-<version> \
$NOSAVE $PE_AARCH_IMAGE
```

- If the ARM Allinea Studio compiler and license were obtained directly from ARM:

Install the Allinea/19.0 compiler.

**IMPORTANT:** The instructions and commands in this step assume the name of the tar for the Allinea/19.0 compiler will be:

```
ARM-Compiler-for-HPC.19.0_SUSE_12_aarch64.tar
```

```
smw# mkdir /var/opt/cray/imps/image_roots/$PE_AARCH_IMAGE/tmp/tmp.allinea
smw# cp -i -a ARM-Compiler-for-HPC.19.0_SUSE_12_aarch64.tar \
/var/opt/cray/imps/image_roots/$PE_AARCH_IMAGE/tmp/tmp.allinea/.
smw# image chroot $PE_AARCH_IMAGE
# cd /tmp/tmp.allinea
# tar xf ARM-Compiler-for-HPC.19.0_SUSE_12_aarch64.tar
# cd ARM-Compiler-for-HPC.19.0_AArch64_SUSE_12_aarch64
# ./arm-compiler-for-hpc-19.0_Genericic-AArch64_SUSE-12_aarch64-linux-rpm.sh \
-a -i /opt/allinea/19.0.0
# cd
# /bin/rm -r /tmp/tmp.allinea
# exit # image chroot
```

Install Cray's Allinea programming environment.

```
smw# image install -r cdt-aarch64-prgenv-allineasup-<version> \
$NOSAVE $PE_AARCH_IMAGE
```

d. Skip to step 7 on page 303.

6. Install additional PE content using the `image install` command (for Upgrade Installs only).

a. Find the recipe to be installed by using the `recipe list` command to list all the recipes.

```
smw# recipe list | grep cdt-aarch64
```

b. Use the following command to update the PE image for the installed product.

If installing CMS-CDT-AARCH64-<version>.iso, use the following command:

```
smw# image install -r cdt-aarch64-<version> $NOSAVE $PE_AARCH_IMAGE
```

c. If installing the ARM Allinea Studio compiler, perform one of the following procedures based on who the compiler and license was obtained from:

- If the ARM Allinea Studio compiler and license were obtained from Cray:

When installing CMS-CDT-AARCH64-PrgEnv-allinea-<version>.iso, ensure that CMS-CDT-AARCH64-<version>.iso is already installed, and then use the following command:

```
smw# image install -r cdt-aarch64-prgenv-allinea-<version> \
$NOSAVE $PE_AARCH_IMAGE
```

- If the ARM Allinea Studio compiler and license were obtained directly from ARM:

Install the Allinea/19.0 compiler.

**IMPORTANT:** The instructions and commands in this step assume the name of the tar for the Allinea/19.0 compiler will be:

```
ARM-Compiler-for-HPC.19.0_SUSE_12_aarch64.tar
```

```
smw# mkdir /var/opt/cray/imps/image_roots/$PE_AARCH_IMAGE/tmp/tmp.allinea
smw# cp -i -a ARM-Compiler-for-HPC.19.0_SUSE_12_aarch64.tar \
/var/opt/cray/imps/image_roots/$PE_AARCH_IMAGE/tmp/tmp.allinea/.
smw# image chroot $PE_AARCH_IMAGE
# cd /tmp/tmp.allinea
# tar xf ARM-Compiler-for-HPC.19.0_SUSE_12_aarch64.tar
```

```
# cd ARM-Compiler-for-HPC 19.0_AArch64_SUSE_12_aarch64
# ./arm-compiler-for-hpc-19.0_Generic-AArch64_SUSE-12_aarch64-linux-rpm.sh \
-a -i /opt/allinea/19.0.0
# cd
# /bin/rm -r /tmp/tmp.allinea
# exit # image chroot
```

Install Cray's Allinea programming environment.

```
smw# image install -r cdt-aarch64-prgenv-allineasup-<version> \
$NOSAVE $PE_AARCH_IMAGE
```

Repeat this step for each PE ISO installed in step 3 on page 300.

7. (Optional) Use the `image install` command to set the default to the PE components that are on this PE ISO.

Substitute the correct recipe version in the following commands.

- If installing CMS-CDT-AARCH64-<version>.iso, use the following command:

```
smw# image install -r cdt-aarch64-set-default-<version> \
$NOSAVE $PE_AARCH_IMAGE
```

- Sites that are installing CMS-CDT-AARCH64-PrgEnv-allinea-<version>.iso, ensure that CMS-CDT-AARCH64-<version>.iso is already installed, and then use the following command:

```
smw# image install -r cdt-aarch64-prgenv-allinea-set-default-<version> \
$NOSAVE $PE_AARCH_IMAGE
```

- Sites that are installing CMS-CDT-AARCH64-PrgEnv-allineasup-<version>.iso, ensure that CMS-CDT-AARCH64-<version>.iso is already installed, and then use the following command:

```
smw# image install -r cdt-aarch64-prgenv-allineasup-set-default-<version> \
$NOSAVE $PE_AARCH_IMAGE
```

8. Modify `/etc/*rc.local` files before pushing the AArch64 PE image root to the boot node.

`/etc/*rc.local` files are generated as a part of Enable PE (step 10 on page 303). Site configuration files are provided to tailor these scripts to a site's particular needs. In the `/var/opt/cray/imps/image_roots/$PEIMAGE` directory:

- `/etc/opt/cray/pe/admin-pe/site-config` is provided to define syntax agnostic commands such as loading modules.
- `/etc/opt/cray/pe/admin-pe/site-config.[csh|sh]` is provided for commands requiring shell specific syntax.

Changes to these files will persist across updates to the pe-setup package that initially installs them.

9. Push the AArch64 PE image.

Push the PE image to the boot node of the partition where it is to be used:

```
smw# image sqpush -d boot $PE_AARCH_IMAGE
```

10. Configure and enable the AArch64 PE bind mount profile.

For a fresh install, configure and enable the `PE_aarch64` bind mount profile in the Cray Image Binding configuration service. Image binding cannot be performed when there is a job running on a compute node.

- a. Ensure that the `cray_image_binding` configuration service is enabled.

```
smw# cfmset modify --set true cray_image_binding.enabled p0
smw# cfmset get cray_image_binding.enabled p0
true
```

- b. Change the value of the `PE_aarch64` profile image field to match the name of the image that was used to set `$PE_AARCH_IMAGE`.

```
smw# cfmset modify --set pe_cle_6.0.up07_sles_12sp3_aarch64 \
cray_image_binding.settings.profiles.data.PE_aarch64.image p0

smw# cfmset get cray_image_binding.settings.profiles.data.PE_aarch64.image p0
pe_cle_6.0.up07_sles_12sp3_aarch64
```

- c. Enable the `PE_aarch64` profile.

Has the AArch64 PE image root been pushed to the boot node? If not, first do step 9 on page 303, and then return to this step.

```
smw# cfmset modify --set true \
cray_image_binding.settings.profiles.data.PE_aarch64.enabled p0

smw# cfmset get cray_image_binding.settings.profiles.data.PE_aarch64.enabled p0
true
```

### 4.8.3.3 Complete the PE Installation

#### Prerequisites

Cray Programming Environment (PE) software has been installed for x86-64 nodes and/or AArch64 nodes.

#### Procedure

1. Update the CLE config set.

Changes made to the `cray_image_binding` configuration service in the previous procedures modified the CLE config set without running pre- and post-configuration scripts, so that config set was marked invalid. This step uses `cfmset update` in prepare mode (no user interaction) to ensure that all configuration scripts are run.

```
smw# cfmset update -m prepare p0
```

2. Validate the config set.

```
smw# cfmset validate p0
```

3. (Fresh install only) Reboot the system with PE.

A reboot is done only for an initial installation because of the initial setup. It is not required for the monthly PE updates.

```
smw# su - crayadm
crayadm@smw> xtbootsys -s last -a auto.hostname.stop
crayadm@smw> xtbootsys -a auto.hostname.start
```

If this site does not have an `auto.hostname.stop` file, use the following command to shut down the system.

```
crayadm@smw> xtbootsys -s last -a auto.xtshutdown
```

4. Build a sample MPI job that leverages the PE product by compiling and executing the application.

a. Test basic aprun functionality.

1. Log in to the login node.

```
crayadm@smw> ssh boot  
crayadm@boot> ssh login
```

2. Run apstat to get the number of nodes to use for the following commands:

```
crayadm@login> NUMNODES=$((apstat -v | grep XT | awk '{print $3}')); \  
echo NUMNODES is $NUMNODES
```

```
crayadm@login> aprun -n $NUMNODES -N2 python -c "print 'hello world.'"
```

b. Compile a sample MPI program.

1. If PrgEnv-Cray is loaded as a default module, unload it.

```
crayadm@login> module unload PrgEnv-cray
```

2. Load modules.

```
crayadm@login> module load PrgEnv-gnu cray-mpich  
crayadm@login> cd /tmp  
crayadm@login> export CRAY_CPU_TARGET=x86-64
```

3. Obtain sample MPI code for compile.

4. Compile sample MPI code.

5. Execute sample MPI code.

c. Log out of the login node and boot node, and exit the su session to return to being root on the SMW.

```
crayadm@login> exit  
crayadm@boot> exit  
crayadm@smw> exit  
smw#
```

5. Make a snapshot post PE installation.

Cray recommends saving a snapshot of the system immediately after the PE software installation is complete. If any root users make bad changes after the software install is complete, revert to this snapshot to avoid a redo of the entire software install.

```
smw# snaputil list
```

```
smw# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')  
smw# echo $SNAPSHOT
```

```
smw# snaputil create ${SNAPSHOT}.postpe-${TODAY}
```

6. Back up the CLE and global config sets post PE installation.

```
smw# cfgset create --clone global global-postpe-${TODAY}
```

This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-postpe-${TODAY}
```

#### 4.8.4 Install and Configure a Workload Manager (WLM)

Cray XC Series systems support the use of workload manager (WLM) software products. The SMW 8.0.UP07 release supports these three WLM products: PBS, Moab/TORQUE, and Slurm. Each product requires installation and configuration prior to use.

For the most up-to-date information regarding workload manager software compatibility with CLE releases, look on the CrayPort website at <http://crayport.cray.com>.

During the initial installation process, the `cray_ccm` and `cray_cnat` configuration services were disabled. Depending on the requirements of this system, these config services may need to be enabled and configured during or after WLM installation.

<b>PBS Professional™</b>	<p>PBS Professional is a commercial product licensed by Altair Engineering, Inc.</p> <ul style="list-style-type: none"> <li>For general product information: <a href="http://www.altair.com">http://www.altair.com</a></li> <li>For PBS Professional documentation: <a href="http://www.pbsworks.com/PBSProductGT.aspx?n=PBS-Professional&amp;c=Overview-and-Capabilities&amp;d=PBS-Professional,-Documentation">http://www.pbsworks.com/PBSProductGT.aspx?n=PBS-Professional&amp;c=Overview-and-Capabilities&amp;d=PBS-Professional,-Documentation</a></li> <li>Note that PBS Professional uses a license manager, which requires a network connection between the license server and the SDB node on a Cray system.</li> </ul>
<b>Moab™ and TORQUE</b>	<p>Moab and TORQUE are commercial products licensed by Adaptive Computing.</p> <ul style="list-style-type: none"> <li>For general product information: <a href="http://www.adaptivecomputing.com">http://www.adaptivecomputing.com</a></li> </ul>
<b>Slurm</b>	<p>Slurm (Simple Linux Utility for Resource Management) is an open source application that is commercially supported by SchedMD, among others.</p> <ul style="list-style-type: none"> <li>For general product information: <a href="http://www.schedmd.com/">http://www.schedmd.com/</a></li> <li>For Cray-specific installation/configuration instructions: <i>XC™ Series Slurm Installation Guide</i> (S-2538)</li> </ul>

#### 4.8.5 Install and Configure eLogin

Beginning with the CLE 6.0.UP06 release, management support for eLogin nodes is provided entirely by the SMW; no other management node is needed.

In CLE 5.2 releases, external login nodes, then called CDL (Cray Development and Login) nodes, were managed on a CIMS (Cray Integrated Management Server) node running Bright Computing software with Cray additions to configure CDL nodes and build the software image provisioned to the nodes by Bright Computing software. In CLE 6.0 releases prior to UP06, the SMW was used to prepare the image root and data in the config set for eLogin nodes, but the CMC (Cray Management Controller) node, running CSMS (Cray System Management Software) and OpenStack software, was used to provision and manage eLogin nodes.

Now, the SMW (running SLES 12 SP3) is used not only for the config set and eLogin image root but also to provision the image root to the node via PXE boot and perform other management functions.

- Stand-alone SMW**

For systems that use eLogin, complete the SMW/CLE fresh install process by installing and configuring eLogin using *XC™ Series SMW-managed eLogin Installation Guide* (S-3020) for release CLE 6.0.UP07.



- **SMW HA**

For systems that are being installed with SMW high availability (SMW HA), the SMW/CLE fresh install process is complete for the first SMW. The SMW HA system will be configured for eLogin later in the SMW HA fresh install process. Return to the flowchart at the beginning of Chapter 2 "Install and Configure an SMW HA System" in *XC™ Series SMW HA Installation Guide (SLEHA12.SP3.UP07) S-0044* for guidance on how to continue the SMW HA installation process.

## 5 Update SMW/CLE Software

---

Cray provides periodic updates and upgrades to each SMW and CLE release. In an update release, only the minor version numbers (following UP) change, for example, from CLE 6.0.UP01 to CLE 6.0.UP02. In an upgrade release, the major and possibly the minor version numbers change, for example, from SMW 8.0.UP01 to SMW 8.1.UP00.

Follow the procedures in this chapter to update to CLE 6.0.UP07 / SMW 8.0.UP07. The procedures provided here include updating the base operating system running on the SMW, if needed.

**update path** To use these procedures, this system must be running CLE 6.0 / SMW 8.0 (UP05 or UP06) software, and the SMW must be running SUSE Linux Enterprise Server (SLES) version 12 SP3.

The installers for CLE 6.0 / SMW 8.0 are a rewrite from the previous generation, and they use some newer technology to make the update installation process faster and more flexible and to minimize system downtime. These improvements include using a Btrfs file system for staging upgrades, `zypper` repositories for managing packages, and a flexible installer task processor.

**SMW HA only:** For a system that has been configured for SMW high availability (HA), the active SMW must be updated first and then powered down to fail over to the passive SMW, which then becomes the active SMW and can be updated. **Do not use this guide for updating an SMW HA system.** Instead, use *XC™ Series SMW HA Installation Guide (SLEHA12.SP3.UP07) S-0044*.

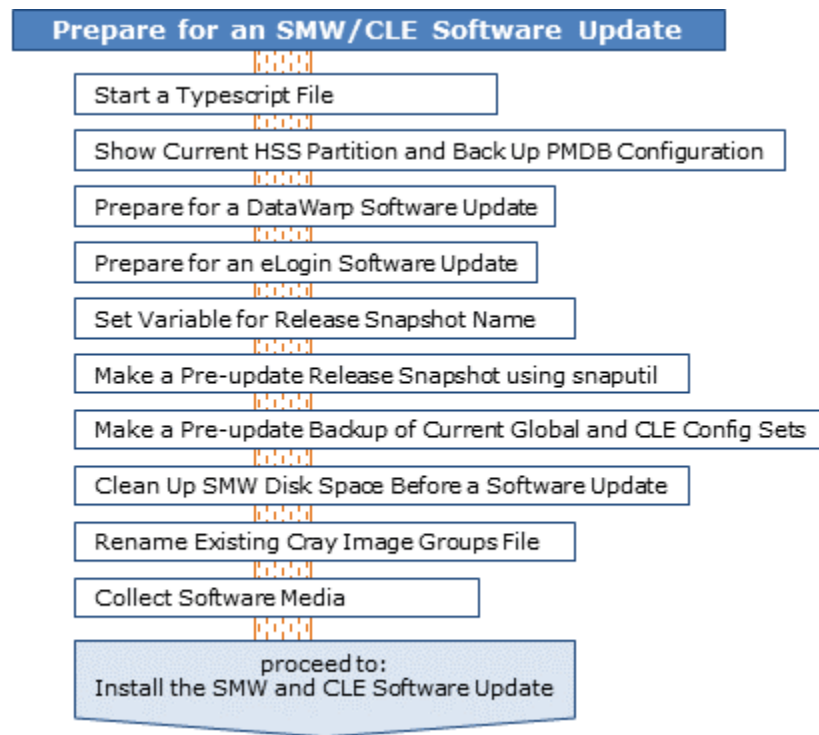
### 5.1 Prepare for an SMW/CLE Software Update

To prepare for an update of the SMW/CLE software, do the following:

- Read:
  - *SMW Release Errata* and *SMW README*.
  - *CLE Release Errata* and the *CLE README*.
  - Field Notices (FN) related to kernel security fixes to identify any changes to this release package. Apply any needed changes before installing the new software.
- If local changes have been made to any automation files, such as `/opt/cray/hss/default/etc/auto.xtshutdown`, back them up before beginning the SMW/CLE update.
- If using the Cray simple event correlator (SEC), determine whether the `/opt/cray/sec/default/SEC_VARIABLES` file has local changes. If it does, make a backup copy of this file before beginning the SMW/CLE update. For more information, see *XC™ Series SEC and check\_xt Software Configuration Guide (S-2542)*.
- Back up any local scripts.

When those preparation activities are done, complete preparation for the software update using the procedures that follow.

Figure 35. Visual Guide to Preparing for an SMW/CLE Software Update



### 5.1.1 Start a Typescript File

#### About this task

Sites can make as few or as many typescripts as they deem useful. Cray recommends starting a typescript file at these milestones:

- just before installing a new software release
- just before configuring the newly installed software

#### Procedure

1. Log in as root to the SMW.
2. (First time only) Create a release directory for the typescript file.

```
smw# mkdir -p /var/adm/cray/release
```

3. Change to the release directory.

```
smw# cd /var/adm/cray/release
```

4. Set a variable equal to today's date.

```
smw# export TODAY=`date +%Y%m%d`
smw# echo ${TODAY}
```

5. Start a typescript file.

```
smw# script -af ${TODAY}.suffix
```

For *suffix*, substitute a unique string to distinguish among typescript files, such as `install.1` or `update.2`.

6. Change prompt to include a timestamp.

```
smw# PS1="\[\e[1;31m\]\u@\h:\w \t # \[\e[0m\]\[\e[00m\]"
```

## 5.1.2 Show Current HSS Partition and Back Up PMDB Configuration

### About this task

This procedure captures some configuration information on the system prior to updating software so that it can be re-created after the update, if desired.

### Procedure

1. Show the current HSS partition configuration.

```
smw# xtcli part_cfg show
```

2. Back up any PMDB config files that have been customized.

PMDB config files are stored in `/var/lib/pgsql/data`.

## 5.1.3 Prepare for a DataWarp Software Update

### Prerequisites

This procedure applies only to systems currently running DataWarp that are updating from CLE 6.0.UP05.

### About this task



**WARNING:** CLE 6.0.UP06 included significant changes to DataWarp that are backward incompatible with previous releases. Any existing DataWarp file system data will be lost if not backed up **prior** to updating CLE from a pre-UP06 release. Existing state files containing current node and pool information will not be compatible with the updated DataWarp service.

This procedure backs up state files and provides an example of how to determine which users have existing DataWarp instances. Although the primary use cases for DataWarp are as a temporary buffer, Cray recommends notifying users of the potential for data loss and allocating sufficient time for them to back up their data before

performing the CLE update. Cray does not recommend that administrators attempt to back up all existing DataWarp file systems.

## Procedure

1. Log in to the `sdb` node as `root`, and back up the current DataWarp state. Note the location of this backup as it is used when updating DataWarp after the CLE update.

```
sdb# module load dws
sdb# dwbackup > /persistent_storage/my_dws_backup.json
```

2. Display existing DataWarp instances and user information.

Determining which users have existing DataWarp instances is done through the site-specific workload manager (WLM). Each WLM has its own syntax for this, and it is beyond the scope of this guide to document all cases. The following example is provided with the caveat that it may be out of sync with changes made by the WLM vendor. For detailed information, refer to the WLM documentation.

In Slurm, for example, current DataWarp information is available using the `scontrol show burst` command.

```
login> scontrol show burst
Name=General DefaultPool=wlm_pool Granularity=384MiB TotalSpace=48.03TiB
UsedSpace=36.3TiB
  Flags=EnablePersistent,TeardownFailure
  StageInTimeout=86400 StageOutTimeout=86400 ValidateTimeout=5 OtherTimeout=300
  GetSysState=/opt/cray/dw_wlm/default/bin/dw_wlm_cli
  Allocated Buffers:
    Name=big CreateTime=2017-12-19T16:46:05 Pool=wlm_pool Size=20.2T
  State=allocated UserID=sally(1000)
    Name=medium CreateTime=2018-01-26T08:32:25 Pool=dwcache Size=10.5T
  State=allocated UserID=sally(1000)
    Name=small CreateTime=2018-02-18T11:57:03 Pool=dwcache Size=5.6T
  State=allocated UserID=rich(1001)
  Per User Buffer Use:
    UserID=sally(1000) Used=30.7T
    UserID=rich(1001) Used=5.6T
```

This shows that users `sally` and `rich` have existing DataWarp instances.

3. Inform users that data in currently existing DataWarp instances will be lost unless backed up.
4. Follow the DataWarp update procedure after completing the update of CLE. See *XC™ Series DataWarp™ Installation and Administration Guide* (S-2564) for details.

### 5.1.4 Prepare for an eLogin Software Update

#### Prerequisites

This system has an SMW-managed eLogin deployment from CLE 6.0.UP06 or a later release.

## About this task

If this system has SMW-managed eLogin nodes running CLE 6.0.UP06 or a later release, it is necessary to capture all node registry information for each eLogin node in this system prior to updating the SMW with the SMW 8.0.UP07 / CLE 6.0.UP07 software release.

## Procedure

Capture eLogin node registry information in a `.csv` file.

The following example writes CLE 6.0.UP06 registry information into a file named `/tmp/cle60up06_esd_nodeinfo`. Substitute the appropriate release of the information being captured.

```
smw# enode list --fields all --format csv --output /tmp/cle60up06_esd_nodeinfo
```

After the SMW/CLE software update is complete, the `/tmp/cle60up06_esd_nodeinfo` file will be used to re-register all eLogin nodes using the eLogin update process documented in the CLE 6.0.UP07 version of *XC™ Series SMW-managed eLogin Installation Guide (S-3020)*.

### 5.1.5 Set Variable for Release Snapshot Name

## About this task

This procedure sets a variable for the name of the snapshot that will be used to install and configure the software update. Setting a variable now enables better command substitution in later commands dealing with snapshots.

(SMW HA only) This is especially important for SMW HA systems because it makes it easier to use the exact same snapshot name for both SMWs. Using different snapshots results in HSS database (MySQL) inconsistencies, which causes problems at failover.

## Procedure

Set the `SNAPSHOT` environment variable for the release snapshot to the name of the release to be installed and today's date.

```
smw# export SNAPSHOT=SMW-8.0UP07_CLE-6.0UP07.${TODAY}
smw# echo $SNAPSHOT
```

### 5.1.6 Make a Pre-update Release Snapshot using snaputil

## Prerequisites

This procedure assumes that the variable for the release snapshot name was set in [Set Variable for Release Snapshot Name](#) on page 312.

## About this task

This procedure uses `snaputil` to make an archival release snapshot prior to any update activities.

For more snapshot information, see [About Snapshots and Config Set Backups](#) on page 21.

## Procedure

1. List the available snapshots on the system.

```
smw# snaputil list
```

2. Create the pre-update archival release snapshot.

If the running system is what will be updated, create a snapshot from the currently booted system snapshot (denoted by "cur"), which is what `snaputil` uses by default. (Note that the default snapshot, denoted by "def," is what the system will **boot from** by default.)

```
smw# snaputil create ${SNAPSHOT}.preupdate-${TODAY}
```

If a different snapshot will be used for the software update, specify it using the `--from` argument with the `snaputil` command.

```
smw# snaputil create ${SNAPSHOT}.preupdate --from different_snapshot
```

### 5.1.7 Make a Pre-update Backup of Current Global and CLE Config Sets

#### About this task

This procedure uses `cfgset` to make an archival config set backup prior to any update activities.

**Best Practice.** Make a snapshot and back up config sets at the same time to keep them in sync. Name the snapshot and config set backup using the same suffix and date/time stamp to help administrators identify which snapshot and config set backup pairs belong together.

For more config set backup information, see [About Snapshots and Config Set Backups](#) on page 21.

## Procedure

1. Back up the current global config set.

```
smw# cfgset create --clone global global-preupdate-${TODAY}
```

2. Back up the current CLE config set.

This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-preupdate-${TODAY}
```

### 5.1.8 Clean Up SMW Disk Space Before a Software Update

#### About this task

Without proactive management, old snapshots, image roots, boot images, and logs can accumulate over time, interfering with the proper operation of the system. Space management should be a normal part of the system

administration of the system, but it is particularly important prior to beginning a software update. The following steps focus attention on areas that make the most difference.

## Procedure

### 1. Check disk space.

It is important to ensure the system does not run low on disk space. While the default file-system sizes allow for a reasonable number of previous releases, use the following command to find out how much disk space is available.

```
smw# df -h
```

### 2. Remove unneeded snapshots.

Because `snaptail(8)` snapshots use Copy On Write (COW) to reduce disk usage, the specific amount of disk space used by each snapshot can vary. And because Btrfs quotas, the mechanism used to monitor space utilization by snapshots, are typically disabled due to severe performance issues, it can be difficult to know exactly how much space a snapshot uses.

The most pragmatic approach to snapshot space management is to simply delete unneeded snapshots. The amount of space a snapshot uses is the delta between it and its parent snapshot.

- A snapshot with few or no changes in it will use little additional space.
- Snapshots of previous releases use more space than snapshots that simply contain configuration changes.
- Snapshots that contain SUSE updates will use more space.

To determine the lineage of a snapshot, use the following command:

```
smw# snaputil show snapshot_name
```

This command also shows the CLE and SMW software version installed in that snapshot. Because `snaptail(8)` snapshots span file systems, monitor the space available on `/` and `/var/opt/cray/repos` before and after removing snapshots to see the effect of the removal.

### 3. Remove unneeded repositories.

The software repositories contained in `/var/opt/cray/repos` may require some cleanup. When a software update is performed, some repositories keep the same name, such as the base SUSE repositories, but some repositories are versioned, so a new one is created for each update. For example, the repositories containing the CLE software have the CLE version as part of the repository name, so a new repository is provided every release. The installation process disables the repositories from previous releases, but it does not delete them. This means the new snapshot typically will contain the repositories not only for that release, but for previous release(s) as well, albeit disabled.

Although it seems that these old repositories are taking up additional space, because of the way COW works, if a previous snapshot contains the old software, the old repos in the new snapshot do not actually take up additional space. However, if the system administrator deletes the snapshot of the old software, the space would not be freed because those repositories exist in the new snapshot. To free that space, delete both the old snapshot and the old repositories in all other snapshots.

### 4. Clean up file systems not included in snapshots.

The following file systems on the SMW are not part of snapshots:

- `/var/opt/cray/imps`



This file system contains the config sets, image roots, and boot images for the system.

- **Config sets.** Config sets can be listed and removed with the `cfgset(8)` utility. While config sets are typically relatively small, determine their space usage with the following command:

```
smw# du -sh /var/opt/cray/imps/config/sets/*
```

- **Boot images.** Boot images are contained in `/var/opt/cray/imps/boot_images`. Because there is no Cray-specific utility to manage these files, use standard Linux utilities to list and delete them. These files contain an encapsulated copy of the images and are what is actually sent to the nodes during the boot process.
- **Image roots.** Image roots can be managed with the `image(8)` utility. Image roots are an intermediate step to the creation of the boot images and take a considerable amount of space. Determine how much space each image root is using with the following command:

```
smw# du -sh /var/opt/cray/imps/image_roots/*
```

- /home
- /tmp
- /boot

Because `/boot` contains kernels and `initrd` files for all snapshots, Cray recommends NOT removing files from this directory. If files must be removed, use the `snaptutil show` utility to show which kernel is being used by each snapshot, to avoid deleting kernels that are in use.

- /var/opt/cray/disk/1

This file system contains a number of subdirectories:

- `debug`: used in support of system dumps.
- `dump`: used in support of system dumps.
- `logs`: contains system logs. This directory is maintained by the `xttrim(8)` utility and configured with `xttrim.conf(5)`. See the man pages to adjust the compression and deletion settings.

## 5. Clean up file systems on the boot and SDB nodes.

The boot and SDB nodes have the following persistent file systems:

- `boot:/cray_home`
- `boot:/non_volatile`
- `boot:/var/opt/cray/imps`
- `sdb:/alps_shared`
- `sdb:/var/lib/mysql`
- `sdb:/var/opt/cray/ncmd`

Of these file systems, only `boot:/home` and `boot:/var/opt/cray/imps` need to be proactively managed. The `/var/opt/cray/imps` file system on the boot node contains image roots that are used in support of booting netroot nodes. Unlike on the SMW, the image roots present on the boot node are used directly during the boot of netroot nodes. There are no Cray tools to list and remove image roots on the boot node, so Linux commands must be used to manage these files.

## 6. (Conditional) Clean up and manage the `/var/opt/cray/imps/localtemp` directory.

This step applies only to systems that are being updated from CLE 6.0.UP05 or an earlier release.

The `/var/opt/cray/imps/localtemp` directory can fill up with temporary files if not properly managed. These files are created as a result of running various image and configuration management tools (`imgbuilder`, `image`, `recipe`, `pkgcoll`, `repo` and `cfgset`). If not cleaned up, these files can negatively affect the time required to run `SMWinstall` and update the system.

Beginning with the CLE 6.0.UP06 release, Cray adds any relevant data from these files to the normal logs and removes the files. However, for systems being updated from CLE 6.0.UP05 or an earlier release, the following steps must be performed to clean up the temporary files and directories and to avoid filling up `/var/opt/cray/imps/localtemp` going forward.

- a. Remove temporary files and directories from the `/var/opt/cray/imps/localtemp` directory.
- b. Modify the `delete_callback_output` value in the `/etc/opt/cray/imps/imps.json` file.

Note that this change must be made to the `imps.json` file in the current snapshot, which is typically the snapshot used to create the target snapshot into which the SMW/CLE software update will be installed. If some other snapshot will be used to create the installation target snapshot (not common), make this change to the `imps.json` file in that snapshot.

```
vi /etc/opt/cray/imps/imps.json
```

In the file, find the `delete_callback_output` variable and set it to `true`.

```
{
  "system_config": {
    "service": {
      "imps": {
        ...
        "image_recipes": {
          "delete_callback_output": true,
          ...
        }
        ...
      }
    }
  }
}
```

### 5.1.9 Rename Existing Cray Image Groups File

#### Prerequisites

The `cray_image_groups.yaml` file is present from a previous installation of an SMW 8.0 / CLE 6.0 release.

#### About this task

The Cray image groups file (`cray_image_groups.yaml`) is included with the new set of templates in every SMW/CLE software installation and update. When the installer runs it will place the following two files into the global config set:

- `cray_image_groups.yaml` (ONLY if a file with that name is not already there—the existing one is never overwritten)
- `cray_image_groups.yaml.last` (ALWAYS, and the contents are identical to the contents of the new `cray_image_groups.yaml`)

For a software update from CLE 6.0.UP06 or an earlier release, rename (move) the existing `cray_image_groups.yaml` so the installer will add the new one with all the correct content for the new release. Retain the renamed file for reference in a later procedure, when the new `cray_image_groups.yaml` is customized for this system.

## Procedure

Rename the existing `cray_image_groups.yaml`.

```
smw# mv /var/opt/cray/imps/config/sets/global/config/cray_image_groups.yaml \
/var/opt/cray/imps/config/sets/global/config/cray_image_groups.yaml.up0x
```

This example command appends `up0x` to the file name, where `up0x` designates the CLE 6.0 release of the existing image groups file. For example, if this is the CLE 6.0.UP06 file, rename it `cray_image_groups.yaml.up06`.

### 5.1.10 Collect Software Media

#### About this task

The Cray release distribution consists of one DVD and several other pieces of media that may be on DVDs or furnished as ISO files. These ISO files are available for download at CrayPort (<https://crayport.cray.com>).

The installer requires several ISO files to be available for setting up and installing packages from SLE repositories. The names of these ISOs are hard-coded in the installer configuration, but the containing directory can be anywhere that makes sense for this site.

**IMPORTANT:** The installer expects these ISO files to be located in the default location, `/root/isos`. If that default location is not used for this system, specify the correct location for the ISO files by using the `--iso-dir` option with the `SMWinstall` command.

## Procedure

1. Make an ISO directory on the SMW that is exempt from snapshots, and link it to the default ISO location.

Instead of placing the ISOs directly in `/root/isos`, create a directory in the Btrfs subvolume `/var/adm/cray`, which is exempt from snapshots, place the ISOs there, and link the two directories. This prevents the large ISO files from unnecessarily increasing the size of snapshots.

```
smw# mkdir -p /var/adm/cray/release/isos
smw# ln -s /var/adm/cray/release/isos /root/isos
```

The "ISO directory" referred to in the following steps is `/var/adm/cray/release/isos`.

2. Download the SLES 12 SP3 distribution ISOs to the ISO directory on the SMW.

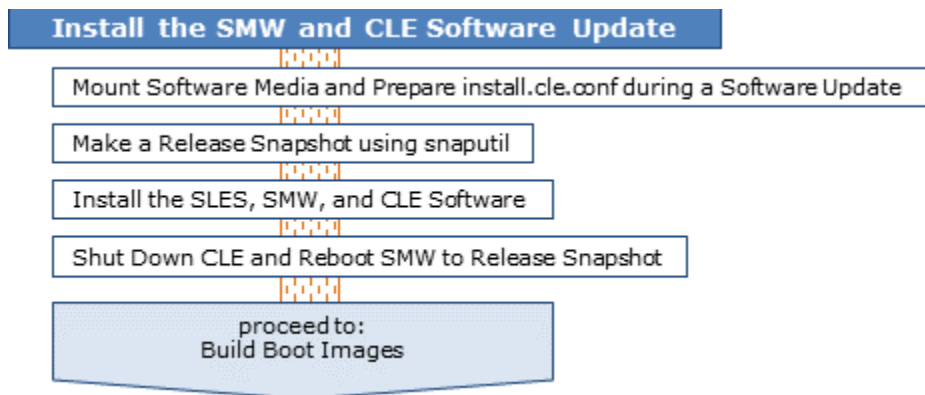
- `SLE-12-SP3-Server-DVD-x86_64-GM-DVD1.iso`
- `SLE-12-SP3-Server-DVD-aarch64-GM-DVD1.iso`
- `SLE-12-SP3-SDK-DVD-x86_64-GM-DVD1.iso`
- `SLE-12-SP3-SDK-DVD-aarch64-GM-DVD1.iso`

- SLE-12-SP3-WE-DVD-x86\_64-GM-DVD1.iso
  - SLE-12-Modules-v3.iso
3. Download the CentOS 6.5 distribution ISOs to the ISO directory on the SMW.
    - CentOS-6.5-x86\_64-bin-DVD1.iso
  4. Download CLE 6.0 and SMW 8.0 ISOs to the ISO directory on the SMW.
    - SMW release: smw-8.0.7128-201806242300.iso
    - CLE release: cle-6.0.7128-201806242300.iso
  5. Download the SLES 12 security updates ISO (sleupdate-12sp3+180209-201802091328.iso) to the ISO directory on the SMW.
  6. Make a directory on the SMW (if it does not already exist) to hold any patches that may be available on CrayPort.
 

```
smw# mkdir -p /var/adm/cray/release/patchsets
```
  7. Download any SMW, CLE, and SMW HA patches to the patchset directory on the SMW, as described in the release notes (ERRATA and README files).

## 5.2 Install the SMW and CLE Software Update

Figure 36. Visual Guide to Installing the SMW and CLE Software Update



### 5.2.1 Mount Software Media and Prepare install.cle.conf during a Software Update

#### Prerequisites

The release software media have been collected and placed in the appropriate directories on the SMW.

## About this task

This procedure sets environment variables for the SMW and CLE media, mounts the SMW and CLE media, and updates the current `install.cle.conf` to add any new configuration options from this release and to disable image building when the installer is run.

## Procedure

### —— SET ENVIRONMENT VARIABLES ——

1. Set an environment variable for the SMW media.

- a. Confirm that this is the right SMW media.

```
smw# ls -l /root/isos/smw*iso
-rw-r--r-- 1 root root 427184128 July 10 21:42 smw-8.0.7128-201806242300.iso
```

- b. Set an environment variable for the SMW media.

```
smw# export SMW_RELEASE=/root/isos/smw-8.0.7128-201806242300.iso
smw# echo $SMW_RELEASE
```

2. Set an environment variable for the CLE media.

- a. Confirm that this is the right CLE media.

```
smw# ls -l /root/isos/cle*iso
-rw-r--r-- 1 root root 1146388480 July 10 20:38 cle-6.0.7128-201806242300.iso
```

- b. Set an environment variable for the CLE media.

```
smw# export CLE_RELEASE=/root/isos/cle-6.0.7128-201806242300.iso
smw# echo $CLE_RELEASE
```

3. Set an environment variable for the SLES 12 SP3 security updates media.

- a. Confirm that this is the right SLES 12 SP3 security updates media.

```
smw# ls -l /root/isos/sleupdate*iso
-rw-r--r-- 1 root root 3482064896 July 10 09:19 /root/isos/
sleupdate-12sp3+180209-201802091328.iso
```

- b. Set an environment variable for the SLES 12 SP3 security updates media.

```
smw# export SLE_UPDATE=/root/isos/sleupdate-12sp3+180209-201802091328.iso
smw# echo $SLE_UPDATE
```

### —— MOUNT THE SMW, SLES, AND CLE MEDIA ——

4. Mount the SMW release media.

```
smw# mkdir -p /media/SMW
smw# mount -o loop,ro ${SMW_RELEASE} /media/SMW
```

5. Mount the SLES security updates media.

```
smw# mkdir -p /media/sleupdate
smw# mount -o loop,ro ${SLE_UPDATE} /media/sleupdate
```

## 6. Mount the CLE release media.

```
smw# mkdir -p /media/CLE
smw# mount -o loop,ro ${CLE_RELEASE} /media/CLE
```

———— PREPARE THE `install.cle.conf` FILE ————

The `install.cle.conf` file contains configuration that controls the installer's image building behavior. The purpose of the following two steps is to change the current `install.cle.conf` file to disable image building and include any new configuration options provided in the new release.

## 7. Identify changes to `install.cle.conf` with this release.

Compare `install.cle.conf.example` from the CLE media just mounted to the current version of `install.cle.conf` on the SMW to see if there are additional configuration options available with this release of CLE software.

```
smw# diff /media/CLE/products/cle/install.cle.conf.example \
/var/adm/cray/install.cle.conf
```

The diff output should show at least the following, indicating that the current `install.cle.conf` will automatically generate boot images when the installer (`SMWinstall`) is run. It may show other differences as well, if new configuration options are available with this release of CLE software.

```
13c13
< build_images: no
---
> build_images: yes
```

## 8. Edit the current `install.cle.conf` to disable image building and add new configuration options (if any).

If the output of the `diff` command in the previous step shows `build_images: yes` for the current `install.cle.conf` file, edit that file and set `build_images: no` to disable image building. Images will be built later in the update process.

```
smw# vi /var/adm/cray/install.cle.conf
```

If there are other configuration options in `install.cle.conf.example` that appear in the diff output, add those to `install.cle.conf` as well.

**Why disable image building?** Image building is disabled during installation of the software updates for the following reasons:

- Image recipes that fail recipe validation could cause the software installation to fail. The `imgbuilder` command calls `recipe validate` before calling `image create`, and `recipe validate` calls `zypper --dry-run` to validate that the packages installed by the recipe are actually provided by the repos referenced in the recipe. It is possible for custom recipes (including FIO, and potentially WLM) to fail the new recipe validation step but still be buildable. Performing the software installation without running `imgbuilder` will allow the installation to complete.
- Some administrators prefer to install the software updates with image building disabled so that they can look at the new recipe names prior to updating custom recipes.

## 5.2.2 Make a Release Snapshot using snaputil

### Prerequisites

This procedure assumes the variables for the release snapshot name and the base operating system snapshot name were set in [Set Variable for Release Snapshot Name](#) on page 312.

### About this task

This procedure creates the release snapshot, which is the snapshot into which the base operating system and software updates will be installed while the system is running. The SMW will be booted from this snapshot later in the process. See the `snaputil(8)` man page for more information about using the `snaputil` utility.

### Procedure

1. List the available snapshots on the system.

```
smw# snaputil list
```

2. Create the release snapshot.

If the running system is what will be updated, create a snapshot from the currently booted system (denoted by "cur"), which is what `snaputil` uses by default. (Note that the default snapshot, denoted by "def," is what the system will boot from by default.)

```
smw# snaputil create ${SNAPSHOT}
```

If a different snapshot will be used for the software update, specify it using the `--from` argument with the `snaputil` command.

```
smw# snaputil create ${SNAPSHOT} --from different_snapshot
```

**NOTE:** The HSS database in a snapshot that has not been booted recently may no longer reflect the physical state (what components are where) or administrative state (which nodes are enabled, disabled, set-to-service, and so forth) of the XC system. In such cases, after the SMW is rebooted to that snapshot, run `freshenhss` in the snapshot to restore this information from the last-booted snapshot. See [About Snapshots and Config Set Backups](#) on page 21 for more information.

3. Remove obsolete repositories from release snapshot, if needed.

Because all new repos were provided for CLE 6.0.UP06, installing the SMW ISO during an update from a release earlier than CLE 6.0.UP06 will take up more space than usual, and the disk may run out of room. To help with this problem, use `snaputil chroot` to go into the snapshot and remove any obsolete repos. For more information about removing repos, see the "Remove unneeded repositories" step in [Clean Up SMW Disk Space Before a Software Update](#) on page 313.

## 5.2.3 Install the SLES, SMW, and CLE Software

### Prerequisites

- This system is being updated from SMW 8.0 / CLE 6.0 UP05 or UP06 (if unsure: `cat /etc/opt/cray/release/cle-release`).
- The SMW, SLES update, and CLE media have been mounted.
- Image building has been disabled in `install.cle.conf`.
- The release snapshot, which will be used as the target snapshot for the installation in this procedure, has been created.

### About this task

An update from UP05 or a later release does not include an SMW base operating system update, because those releases and the current release use SLES 12 SP3 as the base OS. This procedure runs the installer to update the Cray SMW software, install SLES security updates, and update CLE software.

### Procedure

Install the software updates into the release snapshot.

**ATTENTION:** Do not run the installer from `/root/isos` when `/root/isos` is a link to `/var/adm/cray/release/isos` because that will cause an installer error.

```
smw# /media/SMW/install.py --target=${SNAPSHOT}
smw# /media/sleupdate/install.py --target=${SNAPSHOT}
smw# /media/CLE/install.py --target=${SNAPSHOT}
```

**Trouble?** The installer automatically disables Btrfs quotas. If the installation appears to hang, Cray suggests allowing it to continue and complete instead of interrupting it.

## 5.2.4 Shut Down CLE and Reboot SMW to Release Snapshot

### Prerequisites

The base OS, the Cray SMW and CLE software, and SLES security updates have been installed into the release snapshot.

### About this task

The default snapshot must be set to the correct snapshot before booting the SMW.

### Procedure

———— SET THE DEFAULT SNAPSHOT ————

1. Set the default snapshot.



```
smw# /boot/install-support/default/snaputil default ${SNAPSHOT}
```

2. Verify that the correct snapshot is the default.

```
smw# /boot/install-support/default/snaputil list
```

———— SHUT DOWN THE CLE SYSTEM ————

3. If the boot node is up, then shut down the CLE system.

Use the shutdown automation file customized for this system.

```
smw# su - crayadm
```

- If this system has a shutdown automation file, use it to shut down the system (substitute the correct host name in the example command).

```
crayadm@smw> xtbootsys -s last -a auto.hostname.stop
```

- If this system does not have a shutdown automation file, use the following command instead.

```
crayadm@smw> xtbootsys -s last -a auto.xtshutdown
```

```
crayadm@smw> exit  
smw#
```

———— REBOOT THE SMW TO THE RELEASE SNAPSHOT ————

4. Exit the typescript file prior to rebooting the SMW.

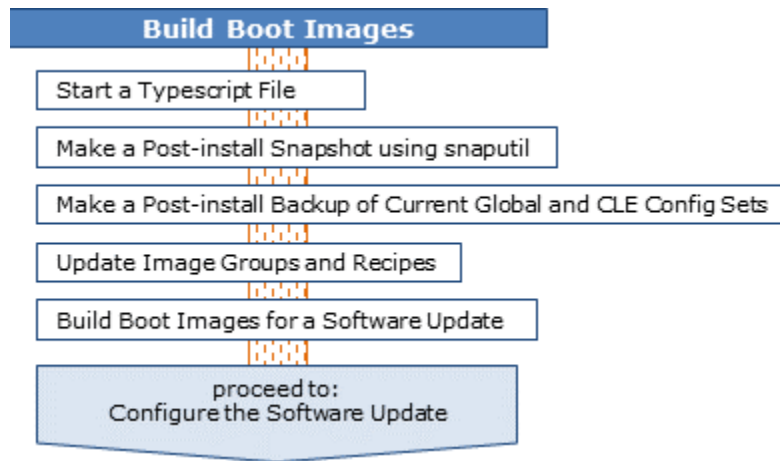
```
smw# exit
```

5. Reboot the SMW.

```
smw# reboot
```

## 5.3 Build Boot Images

Figure 37. Visual Guide to Building Boot Images



### 5.3.1 Start a Typescript File

#### About this task

Sites can make as few or as many typescripts as they deem useful. Cray recommends starting a typescript file at these milestones:

- just before installing a new software release
- just before configuring the newly installed software

#### Procedure

1. Log in as root to the SMW.
2. (First time only) Create a release directory for the typescript file.

```
smw# mkdir -p /var/adm/cray/release
```

3. Change to the release directory.

```
smw# cd /var/adm/cray/release
```

4. Set a variable equal to today's date.

```
smw# export TODAY=`date +%Y%m%d`
smw# echo ${TODAY}
```

5. Start a typescript file.

```
smw# script -af ${TODAY}.suffix
```

For *suffix*, substitute a unique string to distinguish among typescript files, such as `install.1` or `update.2`.

6. Change prompt to include a timestamp.

```
smw# PS1="\[\e[1;31m\]\u@\h:\w \t # \[\e[0m\]\[\e[00m\]"
```

### 5.3.2 Make a Post-install Snapshot using snaputil

#### About this task

This procedure uses `snaputil` to make an archival snapshot of the system after installing a new software release (fresh install or software update) and before configuring that software.

**Best Practice.** Make a snapshot and back up config sets at the same time to keep them in sync. Name the snapshot and config set backup using the same suffix and date/time stamp to help administrators identify which snapshot and config set backup pairs belong together.

For more information, see [About Snapshots and Config Set Backups](#) on page 21.

#### Procedure

1. List the available snapshots on the system.

```
smw# snaputil list
```

2. Set the `SNAPSHOT` environment variable using the currently booted snapshot name.

```
smw# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')
smw# echo $SNAPSHOT
```

Setting a variable for the snapshot name enables better command substitution in later commands dealing with snapshots.

This is especially important for SMW HA systems because it makes it easier to use the exact same snapshot name for both SMWs. Using different snapshots results in HSS database (MySQL) inconsistencies, which causes problems at failover.

3. Create a new snapshot.

```
smw# snaputil create ${SNAPSHOT}.postinstall-${TODAY}
```

### 5.3.3 Make a Post-install Backup of Current Global and CLE Config Sets

#### About this task

This procedure uses the `cfgset` command to create a post-install backup of the global and CLE config sets after installing a new software release (fresh install or software update) and before configuring that software.

## Procedure

1. Back up the current global config set.

```
smw# cfgset create --clone global global-postinstall-${TODAY}
```

2. (Software update only) Back up the current CLE config set.

SKIP THIS STEP if this is a fresh install, because the CLE config set has not yet been created at this point in the process.

This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-postinstall-${TODAY}
```

### 5.3.4 Update Image Groups and Recipes

#### Prerequisites

The `cray_image_groups.yaml` file was renamed in [Rename Existing Cray Image Groups File](#) on page 316 so that a CLE 6.0.UP07 version of that file would be installed.

#### About this task

This procedure updates the `cray_image_groups` configuration file and ensures that the Cray image groups file and custom recipes have recipe names that indicate the current release.

## Procedure

1. Update the UP07 Cray image groups file.

In an earlier procedure, the `cray_image_groups.yaml` file used prior to this software update was renamed to clear the way for the software to install a new `cray_image_groups.yaml` for UP07. Compare the previous version with the UP07 version before making any changes to the UP07 version.

```
smw# vi /var/opt/cray/imps/config/sets/global/config/cray_image_groups.yaml
```

Note that the Cray-provided `cray_image_groups.yaml` file has changed in the following ways:

- Image specifications (stanzas) for AArch64 images have been added. If this system has no AArch64 nodes, these stanzas will be removed later in this step.
- Variable substitution for the CLE release name (`{cle_release_lowercase}`) in the `recipe` and `dest` fields is now supported (new in CLE 6.0.UP07). Any `cray_image_groups.yaml` file that uses this new variable in recipe and image names cannot be used in systems running CLE 6.0.UP06 or an earlier release.
- The `arch` field (new in CLE 6.0.UP06) specifies the architecture to be used for the image root. If this field is not present, the recipe's default architecture is used.
- The `export_format` field (new in CLE 6.0.UP06) specifies the image export format. Use of a file extension (e.g., `.cpio`) in the `dest` field is deprecated.

This example image specification (stanza) shows these changes in bold.

```
- recipe: "compute_cle_{cle_release_lowercase}_sles_12sp3_ari"
  dest: "compute{note}_cle_{cle_release_lowercase}-build{cle_build}{patch}_sles_12sp3-x86_64-created{date}"
```

```
arch: "x86_64"
export_format: "cpio"
nims_group: "compute"
```

For more information, see [About Image Groups and How to Customize Them](#) on page 38.

a. Save a copy of the old default image group.

Make a copy of the default image group from the old file, `cray_image_groups.yaml.up0x`, rename it (for example, `default_up05` if updating from UP05), and add it to the new UP07 `cray_image_groups.yaml` file. By preserving the old default group in the new image groups file, this site will be able to run `imgbuilder` while booted from the associated earlier release, if necessary, specifying the old default group in the command line.

b. Save a copy of the new default image group.

Make a copy of the new default image group prior to making any changes to it, and name the copy `default_up07_original`.

c. Customize the new `cray_image_groups.yaml` file to reflect the needs of this site.

Review the contents of the old image groups file, `cray_image_groups.yaml.up0x`, to see what should be customized in the new UP07 `cray_image_groups.yaml` file.

Possible customizations:

- Migrate image stanzas from the old default image group to the default image group in the UP07 `cray_image_groups.yaml` file.
- Migrate site-specific image groups from the old image groups file (`cray_image_groups.yaml.up0x`) to the UP07 `cray_image_groups.yaml` file.
- Within UP07 `cray_image_groups.yaml`:
  - Remove architecture-specific image stanzas from the default image group if this system does not have the associated hardware. That is, if there are no AArch64 nodes in this system, remove the AArch64 stanzas, and if there are no x86-64 nodes, remove the x86-64 stanzas.
  - Move image stanzas from other image groups to the default image group.
  - Customize image names (`dest` field) for this system.

When migrating image specifications and image groups to the new `cray_image_groups.yaml`, ensure that the `arch` and `export_format` fields are present and recipe/image names are consistent, otherwise image building may fail. Also, Cray recommends converting pre-UP07 recipe/image names to use the `{cle_release_lowercase}` variable for ease of use later, but it is not required.

2. (Custom recipes only) Update recipe and package collection names in existing custom recipes.

If this site has custom recipes, such as for the installation of workload manager (WLM) software, or local site repositories, package collections, or RPMs, then clone the custom recipes and update the clones to reference the UP07 recipes and package collections.

**NOTE:** Beginning with the CLE 6.0.UP07 release, CLE repo, package collection, image recipe, and image names now have a "dot" between `6.0` and `upxx` in the CLE version. Update all custom repos, package collections, and recipes that reference CLE repos, package collections, and recipes to reflect this change.

For example, all instances of `6.0up07` in CLE repo/package/recipe/image names have changed to `6.0.up07`. This new format tracks the official release format as provided in the CLE release RPM.

Note that WLM recipes may reference Lustre 2.5. If so, change all instances of 'lus25' to 'lus27' because the current release uses Lustre 2.7 instead of Lustre 2.5.

To update a custom recipe, use the following steps—do not edit the recipe JSON file directly.

- a. View the contents of the custom recipe.

This example command uses a custom recipe named *my\_old\_recipe*. Substitute the correct recipe name for this site/system.

```
smw# recipe show my_old_recipe
```

- b. Create a new custom recipe.

This example command creates a new custom recipe named *my\_up07\_recipe*. Substitute the correct recipe name for this site/system.

```
smw# recipe create --clone my_old_recipe my_up07_recipe
```

- c. Review and set the distribution value for the custom recipe.

The CLE 6.0.UP06 release introduced recipe and package collection schema changes enabling recipes and package collections to be tagged for a particular Linux distribution. When validating a recipe or when building an image, the distribution value is used to ensure that all the subrecipes and package collections referenced are consistent with regard to distribution. Use the following commands to review and set the distribution value appropriately.

```
smw# recipe show --fields dist my_up07_recipe
smw# recipe update -t DIST my_up07_recipe
```

The first time custom recipes or custom packages are edited after an update from a pre-UP06 release, all local recipes or all local package collections are updated to the new schema automatically. The old schema of the old recipes and package collections is not preserved, but old recipes and packages should be available through pre-UP06 snapshots.

- d. Remove earlier-release recipes, package collections, and repositories contained by the new custom recipe.

These example commands remove an older-release subrecipe (*old\_subrecipe*), package collection (*old\_packagecollection*), and repo (*old\_repo*) from the newly created custom recipe *my\_up07\_recipe*. Substitute the correct recipe, subrecipe, package collection, and repo names for this site/system.

```
smw# recipe update --remove-recipe old_subrecipe my_up07_recipe
smw# recipe update --remove-coll old_packagecollection my_up07_recipe
smw# recipe update --remove-repo old_repo my_up07_recipe
```

- e. Determine the names of recipes, package collections, and repositories available in the UP07 release.

```
smw# recipe list
smw# pkgcoll list
smw# repo list
```

- f. Add the names of UP07 recipes, package collections, and repositories to the new custom recipe.

Replace the recipes, package collections, and repositories that were removed from the custom recipe with their counterparts in the new release, found in the previous substep.

These example commands add a current-release subrecipe (*new\_subrecipe*), package collection (*new\_packagecollection*), and repo (*new\_repo*) to the newly created custom recipe *my\_up07\_recipe*. Substitute the correct recipe, subrecipe, package collection, and repo names for this site/system.

```
smw# recipe update --add-recipe new_subrecipe my_up07_recipe
smw# recipe update --add-coll new_packagecollection my_up07_recipe
smw# recipe update --add-repo new_repo my_up07_recipe
```

- g. Ensure that any site custom recipes are in the default image group or a site-specific stanza in `/var/opt/cray/imps/config/sets/global/config/cray_image_groups.yaml` so that they will get built when `imgbuilder` is run.

For more information and tips about creating a new recipe, see [Install Third-Party Software with a Custom Image Recipe](#) on page 417.

### 5.3.5 Build Boot Images for a Software Update

#### Prerequisites

- The software updates have been installed with image building disabled.
- The SMW has rebooted to the release snapshot.

#### About this task

Because image building was disabled during installation of the software updates, images must be built manually by calling `imgbuilder`. This procedure backs up the NIMS map and builds the new boot images.

#### Procedure

1. Back up the active NIMS map.

- a. Determine which NIMS map is active.

```
smw# cmap list | grep True
p0                p0                True
```

- b. Clone the active NIMS map.

```
smw# cmap create --clone p0 p0_backup
```

**Trouble?** The `cmap` command will first verify that the CLE config set associated with the NIMS map exists. If it does not exist, the command will fail with an error message to that effect.

- If the config set is expected to be missing (for example, during an installation when the CLE config set has not yet been created), then repeat the `cmap create` command with the `--no-verify` flag.
- If the config set is NOT expected to be missing, then create/locate the missing config set, set it as the default config set for the NIMS map, and repeat the `cmap create` command.

2. Build images and map them to NIMS groups.

Use one of the following commands. Note that the `imgbuilder` command also creates a new NIMS map if an existing one is not specified.

**TIP:** When there are existing image roots, as with a software update or a preinstallation, `imgbuilder` will prompt for confirmation to "Build anyway?" because building new images will replace existing ones. If the answer will be "y" for all images, the command can be run with the `--force` option to make admin interaction during image building unnecessary.

- To build all images in the `default` image group in parallel (this is typically quicker):

```
smw# imgbuilder --map --processes=NUM
```

Replace `NUM` with the number of images to be built concurrently (capped at the number of CPUs as reported by `lscpu`).

- To build all images in the `default` image group serially:

```
smw# imgbuilder --map
```

For more information about building images in parallel, see [About Parallel Image Creation](#) on page 40.

**Trouble?** If any recipe validation errors are encountered when calling `imgbuilder`, do one of the following:

- Option 1: Identify the recipes that are failing and add any repos needed to provide the packages being installed.
- Option 2: Run `imgbuilder` with the `--skip-validation` option.

**Trouble?** If `imgbuilder` complains about a NIMS group not existing in the current NIMS map, check `cray_image_groups.yaml` for any `nims_group` entries that are not used in this NIMS map. Remove any that are not used.

3. (Optional) Re-enable image building and enable parallel image building in `install.cle.conf`.

```
smw# vi /var/adm/cray/install.cle.conf
```

- a. Change `build_images: no` to `build_images: yes` to re-enable image building during installation.
- b. If parallel image building during installation is desired, set the value of `build_images_processes` to a number greater than 1 and less than the number of CPUs on the SMW.

For more information about building images in parallel, see [About Parallel Image Creation](#) on page 40.

**Save Time.** No need to wait until image creation is complete. Make the necessary configuration changes while images are being built. Proceed to [Configure the Software Update](#) on page 330.

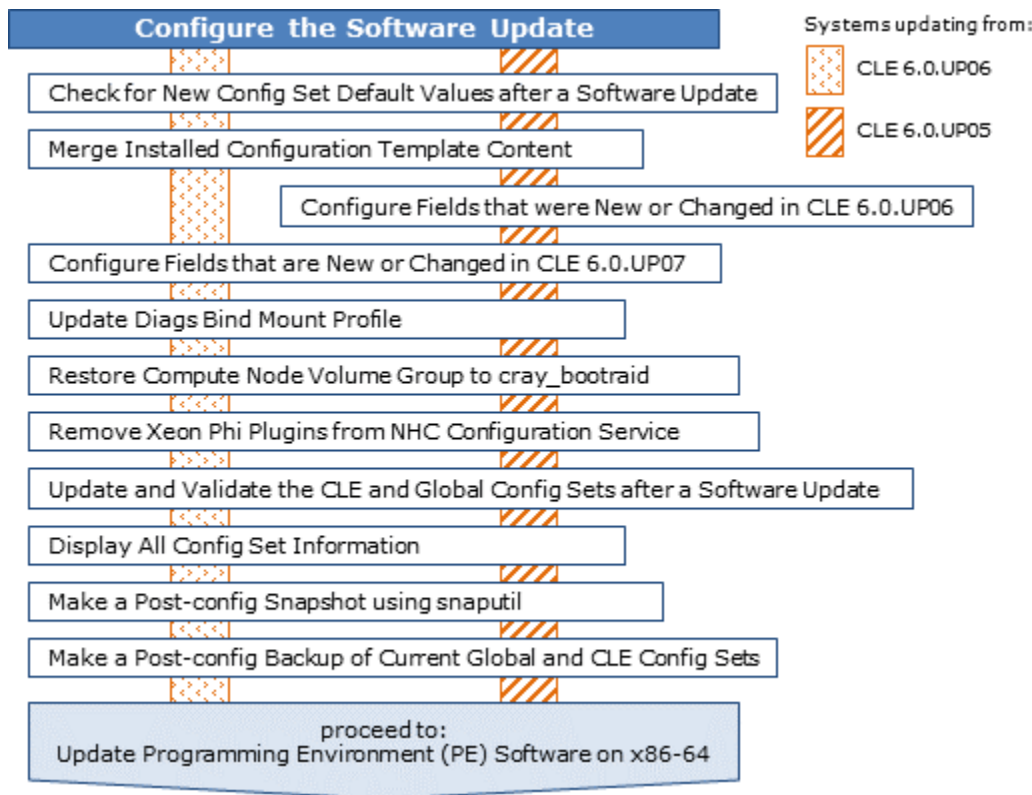
## 5.4 Configure the Software Update

Use the following procedures to make configuration changes to the config sets on the SMW, which has been rebooted to the new release snapshot.

Note that systems updating from CLE 6.0.UP05 require an extra procedure: before configuring items that are new or changed in UP07, it is necessary to first configure items that were new or changed in UP06.



Figure 38. Visual Guide to Configuring the Software Update



### 5.4.1 Check for New Config Set Default Values after a Software Update

#### Prerequisites

- CLE and global config sets from the old CLE 6.0 release have been backed up.
- New software (SMW, CLE, and SLE updates) has been installed into a snapshot and the SMW has been booted to that snapshot.

#### About this task

When invoked with `cfgset update`, the configurator will update the values of unconfigured settings in the config set if there are new default values or new pre-populated data values for those settings in the configurator templates installed on the SMW during a software update. Only settings that are marked as 'unconfigured' in the config set and have new values in the configurator templates will be updated. The `cfgset update` command prints an 'INFO' message for each unconfigured setting that it updates, and it enters in the changelog the name of each updated setting and its old and new value.

This procedure checks for values that will be updated by the configurator BEFORE the `cfgset update` command is run on the current config set. By creating a clone of the config set and updating it first, system administrators are able to preview the changes that will occur and take action if an old value should be kept. The cloned and updated config set (the "UP07 preview" config set) is used only to preview changes that the configurator will make. It is not used for any other purpose.

**IMPORTANT:** Perform this procedure for all CLE config sets that are in use and for the global config set. The examples show the commands for a CLE config set `p0`. Substitute the name of the config set being previewed.

## Procedure

1. Clone the config set to be previewed.

```
smw# cfgset create --clone p0 p0.up07-preview
```

2. Update the cloned config set with `--no-scripts` and `--mode prepare`.

Note that the configurator will produce WARNING messages because no pre- or post-configuration scripts have been run, and the config set will be marked as invalid. Ignore those messages. Instead, pay attention to the INFO messages about updating unconfigured data.

```
smw# cfgset update --no-scripts --mode prepare p0.up07-preview
INFO - Checking directory access
INFO - Checking configuration services
INFO - Checking management node templates prior to merging
INFO - Merging services and validating schema
INFO - Updated unconfigured data for
'cray_local_users.settings.users.data.root.passwordless_ssh' with new value
from template. See changelog for details.
INFO - Updated unconfigured data for
'cray_simple_shares.settings.NFS.data./var/opt/cray/imps.fs_mount_opt' with new
value from template. See changelog for details.
INFO - Updated unconfigured data for
'cray_drc.settings.server.data.database_filename' with new value from template.
See changelog for details.
...
INFO - Changelog will be written to
/var/opt/cray/imps/config/sets/p0.update-preview/changelog/
changelog_2017-05-02T15:56:47.yaml
...
```

3. View the changelog to see any fields that were updated and their changed values.

When using this example command, substitute the path to the changelog for this configurator session, which is found in an INFO statement near the end of the session output. For this example, the

`path_to_changelog` would

be

`/var/opt/cray/imps/config/sets/p0.up07-preview/changelog/changelog_2017-05-02T15:56:47.yaml`

.

```
smw# cat path_to_changelog
```

```
cray_local_users.settings.users.data.root.passwordless_ssh:
  previous: false
  current: true
cray_simple_shares.settings.NFS.data./var/opt/cray/imps.fs_mount_opt:
  previous: ''
  current: ro
cray_drc.settings.server.data.database_filename:
  previous: drc.db
  current: deprecated
...
```

#### 4. Keep the old value of an unconfigured field.

If the preview shows that the configurator will change a value that this site would prefer to keep, use the `cfgset modify` command to set the correct value in the current config set (not the preview config set). Use the full name of the setting, as found in the changelog.

For example, suppose this site wants

`cray_local_users.settings.users.data.root.passwordless_ssh` to have the value `false`. The administrator did not configure it previously because the default value was the desired value. Now that the default has changed, that setting must be configured and set to `false` so that the new default value will not be used.

```
smw# cfgset modify --set false \  
cray_local_users.settings.users.data.root.passwordless_ssh p0
```

Enter `cfgset modify -h` for information about this command. Note that using this method of changing the value of a setting will cause the config set to be marked as invalid because no pre- and post-configuration scripts are run. This is not a problem, because the config set will be updated and validated later in the software update process.

Repeat this step for each setting that needs to keep an old value.

Repeat this procedure for the global config set and for all CLE config sets that are in use on this system.

## 5.4.2 Merge Installed Configuration Template Content

### Prerequisites

- CLE and global config sets from the old CLE 6.0 release have been backed up.
- New software (SMW, CLE, and SLE updates) has been installed into a snapshot and the SMW has been booted to that snapshot.
- The global and all CLE config sets have been previewed and modified where needed

### About this task

It is necessary to merge content from the newly installed configuration templates into the existing config sets in preparation for modifying them in the procedures that follow.

### Procedure

#### 1. Merge installed template content with the CLE config sets.

Proceed with this step only after the global and all CLE config sets have been previewed and modified where needed.

```
smw# cfgset update --no-scripts --mode prepare p0
```

Note that the configurator will produce WARNING messages because no pre- or post-configuration scripts have been run, and the config set will be marked as invalid. Ignore those messages; the config set will be updated without the `--no-scripts` flag later in the process, which will address those issues.

#### 2. Merge installed template content with the global config set.

Proceed with this step only after the global and all CLE config sets have been previewed and modified where needed.

```
smw# cfgset update --no-scripts --mode prepare global
```

Expect the same WARNING messages and continue to ignore them.

When finished merging new content to the CLE and global config sets, continue as follows:

**UP05** For sites updating from UP05, proceed to:

[Configure Fields that were New or Changed in CLE 6.0.UP06](#) on page 334

**UP06** For sites updating from UP06, proceed to:

[Configure Fields that are New or Changed in CLE 6.0.UP07](#) on page 339

### 5.4.3 Configure Fields that were New or Changed in CLE 6.0.UP06

#### Prerequisites

- This system is being updated from SMW 8.0 / CLE 6.0 UP05 (if unsure, `cat /etc/opt/cray/release/cle-release`).
- CLE and global config sets from the previously installed CLE 6.0 release have been backed up.
- Installed configuration template content has been merged into existing config sets.

#### About this task

If this system is being updated from CLE 6.0.UP05, then use this procedure to configure the following changes introduced in the CLE 6.0.UP06 release:

- `/etc/hosts` file changes  
Hosts data manually added by sites directly to the `/etc/hosts` file is no longer preserved. Now, the `/etc/hosts` file for both internal CLE nodes and eLogin nodes is composed of four distinct components: the auto-generated hosts data (as before), and several new files that can be modified by sites. Sites control the auto-generated data only through config set data. Sites can manually add data to the new files, which are appended and prepended to the auto-generated data when the config set is updated. This procedure migrates hosts data previously added manually by sites.
- New global configuration service, `cray_global_local_users`, with a pre-populated group  
The new `cray_global_local_users` has a pre-populated group (`craylogreaders`) intended for use with log files and directories. Because this is an entirely new config service, that pre-populated group was automatically transferred to the config set when installed config template content was merged in an earlier procedure. However, one field in that group must be configured. This procedure configures that field and provides an opportunity to modify two settings in `cray_logging` to use the new `craylogreaders` group.
- New CLE configuration services and new settings in an existing CLE configuration service to support SMW-managed eLogin
  - `cray_storage`: extended to specify the storage layout for local storage on eLogin nodes.

- `cray_cfgset_exclude`: added to specify which parts of the config set should be excluded when the config set is pushed from the SMW to eLogin nodes.
- `cray_kdump`: added to configure the kernel dump tool on eLogin nodes.

This procedure does NOT include steps for configuring those services/settings. Instead, it disables the two new services for this stage of the process. Sites with eLogin nodes will do preparation and configuration later, at the end of the SMW/CLE software update process, using *XC™ Series SMW-managed eLogin Installation Guide* (S-3020).

- New platform keyword for disabled nodes
  - All platform keywords, such as `platform:compute`, `platform:service-ARM`, and `platform:compute-HW12`, include nodes that have been disabled.
  - A new `platform:disabled` keyword can be used by administrators to identify disabled nodes.
  - (not new but previously undocumented) Groups of nodes can be excluded using a negation operator: `~` (the tilde symbol). All non-negated keywords are resolved first, then negated ones are removed.
- Persistent storage for the `ncmd` daemon is now managed in the `cray_bootraid` configuration service (global config set) rather than in the `cray_persistent_data` config service (CLE config set).
- The `cray_multipath` configuration service in both the CLE and global config sets now supports WWIDs (World Wide Identifiers) that are 33 characters long, which is needed for DAL Lustre configuration.

**IMPORTANT:** Perform the CLE config set steps for all CLE config sets that are in use on this system. The examples show the commands for a CLE config set `p0`. Substitute the name of the config set being modified.

## Procedure

————— MIGRATE ANY SITE DATA MANUALLY ADDED TO `/etc/hosts` —————

1. Migrate site data that was manually added to the `/etc/hosts` file.

If a site manually entered data in `/etc/hosts` prior to CLE 6.0.UP06, migrate those entries to:

- `hosts.head` and/or `hosts.tail` for data that needs to be in the `hosts` file that will be installed on internal CLE nodes
- `hosts_ext.head` and/or `hosts_ext.tail` for data that needs to be in the `hosts` file that will be installed on eLogin nodes

Those files are located in the following directory on the SMW:

```
/var/opt/cray/imps/config/sets/<CONFIG_SET>/files/roles/common/etc/
```

The contents of those files will be prepended (head) or appended (tail) to the auto-generated hosts data in `/etc/hosts` when the config set is updated later in the software update process. For additional information, see [About the /etc/hosts File](#) on page 43.

————— UPDATE `cray_global_local_users` —————

2. Configure the `roles` field of the `craylogreaders` group in `cray_global_local_users`.

Intended for use with ownership of log files and directories, the `craylogreaders` group has more restricted permissions than the `crayadm` group.

```
smw# cfgset modify --add smw cray_global_local_users.settings.groups.data.craylogreaders.roles global
smw# cfgset get cray_global_local_users.settings.groups.data.craylogreaders.roles global
smw
```

### 3. (Optional) Create a new group in `cray_global_local_users`, if needed.

Cray recommends using the pre-populated `craylogreaders` group to specify ownership of log files and directories in `cray_logging`, which is updated in a later step in this procedure. However, if this site wishes to create and use a custom group, use the commands in the following example.

Replace `my_group` in all lines with the name (key) for this new group. Replace `'12345'` with any unused GID value. To see what GIDs are already in use on this SMW, look in `/etc/groups`.

```
smw# cfgset modify --add my_group cray_global_local_users.settings.groups.data global
smw# cfgset modify --set '12345' cray_global_local_users.settings.groups.data.my_group.gid global
smw# cfgset modify --set 'custom group for purpose A' \
cray_global_local_users.settings.groups.data.my_group.description global
smw# cfgset modify --set false cray_global_local_users.settings.groups.data.my_group.deleted global
smw# cfgset modify --add smw cray_global_local_users.settings.groups.data.my_group.roles global
```

## ————— UPDATE LOGGING SETTINGS —————

### 4. Use the current default permissions for log files and directories, which are not backward compatible, or change permissions to maintain backward compatibility.



**WARNING:** The current default permissions for log files and directories are not backward compatible. If used, this system will no longer be able to boot and run CLE 6.0 releases older than CLE 6.0.UP06.

Sites must choose whether to go forward with the current default permissions, which make possible the use of the new `craylogreaders` group for log ownership, or change the permissions to maintain backward compatibility.

- **If backward compatibility is NOT an issue**, use the current default permissions (no action needed), and use one of the following ownership groups for log files and directories:
  - `crayadm`, the default group
  - (recommended) `craylogreaders`
  - custom group defined in `cray_global_local_users`, which is in the global config set
- **If backward compatibility is necessary**, keep `crayadm` as the ownership group for log files and directories, and change the permissions for log files and directories to values compatible with previous releases.

Note that the ownership group specified for log files and directories, whether the default or some other group, will be set as a secondary group for the `crayadm` and `postgres` users.

- a. Determine whether `cray_logging` in the CLE config set (`p0` in the example) inherits from `cray_logging` in the global config set.

```
smw# cfgset get cray_logging.inherit p0
```

- If the result is `true`, then `cray_logging` in the global config set is the only config service that needs to be changed.
- If the result is `false`, then `cray_logging` needs to be changed in the CLE config set as well as the global config set.

b. Update `cray_logging` in the global config set.

- If backward compatibility is NOT needed, use the current default permissions (no action needed), and set group ownership for log files and directories. To use `crayadm` (the default) as the ownership group, do nothing. To use `craylogreaders` or a custom group instead, use the following commands.

```
smw# cfgset modify --set craylogreaders cray_logging.settings.dirs.data.group global
```

```
smw# cfgset get cray_logging.settings.dirs.data.group global
craylogreaders
```

```
smw# cfgset modify --set craylogreaders cray_logging.settings.logs.data.group global
```

```
smw# cfgset get cray_logging.settings.logs.data.group global
craylogreaders
```

Note that the value of `group` in the `dirs` setting must always be the same as the value of `group` in the `logs` setting.

When group ownership changes are applied, they will affect all log directories and all newly created log files going forward, but they are not retroactive. The group ownership of previously created log files is not changed. If this site wishes to change ownership of older log files as well, those changes must be made manually.

- If backward compatibility is needed, keep `crayadm` as the ownership group for log files and directories (no action needed), and change the permissions for log files and directories to values compatible with previous releases. Use the values shown in the example.

```
smw# cfgset modify --set 0775 cray_logging.settings.dirs.data.mode global
```

```
smw# cfgset get cray_logging.settings.dirs.data.mode global
0775
```

```
smw# cfgset modify --set 0644 cray_logging.settings.logs.data.mode global
```

```
smw# cfgset get cray_logging.settings.logs.data.mode global
0644
```

c. (Conditional) Update `cray_logging` in the CLE config set.

If `cray_logging` in the CLE config set does not inherit from `cray_logging` in the global config set, then perform this step. Otherwise, skip it.

- If backward compatibility is NOT needed, use the current default permissions (no action needed), and set group ownership for log files and directories. To use `crayadm` (the default) as the ownership group, do nothing. To use `craylogreaders` or a custom group instead, use the following commands. This example changes the values to `craylogreaders` in CLE config set `p0`.

```
smw# cfgset modify --set craylogreaders cray_logging.settings.dirs.data.group p0
```

```
smw# cfgset get cray_logging.settings.dirs.data.group p0
craylogreaders
```

```
smw# cfgset modify --set craylogreaders cray_logging.settings.logs.data.group p0
```

```
smw# cfgset get cray_logging.settings.logs.data.group p0
craylogreaders
```

Note that the value of `group` in the `dirs` setting must always be the same as the value of `group` in the `logs` setting.

When group ownership changes are applied, they will affect all log directories and all newly created log files going forward, but they are not retroactive. The group ownership of previously created log files is not changed. If this site wishes to change ownership of older log files as well, those changes must be made manually.

- If backward compatibility is needed, keep `crayadm` as the ownership group for log files and directories (no action needed), and change the permissions for log files and directories to values compatible with previous releases. Use the values shown in the example.

```
smw# cfgset modify --set 0775 cray_logging.settings.dirs.data.mode p0
```

```
smw# cfgset get cray_logging.settings.dirs.data.mode p0
0775
```

```
smw# cfgset modify --set 0644 cray_logging.settings.logs.data.mode p0
```

```
smw# cfgset get cray_logging.settings.logs.data.mode p0
0644
```

#### ————— DISABLE NEW ELOGIN CONFIG SERVICES —————

The next two steps disable `cray_cfgset_exclude` and `cray_kdump`. Sites with eLogin nodes will enable and configure them later, at the end of the SMW/CLE software update process, using *XC™ Series SMW-managed eLogin Installation Guide (S-3020)*.

#### 5. Disable `cray_cfgset_exclude`.

The following example disables `cray_cfgset_exclude` and verifies that it has been disabled in CLE config set `p0`.

```
smw# cfgset modify --set false cray_cfgset_exclude.enabled p0
smw# cfgset get cray_cfgset_exclude.enabled p0
false
```

#### 6. Disable `cray_kdump`.

The following example disables `cray_kdump` and verifies that it has been disabled in CLE config set `p0`.

```
smw# cfgset modify --set false cray_kdump.enabled p0
smw# cfgset get cray_kdump.enabled p0
false
```

#### ————— REMOVE `ncmd` MOUNT POINT FROM `cray_persistent_data` —————

#### 7. List the mount points defined in `cray_persistent_data`.

```
smw# cfgset get cray_persistent_data.settings.mounts.data p0
/var/opt/cray/alps
/var/opt/cray/aeld
/var/opt/cray/appterm
/var/opt/cray/ncmd
/var/lib/nfs
```

#### 8. If `/var/opt/cray/ncmd` is in the list, remove it.



This entry is no longer needed, because persistent data for the `ncmd` daemon is now stored in the SDB node volume group on the boot RAID, which is managed in the `cray_bootraid` config service.

```
smw# cfgset modify --remove /var/opt/cray/ncmd cray_persistent_data.settings.mounts.data p0
smw# cfgset get cray_persistent_data.settings.mounts.data p0
/var/opt/cray/alps
/var/opt/cray/aeld
/var/opt/cray/appterm
/var/lib/nfs
```

## 5.4.4 Configure Fields that are New or Changed in CLE 6.0.UP07

### Prerequisites

- This system is being updated from SMW 8.0 / CLE 6.0 UP05 or UP06 (if unsure, `cat /etc/opt/cray/release/cle-release`).
- CLE and global config sets from the previously installed CLE 6.0 release have been backed up.
- Installed configuration template content has been merged into existing config sets.

### About this task

If this system is being updated from CLE 6.0.UP05 or UP06, then use this procedure to configure the following changes introduced in the CLE 6.0.UP07 release:

- A new configuration service, Cray NAT (network address translation) Masquerading (`cray_nat_masq`), defines a gateway node to serve as the default gateway for one or more client nodes, and it provides the option to configure `SuSEfirewall12` to permit network traffic from a client node to a network outside the Cray high-speed network (HSN). A common use case is to allow network access outside the HSN for ARM compute nodes repurposed as login nodes.

Because Cray does not support the addition of hardware as part of a software update, and this new service is intended primarily for use with ARM compute nodes repurposed as login nodes, this service is disabled in this procedure. Instructions for configuring this service later will be included in *XC™ Series System Administration Guide (CLE 6.0.UP07) S-2393*.

- The `atftpd` daemon does not log to `/var/opt/cray/log/smwmessages-<date>` as most daemons do, but logs to `/var/log/atftpd/atftp.log` instead. If this site would prefer `atftpd` to log to `/var/opt/cray/log/smwmessages-<date>`, and has not already changed it, use the step in this procedure to manually change it.
- The platform keyword for disabled nodes, `platform:disabled`, can now be used to create or modify node groups.

**IMPORTANT:** Perform the CLE config set steps for all CLE config sets that are in use on this system. The examples show the commands for a CLE config set `p0`. Substitute the name of the config set being modified.

### Procedure

1. Disable new NAT Masquerading config service.

The following example disables `cray_nat_masq` and verifies that it has been disabled in CLE config set `p0`.

```
smw# cfgset modify --set false cray_nat_masq.enabled p0
smw# cfgset get cray_nat_masq.enabled p0
false
```

2. (Optional) Change the logging location of atftpd to /var/opt/cray/log/smwmessages-<date>.

Cray uses the SUSE atftpd server. The /etc/init.d/atftpd file provided by SUSE is set up to log to /var/log/atftpd/atftp.log, and the logging location is not configurable in the associated /etc/sysconfig/atftpd file. This is different than most SMW daemons, which log to /var/opt/cray/log/smwmessages-<date>.

Perform this step if this site wants atftpd on this system to log to /var/opt/cray/log/smwmessages-<date> and has not already made this change.

- a. Edit the /etc/init.d/atftpd file.

```
smw# sed --in-place=.bak 's/--logfile $ATFTP_LOG_FILE//' /etc/init.d/atftpd
```

- b. Restart the atftpd service.

```
smw# systemctl daemon-reload
smw# systemctl restart atftpd.service
```

3. Use the platform:disabled keyword to create a node group, as needed.

The following example uses the platform:disabled keyword and the negation operator to create a new node group called *enabled\_compute\_nodes*. Note that the example places the negated keyword last in the list of group members; however the ordering of the list does not matter. All non-negated keywords are resolved first, then negated ones are removed.

```
smw# cfgset modify --add enabled_compute_nodes cray_node_groups.settings.groups.data p0
smw# cfgset get cray_node_groups.settings.groups p0

smw# cfgset modify --set "All compute nodes, excluding disabled nodes." \
cray_node_groups.settings.groups.data.enabled_compute_nodes.description p0

smw# cfgset modify --add platform:compute --add ~platform:disabled \
cray_node_groups.settings.groups.data.enabled_compute_nodes.members p0

smw# cfgset get cray_node_groups.settings.groups.data.enabled_compute_nodes.members p0
```

## 5.4.5 Update Diags Bind Mount Profile

### About this task

This procedure updates the diags bind mount profile in the *cray\_image\_binding* config service.

### Procedure

#### ————— UPDATE THE DIAGS BIND MOUNT PROFILE —————

The following steps use the `cfgset modify` command, which does not produce output when successful. An optional `cfgset get` command is provided for each to verify the resulting setting value. Using the `cfgset modify` command causes the config set to be marked as invalid because no pre- and post-configuration scripts are run; therefore the config set must be updated and validated afterwards.

1. Ensure that the `cray_image_binding` configuration service is enabled.

```
smw# cfgset modify --set true cray_image_binding.enabled p0
smw# cfgset get cray_image_binding.enabled p0
true
```

2. Change the value of the `diags` profile `image` field.

Determine the name of the image root used in the `diags` profile in `cray_image_binding`.

```
smw# cfgset get cray_image_binding.settings.profiles.data.diags.image p0
```

The image name in the `diags` profile `image` field MUST match the name of the `diags` image root that will be pushed to the boot node later in the update process. If the `diags` profile `image` field is set to the wrong image, change it.

```
smw# image list | grep diag
```

In the following command, replace `DIAGS_IMAGE` with the actual `diags` image root on this system.

```
smw# cfgset modify --set DIAGS_IMAGE \
cray_image_binding.settings.profiles.data.diags.image p0
smw# cfgset get cray_image_binding.settings.profiles.data.diags.image p0
```

3. Enable the `diags` profile.

```
smw# cfgset modify --set true \
cray_image_binding.settings.profiles.data.diags.enabled p0
smw# cfgset get cray_image_binding.settings.profiles.data.diags.enabled p0
true
```

## 5.4.6 Restore Compute Node Volume Group to `cray_bootraid`

### About this task

**NOTICE:** This procedure is only for systems that have or will have SSD-endowed compute nodes. If that is not the case, skip this procedure.

During a fresh install of SMW/CLE software, sites without SSD-endowed compute nodes are instructed to either remove or disable the compute node volume group that Cray provides as pre-populated data in the `cray_bootraid` configuration service. However, if a site decides to add SSD-endowed compute nodes later, that volume group (VG) will be needed. This procedure restores (if removed) and configures the compute node volume group.

### Procedure

1. Determine whether the compute node VG (`compute_node_local`) is missing or unconfigured/disabled.

This command retrieves the value of the compute node VG owner.

```
smw# cfgset get cray_bootraid.settings.storage_sets.data.\
cledefault.volume_groups.compute_node_local.owner global
```

The configurator will return one of the following results:

- `compute`: The compute node VG is present and this field is set to the correct value.

- blank line or `null`: The compute node VG is present but this field is set to the empty string or `null`, which indicates that this VG is unconfigured or was disabled.
- error: The following error message indicates that the compute node VG is missing.

```
Error: could not get 'global':
path=cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.owner
Path entry 'compute_node_local' not found in schema.
```

2. If the compute node VG is present and the owner is `compute`, examine the devices setting.

```
smw# cfgset get cray_bootraid.settings.storage_sets.data.\
cledefault.volume_groups.compute_node_local.devices global
```

- If the output of this command lists `select: nvme0n1` or a system-specific list of devices, then the compute node VG is configured and nothing more needs to be done.  
Skip the rest of this procedure.
- If there is no output, then the list of devices is empty and needs to be configured.  
Continue to the next step.

———— CONTINUE IF COMPUTE NODE VG IS MISSING OR UNCONFIGURED ————

3. Prepare global configuration worksheets for editing.

- a. Create fresh global configuration worksheets.

```
smw# cfgset update --no-scripts --mode prepare global
```

Note that the configurator will produce WARNING messages because no pre- or post-configuration scripts have been run, and the config set will be marked as invalid. Ignore those messages; the config set will be updated without the `--no-scripts` flag later in the process, which will address those issues.

- b. Copy the worksheets to a work area for editing.

```
smw# mkdir -p /var/adm/cray/release
smw# cp -a /var/opt/cray/imps/config/sets/global/worksheets \
/var/adm/cray/release/global_worksheet_workarea
```

4. Edit the `cray_bootraid` worksheet.

```
smw# cd /var/adm/cray/release/global_worksheet_workarea
smw# vi cray_bootraid_worksheet.yaml
```

5. In the `cray_bootraid` worksheet, locate the pre-populated data below this line.

```
# ** 'storage_sets' DATA **
```

The first line of data defines the name of the CLE default storage set: `cledefault`.

```
cray_bootraid.settings.storage_sets.data.name.cledefault: null
```

Below that are lines that define the volume groups within `cledefault`: a boot node volume group with stanzas for three volumes (`home`, `imps`, and `nvolatile`), and an SDB node volume group with stanzas for two volumes (`db` and `alps`). If present, the compute node volume group would be below the last SDB node VG volume stanza.

## 6. Add or configure the compute node VG.

If not present, add the compute node volume group data between the last line of the SDB node volume group alps volume and the first line of the SMW default storage set (smwdefault).

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.sdb_node_vg.volumes.alps.mount_options: ''

*****
***** PLACE COMPUTE NODE VOLUME GROUP DATA HERE *****
*****

cray_bootraid.settings.storage_sets.data.name.smwdefault: null
```

Add the following lines, exactly as they are shown in the example. (Note that if using the PDF version of this guide, spurious line breaks may occur because of PDF page width limitations. Remove all such line breaks when pasting this into the worksheet so that the YAML formatting will be correct.)

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.sdb_node_vg.volumes.alps.mount_options: ''

cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.name.compute_node_local: null
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.owner: compute
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.devices:
- 'select: nvme0n1'

cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.name.temporary: null
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.temporary.description:
  Temporary, but managed files.
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.temporary.type: lvm
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.temporary.fs_type:
  ext3
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.temporary.fs_size: 40%
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.temporary.fs_mount_point:
  /temporary
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.temporary.snapshot:
  false
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.temporary.fs_remove_data:
  true
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.temporary.fs_cncu_enable:
  true
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.temporary.mount_options:
  ''

cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.name.swap: null
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.swap.description: ''
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.swap.type: lvm
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.swap.fs_type: swap
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.swap.fs_size: '30'
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.swap.fs_mount_point:
  swap
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.swap.snapshot: false
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.swap.fs_remove_data:
  false
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.swap.fs_cncu_enable:
  false
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.swap.mount_options: ''

cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.name.unmanaged: null
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.unmanaged.description:
  ''
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.unmanaged.type: lvm
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.unmanaged.fs_type:
  ext3
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.unmanaged.fs_size: ALL
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.unmanaged.fs_mount_point:
  /unmanaged
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.unmanaged.snapshot:
  false
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.unmanaged.fs_remove_data:
  false
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.unmanaged.fs_cncu_enable:
  false
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.compute_node_local.volumes.unmanaged.mount_options:
  ''

cray_bootraid.settings.storage_sets.data.name.smwdefault: null
```

If the compute node VG is already present, compare the compute node VG values to the ones shown in the previous example and modify them to match.

7. Upload the modified `cray_bootstrap` worksheet to the global config set.

```
smw# cfgset update global \  
-w /var/adm/cray/release/global_worksheet_workarea/cray_bootstrap_worksheet.yaml
```

## 5.4.7 Remove Xeon Phi Plugins from NHC Configuration Service

### About this task

This procedure checks for several Node Health Checker (NHC) plugins that are no longer supported. If they are present in the CLE config set, this procedure removes them. For a full explanation, see "Remove Xeon Phi Plugin Values From an Existing Config Set to Prevent NHC from Failing" in *XC™ Series Node Health Checker (NHC) Plugin Test User Guide (S-0023)*.

### Procedure

1. Check for Xeon Phi plugin entries on the existing config set using `cfgset get`.

```
smw# cfgset get cray_node_health.settings.plugins.data p0  
Default Alps  
Plugin DVS Requests  
Plugin Datawarp  
Default Application  
Default Reservation  
Plugin Nvidia  
Plugin ugni  
Plugin Sigcont  
Plugin Hugepage Check  
Plugin CCM Test  
Xeon Phi Plugin App Test  
Xeon Phi Plugin Reservation  
Xeon Phi Plugin Memory  
Xeon Phi Plugin Alps
```

2. If there are any Xeon Phi plugin entries, remove them from the config set using `cfgset modify`.

```
smw# cfgset modify --remove 'Xeon Phi Plugin App Test' \  
cray_node_health.settings.plugins.data p0  
  
smw# cfgset modify --remove 'Xeon Phi Plugin Reservation' \  
cray_node_health.settings.plugins.data p0  
  
smw# cfgset modify --remove 'Xeon Phi Plugin Memory' \  
cray_node_health.settings.plugins.data p0  
  
smw# cfgset modify --remove 'Xeon Phi Plugin Alps' \  
cray_node_health.settings.plugins.data p0
```

3. Verify that all Xeon Phi plugins have been removed from the config set:

```
smw# cfgset get cray_node_health.settings.plugins.data p0  
Default Alps  
Plugin DVS Requests  
Plugin Datawarp  
Default Application
```

```

Default Reservation
Plugin Nvidia
Plugin ugni
Plugin Sigcont
Plugin Hugepage Check
Plugin CCM Test

```

## 5.4.8 Update and Validate the CLE and Global Config Sets after a Software Update

### Prerequisites

- Modifications to the CLE and global config sets using `cfgset modify`, if any, are complete.
- Modified configuration worksheets, if any, have been uploaded to the config sets.

### About this task

This procedure uses `cfgset update` in auto mode to ensure that all configuration scripts are run and to prompt for unconfigured settings. Then `cfgset validate` is run to check for correctness and consistency.

Configurator navigation tips:

- For context-sensitive command help, enter `?`.
- To add a single value, enter the data and press **Enter**.
- To add a list, enter `+`, enter each list item on its own line, press **Ctrl-d** when done entering list items, and then press **Enter** to set the list entries.
- To skip a setting, press the `>` key. Note that skipping an unconfigured setting leaves it unconfigured, which means the configurator will assign it the default value and will prompt for it again if invoked with the same command.
- To correct an error in a previous setting, press the `<` key to go back to the previous setting, correct it, then continue forward. Use `<` to back up several settings, if needed.

### Procedure

#### 1. Update the CLE config set.

The configurator will prompt only for unconfigured settings that are level `required` or level `basic`. The example commands use a CLE config set named `p0`. Substitute the correct config set name for this system.

```
smw# cfgset update p0
```

Repeat this step for all CLE config sets in use on this system.

#### 2. Validate the CLE config set.

```
smw# cfgset validate p0
```

If validation cross-checks find any errors or inconsistencies in the configuration, use the information in the output to correct those now (using the `cfgset modify` command), and then return to the previous step to update the config set again.

Repeat this step for all CLE config sets in use on this system.

**3. Update the global config set.**

The configurator will prompt only for unconfigured settings that are level `required` or level `basic`.

```
smw# cfgset update global
```

**4. Validate the global config set.**

```
smw# cfgset validate global
```

If validation cross-checks find any errors or inconsistencies in the configuration, use the information in the output (error and/or warning messages) to correct those now using the `cfgset modify` command, and then return to the previous step to update the config set again.

Because this validation step occurs after making changes to the global config set, but before Ansible plays are run to apply those changes, changes such as the addition of a new group (e.g., `craylogreaders`) may be flagged as not yet existing on the SMW. This will result in a warning message if the "missing" group or other setting has been defined correctly in the config set. This will result in an error message if it has not been defined correctly.

**5. Apply configuration changes on the SMW.**

When these changes are applied, any changes to group ownership of log files and directories will affect all log directories and all newly created log files, but the group ownership of previously created log files is not changed. If this site wishes to change ownership of older log files as well, those changes must be made manually.

**a. Run `cray-ansible` on the SMW.**

```
smw# /etc/init.d/cray-ansible start
```

**b. Verify that log group ownership changes, if any, have been applied.**

```
smw# cd /var/opt/cray/log
smw# find . -type f -user root -group crayadm \
-exec chown root:craylogreaders "{}" ";"
smw# find . -type d -user root -group crayadm \
-exec chown root:craylogreaders "{}" ";"
```

**c. Restart `rsyslog`.**

```
smw# systemctl restart rsyslog
```

**d. Restart PostgreSQL on the SMW.**

```
smw# systemctl restart postgresql
```

**e. Verify that `nimsd` is working.**

```
smw# cnode list
```



## 5.4.9 Display All Config Set Information

### About this task

This procedure is not required, but it may aid in troubleshooting. It displays all of the configuration settings and writes them to a file for the CLE settings, a file for the global settings, and the typescript file started at the beginning of this session.

### Procedure

1. Display the CLE config set (p0 in this example).

Repeat this step for each CLE config set that will be used to boot the system.

```
smw# cfgset search -l advanced p0 | tee \
/var/adm/cray/release/p0.${TODAY}.update.advanced.conf
```

2. Display the global config set.

```
smw# cfgset search -l advanced global | tee \
/var/adm/cray/release/global.${TODAY}.update.advanced.conf
```

## 5.4.10 Make a Post-config Snapshot using snaputil

### About this task

This procedure uses `snaputil` to make an archival snapshot of the system after configuring SMW/CLE software and before booting the CLE system.

**Best Practice.** Make a snapshot and back up config sets at the same time to keep them in sync. Name the snapshot and config set backup using the same suffix and date/time stamp to help administrators identify which snapshot and config set backup pairs belong together.

For more information, see [About Snapshots and Config Set Backups](#) on page 21.

### Procedure

1. List the available snapshots on the system.

```
smw# snaputil list
```

2. Set the `SNAPSHOT` environment variable using the currently booted snapshot name.

```
smw# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')
smw# echo $SNAPSHOT
```

Setting a variable for the snapshot name enables better command substitution in later commands dealing with snapshots.

This is especially important for SMW HA systems because it makes it easier to use the exact same snapshot name for both SMWs. Using different snapshots results in HSS database (MySQL) inconsistencies, which causes problems at failover.

3. Create a new snapshot.

```
smw# snaputil create ${SNAPSHOT}.postconfig-${TODAY}
```

### 5.4.11 Make a Post-config Backup of Current Global and CLE Config Sets

#### About this task

This procedure uses the `cfgset` command to create a post-install backup of the global and CLE config sets after configuring SMW/CLE software and before booting the CLE system.

#### Procedure

1. Back up the current global config set.

```
smw# cfgset create --clone global global-postconfig-${TODAY}
```

2. Back up the current CLE config set.

This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-postconfig-${TODAY}
```

## 5.5 Update Cray Programming Environment (PE) Software on x86-64

### Prerequisites

Cray Programming Environment (PE) software should be updated with the PE Installer.

#### About this task

The same PE image can be used for several of the monthly releases of PE software, but a fresh image must be created and used with each new CLE release. This procedure creates a fresh PE image for this CLE software release, and then updates PE software content and makes it available on x86-64 compute and login nodes.

#### Procedure

1. Set an environment variable for the PE image name.

The old name for the PE image set in the Cray Image Binding Service of the CLE config set will need to be changed to this new name.

```
smw# export PE_X86_IMAGE=pe_cle_6.0.up07_sles_12sp3_x86-64
smw# echo $PE_X86_IMAGE
```

2. Create fresh PE image root on the SMW for this software release.

- a. Get the name of the PE image recipe on the system.

```
smw# recipe list | grep ^pe
pe_base_cle_6.0.up07_sles_12sp3
```

- b. Create the PE image (\$PE\_X86\_IMAGE) using that recipe name.

```
smw# image create -r pe_base_cle_6.0.up07_sles_12sp3 $PE_X86_IMAGE
```

### 3. Configure and enable the PE bind mount profile.

- a. Ensure that the `cray_image_binding` configuration service is enabled.

```
smw# cfgset modify --set true cray_image_binding.enabled p0
smw# cfgset get cray_image_binding.enabled p0
true
```

- b. Change the value of the PE profile image field to match the name of the image that was used to set \$PE\_X86\_IMAGE.

```
smw# cfgset modify --set pe_cle_6.0.up07_sles_12sp3_x86-64 \
cray_image_binding.settings.profiles.data.PE.image p0

smw# cfgset get cray_image_binding.settings.profiles.data.PE.image p0
pe_cle_6.0.up07_sles_12sp3_x86-64
```

- c. Ensure that the PE profile is enabled.

```
smw# cfgset modify --set true \
cray_image_binding.settings.profiles.data.PE.enabled p0

smw# cfgset get cray_image_binding.settings.profiles.data.PE.enabled p0
true
```

### 4. Update the `IMAGE_DIRECTORIES` field in the installer configuration file (/var/adm/cray/release/pe/install-cdt.yaml).

If PE is to be installed in the new PE image only, update `IMAGE_DIRECTORIES` as follows:

```
IMAGE_DIRECTORIES :
- /var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64
```

If PE is to be installed in both the old and new images, update `IMAGE_DIRECTORIES` as follows:

```
IMAGE_DIRECTORIES :
- /var/opt/cray/imps/image_roots/pe_compute_cle_6.0up06_sles_12sp3
- /var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64
```

### 5. Copy the most recent PE ISOs to the SMW and mount the ISOs.

Starting with the CDT 16.06 release, the full CDT release is now provided on multiple DVDs rather than on a single one. One DVD will be provided for each of these files:

- CDT-base-<version>.iso
- CDT-PrgEnv-cray-<version>.iso (not provided for CDT-NCC)
- CDT-PrgEnv-intel-<version>.iso
- CDT-PrgEnv-pgi-<version>.iso

- a. Remove the following directory in case it exists from a previous installation, where `ISO_MOUNT_DIR` is the variable in the `.yaml` configuration file that points to the directory where the contents of the ISO are

being copied. In the following instructions, `$ISO_MOUNT_DIR` refers to the directory specified in the `ISO_MOUNT_DIR` field in `install-product.yaml`.

```
# rm -f -r $ISO_MOUNT_DIR
```

- b. Perform the following steps for each ISO file downloaded to combine the contents into a single installation directory.

The possible ISO files and their respective required vs optional status are:

- `product-base-version.iso` (REQUIRED)
- `product-PrgEnv-cray-version.iso` (OPTIONAL but not provided for CDT-NCC)
- `product-PrgEnv-intel-version.iso` (OPTIONAL)
- `product-PrgEnv-pgi-version.iso` (OPTIONAL)

If `install-product.yaml` sets `INSTALL_CCE_LIBRARIES : YES` then `product-PrgEnv-cray-version.iso` should be mounted and rsynced.

(NOTE: Above `.iso` is not provided in CDT-NCC packages)

If `install-product.yaml` sets `INSTALL_INTEL_LIBRARIES : YES` then `product-PrgEnv-intel-version.iso` should be mounted and rsynced.

If `install-product.yaml` sets `INSTALL_PGI_LIBRARIES : YES` then `product-PrgEnv-pgi-version.iso` should be mounted and rsynced.

1. Mount the base ISO listed above.

```
# mount -r -o loop product-xxx-version.iso /mnt
```

2. Use the `rsync` command to copy the ISO file content to `ISO_MOUNT_DIR`, the directory where the contents of the ISO are being copied:

```
# rsync -a -v /mnt/ $ISO_MOUNT_DIR/
```

3. Unmount the ISO.

```
# umount /mnt
```

4. Repeat these steps for each optional ISO to be installed. Again, the base ISO is required but the remaining ISO files (PrgEnv-cray (if CDT, not CDT-NCC), PrgEnv-intel, and/or PrgEnv-pgi) are optional.

6. Update the `craype-installer` RPM on the SMW, from the PE ISO.

```
smw# rpm -Uvh \
/var/adm/cray/release/pe/mount_iso/installer/craype-installer-*.x86_64.rpm
```

7. Install PE software from the most recent PE installation media and installer.

- a. Run the PE installer.

```
smw# module load craype-installer
smw# craype-installer.pl --install --install-yaml-path ./install-cdt.yaml
```

When the installation completes, the following output will be shown, summarizing the installed packages.

```
1) atp-1.7.5-0_3605.x86_64 (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
2) cray-cddb-1.0.3-0_3575.x86_64 (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
```

```
3) cray-dwarf-14.2.0-0.x86_64 (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
<snip>
71) perftools-clients-6.2.2-1.x86_64 (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
```

- b. Set the default versions for PE with `set_default` scripts, if the above install succeeds.

```
smw# craype-installer.pl --set-default --install-yaml-path ./install-cdt.yaml
```

- c. Unmount the ISO.

```
smw# umount ./mount_iso
```

- d. Clean up the PE ISO and PE RPMs.

These RPMs are large and use up disk space, so they can be removed.

```
smw# rm *.iso *.rpm *.tar.gz
```

8. Install as many older monthly PE releases to this UP07 PE image root as desired.

For each of the older monthly PE release ISOs, do the following steps to install them to the new `$PE_X86_IMAGE` image root.

- a. Mount the PE ISO.

```
smw# mount -o loop,ro <downloaded PE ISO> /var/adm/cray/release/pe/mount_iso
```

- b. Install PE software from the most recent PE installation media and installer.

Run the PE installer. This will install the older PE software release to the new `$PE_X86_IMAGE` image root.

```
smw# craype-installer.pl --install --install-yaml-path ./install-cdt.yaml
```

When the installation completes, the following output will be shown, summarizing the installed packages.

```
1) atp-1.7.5-0_3605.x86_64 (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
2) cray-cddb-1.0.3-0_3575.x86_64 (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
3) cray-dwarf-14.2.0-0.x86_64 (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
<snip>
71) perftools-clients-6.2.2-1.x86_64 (/var/opt/cray/imps/image_roots/pe_cle_6.0.up07_sles_12sp3_x86-64)
```

- c. If desired, set this older PE release as the default version.

If this version of PE should be the default and not the most recent version of PE software installed earlier, then set it to default with this command.

```
smw# craype-installer.pl --set-default --install-yaml-path ./install-cdt.yaml
```

- d. Unmount the ISO.

```
smw# umount ./mount_iso
```

- e. Clean up the PE ISO and PE RPMs.

These RPMs are large and use up disk space, so they can be removed.

```
smw# rm *.iso *.rpm *.tar.gz
```

9. Update the CLE config set.

An earlier step modified the config set without running pre- and post-configuration scripts, so that config set was marked invalid. This step uses `cfgset update` in prepare mode (no user interaction) to ensure that all configuration scripts are run.

```
smw# cfgset update -m prepare p0
```

#### 10. Validate the config set.

```
smw# cfgset validate p0
```

#### 11. Make a snapshot post PE update.

Cray recommends saving a snapshot of the system immediately after the PE software update is complete. If any `root` users make bad changes after the software update is complete, revert to this snapshot to avoid a redo of the entire software update.

```
smw# snaputil list
```

```
smw# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')  
smw# echo $SNAPSHOT
```

```
smw# snaputil create ${SNAPSHOT}.postpe-${TODAY}
```

#### 12. Back up the CLE and global config sets post PE update.

```
smw# cfgset create --clone global global-postpe-${TODAY}
```

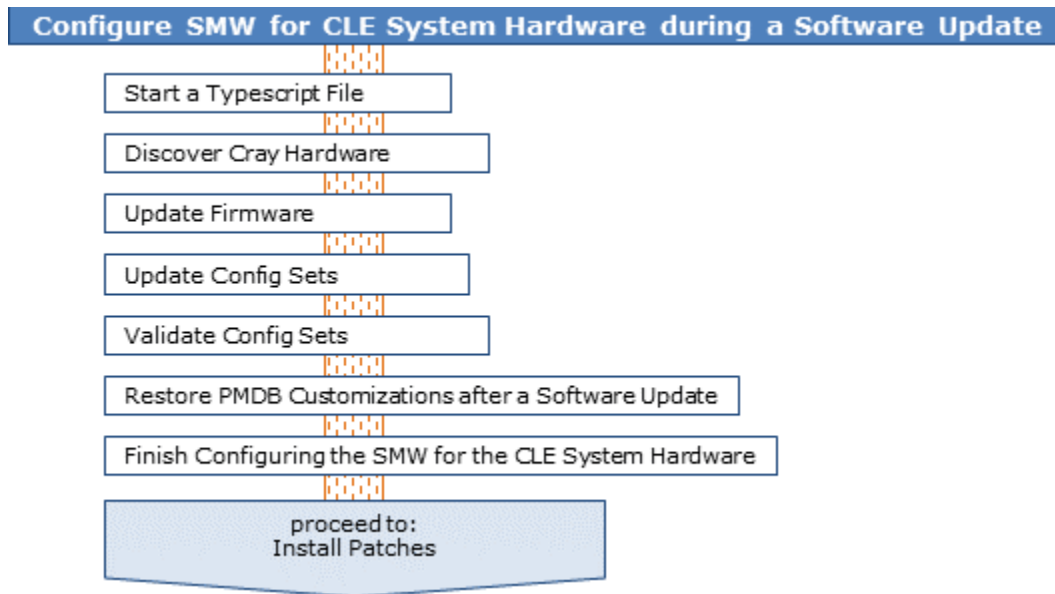
This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-postpe-${TODAY}
```

## 5.6 Configure SMW for CLE System Hardware during a Software Update

In this part of the software update process, use these procedures to discover hardware, update firmware, update and validate config sets, and check the status of all SMW components.

Figure 39. Visual Guide to Configuring the SMW for CLE Hardware during a Software Update



## 5.6.1 Start a Typescript File

### About this task

Sites can make as few or as many typescripts as they deem useful. Cray recommends starting a typescript file at these milestones:

- just before installing a new software release
- just before configuring the newly installed software

### Procedure

1. Log in as root to the SMW.
2. (First time only) Create a release directory for the typescript file.

```
smw# mkdir -p /var/adm/cray/release
```

3. Change to the release directory.

```
smw# cd /var/adm/cray/release
```

4. Set a variable equal to today's date.

```
smw# export TODAY=`date +%Y%m%d`
smw# echo ${TODAY}
```

5. Start a typescript file.

```
smw# script -af ${TODAY}.suffix
```

For *suffix*, substitute a unique string to distinguish among typescript files, such as `install.1` or `update.2`.

6. Change prompt to include a timestamp.

```
smw# PS1="\[\e[1;31m\]\u@\h:\w \t # \[\e[0m\]\[\e[00m\]"
```

## 5.6.2 Discover Cray Hardware

### About this task

**About Hardware Discovery.** This procedure uses `xtdiscover` to ensure that any changes made to the HSS database schema for new features are captured. To display the configuration, use the `xtcli` command after running `xtdiscover`. For more detailed information, see the `xtdiscover(8)` man page.

**About STONITH.** This procedure prepares STONITH, a Linux service that automatically powers down a node that has failed or is suspected of failure. If either boot node failover or SDB node failover will be used, then STONITH needs to be set on the primary blade and backup blade.

**IMPORTANT:** The primary boot node should not be on the same blade as the backup boot node or an SDB node. Likewise, the primary SDB node should not be on the same blade as the backup SDB node or a boot node. Four different blades should be used if there are two boot nodes and two SDB nodes.

**Trouble?** If a step in this procedure fails because of a hardware issue, such as a cabinet failing to power up, resolve that issue and then go back to the last successful step in the procedure and continue from there. Do not skip steps or continue out of order.

### Procedure

1. Power down the system.

```
smw# xtcli power down s0
Turning off power to cabinet and waiting for confirmation...
```

Component	Flags:	Result
-----	-----	-----
c0-0	noflags :	Success
c0-0c0s0	noflags :	Success
c0-0c0s1	noflags :	Success
c0-0c0s2	noflags :	Success
c0-0c0s3	noflags :	Success

2. Reboot the cabinet controllers (CC), then verify that all CCs are up.

- a. Reboot the cabinet controllers.

```
smw# xtccreboot -c all
xtccreboot: reboot sent to specified CCs
smw# sleep 180
```

- b. Are all cabinet controllers up now? Repeat this command until all of the cabinet controllers report in.

```
smw# xtalive -a llsysd -l 11 s0
The expected responses were received.
```



### 3. Power up the system, then verify the blades are powered on.

#### a. Power up the system.

```
smw# xtcli power up s0
Turning on power to cabinet and waiting for confirmation...
```

Component	Flags:	Result
-----	-----	-----
c0-0	noflags :	Success
c0-0c0s0	noflags :	Success
c0-0c0s1	noflags :	Success
c0-0c0s2	noflags :	Success
c0-0c0s3	noflags :	Success

#### b. Verify the blades are powered on and the necessary daemons are responding.

```
smw# sleep 60
smw# xtalive
```

Note that at this point the `xtcli status` output shows that all nodes are "off" because they have not yet been bounced.

### DISCOVER CRAY SYSTEM HARDWARE

### 4. Run the `xtdiscover` command.

`xtdiscover` may pause with instructions to bounce the system.

```
smw# xtdiscover
***** xtdiscover started *****
...
```

In a separate window, please bounce the system now to continue discovery.

### 5. If `xtdiscover` pauses with instructions to bounce the system, open a separate window and, as `crayadm`, run the command to bounce the system.

```
crayadm@smw> /opt/cray/hss/default/etc/xtdiscover-bounce-cmd
```

### 6. If it was necessary to bounce the system, then when the `xtbounce` command from the previous step has finished, return to the `xtdiscover` window and enter "c" to continue the hardware discovery.

```
After bounce completes, enter 'c' to complete discovery
or 'q' or 'a' to abort [c]: c
```

### 7. Commit the results of `xtdiscover` to the database.

When asked whether to commit the `xtdiscover` results to the database, enter **y**.

If `xtdiscover` reports that it saved configuration changes in this file, run this command to show what differences were detected:

```
smw# cat /opt/cray/hss/default/etc/xtdiscover-config-changes.diff
```

(optional) PREPARE STONITH FOR BOOT NODE AND SDB NODE FAILOVER

8. For sites using boot node failover, set STONITH for the blade containing the primary boot node and the blade containing the backup boot node.

Skip this step if there will be no boot node failover for this system.

In the example, the primary boot node is c0-0c0s0n1, so its blade is c0-0c0s0, and the backup boot node is c0-2c0s4n1, so its blade is c0-2c0s4.

```
smw# xtdaemonconfig c0-0c0s0 stonith=true
smw# xtdaemonconfig c0-2c0s4 stonith=true
```

9. For sites using SDB failover, set STONITH for the blade containing the primary SDB node and the blade containing the backup SDB node.

Skip this step if there will be no SDB node failover for this system.

In the example, the primary SDB node is c0-0c0s1n1, so its blade is c0-0c0s1, and the backup SDB node is c0-4c0s3n1, so its blade is c0-4c0s3.

```
smw# xtdaemonconfig c0-0c0s1 stonith=true
smw# xtdaemonconfig c0-4c0s3 stonith=true
```

#### DISCOVER HSN ROUTING CONFIGURATION

10. Discover the routing configuration of the high-speed network (HSN).

After `xtdiscover` finishes, run the `rtr` command as `crayadm` to determine the exact configuration of the HSN.

- a. Switch to `crayadm`.

```
smw# su - crayadm
crayadm@smw> PS1="\u@\h:\w \t> "
```

- b. Run the `rtr --discover` command.

```
crayadm@smw> rtr --discover
```

If the system was not bounced previously, the following message may be displayed. If so, enter **y**.

```
System was not bounced in diagnostic mode, should I re-bounce? Continue (y/n)?
```

If this is a partitioned system, first deactivate the partitions, run `rtr` for the full system, and then activate the partitions again. This is most important when `xtdiscover` has identified a hardware change.

```
crayadm@smw> xtcli part_cfg deactivate p1
crayadm@smw> xtcli part_cfg deactivate p2
crayadm@smw> xtcli part_cfg activate p0

crayadm@smw> rtr --discover

crayadm@smw> xtcli part_cfg deactivate p0
crayadm@smw> xtcli part_cfg activate p1
crayadm@smw> xtcli part_cfg activate p2
```

## 5.6.3 Update Firmware

### Prerequisites

This procedure assumes that Cray hardware discovery has been completed successfully.

### About this task

This procedure first checks whether the firmware of cabinet and blade components (controllers) needs to be updated, then updates the firmware only if there are revision mismatches. For the current list of cabinet and blade controllers, see the `xtzap` man page.

### Procedure

**NOTE:** These commands are performed from the `crayadm` account, as indicated by the command prompts.

#### 1. Check firmware.

Check whether any firmware needs to be updated on the various controllers.

```
crayadm@smw> xtzap -r -v s0
```

If the firmware on any controllers is out of date, the output looks like this, and the firmware needs to be updated (reflashed).

Individual Revision Mismatches:

Type	ID	Expected	Installed
cc_bios	c0-0	0013	0012
bc_bios	c0-0c0s0	0013	0012
bc_bios	c0-0c0s1	0013	0012
bc_bios	c0-0c0s2	0013	0012
bc_bios	c0-0c0s3	0013	0012

#### 2. Update firmware, if any components are not current.



**CAUTION:** The `xtzap` command is normally intended for use by Cray Service personnel only. Improper use of this restricted command can cause serious damage to the computer system.

Run `xtzap -a` to update all components.

```
crayadm@smw> xtzap -a s0
```

Note that it is possible to update firmware in cabinets or blades only rather than the entire system. For more information, see *XC™ Series System Administration Guide (S-2393)*.

#### 3. Run `xtbounce --linktune` if any components were not current.

Force `xtbounce` to do a `linktune` on the full system before checking firmware again.

```
crayadm@smw> xtbounce --linktune=all s0
```

4. Confirm that all components with out-of-date firmware have been updated.

Check firmware again after updating and linktuning those components.

```
crayadm@smw> xtzap -r -v s0
```

## 5.6.4 Update Config Sets

### About this task

It is necessary to update all config sets at several points in the fresh install or software update process, such as after hardware discovery. If any nodes or blades were enabled or added prior to running `xtdiscover` and the config sets are not updated afterward, then the system `/etc/hosts` files will not have entries generated for the respective nodes and the nodes will not boot (the boot error will indicate "not in any tier" in an ansible failure). The update ensures that pre- and post-configuration scripts have been properly executed for the global and CLE config sets.

### Procedure

1. Update the global config set.

```
smw# cfgset update global
```

2. Update the CLE config set.

```
smw# cfgset update p0
```

Repeat this step for all CLE config sets used in this system.

## 5.6.5 Validate Config Sets

### About this task

It is important to validate any config set that has been modified, because there is currently no mechanism to prevent the system from trying to use an invalid config set. Validation is useful for determining if the config set is minimally viable for use with the system it is intended to configure.

**IMPORTANT:** Validation ensures that a config set passes all rules stored on the system. A validated config set does not necessarily equate to a config set with configuration data that will result in a properly configured system.

When validating a config set, the configurator checks the following:

- Config set has the proper directory structure and permissions.
- All configuration templates have correct YAML syntax.
- All configuration templates adhere to the configurator schema.
- All fields of type `lookup` reference values and settings that exist in the available configuration services.
- All level `required` fields in enabled services are configured (i.e., their state is `set`).

- Pre-configuration and post-configuration callback scripts ran successfully during the latest config set update.
- `cfgset validate` has run all validation rules installed on the system.

For more information on how `lookup` fields work, see the "Advanced: Lookup" section in "Configurator Data Types and How to Set Them," which is in *XC™ Series Configurator User Guide* (S-2560). For more information about validation rules, see "Validate a Config Set and List Validation Rules," also in that publication.

## Procedure

1. Validate the global config set.

```
smw# cfgset validate global
```

2. Validate the CLE config set.

This example uses CLE config set `p0`. Substitute the correct config set name for this site.

```
smw# cfgset validate p0
```

## 5.6.6 Restore PMDB Customizations after a Software Update

### About this task

If the Power Management Database (PMDb) is located on the SMW (an internal PMDB), then when an SMW software update or upgrade includes a PostgreSQL update or upgrade (the SMW 8.0.UP07 release includes PostgreSQL 9.6), a new PostgreSQL data directory is automatically created, and the old PostgreSQL data directory is automatically removed to avoid running out of space. In the process, any site-specific settings and scripts may be lost. Use the first step to restore any site customizations to the PMDB, and then skip the rest of this procedure.

If the PMDB is located on an external node instead (a remote PMDB), then when an SMW software update or upgrade includes a PostgreSQL update or upgrade (the SMW 8.0.UP07 release includes PostgreSQL 9.6), a new PMDB image must be built and installed on the external node. If the previous image included site customizations, additional steps must be taken to ensure that those are added to the new image. Skip the first step, and then perform the rest of this procedure.

## Procedure

1. (Internal PMDB only) Restore any customized PMDB config files in `/var/lib/pgsql/data`.

Customized PMDB config files were backed up in an earlier procedure as part of preparing for the SMW software update.

When done with this step, skip the remaining steps and proceed to the next procedure in this guide.

2. (Remote PMDB only) Prepare any site customizations to the PMDB image and rebuild the image.

- a. If there are site customizations to the PMDB image, prepare the necessary files and then rebuild the updated image.

Use the instructions in the following two procedures, which are located under "Create a Remote Power Management Database" in *XC™ Series Power Management and SEDC Administration Guide* (S-0043).

- "Prepare the Remote PMDB for Software Updates"
- "Update the Remote Database Node Software"

Skip the next substep, which is identical to the last step of the "Update the Remote Database Node Software" procedure.

- b. If there are NO site customizations to the PMDB image, simply rebuild the updated image.

```
smw# make cray-pmdb-image
```

### 3. (Remote PMDB only) Update the remote PMDB.

The update of the remote PMDB (essentially a fresh install of the updated PMDB image) can be done now, or it can be done after the SMW/CLE software update process is complete.

- To update the remote PMDB now:
  1. Follow the instructions in "Install and Deploy a Remote Power Management Database," which is located under "Create a Remote Power Management Database" in *XC™ Series Power Management and SEDC Administration Guide (S-0043)*.
  2. Return here and continue the SMW/CLE software update process by proceeding to the next procedure in this guide.
- To update the remote PMDB later, proceed to next procedure in this guide. A prompt to update the remote PMDB occurs at the end of the SMW/CLE software update process.

## 5.6.7 Finish Configuring the SMW for the CLE System Hardware

### Prerequisites

Cray hardware has been discovered and component firmware has been updated (if needed).

### About this task

This procedure contains the final steps of configuring the SMW for the CLE system hardware. Note that a full system is referred to as "s0" here. The term "p0" could have been used, because in this context, the two terms are interchangeable. In contrast, commands that operate on config sets use only the term "p0" when referring to a full system. In the config set context, the terms are not interchangeable.

Note that all of the commands in this procedure are run as `crayadm`.

### Procedure

1. Check status on all components.

```
crayadm@smw> xtcli status s0
```

2. Check routing configuration of the system.

```
crayadm@smw> rtr -R s0
```

Note that the `rtr -R` command produces no output unless there is a routing problem.

3. Examine the hardware inventory and verify that all nodes are visible to the SMW.

```
crayadm@smw> xthwinv s0 > xthwinv.out
```

```
crayadm@smw> xthwinv -x s0 > xthwinv.xml
```

#### 4. Check microcontroller information.

Execute the `xtmcinfo -u` command to retrieve microcontroller information from cabinet control processors and blade control processors. Ensure that all blade controllers have output and show similar uptime values.

```
crayadm@smw> xtmcinfo -u s0
```

#### 5. Exit from `crayadm` back to `root`.

```
crayadm@smw> exit
smw#
```

## 5.7 Install Patches

### About this task

This procedure finds, downloads, and installs patches for a Cray XC Series system. This is done just prior to booting the CLE system.

System administrators that prefer to boot the CLE system first and perform post-boot tests before installing patches may defer this procedure until after the first system boot, unless the release notes indicate that one or more patches are required for a successful boot.

### Procedure

1. Check CrayPort for patches released by Cray.
2. Make a directory on the SMW (if it does not already exist) to hold any patches that may be available on CrayPort.

```
smw# mkdir -p /var/adm/cray/release/patchsets
```

3. Download any SMW, CLE, and SMW HA patches to the patchset directory on the SMW, as described in the release notes (ERRATA and README files).
4. Install SMW and CLE patches.
  - To install a single patch, follow the instructions provided in the patch `.readme` file.
  - To load and install multiple patches, complete the following substeps. When installing more than one patch, Cray recommends postponing the building and mapping of images until the last patch is installed.
  - SMW patches are typically installed before CLE patches; however, if CLE patches are installed first, SMW patches can be installed while images are built (this step and the next step done in parallel).
  - If there are any SMW HA patches (for example, `SMWHA_12.0.UPxx.PSxx`), do not install them at this time. They will be installed after SMW HA software is installed/updated, following instructions in *XC™ Series SMW HA Installation Guide (S-0044)*.

**NOTE:** (SMW HA only) Make a note of all SMW and CLE patch sets that will be applied on the first SMW (SMW HA patch sets are not applied at this time). The second SMW must have exactly the same patch sets.

- a. Temporarily suppress building and mapping images.

```
smw# export PATCHSET_BUILD_IMAGES=false
smw# echo $PATCHSET_BUILD_IMAGES

smw# export PATCHSET_NIMS_TIMING=deferred
smw# echo $PATCHSET_NIMS_TIMING
```

- b. Follow all of the instructions in the patch .readme files.

These instructions will include running the `.load` script and the `.install` script for each patch, and there may be additional steps for some patches, such as running `xtzap` again to update firmware from an SMW patch.

Note that a "script" file might not be a runnable script. If necessary, copy and paste the commands into the command line and run them manually.

## 5. Build and map new images using `imgbuilder`.

**TIP:** When there are existing image roots, as with a software update or a preinstallation, `imgbuilder` will prompt for confirmation to "Build anyway?" because building new images will replace existing ones. If the answer will be "y" for all images, the command can be run with the `--force` option to make admin interaction during image building unnecessary.

To build and map images, use one of the following commands.

- To build all images in the `default` image group in parallel (this is typically quicker):

```
smw# imgbuilder --map --processes=NUM
```

Replace `NUM` with the number of images to be built concurrently (capped at the number of CPUs as reported by `lscpu`).

- To build all images in the `default` image group serially:

```
smw# imgbuilder --map
```

For more information about building images in parallel, see [About Parallel Image Creation](#) on page 40.

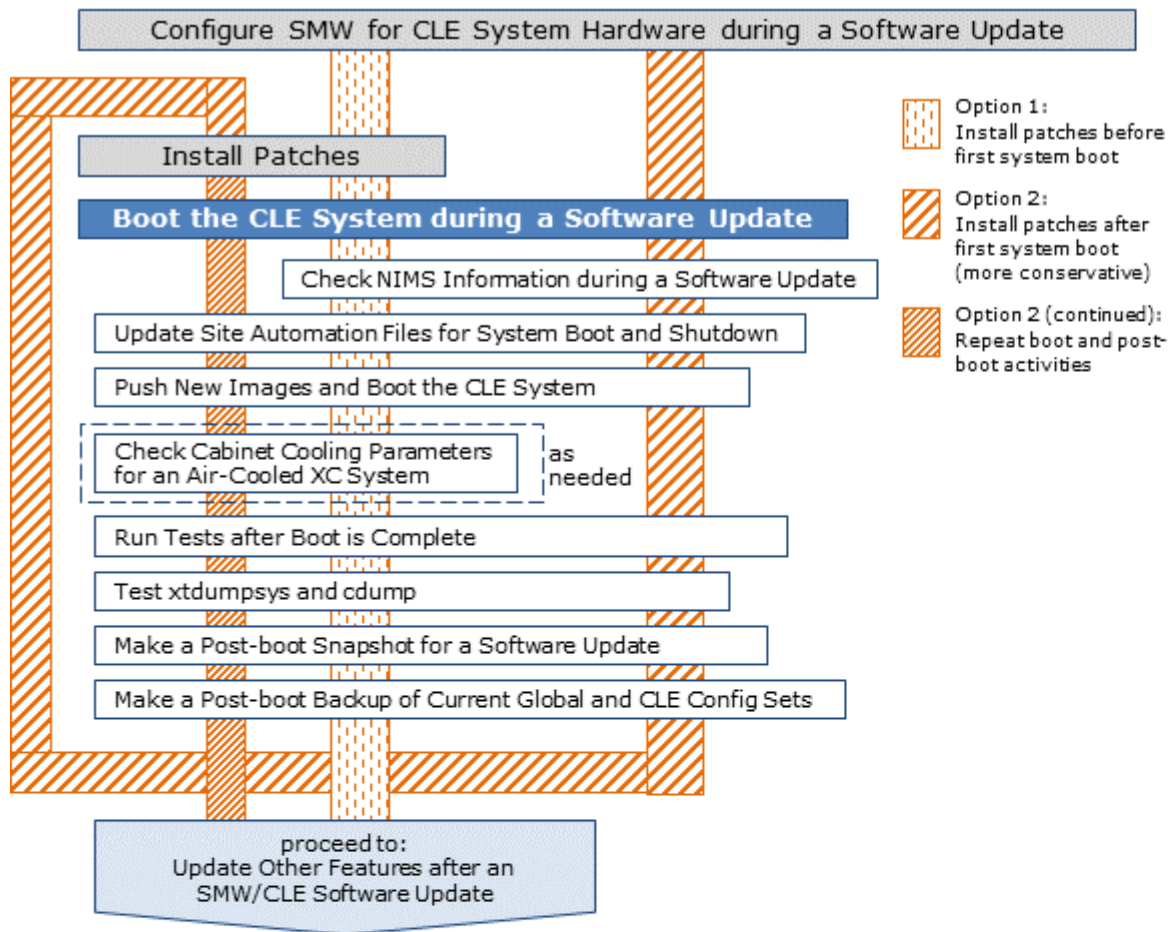
## 5.8 Boot the CLE System during a Software Update

The SMW/CLE software update process is nearly complete. The following procedures boot the CLE system and perform post-boot activities such as running tests and making a snapshot.

If patches have just been installed, continue with the procedures shown in Option 1 in the figure. If this site has chosen to defer the installation of patches until after the first system boot, follow the flow in Option 2 instead, which loops back to "Install Patches" and repeats the boot and post-boot steps.



Figure 40. Visual Guide to Booting the CLE System during a Software Update



## 5.8.1 Check NIMS Information during a Software Update

### About this task

This procedure lists NIMS (Node Image Mapping Service) information: which maps are active on the SMW and what NIMS information is stored for each node.

### Procedure

1. Check active NIMS maps.

```
smw# cmap list
```

2. Set a map to be active.

If a new NIMS map has been created as part of the software update process, ensure that the new map is active.

```
smw# cmap setactive map_name
```

3. Check the default config set of the active NIMS map.

```
smw# cmap list --fields default_config_set map_name
```

If this is not the desired default config set, use [Set Default Config Set for a NIMS Map](#) on page 408 to change it. If selected nodes need to use a different config set, see [Set Config Set for a Node](#) on page 409.

#### ————— CHECK NIMS INFORMATION —————

When checking NIMS information, things to look for include:

- Are nodes in the correct NIMS groups?
- Does each node have the correct boot image?
- Does each node have the correct config set assigned?
- If netroot used, do the netroot nodes have an "initrd-compute-large" or "initrd-login-large" boot image set as the image? And do they have the associated netroot image root (same as boot image but without "initrd-" or ".cpio") set as the netroot kernel parameter?

4. Check NIMS information for all nodes.

```
smw# cnode list
```

5. Check NIMS information for each NIMS group.

```
smw# cnode list --filter group=admin
smw# cnode list --filter group=service
smw# cnode list --filter group=login
smw# cnode list --filter group=compute
```

Check any additional NIMS groups that may have been created for netroot compute and login nodes (typically created only when netroot is used on a subset of compute and login nodes instead of all of them, so that the NIMS compute and login groups cannot be used for that subset of nodes).

```
smw# cnode list --filter group=compute_netroot
smw# cnode list --filter group=login_netroot
```

Check any additional NIMS groups that may have been created for DataWarp with Fusion IO SSDs.

```
smw# cnode list --filter group=fio-service
```

Check any additional NIMS groups that may have been created with WLM (workload manager) or other site names.

```
smw# cnode list --filter group=wlm-admin
smw# cnode list --filter group=wlm-service
smw# cnode list --filter group=wlm-login
```

## 5.8.2 Update Site Automation Files for System Boot and Shutdown

### About this task

This procedure describes how to update the site/system boot and shutdown automation files in preparation for booting the CLE system.

### Procedure

1. Update any site/system boot and shutdown automation files.

If this site has system-specific boot and shutdown automation files, compare their contents to the newly distributed `auto.generic` and `auto.xtshutdown` files, and then edit the site files to merge in any new content.

```
smw# diff /opt/cray/hss/default/etc/auto.generic \
/opt/cray/hss/default/etc/auto.hostname.start
```

```
smw# diff /opt/cray/hss/default/etc/auto.xtshutdown \
/opt/cray/hss/default/etc/auto.hostname.stop
```

2. Add the `xtfailover_halt` command to `auto.hostname.stop` if boot or SDB node failover is used and that command is not already present.

If boot or SDB node failover is used, the `xtfailover_halt` command must be included and enabled in the shutdown automation file (`auto.hostname.stop`). Add the following lines to `auto.hostname.stop`, if not already present.

To enable `xtfailover_halt`, uncomment the `lappend actions` line. This ensures that the `xtbootsys` shutdown process sends a STOP NMI to the failover nodes.

```
# Enable the following line if boot or sdb failover is enabled:
lappend actions { crms_exec \
"/opt/cray/hss/default/bin/xtfailover_halt --partition $data(partition,given) --
shutdown" }
```

## 5.8.3 Push New Images and Boot the CLE System

### Prerequisites

This procedure assumes the following:

- The XC system is not yet booted.
- New images have been created that must be pushed to the boot node prior to booting the XC system (e.g., diagnostics (diags), PE, and netroot image roots).

### About this task

This procedure boots the boot node; pushes any new diags, PE, and netroot compute/login image roots to the boot node; shuts down the system; and then boots the XC system with `xtbootsys`.

The example commands for pushing the image roots to the boot node use `image sqpush` instead of `image push`. For more information, see [About Image Pushes: push versus sqpush](#) on page 42.

### Procedure

1. Boot the boot node.

```
smw# su - crayadm
crayadm@smw> xtbootsys -a auto.bootnode
crayadm@smw> exit
smw#
```

Wait until the boot node is completely booted before continuing to the next step.

2. If this system has a diags image root for this release, push it to the boot node now.

```
smw# image list | grep diag
```

In the following command, replace `DIAGS_IMAGE` with the actual diags image root on this system.

```
smw# image sqpush -d boot DIAGS_IMAGE
```

**Trouble?** If passwordless `ssh` has not been prepared between `root@smw` and `root@boot`, then the system will prompt for the password for `root@boot` twice.

3. If this system has a PE image root for this release, push it to the boot node now.

```
smw# image list | grep pe
```

In the following command, replace `PE_IMAGE` with the actual PE image root on this system.

```
smw# image sqpush -d boot PE_IMAGE
```

4. If this system has a netroot compute image root for this release, push it to the boot node now.

```
smw# image list | grep compute-large
```

In the following command, replace `NETROOT-COMPUTE_IMAGE` with the actual netroot compute image root on this system.

```
smw# image sqpush -d boot NETROOT-COMPUTE_IMAGE
```

5. If this system has a netroot login image root for this release, push it to the boot node now.

```
smw# image list | grep login-large
```

In the following command, replace `NETROOT-LOGIN_IMAGE` with the actual netroot login image root on this system.

```
smw# image sqpush -d boot NETROOT-LOGIN_IMAGE
```

6. Shut down the CLE system.

```
smw# su - crayadm
```

If this site has an automation file for shutting down the system, use it now. In the following command, replace `auto.hostname.stop` with the shutdown automation file used for this system.

```
crayadm@smw> xtbootsys -s last -a auto.hostname.stop
```

If there is no automation file, use the following command to shut down the system.

```
crayadm@smw> xtbootsys -s last -a auto.xtshutdown
```

7. Boot the CLE system.

Replace `auto.hostname.start` with the boot automation file used for this system.

```
crayadm@smw> xtbootsys -a auto.hostname.start
```

**Trouble?** If there are any problems booting CLE, see the *XC™ Series Boot Troubleshooting Guide (S-2565)* for techniques to determine what might be causing the problem.

## 5.8.4 Check Cabinet Cooling Parameters for an Air-Cooled XC System

### Prerequisites

- This is an XC Series air-cooled (XC-AC) system running SMW 8.x / CLE 6.x software.
- Patches have been installed.
- The system is booted.

### About this task

Cray provides the `xtaccheckcool` tool to enable sites to check and/or set the cooling parameters of an XC-AC system. Use this tool in the following circumstances:

- after a fresh install
- after a software update
- as part of customizing a preinstalled system
- after adding hardware to a system
- periodically, to verify that parameters are set correctly for an operational system

## Procedure

1. Display current and recommended cabinet cooling parameters.

Substitute the elevation (in feet) of this site.

```
smw# xtaccheckcool -e elevation_in_feet
```

2. Write the recommended cabinet cooling parameters to all cabinet controllers, as needed.

This command uses the `-w` option to write the recommended cooling parameters for the specified elevation to every cabinet in the system. Substitute the elevation (in feet) of this site.

```
smw# xtaccheckcool -w -e elevation_in_feet
```

To target a specific cabinet, use the `-t cname` option, where `cname` is the cname of the cabinet.

For more information and example output, see the `xtaccheckcool(8)` man page.

### 5.8.5 Run Tests after Boot is Complete

#### Prerequisites

This procedure assumes the following:

- The system has completed booting.
- The compute nodes are "interactive" (i.e., not under workload manager control).
- ALPS is available.

If ALPS is not available and Slurm is used as the workload manager (WLM), then the compute nodes can be either "interactive" or "batch" and `srun` (the Slurm command equivalent to `aprun`) should be used instead of the `aprun` commands in the steps that follow.

#### About this task

Log in to the login node as `crayadm`. This can be done from the SMW to the boot node to the login node, or directly from another computer to the login node without passing through the SMW and boot node. Then perform these rudimentary functionality checks.

## Procedure

1. Run `apstat` to get the number of nodes to use for the following commands.

```
crayadm@login> NUMNODES=$((apstat -v | grep XT | awk '{print $3}'))
crayadm@login> echo NUMNODES is $NUMNODES
```

2. Verify that all nodes run (from `/tmp`).

```
crayadm@login> cd /tmp
crayadm@login> aprun -b -n $NUMNODES -N 1 /bin/cat /proc/sys/kernel/hostname
```

3. Verify that the home directory is working by running a job.

```
crayadm@login> cd ~
crayadm@login> aprun -b -n $NUMNODES -N 1 /bin/cat /proc/sys/kernel/hostname
```

4. Verify that the Lustre directory is working by running a job.

```
crayadm@login> cd /lustre_file_system
crayadm@login> aprun -b -n $NUMNODES -N 1 /bin/cat /proc/sys/kernel/hostname
```

#### CHECK CURRENT STATE OF COMPUTE NODE SSDs

The next step is intended only for XC systems that have compute nodes with SSDs, for example, systems with DataWarp SSDs or Intel® Xeon Phi™ "Knights Landing" processors.

5. Run `xtcheckssd` to ensure that SMW databases have the current state of compute node SSDs.

```
root@login# pcmd -r -n ALL_COMPUTE "/opt/cray/ssd/bin/xtcheckssd"
```

## 5.8.6 Test xtdumpsys and cdump

### Prerequisites

This procedure assumes that the system has been booted.

### About this task

This procedure tests the `xtdumpsys` and `cdump` tools. The example output is for illustrative purposes only. Actual output may differ for the current release.

### Procedure

1. Start an `xtdumpsys` typescript.

Start a new window. Start a typescript session for `xtdumpsys` in that new window.

```
smw# su - crayadm
crayadm@smw> export TODAY=`date +%Y%m%d`
crayadm@smw> . /etc/opt/cray/release/cle-release
crayadm@smw> mkdir -p /home/crayadm/dump/${TODAY}_${BUILD}
crayadm@smw> cd /home/crayadm/dump/${TODAY}_${BUILD}
crayadm@smw> script -af hss.xtdumpsys
```

2. Start `xtdumpsys`.

Start the dump, but do not press **Ctrl-d** until step 5 on page 370. When `xtdumpsys` asks for a dump reason, it will have created the dump directory.

```
crayadm@smw> xtdumpsys
INFO: Beginning dump
INFO: Gathering system partition information
```

```

INFO: Gathering system hardware information
INFO: No session specified, defaulting to current.
INFO: Moving temporary log files to the dump directory.
INFO:
#####
INFO: # Your dump is available in /var/opt/cray/dump/p0-YYYYMMDDtHHMMSS-
NNNNNNNNNN #
INFO:
#####
Enter reason for dump:
(an EOF terminates input, usually CTRL-D)

```

### 3. Start a `cdump` typescript in a different window.

Start another window. Start a typescript session for `cdump` in that window.

```

smw# su - crayadm
cdump crayadm@smw> export TODAY=`date +%Y%m%d`
cdump crayadm@smw> . /etc/opt/cray/release/cle-release
cdump crayadm@smw> cd /home/crayadm/dump/${TODAY}_${BUILD}/
cdump crayadm@smw> script -af hss.cdump

```

### 4. Dump a node with `cdump`.

Change to the directory created in the `xtumpsys` window (after `INFO: # Your dump is available in`), then use `cdump` to dump a compute node that successfully booted.

```

cdump crayadm@smw> cd /var/opt/cray/dump/p0-YYYYMMDDtHHMMSS-NNNNNNNNNN
cdump crayadm@smw> mkdir cumps; cd cumps

```

This example uses the `c0-0c0s3n0` node.

```

cdump crayadm@smw> cdump -AmD -r xt-hsn@boot c0-0c0s3n0
Wed Mar 1 09:08:08 CDT 2017 start cdump
...
makedumpfile Completed.
- done
Wed Mar 1 09:08:08 CDT 2017 cdump: # of nodes 1
  success 1
  failed 0
  skipped 0
cdump crayadm@smw> exit

```

For a partitioned system, use the host name to specify which boot node.

### 5. Continue `xtumpsys`: enter a reason.

After `cdump` completes, return to the `xtumpsys` window and enter a reason.

```

xtumpsys window> testdump

```

Then enter an end-of-file (**Ctrl-d**) to end the dump reason.

```

xtumpsys window> <Ctrl-d>
testdump
INFO: Dump reason:
...
INFO:
#####
INFO: # Your dump is available in /var/opt/cray/dump/

```



```
p0-20170301t081927-1304240904 #
INFO:
#####
INFO: No post-processing plugin found at '/etc/opt/cray/dumpsys/
postprocessing.py'
INFO: Example plugins can be found at '/opt/cray/dumpsys/
1.2.5-1.0000.35873.20.1/bin/plugins/examples/postprocessing.py.*'
INFO: Cleaning up

xtdumpsys crayadm@smw> exit
```

## 6. Remove dump directory, if desired.

If there are no errors, it is probably safe to delete the dump directory.

```
xtdumpsys crayadm@smw> rm -rf /var/opt/cray/dump/pX-YYYYMMDDtHHMMSS-NNNNNNNNNN
crayadm@smw> exit
```

## 5.8.7 Make a Post-boot Snapshot for a Software Update

### About this task

This procedure uses `snaputil` to make an archival snapshot of the system after booting the CLE system.

**Best Practice.** Make a snapshot and back up config sets at the same time to keep them in sync. Name the snapshot and config set backup using the same suffix and date/time stamp to help administrators identify which snapshot and config set backup pairs belong together.

For more information, see [About Snapshots and Config Set Backups](#) on page 21.

### Procedure

#### 1. List the available snapshots on the system.

```
smw# snaputil list
```

#### 2. Set the `SNAPSHOT` environment variable using the currently booted snapshot name.

```
smw# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')
smw# echo $SNAPSHOT
```

Setting a variable for the snapshot name enables better command substitution in later commands dealing with snapshots.

This is especially important for SMW HA systems because it makes it easier to use the exact same snapshot name for both SMWs. Using different snapshots results in HSS database (MySQL) inconsistencies, which causes problems at failover.

#### 3. Change the `zypper` repo type.

This step is recommended, but not required, for both a standalone SMW and the first SMW of an SMW HA pair.

```
smw# sed -i 's/type=rpm-md/type=plaindir/' /etc/zypp/repos.d/*.repo
smw# zypper refresh
```

4. Create a new snapshot.

```
smw# snaputil create ${SNAPSHOT}.postboot-${TODAY}
```

5. (SMW HA only) If this is an SMW HA system, change the `zypper` repo type and make a post-boot snapshot on the second SMW also.

- a. Force a failover so a snapshot can be created on the other SMW.

Confirm that the failover has completed, `smw2` is now active, and all services are running again.

```
smw1# crm resource move ClusterIP smw2
smw1# sleep 300
smw1# crm resource unmove ClusterIP
smw1# crm status
```

- b. Change the `zypper` repo type on the second SMW.

This is required for the second SMW of an SMW HA system, because the installer incorrectly sets the `zypper` repo type on the second SMW during installation.

```
smw2# sed -i 's/type=rpm-md/type=plaindir/' /etc/zypp/repos.d/*.repo
smw2# zypper refresh
```

- c. Create a new snapshot on the second SMW.

```
smw2# export TODAY=`date +%Y%m%d`
smw2# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')
smw2# snaputil create ${SNAPSHOT}.postboot-${TODAY}
```

- d. Force a failover back to the first SMW, confirm that the failover has completed, `smw1` is active again, and all services are running.

```
smw2# crm resource move ClusterIP smw1
smw2# sleep 300
smw2# crm resource unmove ClusterIP
smw2# crm status
```

## 5.8.8 Make a Post-boot Backup of Current Global and CLE Config Sets

### About this task

This procedure uses the `cfgset` command to create a post-boot backup of the global and CLE config sets.

### Procedure

1. Back up the current global config set.

```
smw# cfgset create --clone global global-postboot-${TODAY}
```

2. Back up the current CLE config set.

This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-postboot-${TODAY}
```

## 5.9 Update Other Features after an SMW/CLE Software Update

### Prerequisites

This system has one or more of the following features:

- DataWarp
- eLogin nodes
- a remote Power Management Database (PMDB)

### About this task

If this system does not have DataWarp, eLogin, or a remote PMDB, then skip this procedure. The SMW/CLE update process is complete.

If this system does have one or more of those features, then this is the final stage in the SMW/CLE update process. To update DataWarp, eLogin, or a remote PMDB for a Cray XC™ Series system, use the publications listed in the following steps.

### Procedure

**1. Update DataWarp.**

Skip this step if this system does not use DataWarp.

To update DataWarp for this system, follow the instructions in "DataWarp Update Following CLE Update" in *XC™ Series DataWarp™ Installation and Administration Guide* (S-2564).

**2. Update eLogin.**

Skip this step if this system does not have any eLogin nodes.

Beginning with the CLE 6.0.UP06 release, management support for eLogin nodes is provided entirely by the SMW; no other management node is needed.

Now, the SMW (running SLES 12 SP3) is used not only for the config set and eLogin image root but also to provision the image root to the node via PXE boot and perform other management functions.

To update or migrate the eLogin deployment from an earlier release, or to set up eLogin for the first time, follow the instructions in *XC™ Series SMW-managed eLogin Installation Guide* (S-3020) for release CLE 6.0.UP07.

This applies to an SMW HA system as well.

**3. Update the remote PMDB.**

Skip this step if this system does not have a remote PMDB or the remote PMDB was updated earlier in the process.

An update of a remote PMDB is essentially a fresh install of the updated PMDB image. This step assumes that an updated PMDB image has been rebuilt.

To install the updated remote PMDB image, follow the instructions in "Install and Deploy a Remote Power Management Database," which is located under "Create a Remote Power Management Database" in *XC™ Series Power Management and SEDC Administration Guide (S-0043)*.

## 6 Customize Preinstalled SMW/CLE Software

---

Cray ships System Management Workstation (SMW) systems that are installed and configured with Cray-specific hostnames and IP addresses, among other things. To complete the configuration on site, reconfigure the system using these procedures.

Note that many commands require root privilege.



**CAUTION: Boot failure possible if using `cfgset` under certain conditions.**

The `cfgset create` and `cfgset update` commands always call pre- and post-configuration scripts. Some of these scripts require HSS daemons and other CLE services to be running. This can cause problems under these conditions:

- If `xtdiscover` is running, `cfgset` may hang or produce incorrect data that can result in system boot failure.
- If `xtbounce` is in progress or if the SMW is not connected to XC hardware, `cfgset` will fail.

In these circumstances, use the `--no-scripts` option with `cfgset create` or `cfgset update` to avoid running the scripts. Because using that option results in an invalid config set, remember to run `cfgset update` without the `--no-scripts` option afterwards, when circumstances permit, to ensure that all pre- and post-configuration scripts are run.

1. [Update Site Information and Install Needed Patches](#) on page 376.
2. [Change the Default System Management Workstation \(SMW\) Passwords](#) on page 378 (includes instructions for logging in as root).
3. [Change the Time Zone](#) on page 379.
4. (Optional) [Configure the SMW Firewall](#) on page 382.
5. [Configure LAN on the SMW](#) on page 383.
6. [Change Networks, IP Addresses in Global Config Set](#) on page 383.
7. [Change Networks and IP Addresses in CLE Config Set](#) on page 385.
8. Configure iDRAC network information.
  - For a Dell R630 or R640 SMW: [Set Up iDRAC for a Dell R630 or R640 SMW](#) on page 388.
  - For a Dell R815 SMW: [Set Up iDRAC for a Dell R815 SMW](#) on page 391.
9. [Change the Default iDRAC Password](#) on page 395.
10. [Enable Write Cache on SMW Boot RAID Volume](#) on page 395 (required for ALPS)
11. (Optional) [Configure the SEC and check\\_xt Monitoring and Notification Utilities](#) on page 268.
12. (Optional) [Configure Site Lightweight Log Manager \(LLM\)](#) on page 397.
13. (Optional) [Prevent Unintentional Re-creation of Mail Configuration Files](#) on page 291.

14. [Make a Post-customize Snapshot using snaputil](#) on page 400.
15. [Make a Post-customize Backup of Current Global and CLE Config Sets](#) on page 401.

## 6.1 Update Site Information and Install Needed Patches

### Prerequisites

This procedure uses the `xtshowrev` tool. If that module is not yet loaded, see [Prepare Site and Software Revision Information Reporting using xtgetrev and xtshowrev](#) on page 251.

### About this task

The first task in customizing a preinstalled system is to ensure that the site name and serial number are set correctly and determine which patches were installed.

### Procedure

1. Determine which patches were installed in the factory.

Use the `xtshowrev` command. The example shows output for an older release, but its purpose is to indicate where to look for CLE, SLE, and SMW patch information.

```
smw# xtshowrev
Site:          CRAY/INTERNAL
S/N:          9999
System Type:   XC40
Install Date:  2016-06-01
System Name:   panda1
CNL/CLE Release: 6.0.UP01
XT Release:    6.0.96
CLE Kernel:    3.12.51-52.31.1_1.0600.9146
CLE OS:        SLES12
CLE Patch Sets: 01 02 03      <----- CLE patches applied
CLE FNs:
Lustre Version: 2.5.4
OS Type:        CLE
SMW Release:    8.0.UP01
SMW Build:      8.0.96
HSS Release:    8.0__446__ge75851a-49.1
SMW Kernel:     3.12.51-52.39
SMW OS:         SLES12
SLE Patch Sets: <----- SLE patches applied
SMW Patch Sets: <----- SMW patches applied
SMW FNs:        5844c
SEC Release:    Cray_SEC 8.0__6__g689802a (sec 2.7.6)
Current Date:   2016-06-01 12:59:21
crayadm@smw>
```

2. Update site information in the `site_config` file.

For an initial install, the `xtgetrev` command is used to enter site information. For a preinstalled system, enter this information by manually editing the `site_config` file instead.

```
smw# vi /etc/opt/cray/release/pkginfo/site_config
---
site name: CRAY/INTERNAL      <----- change this
serial number: 9999           <----- change this
system name: pandal          <----- change this, if needed
system type: XC40             <----- change this, if needed
install date: 2016-06-01
os type: CLE
```

It is especially important to change/enter the serial number because that is the key into the Site Configurations Database, and it is used to determine whether a site has access to future patches.

### 3. Check for patches released by Cray.

Day-one patches are noted in the Errata docs that are included with the release. For other patches, check CrayPort, which is updated with available patches for the entitled site serial numbers when a patch is released. If patches need to be applied, continue with the remaining steps.

CONTINUE ONLY IF PATCHES NEED TO BE APPLIED

### 4. Make a directory on the SMW (if it does not already exist) to hold any patches that may be available on CrayPort.

```
smw# mkdir -p /var/adm/cray/release/patchsets
```

### 5. Download any SMW, CLE, and SMW HA patches to the patchset directory on the SMW, as described in the release notes (ERRATA and README files).

### 6. Install SMW and CLE patches.

- To install a single patch, follow the instructions provided in the patch `.readme` file.
- To load and install multiple patches, complete the following substeps. When installing more than one patch, Cray recommends postponing the building and mapping of images until the last patch is installed.
- SMW patches are typically installed before CLE patches; however, if CLE patches are installed first, SMW patches can be installed while images are built (this step and the next step done in parallel).
- If there are any SMW HA patches (for example, SMWHA\_12.0.UPxx.PSxx), do not install them at this time. They will be installed after SMW HA software is installed/updated, following instructions in *XC™ Series SMW HA Installation Guide* (S-0044).

**NOTE:** (SMW HA only) Make a note of all SMW and CLE patch sets that will be applied on the first SMW (SMW HA patch sets are not applied at this time). The second SMW must have exactly the same patch sets.

#### a. Temporarily suppress building and mapping images.

```
smw# export PATCHSET_BUILD_IMAGES=false
smw# echo $PATCHSET_BUILD_IMAGES

smw# export PATCHSET_NIMS_TIMING=deferred
smw# echo $PATCHSET_NIMS_TIMING
```

#### b. Follow all of the instructions in the patch `.readme` files.

These instructions will include running the `.load` script and the `.install` script for each patch, and there may be additional steps for some patches, such as running `xtzap` again to update firmware from an SMW patch.

Note that a "script" file might not be a runnable script. If necessary, copy and paste the commands into the command line and run them manually.

## 7. Build and map new images using `imgbuilder`.

**TIP:** When there are existing image roots, as with a software update or a preinstallation, `imgbuilder` will prompt for confirmation to "Build anyway?" because building new images will replace existing ones. If the answer will be "y" for all images, the command can be run with the `--force` option to make admin interaction during image building unnecessary.

To build and map images, use one of the following commands.

- To build all images in the `default` image group in parallel (this is typically quicker):

```
smw# imgbuilder --map --processes=NUM
```

Replace `NUM` with the number of images to be built concurrently (capped at the number of CPUs as reported by `lscpu`).

- To build all images in the `default` image group serially:

```
smw# imgbuilder --map
```

For more information about building images in parallel, see [About Parallel Image Creation](#) on page 40.

## 6.2 Change the Default System Management Workstation (SMW) Passwords

### About this task

The SMW contains its own `/etc/passwd` file that is separate from the password file for the rest of the CLE system. After logging on to the SMW for the first time, Cray recommends changing the default passwords, as described in the following instructions.

### Procedure

#### 1. Log in to SMW as root.

When the login screen is displayed with the `crayadm` account as the account which will be logged in:

- Select **Not listed?**, then enter `root` for the username.
- Either press **Enter** or select **Sign In**.
- Enter the password for root.

#### 2. Change default passwords on the SMW by executing the following commands.

```
smw# passwd root
```

```
smw# passwd crayadm
```

```
smw# passwd mysql
```



It is also necessary to change the iDRAC password, which uses a different procedure. See [Change the Default iDRAC Password](#) on page 395.

## 6.3 Change the Time Zone

### Prerequisites

This procedure assumes that the XC system is booted.

### About this task

This procedure changes the time zone of an XC system by changing some configuration and then rebooting components. Most of these commands must be performed as root.

### Procedure

1. Check the current time zone.

- a. Check time zone on SMW.

```
smw# date
```

- b. Check time zone on cabinet and blade controllers.

```
smw# xtrsh -l root -s date
```

- c. Check time zone on boot node.

```
smw# ssh boot date
```

- d. Check time zone on SDB node.

This command works from the SMW if the SDB node is a tier1 node with an Ethernet connection to the SMW.

```
smw# ssh sdb date
```

- e. Check time zone on all service nodes.

```
smw# ssh sdb pcmd -r -n ALL_SERVICE_NOT_ME "date"
```

- f. Check time zone on all compute nodes.

```
smw# ssh sdb pcmd -r -n ALL_COMPUTE "date"
```

Continue to the next step only if the time zone needs to be changed.

2. Change the SMW local time zone, if needed.

The default time zone on the SMW is **America/Chicago**. To change it:

- a. Execute this command:

```
smw# yast2 timezone
```

yast2 opens a new window for changing the time zone, then a pop-up window appears with this message: "file /etc/ntp.conf has been changed manually. YaST might lose some of the changes."

- b. Select the **Do not show this message anymore** checkbox, then select **Continue**.
- c. Choose the time zone either by selecting a region on the map or by using the drop-down menus for **Region** and **Time Zone**.
- d. Select **Other Settings** if the time is incorrect, then select the **Manually** radio button and enter **Current Time** and **Current Date**. Select **Accept** when done.
- e. Select **OK** when done with time zone settings.

The change on the SMW is immediate, but any users on the system need to log out and then log in again to get the new environment. This does not change the time zone for the CLE nodes or the cabinet and blade controllers. Continue to step 3 to make those changes.

### 3. Change the time zone in the global config set.

- a. Set `cray_time.settings.service.data.timezone` to the desired time zone.

A list of possible time zones is available on the SMW in `/usr/share/zoneinfo/zone1970.tab`.

```
smw# cfgset update -s cray_time -m interactive global
```

- b. Validate the config set.

```
smw# cfgset validate global
```

### 4. Change the time zone in the CLE config set.

If the CLE config set has `cray_time.inherit` set to true, then the time zone and other time settings from the global config set will be inherited by the CLE config set. If the CLE config set has `cray_time.inherit` set to false, then use these commands to change the setting and validate the config set.

- a. Set `cray_time.settings.service.data.timezone` to the desired time zone.

A list of possible time zones is available on the SMW in `/usr/share/zoneinfo/zone1970.tab`.

```
smw# cfgset update -s cray_time -m interactive p0
```

- b. Validate the config set.

```
smw# cfgset validate p0
```

### 5. Put the SMW time zone setting where the cabinet and blade controllers can access it.

```
smw# cp /etc/localtime /opt/tftpboot/localtime
```

### 6. Reboot to set the new time zone for all components.

- a. Shut down CLE.

```
smw# su - crayadm
crayadm@smw> xtbootsys -s last -a auto.hostname.stop
```

If this site does not have an `auto.hostname.stop` file, use the following command to shut down the system.

```
crayadm@smw> xtbootsys -s last -a auto.xtshutdown
```

- b. Reboot the SMW and verify that the time zone has been reset.

```
crayadm@adm> exit  
smw# reboot
```

After the SMW reboots, check that the SMW has the desired time zone setting.

```
smw# date
```

- c. Reboot the cabinet controllers, then verify that all cabinet controllers are up.

```
smw# xtccreboot -c all  
  
smw# sleep 120  
  
smw# xtalive -a llsysd -l 11 s0
```

Repeat the `xtalive` command until all cabinet controllers are alive.

- d. Reboot the blade controllers, then verify that all blade controllers are up.

```
smw# xtccreboot -b all  
  
smw# sleep 120  
  
smw# xtalive s0
```

Repeat the `xtalive` command until all blade controllers are alive.

- e. Boot CLE nodes for the new time zone using the site boot automation file.

```
crayadm@smw> xtbootsys -a auto.hostname.start
```

## 7. Check the current time zone again.

- a. Check time zone on SMW.

```
smw# date
```

- b. Check time zone on cabinet and blade controllers.

```
smw# xtrsh -l root -s date
```

- c. Check time zone on boot node.

```
smw# ssh boot date
```

- d. Check time zone on SDB node.

This command works from the SMW if the SDB node is a tier1 node with an Ethernet connection to the SMW.

```
smw# ssh sdb date
```

- e. Check time zone on all service nodes.

```
smw# ssh sdb pcmd -r -n ALL_SERVICE_NOT_ME "date"
```

- f. Check time zone on all compute nodes.

```
smw# ssh sdb pcmd -r -n ALL_COMPUTE "date"
```

If these checks show the correct time zone, then the time zone has been successfully changed.

## 6.4 Configure the SMW Firewall

### Prerequisites

This procedure assumes that SLES 12 has been installed as the base operating system on the SMW.

### About this task

The SUSE firewall settings may need to be adjusted to match site firewall policy and to customize for site IP addresses. This procedure enables and configures the firewall.

**TIP:** It is not necessary to shut down the system before performing this task.

### Procedure

1. Save the SUSE firewall configuration.

Before modifying the SUSE firewall settings, make a copy of the configuration file.

```
smw# cp -p /etc/sysconfig/SuSEfirewall12 /etc/sysconfig/SuSEfirewall12.orig
```

2. Check current firewall settings.

Check current firewall settings and change to support any site requirements. During the process of configuring Cray SMW and CLE software, some of the firewall settings may be adjusted. SSH access is one of the protocols permitted through the firewall from the external network to the SMW.

```
smw# iptables -L
smw# vi /etc/sysconfig/SuSEfirewall12
```

3. Start the firewall immediately.

Invoke the modified configuration.

```
smw# systemctl start SuSEfirewall12_init.service
smw# systemctl start SuSEfirewall12.service
```

4. Ensure that the firewall will start at next boot.

Execute the following commands to start the firewall at boot time.

```
smw# systemctl enable SuSEfirewall12_init.service
smw# systemctl enable SuSEfirewall12.service
```

5. Verify firewall changes.

Verify the changes to the `iptables`.

```
smw# iptables -nvL
```

## 6.5 Configure LAN on the SMW

### About this task

This procedure sets the network configuration for eth0 and the host name for the SMW.

### Procedure

1. Execute this command:

```
smw# yast2 lan
```

The **Network Settings** screen appears with the **Overview** tab highlighted.

2. Select the **eth0** line on the **Overview** tab, then select **Edit**.

The **Network Card Setup** screen appears with the **Address** tab highlighted.

3. Select **Statically Assigned IP address** on the **Address** tab and enter values for IP address, subnet mask, and host name (including the domain name). Then select **Next**.
4. Select the **Hostname/DNS** tab on the **Network Settings** screen.
  1. For the **Hostname and Domain Name** area, enter host name and domain name.
  2. For the **Name Servers and Domain Search List**, enter Name Server 1, Name Server 2, Name Server 3, and Domain Search.
5. Select the **Routing** tab on the **Network Settings** screen, then enter the Default IPv4 Gateway (for the network connected to eth0) and set Device to eth0 using the dropdown menu.
6. Click **OK** after all of the **Network Settings** have been prepared.

## 6.6 Change Networks, IP Addresses in Global Config Set

### Prerequisites

This procedure assumes that SMW software has been installed so that the global config set is present.

### About this task

This procedure suggests some settings to change in the `cray_global_net` configuration service (in the global config set) to add site-specific data. It also includes steps to validate the global config set and run Ansible plays on the SMW to effect the changes.

## Procedure

1. Save a copy of original global worksheets.

Copy the original configuration worksheets into a new directory to preserve them in case they are needed later for comparison.

```
smw# ls -l /var/opt/cray/imps/config/sets/global/worksheets

smw# cp -a /var/opt/cray/imps/config/sets/global/worksheets \
/var/opt/cray/imps/config/sets/global/worksheets.orig
```

2. Make a work area for global worksheets.

Make a work area and copy the global configuration worksheets to that work area for editing.

The worksheets should not be edited in their original location for two reasons: (1) the configurator will not permit updating a config set from worksheets within that config set, and (2) edits would be overwritten when the config set is updated.

```
smw# mkdir -p /var/adm/cray/release
smw# cp -a /var/opt/cray/imps/config/sets/global/worksheets \
/var/adm/cray/release/global_worksheet_workarea
```

3. Change to the work area directory to simplify the editing commands in the following steps.

```
smw# cd /var/adm/cray/release/global_worksheet_workarea
```

4. Update the `cray_global_net` service to change any settings that need site-specific information.

There are two major sections to `cray_global_net`. One describes the networks to which the SMW is connected and the other describes the hosts (`primary_smw` is the only host) and the network interfaces on `primary_smw` that are on those networks.

```
smw# vi cray_global_net_worksheet.yaml
```

- a. Update management network settings.

At a minimum, these settings will need to be changed:

- Information for the management network, which is the customer network connected to the SMW.

```
cray_global_net.settings.networks.data.management.ipv4_network
cray_global_net.settings.networks.data.management.ipv4_netmask
cray_global_net.settings.networks.data.management.ipv4_gateway
cray_global_net.settings.networks.data.management.dns_servers
cray_global_net.settings.networks.data.management.ntp_servers
```

- IP address of the SMW on the management network.

```
cray_global_net.settings.hosts.data.primary_smw.interfaces.customer_ethernet.ipv4_address
```

- b. Update the SMW host ID.

If the customer Ethernet IP address changes, the output from the `hostid` command will be different. After changing this setting in the previous substep

```
cray_global_net.settings.hosts.data.primary_smw.interfaces.customer_ethernet.ipv4_address
```

run the `hostid` command again and update the following setting (the SMW host ID) to the new value.

```
cray_global_net.settings.hosts.data.primary_smw.hostid
```

5. Upload the modified `cray_global_net` worksheet into the global config set.

Note that the full filepath must be specified in this `cfgset` command.

```
smw# cfgset update -w \
/var/adm/cray/release/global_worksheet_workarea/cray_global_net_worksheet.yaml \
global
```

6. Validate the global config set.

```
smw# cfgset validate global
```

7. Run Ansible plays on the SMW.

After the global config set has been updated, reapply any Ansible plays that consume global config set data.

```
smw# /etc/init.d/cray-ansible start
```

**NOTE:** (SMW HA only) Both SMWs require this command. The procedure to install and configure the second SMW includes this command.

Logs from running Ansible plays, such as `cray-ansible`, are stored on the SMW in `/var/opt/cray/log/ansible`.

## 6.7 Change Networks and IP Addresses in CLE Config Set

### Prerequisites

This procedure assumes that the SMW and CLE software has been installed so that a CLE config set is present.

### About this task

The Cray Networking service defines all network information for CLE nodes. This procedure suggests some settings to change in the Cray Networking service configuration worksheet to add site-specific data.

**REMEMBER:** For partitioned systems, each partition generally has its own config set and associated configuration worksheets. Follow this procedure for each partition.

### Procedure

1. Save a copy of original CLE worksheets.

Copy the original configuration worksheets into a new directory to preserve them in case they are needed later for comparison.

```
smw# ls -l /var/opt/cray/imps/config/sets/p0/worksheets

smw# cp -a /var/opt/cray/imps/config/sets/p0/worksheets \
/var/opt/cray/imps/config/sets/p0/worksheets.orig
```

2. Make a work area for CLE worksheets.

Copy the CLE configuration worksheets to a new work area for editing. The worksheets should not be edited in their original location for two reasons: (1) the configurator will not permit updating a config set from worksheets within that config set, and (2) edits would be overwritten when the config set is updated.

```
smw# cp -a /var/opt/cray/imps/config/sets/p0/worksheets \
/var/adm/cray/release/p0_worksheet_workarea
```

Change to the work area directory to simplify the editing commands in the following steps.

```
smw# cd /var/adm/cray/release/p0_worksheet_workarea
```

3. Check the CLE config set for information that may need to be changed.

```
smw# cfgset search -s cray_net -t ipv4 p0
```

4. Edit `cray_net_worksheet.yaml` to change any settings that need site-specific information.

At a minimum, these settings will need to be changed:

- a. Change these values to site-specific values for the "Customer network" to which the login nodes connect.

```
cray_net.settings.networks.data.login.ipv4_network
cray_net.settings.networks.data.login.ipv4_netmask
```

- b. (Only for systems with an external Lustre server) Change these values to site-specific values for each external Lustre server.

```
cray_net.settings.networks.data.lnet.ipv4_network
cray_net.settings.networks.data.lnet.ipv4_netmask
```

- c. Change this value to the IP address of the login node's eth0 interface on the "login" network.

```
cray_net.settings.hosts.data.login_node.interfaces.login_ethernet.ipv4_addresses
```

When making changes, keep this mind:

- Add values for the `dns_servers` and `dns_search` settings to the login network only, not to any other network.
- DO NOT add a value for the `ntp_servers` setting for any network used for CLE nodes, because CLE nodes must source their time/NTP settings from the SMW rather than try to contact NTP servers on the login network.

5. Configure additional hosts, as needed.

If this system has additional service nodes that need to have host name or host name alias or network interface settings, then add a section like this for each of the hosts. The first example shows the host configuration of a DVS node with the host name set to `dvs1`, a host name alias of `dvs`, and one Ethernet interface connected to the login network.

```
cray_net.settings.hosts.data.common_name.dvs_node: null
cray_net.settings.hosts.data.dvs_node.description: DVS node
cray_net.settings.hosts.data.dvs_node.aliases:
- dvs
cray_net.settings.hosts.data.dvs_node.hostid: c0-0c0s0n2
cray_net.settings.hosts.data.dvs_node.host_type: ''
cray_net.settings.hosts.data.dvs_node.hostname: dvs1
cray_net.settings.hosts.data.dvs_node.standby_node: false

cray_net.settings.hosts.data.dvs_node.interfaces.common_name.eth0: null
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.name: eth0
```



```

cray_net.settings.hosts.data.dvs_node.interfaces.eth0.description: Ethernet
    connecting the DVS node to the customer network.
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.vlan_id: ''
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.vlan_etherdevice: ''
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.bonding_slaves: []
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.bonding_module_opts:
    mode=active-backup miimon=100
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.aliases: []
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.network: login
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.ipv4_address: 172.30.50.128
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.mac: ''
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.startmode: auto
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.bootproto: static
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.mtu: ''
cray_net.settings.hosts.data.dvs_node.interfaces.eth0.extra_attributes: []
#cray_net.settings.hosts.data.dvs_node.interfaces.eth0.module: ''
#cray_net.settings.hosts.data.dvs_node.interfaces.eth0.params: ''
#cray_net.settings.hosts.data.dvs_node.interfaces.eth0.unmanaged_interface: false

```

The second example shows the host configuration for an LNet router node that has two different InfiniBand interfaces (ib0 and ib2) to connect to two different networks.

```

cray_net.settings.hosts.data.common_name.clfs_lnet_1: null
cray_net.settings.hosts.data.clfs_lnet_1.description: CLFS router 1 node
cray_net.settings.hosts.data.clfs_lnet_1.aliases: []
cray_net.settings.hosts.data.clfs_lnet_1.roles: []
cray_net.settings.hosts.data.clfs_lnet_1.hostid: 'c0-0cls0n1'
cray_net.settings.hosts.data.clfs_lnet_1.host_type: ''
cray_net.settings.hosts.data.clfs_lnet_1.hostname: lnet1
cray_net.settings.hosts.data.clfs_lnet_1.standby_node: false

cray_net.settings.hosts.data.clfs_lnet_1.interfaces.common_name.ib0: null
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.name: ib0
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.description: InfiniBand
    ib0 connecting the CLFS router 1 node to the lnet network.
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.vlan_id: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.vlan_etherdevice: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.bonding_slaves: []
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.bonding_module_opts:
    mode=active-backup miimon=100
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.aliases: []
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.network: lnet
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.ipv4_address: '10.150.10.65'
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.mac: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.startmode: auto
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.bootproto: static
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.mtu: '65520'
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.extra_attributes:
- IPOIB_MODE='connected'
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.module: ''
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.params: ''
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib0.unmanaged_interface: false

cray_net.settings.hosts.data.clfs_lnet_1.interfaces.common_name.ib2: null
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.name: ib2
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.description: InfiniBand
    ib2 connecting the CLFS router 1 node to the lnet1 network.
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.vlan_id: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.vlan_etherdevice: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.bonding_slaves: []
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.bonding_module_opts:
    mode=active-backup miimon=100
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.aliases: []
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.network: lnet1
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.ipv4_address: '10.151.10.65'
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.mac: ''
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.startmode: auto
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.bootproto: static
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.mtu: '65520'
cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.extra_attributes:
- IPOIB_MODE='connected'
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.module: ''
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.params: ''
#cray_net.settings.hosts.data.clfs_lnet_1.interfaces.ib2.unmanaged_interface: false

```

6. Upload the modified worksheet into the CLE config set.

Note that the full filepath must be specified in this `cfgset` command.

```
smw# cfgset update -w \  
/var/adm/cray/release/p0_worksheet_workarea/cray_net_worksheet.yaml p0
```

7. Validate the CLE config set.

```
smw# cfgset validate p0
```

8. Reboot the CLE nodes to effect the changes made to the CLE config set.

```
smw# su - crayadm  
crayadm@smw> xtbootsys -a auto.hostname.start
```

For information about `auto.hostname.start`, see [About Boot Automation Files](#) on page 34.

**Trouble?** If there are any problems booting CLE, see *XC™ Series Boot Troubleshooting Guide (S-2565)* for techniques to determine what might be causing the problem.

## 6.8 Set Up iDRAC for a Dell R630 or R640 SMW

### Prerequisites

This procedure requires the following:

- Physical access to the SMW console
- iDRAC6 IP address, subnet mask, and default gateway
- SMW `root` account password

### About this task

An integrated Dell Remote Access Controller (iDRAC) enables remote management of the System Management Workstation (SMW). This procedure sets up and enables an iDRAC for a Dell R630 or R640 SMW. For an R815 model, see [Set Up iDRAC for a Dell R815 SMW](#) on page 391.

### Procedure

1. If the SMW is up, `su` to `root` and shut it down.

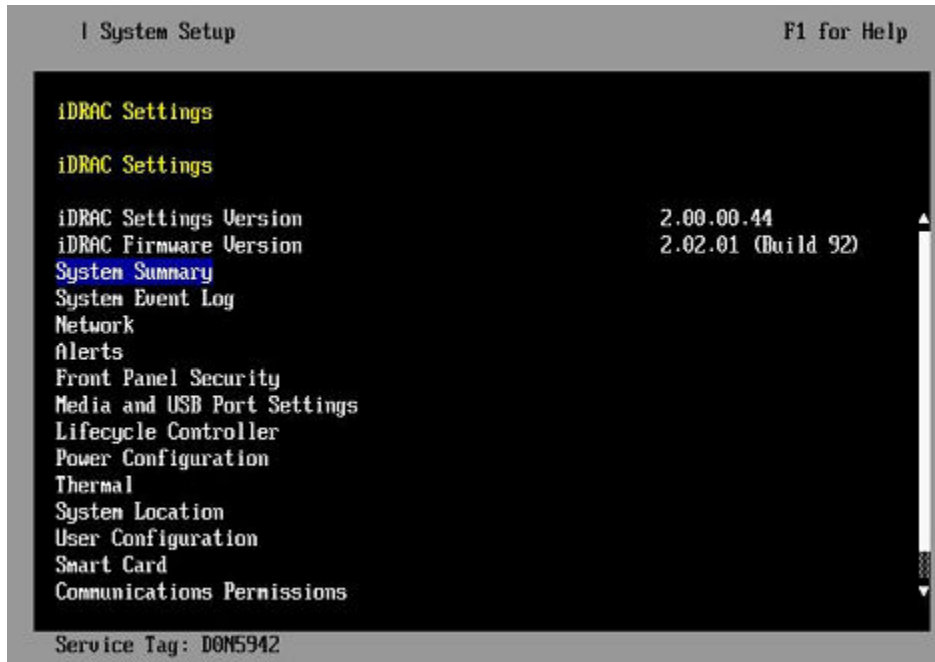
```
crayadm@smw> su - root  
smw# shutdown -h now;exit
```

2. Connect an Ethernet cable to the iDRAC port. The cable is located on back of the R815 SMW in the lower left corner.
3. Power up the SMW.
4. Change the iDRAC settings.

Select **iDRAC Settings** on the **System Setup Main Menu**, then press **Enter**.

The **iDRAC Settings** screen appears.

Figure 41. Dell R630/R640 iDRAC6 Settings Screen



5. Change the iDRAC network.

- a. Select **Network** to display a long list of network settings.

- b. Change the DNS DRAC name.

Use the arrow key to scroll down to **DNS DRAC Name**, then enter an iDRAC host name that is similar to the SMW node host name (e.g., cray-drac).

- c. Change the static DNS domain name.

Use the arrow key to scroll down to **Static DNS Domain Name**, then enter the DNS domain name and press **Enter**.

- d. Change the IPv4 settings.

Use the arrow key to scroll down to the **IPV4 SETTINGS** list.

1. Ensure that IPv4 is enabled.

- a. If necessary, select **Enable IPV4**, then press **Enter**.

- b. Select **<Enabled>** in the pop-up window, then press **Enter** to return to the previous screen.

2. Ensure that DHCP is disabled.

- a. If necessary, select **Enable DHCP**, then press **Enter**.

- b. Select **<Disabled>** in the pop-up window, then press **Enter** to return to the previous screen.

3. Change the IP address.

- a. Select **Static IP Address**.

- 390

- c. Press **Esc** to exit the **Media and USB Port Settings** menu.
9. Set the password for the iDRAC root account.
  - a. Use the arrow key to highlight **User Configuration** on the **iDRAC Settings** screen, then press **Enter**.
  - b. Confirm that User Name is root. Select **User Name**, then enter the "root" user name.
  - c. Select **Change Password**, then enter a new password.
  - d. Reenter the new password in the next pop-up window to confirm it (the default password is "calvin").
  - e. Press the **Esc** key to exit the **User Configuration** screen.
10. Exit iDRAC settings.
  - a. Press the **Esc** key to exit the **iDRAC Settings** screen.  
A "Settings have changed" message appears.
  - b. Select **Yes**, then press **Enter** to save the changes.  
A "Success" message appears.
  - c. Select **Ok**, then press **Enter**.  
The main screen (**System Setup Main Menu**) appears.

## 6.9 Set Up iDRAC for a Dell R815 SMW

### Prerequisites

This procedure requires the following:

- Physical access to the SMW console
- iDRAC6 IP address, subnet mask, and default gateway
- SMW `root` account password

### About this task

An integrated Dell Remote Access Controller (iDRAC) enables remote management of the System Management Workstation (SMW). This procedure sets up and enables an iDRAC for an R815 SMW. For an R630 model, see [Set Up iDRAC for a Dell R630 or R640 SMW](#) on page 388.

### Procedure

1. If the SMW is up, `su` to `root` and shut it down.

```
crayadm@smw> su - root
smw# shutdown -h now;exit
```

2. Connect an Ethernet cable to the iDRAC port. The cable is located on back of the R815 SMW in the lower left corner.

3. Power up the SMW.

4. Change the iDRAC settings.

Watch the screen carefully as text scrolls until the **iDRAC6 Configuration Utility 1.57** line is visible. When the line **Press <Ctrl-E> for Remote Access Setup within 5 sec...** displays, press **Ctrl-E** within 5 seconds.

```

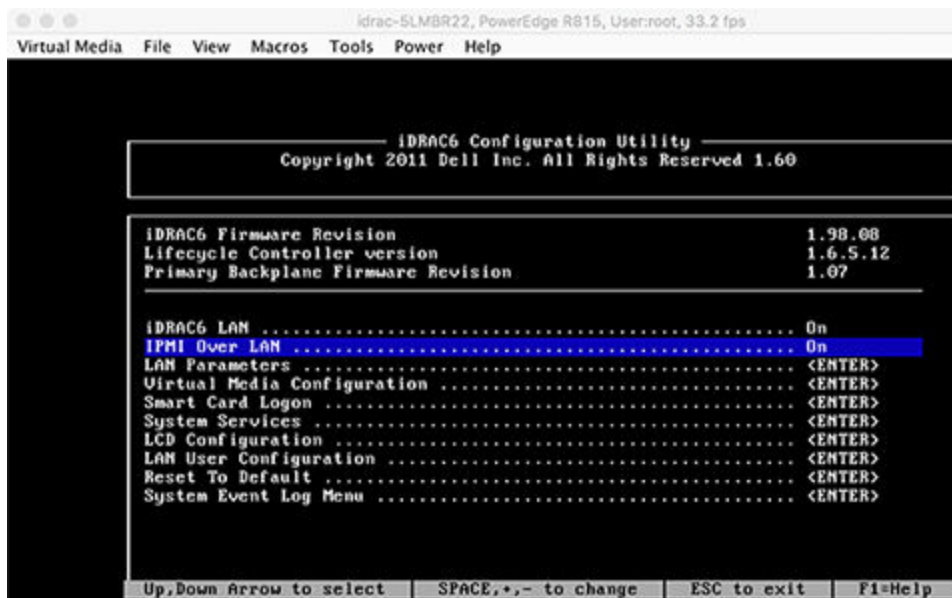
0 5 0 ATA WDC WD5000BPVT-0 1A01 465 GB
LSI Corporation MPT2 boot ROM successfully installed!
iDRAC6 Configuration Utility 1.57
Copyright 2010 Dell Inc. All Rights Reserved
iDRAC6 Firmware Revision version: 1.54.15
Primary Backplane Firmware Revision 1.07
-----
IPv6 Settings
-----
IPv6 Stack : Disabled
Address 1 : ::
Default Gateway : ::
-----
IPv4 Settings
-----
IPv4 Stack : Enabled
IP Address : 172. 31. 73.142
Subnet mask : 255.255.255. 0
Default Gateway : 172. 31. 73. 1
Press <Ctrl-E> for Remote Access Setup within 5 sec...

```

The **iDRAC6 Configuration Utility** menu appears.

5. Set **iDRAC6 LAN** to **ON**.

Figure 42. Dell R815 SMW iDRAC6 Configuration Utility Menu



6. Configure the iDRAC LAN parameters.

Select **LAN Parameters**, then press **Enter**.

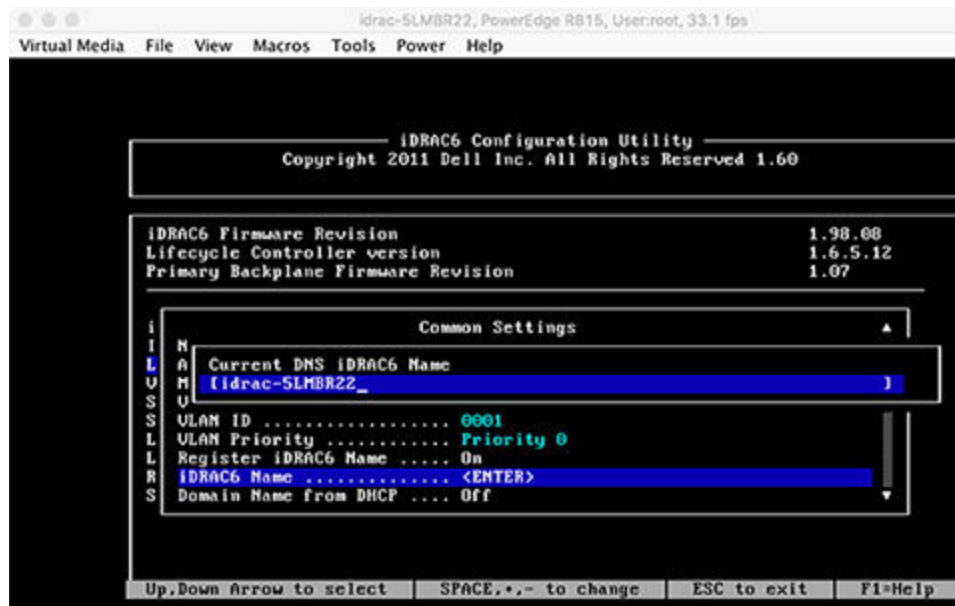
## a. Configure iDRAC6 name.

Use the arrow key to scroll down and select **iDRAC6 Name**, then press **Enter**. Enter a value for **Current DNS iDRAC6 Name** (e.g., smw-drac), then press **Esc**.

**Trouble?** If unable to set the iDRAC6 name, try this:

1. Temporarily set **Register iDRAC6 Name** to **On**.
2. Press **Enter** to set **iDRAC6 Name**. Select current or suggested name (edit enabled). When done, press **Esc**.
3. Return to **Register iDRAC6 Name** and set it to **Off**.

Figure 43. Dell R815 SMW iDRAC6 LAN Parameters: iDRAC6 Name



## b. Configure domain name.

Use the arrow key to scroll down and select **Domain Name**, then press **Enter**. Enter a value for **Current Domain Name** (e.g., us.cray.com), then press **Enter**.

## c. Configure host name string.

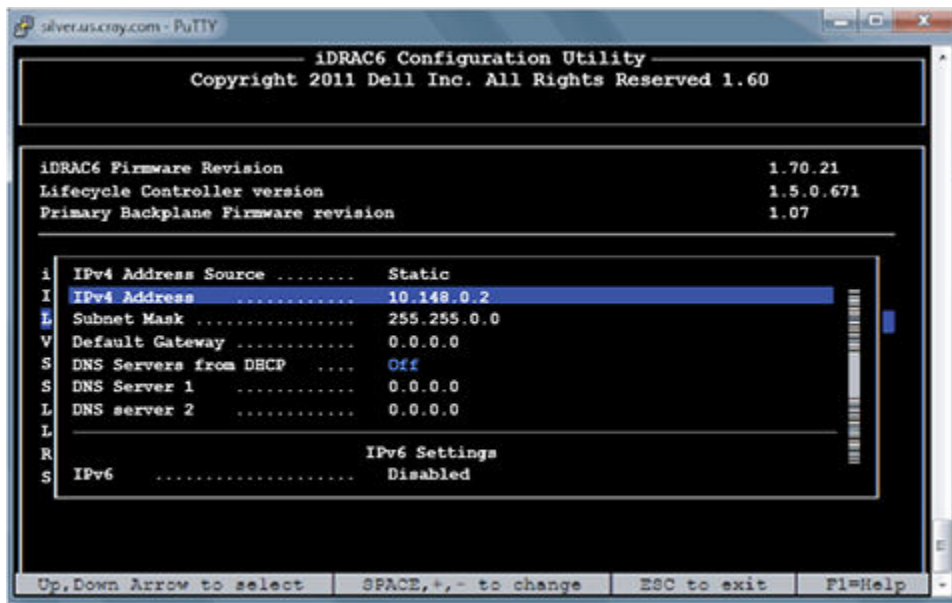
Use the arrow key to scroll down and select **Host Name String**, then press **Enter**. Enter a value for **Current Host Name String** (e.g., smw-drac), then press **Esc**.

## d. Configure IPv4 settings.

Use the arrow key to scroll down into the **IPv4 Settings** group and confirm that the **IPv4 Address Source** is set to **static**. Then enter values for the following:

- IPv4 Address** (the SMW DRAC IP address)
- Subnet Mask** (the SMW iDRAC subnet mask)
- Default Gateway** (the SMW iDRAC default gateway)
- DNS Server 1** (the first site DNS server)
- DNS Server 2** (the second site DNS server)

Figure 44. Dell R815 SMW iDRAC6 IPv4 Parameter Settings



- e. Configure IPv6 settings.

Use the arrow key to scroll down into the **IPv6 Settings** group and ensure that **IPv6** is disabled.

- f. Press **Esc** to exit **LAN Parameters** and return to the **iDRAC6 Configuration Utility** menu.

## 7. Configure iDRAC virtual media.

- a. Select **Virtual Media Configuration**, then press **Enter**.
- b. Select the **Virtual Media** line and press the space key until it indicates **Detached**.
- c. Press **Esc** to exit the **Virtual Media Configuration** menu.

## 8. Set the password for the iDRAC LAN root account.

Using the arrow keys, select **LAN User Configuration**, then press **Enter**. The following configuration is for both SSH and web browser access to the iDRAC.

- a. Select **Account User Name** and enter the account name `root`.
- b. Select **Enter Password** and enter the intended password.
- c. Select **Confirm Password** and enter the intended password again.
- d. Press **Esc** to return to the **iDRAC6 Configuration Utility** menu.

## 9. Exit the iDRAC configuration utility.

- a. Press **Esc** to exit the **iDRAC6 Configuration Utility** menu.
- b. Select **Save Changes and Exit**.

The **BIOS Boot Manager** menu appears.



## 6.10 Change the Default iDRAC Password

### About this task

This procedure describes how to log in to the iDRAC web interface and change a user password.

### Procedure

1. Bring up a web browser.
2. Go to: `https://cray-drac`, where `cray-drac` is the name of the iDRAC.  
A login screen appears.
3. Log in to the web interface as `root`.
4. Select **iDRAC settings** on the left navigation bar.
5. Expand **iDRAC settings** on the left navigation bar.
6. Select **User Authentication**.
7. Select the user whose password is changing. To change the root password, select `userid 2`.
8. Select **Next**.
9. Select the **Change Password** box and enter the new password in the boxes below it.
10. Select **Apply** to complete the password change.

The password change is complete.

**Alternative.** Another approach to changing the iDRAC root password is to use `ipmitool` on the SMW command line interface.

```
smw# ipmitool -U root -I lanplus -H drac-ip-addr -P old-drac-password \
user set password 2 new-drac-password
```

## 6.11 Enable Write Cache on SMW Boot RAID Volume

### Prerequisites

This procedure assumes the following:

- Familiarity with Cray FN5338 "Write cache support statement"
- The SANtricity Storage Manager has been installed.
- The SMW boot RAID has been configured for a NetApp, Inc. storage device.

- The user is logged on to the SMW as `crayadm`.

### About this task

To reduce risk to end-user data integrity, all SMW boot RAID storage arrays are shipped (or configured, for a fresh install) with write cache support disabled. However, there are situations in which sites may want or need to enable write cache on one or more storage volumes to better handle high write rates. ALPS requires write caching, for example, so the volume that contains the ALPS shared file system will need to have write cache enabled.

This procedure enables write cache on a single volume of an existing NetApp, Inc. storage device. Repeat this procedure for each volume to be write-cache enabled.

To mitigate the risk to data integrity, write caching should be used only with battery backup and controller mirroring.

### Procedure

1. Start the SANtricity Storage Manager.

```
crayadm@smw> /usr/bin/SMclient
```

2. Select the storage array containing the volume to be write-cache enabled.
3. Locate the volume to be write-cache enabled, right-click it, then select **Change > Cache Settings**.
4. Enable the desired cache settings.

## 6.12 Configure the SEC and check\_xt Monitoring and Notification Utilities

### About SEC

The Simple Event Correlator (SEC) is an SMW utility that parses every line being appended to system log files, watching for specific strings that represent the occurrence of significant system events. When a specified string is detected, SEC sends notification that this has happened, either by email, IRC, writing to a file, or some user-configurable combination of all three.

SEC is enabled by default, and by default is configured to generate email notifications to `crayadm`. The types of notifications generated and the recipients to whom notifications are sent are defined in the SEC configuration file, `/etc/opt/cray/cray_sec_actions_config`.

The System Management Workstation (SMW) release includes `sec-2.7.6` and an SEC support package, `cray-sec-8.0.0`. The SEC support package contains control scripts to manage the starting and stopping of SEC around a Cray mainframe boot session, in addition to other utilities and a rule set designed for Cray systems.

## About check\_xt

The SEC package includes the `check_xt` utility, which reports on state changes in the Cray system (such as system boots or compute nodes going down), and logs data about the state of nodes and jobs. `check_xt` is designed to work in conjunction with SEC to send email and text alerts about critical system issues.

Note that unlike SEC, `check_xt` is NOT enabled by default. It must be configured and called using a crontab entry.

## Configure SEC and check\_xt

For procedures to configure SEC and `check_xt`, see *XC™ Series SEC and check\_xt Software Configuration Guide* (S-2542) for release CLE 6.0.UP07.

## 6.13 Configure Site Lightweight Log Manager (LLM)

### About this task

If this site uses the Lightweight Log Manager (LLM) to send logs from the SMW to a site log host, use this procedure to update settings in the `cray_logging` configuration service in the global config set.

### Procedure

1. Update the `cray_logging` service.

```
smw# cfgset update -s cray_logging -m interactive -l advanced global
```

2. (Conditional) Configure the `site_loghost` setting, if this site has a site log host.

Selected	#	Settings
		global_options
	1)	global_log_level
	2)	raid
		<b>site_loghost</b>
	3)	name
	4)	ip_protocol
	5)	ip_port
	6)	syslog_format
...		

- a. Add the log host name.

If this system has a site loghost, select `name` (item 3 in this example) and add its name at the following prompt.

```
cray_logging.settings.site_loghost.data.name
[<cr>=set '', <new value>, ?=help, @=less] $ loghost_name
```

- b. Change other log host settings, as needed.

If this system has a site log host, and any of the remaining settings needs a value different than the default, select that setting and change its value.

3. Use the current default permissions for log files and directories, which are not backward compatible, or change permissions to maintain backward compatibility.



**WARNING:** The current default permissions for log files and directories are not backward compatible with releases older than CLE 6.0.UP06. If current permissions are used, this system will no longer be able to boot and run those older CLE 6.0 releases.

Sites must choose whether to go forward with the current default permissions and possibly use a non-default group for log ownership, or change the permissions to maintain backward compatibility and keep `crayadm` as owner of log files and directories.

- **If backward compatibility is NOT an issue**, use the current default permissions (no action needed), and use one of the following ownership groups for log files and directories:
  - `crayadm`, the default group
  - (recommended) `craylogreaders`, a pre-populated group (new in CLE 6.0.UP06) that has more restricted permissions than the `crayadm` group
  - custom group defined in `cray_global_local_users`, which is in the global config set
- **If backward compatibility is necessary**, keep `crayadm` as the ownership group for log files and directories, and change the permissions for log files and directories to values compatible with previous releases.

Note that the ownership group specified for log files and directories, whether the default or some other group, will be set as a secondary group for the `crayadm` and `postgres` users.

- a. If backward compatibility is NOT needed, use the current default permissions (no action needed), and set group ownership for log files and directories.

To use `crayadm` (the default) as the ownership group, do nothing. To use `craylogreaders` or a custom group instead, select the `group` field of the `dirs` setting (item 35 in this example) and change its value to **craylogreaders** or the name of the custom group (the custom group must be defined before it can be used).

Selected	#	Settings
...		dirs
	34)	user
	<b>35)</b>	<b>group</b>
	36)	mode
...		logs
	40)	rotation
	41)	consolidated
	42)	debugraw
	43)	debugmax
	44)	user
	<b>45)</b>	<b>group</b>
	46)	mode

```
cray_logging.settings.dirs.data.group
[<cr>=set 'crayadm', <new value>, ?=help, @=less] $ craylogreaders
```

Because the `group` field of the `logs` setting must be set to the same value, select it (item 45 in this example) and change its value to what was set for `dirs` (**craylogreaders** in this example).

```
cray_logging.settings.logs.data.group
[<cr>=set 'crayadm', <new value>, ?=help, @=less] $ craylogreaders
```

- b. If backward compatibility is needed, keep `crayadm` as the ownership group for log files and directories (no action needed), and change the permissions for log files and directories to values compatible with previous releases. Use the values shown in the examples.

Select the `mode` field of the `dirs` setting (item 36 in this example) and change its value to **0775**.

Selected	#	Settings
...		
		dirs
	34)	user
	35)	group
	<b>36)</b>	<b>mode</b>
...		
		logs
	40)	rotation
	41)	consolidated
	42)	debugraw
	43)	debugmax
	44)	user
	45)	group
	<b>46)</b>	<b>mode</b>

```
cray_logging.settings.dirs.data.mode
[<cr>=set '0771', <new value>, ?=help, @=less] $ 0775
```

Then select the `mode` field of the `logs` setting (item 46 in this example) and change its value to **0644**.

```
cray_logging.settings.logs.data.mode
[<cr>=set '0660', <new value>, ?=help, @=less] $ 0644
```

#### 4. Validate the global config set

```
smw# cfgset validate global
```

#### 5. Apply configuration changes.

Run `cray-ansible` so Ansible plays that consume config set data will apply that data to the SMW.

```
smw# /etc/init.d/cray-ansible start
```

## 6.14 Prevent Unintentional Re-creation of Mail Configuration Files

This procedure is optional. It applies to systems where postfix or sendmail are configured on the SMW.

To prevent the `master.cf` and `main.cf` postfix configuration files from being re-created during software updates or fixes, edit the `/etc/sysconfig/mail` file on the SMW and ensure that the `MAIL_CREATE_CONFIG` setting is set to "no".

```
smw# vi /etc/sysconfig/mail  
MAIL_CREATE_CONFIG="no"
```

## 6.15 Check Cabinet Cooling Parameters for an Air-Cooled XC System

### Prerequisites

- This is an XC Series air-cooled (XC-AC) system running SMW 8.x / CLE 6.x software.
- Patches have been installed.
- The system is booted.

### About this task

Cray provides the `xtaccheckcool` tool to enable sites to check and/or set the cooling parameters of an XC-AC system. Use this tool in the following circumstances:

- after a fresh install
- after a software update
- as part of customizing a preinstalled system
- after adding hardware to a system
- periodically, to verify that parameters are set correctly for an operational system

### Procedure

1. Display current and recommended cabinet cooling parameters.

Substitute the elevation (in feet) of this site.

```
smw# xtaccheckcool -e elevation_in_feet
```

2. Write the recommended cabinet cooling parameters to all cabinet controllers, as needed.

This command uses the `-w` option to write the recommended cooling parameters for the specified elevation to every cabinet in the system. Substitute the elevation (in feet) of this site.

```
smw# xtaccheckcool -w -e elevation_in_feet
```

To target a specific cabinet, use the `-t cname` option, where *cname* is the cname of the cabinet.

For more information and example output, see the `xtaccheckcool(8)` man page.

## 6.16 Make a Post-customize Snapshot using snaputil

### About this task

This procedure uses `snaputil` to make an archival snapshot of the system after customizing a preinstalled system.

**Best Practice.** Make a snapshot and back up config sets at the same time to keep them in sync. Name the snapshot and config set backup using the same suffix and date/time stamp to help administrators identify which snapshot and config set backup pairs belong together.

For more information, see [About Snapshots and Config Set Backups](#) on page 21.

### Procedure

1. List the available snapshots on the system.

```
smw# snaputil list
```

2. Set the `SNAPSHOT` environment variable using the currently booted snapshot name.

```
smw# export SNAPSHOT=$(snaputil list |grep ^cur| awk '{print $2}')
smw# echo $SNAPSHOT
```

Setting a variable for the snapshot name enables better command substitution in later commands dealing with snapshots.

This is especially important for SMW HA systems because it makes it easier to use the exact same snapshot name for both SMWs. Using different snapshots results in HSS database (MySQL) inconsistencies, which causes problems at failover.

3. Create a new snapshot.

```
smw# snaputil create ${SNAPSHOT}.postcustomize-${TODAY}
```

## 6.17 Make a Post-customize Backup of Current Global and CLE Config Sets

### About this task

This procedure uses the `cfgset` command to create a backup of the global and CLE config sets after customizing a preinstalled system.

### Procedure

1. Back up the current global config set.

```
smw# cfgset create --clone global global-postcustomize-${TODAY}
```

**2. Back up the current CLE config set.**

This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-postcustomize-${TODAY}
```



## 7 Troubleshoot SMW/CLE Software Installation

---

For extensive information about troubleshooting a Cray Linux Environment (CLE) system boot, see *XC™ Series Boot Troubleshooting Guide* (S-2565).

## 8 Miscellaneous Installation and Configuration Procedures

---

Some of the following procedures appear in this guide in the context in which they are used and are collected here for easy reference. Others do not fit within the fresh install, software update, or preinstall customization processes, but are provided here for reference as needed.

### 8.1 Back Up Site Data

This procedure helps sites identify and back up important data from the SMW, boot, and SDB nodes. Back up site data before and after installing new software, depending on circumstances and site policy.

<b>Before installation</b>	When a fresh install is performed on a system, disks are wiped clean. Before beginning any installation procedures, back up configuration files, log files, or other files that need to be preserved.
<b>After installation</b>	Sites may also want to archive important SMW and CLE information even if there are no immediate plans to install or reinstall a software release. Saving such information elsewhere will make a later reinstall easier, whether it is planned or part of disaster recovery.

What data should be saved at a particular site depends on several things, such as what is currently installed and where data is stored. A site might have CLE 5.x / SMW 7.x installed, or it might already have CLE 6.0.x / SMW 8.0.x installed and is now planning to do a fresh install and wants to reuse configuration data files. The information to save would be different in each case. And there could be site data in home directories or other parts of the file system unknown to Cray and therefore not listed here. The following suggestions about what data to preserve assume a reinstallation of a CLE 6.x / SMW 8.x release that wipes out an earlier installation of that release.

#### SMW Data to Save from a CLE 6.x / SMW 8.x release

##### SMW Configuration Data

`/var/opt/cray/imps`

Save the entire directory, which has global config sets (`/var/opt/cray/imps/global`) and CLE config sets (`/var/opt/cray/config/sets/p0`). Saving only the worksheet YAML would miss any site files added for distribution by simple sync or any site Ansible plays. Of particular importance in the global config set is `cray_bootraid_config.yaml` (or `cray_bootraid_worksheel.yaml`) which describes how the storage on the Boot RAID is being used.

<code>/etc/</code>	Save the entire directory. Information related to image recipes is stored in <code>/etc/opt/cray/imps/image_recipes.d</code> (especially any site changes to <code>image_recipes.local.json</code> ) and <code>/etc/opt/cray/imps/package_collections.d</code> .
<code>/opt/cray/hss/default/etc</code>	Save the boot automation files ( <code>/opt/cray/hss/default/etc/auto.*</code> ) and any other files with custom settings.
<code>/var/opt/cray/repos</code>	Save any site repos which have been created in this directory.
<code>/home/crayadm/*fs_defs</code>	Save this file if direct-attached Lustre (DAL) was configured.
<code>/var/adm/cray/release/pe/install-cdt.yaml</code>	Save the PE installer YAML configuration file.
Command output	Save output from these commands: <ul style="list-style-type: none"> <li>Are any nodes disabled?</li> </ul>

```
smw# xtcli status s0
```

- What are the boot and SDB nodes and are any CLE partitions present?

```
smw# xtcli part_cfg show
```

### SMW Operational Data

<code>/home</code>	Save any user data in this directory, especially in <code>/home/crayadm</code> .
<code>/var/opt/cray/disk/1</code>	Save all files in this directory, which has logs, dumps, and debugging information.
<code>/var/opt/cray/imps/image_roots</code> and <code>/var/opt/cray/imps/boot_images</code>	No need to save data in these two directories as long as the image recipes are saved, because these files can be rebuilt from the image recipes. And when they are rebuilt, they can be pushed to the boot node or CMC (for eLogin).
<code>/var/lib/mysql</code>	Perform a <code>mysql dump</code> of <code>/var/lib/mysql</code> . This data will be regenerated by rerunning <code>xtdiscover</code> .

## CLE Data to Save from a CLE 6.x / SMW 8.x release

### CLE Boot Node Data

<code>/var/opt/cray/imps</code>	No need to save the files in this directory. They are all copies of files on the SMW.
<code>/non-volatile</code> and <code>/cray_home</code>	Save the data in these two directories for possible restoration after the fresh install.

**CLE SDB Node Data**

**/alps\_shared**  
**and /var/lib/mysql**

No need to save the data in these two file systems. It will be regenerated at the first boot with the newly installed software. The only side effect is that all ALPS apids will start over at apid 100.

## 8.2 Back Up Current Global and CLE Config Sets

### About this task

Sites can back up the current global and CLE config sets as few or as many times as they deem useful. Cray recommends backing up the config sets at these software installation/configuration milestones, which correspond to the suggested milestones for making a snapshot. It is good practice to make a snapshot and back up the config set at the same time to keep them in sync. Cray also recommends naming the snapshot and config set backup using the same suffix and date/time stamp, which helps administrators identify which snapshot and backup pairs belong together.

In the example commands below, replace *suffix* with a unique suffix to distinguish among config set backups. Here is a list of suggested suffixes and their associated milestones.

<b>preupdate</b>	before beginning any software update activities (software update only)
<b>preconfig</b>	after installing a software update and before updating the global and CLE config sets (software update only)
<b>postinstall</b>	after installing a new software release (fresh install or software update) and before configuring that software
<b>postconfig</b>	after configuring SMW/CLE software and before booting the CLE system
<b>postboot</b>	after booting the CLE system
<b>postpe</b>	after installing or updating Cray PE software
<b>postcustomize</b>	after customizing a preinstalled system

### Procedure

1. Back up the current global config set.

```
smw# cfgset create --clone global global-suffix-`${TODAY}
```

2. Back up the current CLE config set.

This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset create --clone p0 p0-suffix-`${TODAY}
```

## 8.3 Back Up or Restore User, Group, and Permissions Information Files

Administrators can back up and restore the user, group, and permissions information of files contained under the `cfgset` directories using the `cfgset_export_perms.py` and `cfgset_restore_perms.py` scripts, respectively. These scripts are normally found in `/opt/cray/imps/default/etc/` and preserve the user ID (UID), group ID (GID), and file permissions of the specified directory. The default directory to be backed up or restored to is the current working directory. This is typically a `cfgset` directory in `/var/opt/cray/imps/config/sets/`. The output of `cfgset_export_perms.py` can be checked in to a source control program, such as Git, enabling a site to safely back up and restore config sets to a central source control server.

### Back Up Using `cfgset_export_perms.py`

The `cfgset_export_perms.py` script will create a backup file `cfgset_permissions_metadata` that is placed in the current working directory. Any time a change is made to a config set, `cfgset_export_perms.py` should be run to capture the addition or removal of any files and permission or ownership changes. The following example demonstrates backing up the user, group, and permissions information located in the `/var/opt/cray/imps/config/sets/p0` directory and viewing the backup file `cfgset_permissions_metadata_p0` created. Note the use of the `--output` subcommand to specify the backup file name:

```
smw# ./cfgset_export_perms.py --output
cfgset_permissions_metadata_p0 /var/opt/cray/imps/config/sets/p0
smw# ls
cfgset_export_perms.py  cfgset_restore_perms.py  cfgset_permissions_metadata
smw# vim cfgset_permissions_metadata_p0
"",0,0,"0755"
".imps_ConfigSet_metadata",0,0,"0644"
"ansible",0,0,"0755"
"dist",0,0,"0755"
"dist/compute-preload.cray",0,0,"0660"
"dist/login-preload.cray",0,0,"0660"
"dist/README",0,0,"0660"
"dist/simple_examples",0,0,"0755"
"dist/simple_examples/sample_config_tasks.yaml",0,0,"0660"
"dist/simple_examples/roles",0,0,"0755"
"dist/simple_examples/roles/example",0,0,"0755"
"dist/simple_examples/roles/example/defaults",0,0,"0755"
"dist/simple_examples/roles/example/defaults/README",0,0,"0660"
"dist/simple_examples/roles/example/files",0,0,"0755"
"dist/simple_examples/roles/example/files/README",0,0,"0660"
"dist/simple_examples/roles/example/handlers",0,0,"0755"
"dist/simple_examples/roles/example/handlers/README",0,0,"0660"
"dist/simple_examples/roles/example/meta",0,0,"0755"
...
```

### Restore Using `cfgset_restore_perms.py`

The `cfgset_restore_perms.py` script will restore the user, group, and permissions information contained in a backup file given to the script. By default, the script will search for a backup file named `cfgset_permissions_metadata` in the current working directory and restore permissions to the working directory. A different backup file to read from or a different directory to restore permissions to may optionally be specified. The following example demonstrates restoring permissions

to `/var/opt/cray/imps/config/sets/p0` using the `cfgset_permissions_metadata_p0` backup file. Note the use of the `--input` subcommand to specify the backup file to read from:

```
smw# ./cfgset_restore_perms.py --input
cfgset_permissions_metadata_p0 /var/opt/cray/imps/config/sets/global
```

## Integrate Scripts as Git Hooks

The `cfgset_export_perms.py` can be specified as a pre-commit hook. The `cfgset_restore_perms.py` script can be specified as a post-merge or post-checkout hook. It is recommended that a separate Git repo be configured for each config set. To establish a Git repo in the `cfgset` directory, follow the steps below:

1. If Git is not currently installed, install Git using Zypper:

```
# zypper in git
```

2. Establish the Git directory in each config set directory that is to be backed up:

```
smw# cd /var/opt/cray/imps/config/sets/p0
smw# git init
smw# git add -A
smw# git commit -m "Initial check-in of P0 config set"
```

3. Optionally, establish any site-specific Git remotes.

To designate either script as a Git hook, follow these steps:

1. Copy the desired script(s) to the `.git/hooks` directory of the Git repository.
2. To use `cfgset_export_perms.py` as a pre-commit hook, rename the copied script to: `pre-commit`. To use `cfgset_restore_perms.py` as a post-checkout or post-merge hook, rename the copied script(s) to either: `post-checkout` or `post-merge`. It is suggested that `cfgset_restore_perms.py` be installed as both a post-checkout and post-merge hook, as the post-checkout commit hook is only called when doing a `git checkout` operation. Likewise, the post-merge commit hook is called only when performing a `git pull` or similar operation. Note that to use `cfgset_restore_perms.py` as both a post-checkout and post-merge script, the script must be copied to the `.git/hooks` directory twice, with one copied script renamed as `post-checkout` and the other copied script renamed as `post-merge`.
3. Ensure all scripts are executable.

## 8.4 Set Default Config Set for a NIMS Map

### Prerequisites

This procedure assumes that a NIMS map has been created.

### About this task

Every NIMS map has a default config set, which is the config set that all of the nodes in the map will use if the nodes have not been specifically set to use a different config set. When a NIMS map is created, the default config set can be specified using the `--config-set` option. If the default config set is not specified, then it defaults to the name of the partition associated with that NIMS map. For an unpartitioned system, the default config set

would be 'p0,' while for a partitioned system, the default config set for the map associated with partition 1 would be 'p1,' and the default config set for the map associated with partition 2 would be 'p2.'

## Procedure

1. Set the default config set for a NIMS map.

```
smw# cmap update -s config_set_name map_name
```

2. Verify that the NIMS map has the specified default config set.

```
smw# cmap list --fields default_config_set map_name
```

## 8.5 Set Config Set for a Node

### About this task

This procedure sets a config set for an individual node and then verifies the setting. The node will use the specified config set instead of the default config set of the active NIMS map.

## Procedure

1. Set the config set for an individual node.

```
smw# cnode update --set-config-set config_set_name cname
```

2. Verify that the node is set to use the specified config set.

```
smw# cnode list --fields config_set cname
```

## 8.6 Rename a NIMS Map

### About this task

The `cmap` command, which is used to create and manage NIMS maps, does not have a "rename" capability. To achieve the same result, create a new NIMS map with the desired name, which will replace the NIMS map to be renamed (the "old" NIMS map), and then delete the old NIMS map.

## Procedure

1. Create a new NIMS map with the desired name.

If no data has changed in the old NIMS map (no `cnode` commands have been performed), then simply create a new NIMS map with the desired name.

```
smw# cmap create <new_NIMS_map_name>
```

If data may have been changed in the old NIMS map, then clone the old NIMS map, naming the clone the desired name.

```
smw# cmap create --clone <old_NIMS_map_name> <new_NIMS_map_name>
```

**Trouble?** The `cmap` command will first verify that the CLE config set associated with the NIMS map exists. If it does not exist, the command will fail with an error message to that effect.

- If the config set is expected to be missing (for example, during an installation when the CLE config set has not yet been created), then repeat the `cmap create` command with the `--no-verify` flag.
- If the config set is NOT expected to be missing, then create/locate the missing config set, set it as the default config set for the NIMS map, and repeat the `cmap create` command.

2. Delete the old NIMS map that is being replaced by the new one just created.

```
smw# cmap delete <old_NIMS_map_name>
```

## 8.7 Modify a Config Set for Use with Advanced Authentication Configurations

The examples given here show various configurations that enable an XC system to connect to an Active Directory system. Authentication configuration is often implementation-specific, and the examples here should only be used as a reference for modifying a config set for use with an authentication method other than the default LDAP setup.

**NOTE:** Because of possible boot failure due to an invalid or misconfigured config set, it is recommended to run Ansible and examine the output for improper values once the config set has been modified. Improper values may prevent a system from booting. The following process is recommended when modifying authentication systems on a config set:

1. Boot system
2. Modify config
3. Run Ansible to ensure proper values and configuration
4. If improper values are returned, revise the config until proper values are attained
5. Reboot the system only after ensuring that proper values are returned.

### Use Jinja Substitution and Variables in Config Files

In rare cases it may be necessary to dynamically determine values. For these instances, variable substitution can be used to manipulate config values depending on whether certain conditions are met. Variable statements follow Jinja templating: items in double quotes are treated as literals, while non-quoted items are variables. Begin a variable statement with `{{` and end with `}}`. Ansible variables and facts, as well as config set values, can be used in a variable statement. In this example, variable substitution is used to manipulate `auth_provider.value`:

```
cray_auth.settings.common_ldap_options.data.auth_provider.value: '{{ 'none' if
ansible_local.cray_system.platform
== 'compute' else 'krb5' }}
```



## Advanced Authentication Settings: Example 1

This example config shows a system set up for an Active Directory server. In this case, the Active Directory server is a Windows Domain server. Depending on the desired debugger verbosity, a debug level may be set for each config set section. If no `debug_level` is specified, no debugging into will be collected. The `/etc/sss.conf` file is displayed below:

```
fireball:/etc/sss # more sssd.conf
[sss]
config_file_version = 2
services = nss, pam
domains = crayit
debug_level = 7

[nss]
filter_users = root, crayadm
filter_groups = root
debug_level = 7

[pam]
debug_level = 7

[domain/crayit]
override_shell = /bin/bash
override_homedir = /cray_home/%u
ldap_search_base=dc=crayit,dc=com
debug_level = 7
krb5_realm=CRAYIT.COM
krb5_server=crayadserver.us.cray.com
# following allows the system to bind to ldap server via a user/password
ldap_default_authtok=mypassword
ldap_default_authtok_type=password
ldap_default_bind_dn=CN=SLES Test,OU=Users,OU=Cray Objects,DC=crayit,DC=com
#
ldap_group_object_class=group
ldap_user_object_class=user
ldap_uri = ldap://crayadserver.us.cray.com/
id_provider = ldap
auth_provider = krb5
ldap_schema = rfc2307bis

[ssh]
debug_level = 7

[sudo]
```

The corresponding `/etc/nsswitch.conf` file is displayed below. Note that the entries for each key represent the corresponding location a value is first searched for (in this case, `files` is searched first followed by `sss`).

```
fireball:/etc/sss # grep -v ^# /etc/nsswitch.conf | sort -u
passwd: files sss
group: files sss
hosts: files dns
networks:      files dns
services: files sss
protocols:     files
rpc:           files
ethers:        files
netmasks:     files
```

The corresponding `/etc/security/access.conf` file:

A sample test run demonstrating Active Directory authentication using the above sample configuration settings:

```
user@user-lnx:~$ ssh -l sles-test
fireball
```

```
*** Welcome to IMPS service node c0-0c0s1n0 (nid 4) ***
Running 115MB Suse 12 image login-large_cle_rhine_sles_12_x86-64_ari
CLE release 6.0.UP03, build 6.0.3019(201611291150)
36 vcores, boot freemem: 62964mb
```

```
dal_nodes | c0-0c0s1n3
rsip_servers | c0-0c0s1n0
lnet_routers | c0-0c0s1n1
tier2_nodes | c0-0c0s1n2
login_nodes | c0-0c0s1n0
sdb_nodes | c0-0c0s0n2
boot_nodes | c0-0c0s0n1
sles-test@fireball:~> id
uid=10000(sles-test) gid=20000(SLES) groups=20000(SLES)
sles-test@fireball:~> df -k .
Filesystem                1K-blocks    Used Available Use% Mounted on
boot-fireball:/cray_home  52428800  954368   49371136    2% /cray_home
sles-test@fireball:~> pwd
/cray_home/sles-test
sles-test@fireball:~> klist
Ticket cache: FILE:/tmp/krb5cc_10000_lwBoeA
Default principal: sles-test@CRAYIT.COM

Valid starting            Expires                Service principal
11/30/2016 12:47:28      11/30/2016 22:47:28  krbtgt/CRAYIT.COM@CRAYIT.COM
```



```

cray_auth.settings.common_nis_options.data.option.proxy_lib_name: null
cray_auth.settings.common_nis_options.data.proxy_lib_name.value: nis
cray_auth.settings.common_nis_options.data.option.proxy_pam_target: null
cray_auth.settings.common_nis_options.data.proxy_pam_target.value: sssdnisproxy
cray_auth.settings.domain.data.reference.crayit: null
cray_auth.settings.domain.data.crayit.servers:
- ldap://crayadserver.us.cray.com/
cray_auth.settings.domain.data.crayit.schema: rfc2307bis
cray_auth.settings.domain.data.crayit.aux_settings:
- ldap_user_object_class=user
- ldap_group_object_class=group
- ldap_default_bind_dn=CN=SLES Test,OU=Users,OU=Cray Objects,DC=crayit,DC=com
- ldap_default_authtok_type=password
- ldap_default_authtok=mypassword
- krb5_server=crayadserver.us.cray.com
- krb5_realm=CRAYIT.COM
- debug_level = 7
- ldap_search_base=dc=crayit,dc=com
cray_auth.settings.access.data.policy:
- +:root:LOCAL
- +:crayadm:LOCAL
- +:@admin_grp:ALL
- +:@dev_users:ALL
cray_auth.settings.access.data.restrictive: false
cray_auth.settings.access.data.config_computes: true
cray_auth.settings.access.data.config_id_service_groups: []
cray_auth.settings.section.data.section_name.sssd: null
cray_auth.settings.section.data.sssd.options.option_name.debug_level: null
cray_auth.settings.section.data.sssd.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.nss: null
cray_auth.settings.section.data.nss.options.option_name.debug_level: null
cray_auth.settings.section.data.nss.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.ssh: null
cray_auth.settings.section.data.ssh.options.option_name.debug_level: null
cray_auth.settings.section.data.ssh.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.sudo: null
cray_auth.settings.section.data.sudo.options.option_name.debug_level: null
cray_auth.settings.section.data.sudo.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.pam: null
cray_auth.settings.section.data.pam.options.option_name.debug_level: null
cray_auth.settings.section.data.pam.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.autofs: null
cray_auth.settings.section.data.autofs.options.option_name.debug_level: null
cray_auth.settings.section.data.autofs.options.debug_level.value: '7'
cray_auth.config_file: /var/opt/cray/imps/config/sets/p0.integration/config/
cray_auth_config.yaml
cray_auth.configurator.version_template: 1.2.1
cray_auth.configurator.version_product: 0.0.0
cray_auth.configurator.version_configurator: 3.0.0
cray_auth.configurator.template_type:
- cle
fireball-smw:/var/opt/cray/imps/config/sets/p0.integration/worksheets #

```

## Advanced Authentication Settings: Example 2

Example 2 demonstrates an alternative configuration method for using Active Directory. In this example, a `/etc/krb5.keytab` file has been generated utilizing a temporary Samba configuration and a user account that has permissions to join the host to the domain. The `/etc/krb5.keytab` file was put into place on the node via Simple Sync. Because in this example there is no keytab file each computer node, `config_computes` must

be set to false. This example uses the same `/etc/security/access.conf` and `/etc/nsswitch.conf` files as Example 1. The `sssd.conf` file is displayed below:

```
===== /etc/sss/sss.conf =====
```

```
[sss]
debug_level = 7
config_file_version = 2
services = nss, pam
domains = crayit

[nss]
debug_level = 7
filter_users = root, crayadm
filter_groups = root

[domain/crayit]
debug_level = 7
id_provider = ad
auth_provider = ad
access_provider = ad
chpass_provider=ad
ad_server = crayadserver.us.cray.com
ad_domain = CRAYIT.COM
default_shell = /bin/bash
#fallback_homedir = /home/%d/%u
use_fully_qualified_names = False
dyn dns_update = false

[ssh]
debug_level = 7

[sudo]
debug_level = 7

[pam]
debug_level = 7

[autofs]
debug_level = 7
```

The corresponding config set:

```
cray_auth.enabled: true
cray_auth.settings.nsswitch_sources.data.database.automount: null
cray_auth.settings.nsswitch_sources.data.automount.sources: files sss
cray_auth.settings.nsswitch_sources.data.database.group: null
cray_auth.settings.nsswitch_sources.data.group.sources: files sss
cray_auth.settings.nsswitch_sources.data.database.hosts: null
cray_auth.settings.nsswitch_sources.data.hosts.sources: files dns
cray_auth.settings.nsswitch_sources.data.database.netgroup: null
cray_auth.settings.nsswitch_sources.data.netgroup.sources: files sss
cray_auth.settings.nsswitch_sources.data.database.passwd: null
cray_auth.settings.nsswitch_sources.data.passwd.sources: files sss
cray_auth.settings.nsswitch_sources.data.database.services: null
cray_auth.settings.nsswitch_sources.data.services.sources: files sss
cray_auth.settings.nsswitch_sources.data.database.shadow: null
cray_auth.settings.nsswitch_sources.data.shadow.sources: files sss
cray_auth.settings.common_ldap_options.data.option.id_provider: null
cray_auth.settings.common_ldap_options.data.id_provider.value: ad
cray_auth.settings.common_ldap_options.data.option.auth_provider: null
```

```

cray_auth.settings.common_ldap_options.data.auth_provider.value: ad
cray_auth.settings.common_nis_options.data.option.id_provider: null
cray_auth.settings.common_nis_options.data.id_provider.value: proxy
cray_auth.settings.common_nis_options.data.option.auth_provider: null
cray_auth.settings.common_nis_options.data.auth_provider.value: proxy
cray_auth.settings.common_nis_options.data.option.proxy_lib_name: null
cray_auth.settings.common_nis_options.data.proxy_lib_name.value: nis
cray_auth.settings.common_nis_options.data.option.proxy_pam_target: null
cray_auth.settings.common_nis_options.data.proxy_pam_target.value: sssdnisproxy
cray_auth.settings.domain.data.reference.crayit: null
cray_auth.settings.domain.data.crayit.servers: []
cray_auth.settings.domain.data.crayit.schema: ''
cray_auth.settings.domain.data.crayit.aux_settings:
- access_provider = ad
- chpass_provider = ad
- ad_server = crayadserver.us.cray.com
- ad_domain = CRAYIT.COM
- default_shell = /bin/bash
- use_fully_qualified_names = False
- dyndns_update = false
- override_homedir = /cray_home/%u
cray_auth.settings.access.data.policy:
- +:root:LOCAL
- +:crayadm:LOCAL
- +:@admin_grp:ALL
- +:@dev_users:ALL
cray_auth.settings.access.data.restrictive: false
cray_auth.settings.access.data.config_computes: false
cray_auth.settings.access.data.config_id_service_groups: []
cray_auth.settings.section.data.section_name.sssd: null
cray_auth.settings.section.data.sssd.options.option_name.debug_level: null
cray_auth.settings.section.data.sssd.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.nss: null
cray_auth.settings.section.data.nss.options.option_name.debug_level: null
cray_auth.settings.section.data.nss.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.ssh: null
cray_auth.settings.section.data.ssh.options.option_name.debug_level: null
cray_auth.settings.section.data.ssh.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.sudo: null
cray_auth.settings.section.data.sudo.options.option_name.debug_level: null
cray_auth.settings.section.data.sudo.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.pam: null
cray_auth.settings.section.data.pam.options.option_name.debug_level: null
cray_auth.settings.section.data.pam.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.autofs: null
cray_auth.settings.section.data.autofs.options.option_name.debug_level: null
cray_auth.settings.section.data.autofs.options.debug_level.value: '7'
cray_auth.settings.section.data.section_name.domain/crayit: null
cray_auth.settings.section.data.domain/crayit.options.option_name.debug_level: null
cray_auth.settings.section.data.domain/crayit.options.debug_level.value: '7'
cray_auth.config_file: /var/opt/cray/imps/config/sets/p0.integration/config/
cray_auth_config.yaml
cray_auth.configurator.version_template: 1.2.1
cray_auth.configurator.version_product: 1.0.0
cray_auth.configurator.version_configurator: 3.0.0
cray_auth.configurator.template_type:
- cle

```

Sample output demonstrating authentication functionality:



- Use the `zypper` command to install software on a node. Software installed with this method is lost the next time the node is booted. Like the `image chroot` method, this approach can be used when testing software that does not need to persist in the image.

**IMPORTANT:** Do not directly modify a Cray-provided recipe. For information on removing an RPM after building from a Cray-provided recipe, see [Remove an Undesired RPM After Building From a Cray Recipe](#) on page 425.

This procedure describes the recommended method of creating a new image recipe for third-party software that will run on Cray nodes. The procedure explains how to add a Cray-provided image recipe as a subrecipe, then add the third-party repositories, package collections, and RPMs, as well as optional non-RPM content. It then shows how to build an image root, export the image root into a boot image, push the boot image to the boot node (netroot only), test it on a single node, and assign the tested image to all applicable nodes.

For more information on image-related concepts and commands, see "About the Image Management and Provisioning System (IMPS)" in the *XC™ Series System Administration Guide (S-2393)*.

## Procedure

### CREATE REPOSITORY

1. Create a new repository and add the third-party packages (RPMs). Skip this step if the repository already exists on the SMW or is hosted on a remote repository server.

- a. Use the `repo create` command to create the new repository (for example, `my_sles12_repo`).

This command requires the operating system distribution (for example, SLES12, RHEL6, CentOS).

```
smw# repo create --dist SLES12 my_sles12_repo
```

- b. Verify that the new repository was created.

```
smw# repo list my*
my_sles12_repo
```

- c. Add the third-party RPMs to the repository. This example takes all RPMs starting with `myrpm` in the example repository path `/path/to/repos/` and copies them to the example repo `my_sles12_repo`.

```
smw# repo update -a "/path/to/repos/myrpm*.rpm" my_sles12_repo
smw# ls -l /var/opt/cray/repos/my_sles12_repo
-rw-r--r-- 1 crayadm crayadm 485137 Nov 23 08:56 myrpm-11.13.1.1-4.x86_64.rpm
```

- d. (Optional) Check the contents of the repository. This command displays the packages but not the full RPM names.

```
smw# repo show --fields contents
```

Add the `--detailed` option to display the version and architecture for each package in the repository.

- e. Validate the repository.

```
smw# repo validate my_sles12_repo
```

### CREATE PACKAGE COLLECTION

2. Create a package collection and add the RPM package names.



A package collection represents a logical grouping of RPMs. Cray recommends using a package collection because the RPMs can be used in multiple image types (such as compute and service node images). Package collections are stored on the SMW in `/etc/opt/cray/imps/package_collections.d/`.

Cray provides the following package collections for workload manager (WLM) software:

- `service-pbs_cle_6.0.up07_sles_12sp3` (packages needed to build and run PBS)
- `service-torque_cle_6.0.up07_sles_12sp3` (packages needed to build and run Moab/TORQUE)
- `slurm-build_cle_6.0.up07_sles_12sp3` (packages needed to build Slurm)
- `compute-slurm_cle_6.0.up07_sles_12sp3` (packages needed to run Slurm on compute nodes)
- `login-slurm_cle_6.0.up07_sles_12sp3` (packages needed to run Slurm on login nodes)
- `service-slurm_cle_6.0.up07_sles_12sp3` (packages needed to run Slurm on service nodes)

- a. Create an empty package collection (for example, `my_collection`).

```
smw# pkgcoll create --description "Example package collection" my_collection
```

- b. Verify that the package collection was created.

```
smw# pkgcoll list my*
my_collection
```

- c. Add the RPM package name or names (for example, `myrpm`) to the package collection.

**IMPORTANT:** When adding an RPM package to a package collection, use the file name of the RPM without the `.rpm` extension. Otherwise, the package will not install in the image root, even though the package collection may validate.

```
smw# pkgcoll update -p myrpm \
--description "My package collection" my_collection
```

- d. Display information about the package collection.

```
smw# pkgcoll show my_collection
my_collection:
  name: my_collection
  description: My package collection
  packages:
    myrpm
```

- e. Validate the package collection.

This example assumes that `my_collection` is for the x86-64 architecture. Use `--arch aarch64` if the package collection is for that architecture.

```
smw# pkgcoll --arch x86_64 validate my_collection
```

## CREATE RECIPE

3. Create a new recipe and customize it by adding a subrecipe (the Cray-provided image) and the content for the third-party software.

- a. List the existing recipes to determine which image recipe to include.

```
smw# recipe list
compute-large_cle_6.0up03_sles_12_x86-64_ari
compute-large_cle_6.0up04_sles_12sp2_x86-64_ari
```

```

compute-large_cle_6.0up05_sles_12sp3_x86-64_ari
compute-large_cle_6.0up06_sles_12sp3_ari
compute-large_cle_6.0up07_sles_12sp3_ari
compute_cle_6.0up03_sles_12_x86-64_ari
compute_cle_6.0up04_sles_12sp2_x86-64_ari
compute_cle_6.0up05_sles_12sp3_x86-64_ari
compute_cle_6.0up06_sles_12sp3_ari
compute_cle_6.0up07_sles_12sp3_ari
dal_cle_6.0up03_centos_6.5_x86-64_ari
dal_cle_6.0up04_centos_6.5_x86-64_ari
dal_cle_6.0up05_centos_6.5_x86-64_ari
dal_cle_6.0up06_centos_6.5_ari
dal_cle_6.0up07_centos_6.5_ari
ellogin_cle_6.0up06_sles_12sp3_ari
ellogin_cle_6.0up07_sles_12sp3_ari
...

```

- b. Create a new image recipe. This example uses the recipe name *site\_compute*.

```

smw# recipe create --description \
"Example recipe for 3rd-party software on compute nodes" site_compute

```

- c. Add the existing image recipe as a subrecipe. This example uses the Cray-provided recipe *compute\_cle\_6.0up07\_sles\_12sp3\_ari*.

```

smw# recipe update -i compute_cle_6.0up07_sles_12sp3_ari site_compute

```

- d. Add the package collection that contains the third-party RPMs (in this example, *my\_collection*).

```

smw# recipe update -c my_collection \
--rationale "Include my package collection" site_compute

```

- e. Add the repository that contains the third-party RPMs (for example, *my\_sles12\_repo*).

```

smw# recipe update -r my_sles12_repo \
--rationale "Include third-party RPMs" site_compute

```

To add a remote repository that is hosted on an external repository server, specify the repository's Uniform Resource Identifier (URI) starting with `http://` or `https://`.

- f. Add the objects mentioned in the subrecipe that are also needed for the parent recipe.

**IMPORTANT:** The objects mentioned in a subrecipe are used to build that subrecipe but are not available to the parent recipe. If a package (RPM) or package collection is specified in the parent recipe, the custom recipe must explicitly contain the set of repositories where the packages can be found.

1. Determine which repository contains the necessary RPM or RPMs. This example `find` command identifies the Cray repository that contains the RPM `otherrpm`.

```

smw# find /var/opt/cray/repos -name otherrpm\* -ls

```

2. Select the correct repository:

- Choose the repository for the image's operating system distribution — use a SLES repository for a SLES image recipe; use a CentOS repository for a CentOS recipe.
- Most operating system and Cray repositories come in pairs (base and updates), such as `sles_12sp3` and `sles_12sp3_updates`. Be sure to select both the *base* and *base\_updates* repositories if they exist, because RPMs in update repositories will have higher version RPMs that will be used instead of lower version RPMs contained in base repos.

3. Add the required repository or repositories (in this example, `otherrepo`).

```
smw# recipe update -r otherrepo \
--rationale "Additional repo for third-party software" site_compute
```

Repeat the `-r` option to add multiple repositories, such as a `base` and `base_updates` repository pair.

```
smw# recipe update -r sles_12 \
-r sle-server_12sp3 -r sle-server_12sp3_updates \
--rationale "SLES12 update repo" site_compute
```

- g. (Optional) Add post-build actions by manually editing the image recipe in `/etc/opt/cray/imps/image_recipes.d/image_recipes.local.json`.

Post-build actions can add non-RPM content (files or directories) to the image or specify commands to run in the chroot environment of the image root (on the SMW). For example, the post-build actions could include copying a tar file into the image, then using `chroot` to run the commands to untar it and run an install script.

- In the `postbuild_chroot` section, add the commands to run in a `chroot` environment for this image root.
- In the `postbuild_copy` section, add the files to copy into the image.

```
smw# vi /etc/opt/cray/imps/image_recipes.d/image_recipes.local.json
```

```
"site_compute": {
  "description": "Example recipe for 3rd-party software on compute nodes",
  "dist": "SLES12",
  "default-arch": "x86_64",
  "valid-arch": [
    "x86_64",
    "aarch64"
  ],
  "packages": [ ... ],
  "package_collections": [ ... ],
  "recipes": [ ... ],
  "repositories": [ ... ],
  "postbuild_chroot": [
    "common_command_1",
    "common_command_2",
    "{% if arch == 'aarch64' %}aarch64_command_1{% endif %}",
    "{% if arch == 'x86_64' %}x86_64_command_1{% endif %}",
    "common_command_3",
    "common_command_4",
    ...
  ],
  "postbuild_copyfiles": [
    "/path/to/file/copyfile_1",
    "/path/to/file/copyfile_2",
    "{% if arch == 'aarch64' %}/path/to/file/aarch64_copyfile_1{% endif %}",
    "{% if arch == 'aarch64' %}/path/to/file/aarch64_copyfile_2{% endif %}",
    "/path/to/file/copyfile_3",
    "/path/to/file/copyfile_4",
    ...
  ],
  "version": "2.0.0",
  "metadata": {
    "created": "2018-05-31T13:24:14"
    "history": [
      "2018-05-31T13:26:01: Extended recipes attribute with 1 Recipe."
      ...
    ]
  }
}
```

While editing the recipe, do not delete or change the version or metadata fields.

**TIP:** Post-build scripts can use the following environmental variables:

- IMPS\_IMAGE\_NAME
- IMPS\_VERSION
- IMPS\_IMAGE\_RECIPE\_NAME
- IMPS\_POSTBUILD\_FILES

h. Validate the image recipe.

```
smw# recipe validate site_compute
INFO - Validating recipe site_compute is valid for x86_64 architecture.
INFO - Validating Image 'site_compute-validate-2018-05-31_13:31:44'
...
INFO - Building out site_compute
INFO - Calling package manager to validate Recipe 'site_compute'; this can
take a few minutes.
INFO - Removed Image 'site_compute-validate-2018-05-31_13:31:44'.
INFO - Removed Image 'site_compute-validate-2018-05-31_13:31:44'.
INFO - Recipe validates.
```

This command checks that the JSON syntax of the image recipe is correct. It also validates, for the specified architecture, all repositories and package collections referenced by the image recipe, checks that all required packages are included in the recipe, and ensures that it can access any files in the `postbuild_copy` section.

**Caveat.** Recipe validation does NOT validate post-build activities, such as running scripts and copyfiles actions, because without actually installing packages, the scripts/actions cannot be run.

## BUILD AND PACKAGE IMAGE

4. Build the image recipe to create the image root.

Choose a unique name for the image root. Cray recommends using the image recipe name plus the current date/time. This example uses the image root name `site_compute_timestamp`.

**IMPORTANT:** If the image root name is not unique, it will overwrite an existing image root. A unique name is especially important for images that are pushed to the boot node. Do not overwrite the image root that is currently used by running nodes.

The `image create` command builds the image recipe starting with the package manager installation and then proceeds to step through the post-build copy and post-build `chroot` commands (in that order).

If the image to be created should have a different architecture than the recipe's default architecture, add the `--arch ARCH` to the following command, where `ARCH` is one of the valid architectures, such as `x86_64` or `aarch64`.

```
smw# image create -r site_compute site_compute_timestamp
INFO - Repository 'my_sles12_repo' validates.
INFO - Recipe 'site_compute' is valid for building.
INFO - Calling Package manager to build new image root; this will take a few
minutes.
INFO - Rebuilding RPM database for Image 'site_compute_timestamp'.
INFO - RPM database does not need to be rebuilt.
INFO - Running post-build scripts for Image 'site_compute_timestamp'.
INFO - Copying postbuild files to /tmp/tmpmAYzG1 in Image
'site_compute_timestamp'
INFO - * Executing post-build chroot script: 'common_command1'
INFO - post-build chroot script output will be located in /tmp/
```

```
site_compute_postbuild_out_20180513-15:55:11g4WA6p
INFO - Build of Recipe 'site_compute' has completed successfully.
```

## 5. (Optional) Display the build history of the image root.

```
smw# image show site_compute_20180521082046
site_compute_20180521082046:
  name: site_compute_20180521082046
  arch: x86_64
  dist: SLES12
  created: 2018-05-21T08:20:46
  history:
    2018-05-21T08:20:57: Successful build of Recipe 'seed_common_6.0.up07_sles_12sp3' into Image
'site_compute_20180521082046'.
    2018-05-21T08:22:53: Successful build of top level recipe 'compute_cle_6.0.up07_sles_12sp3_ari'.
    2018-05-21T08:22:53: Successful rebuild of RPM database.
  path: /var/opt/cray/imps/image_roots/site_compute_20180521082046
```

## 6. Package the image root into a boot image.

```
smw# image export site_compute_timestamp

INFO - Copying kernel /var/opt/cray/imps/image_roots/site_compute_timestamp/boot/
bzImage-3.12.28-4.6_1.0000.8685-cray_ari_c into /tmp/temp_tempfs_50LJ93/DEFAULT
INFO - Copying parameters file /var/opt/cray/imps/image_roots/site_compute_timestamp/
boot/parameters-ari_c into /tmp/temp_tempfs_50LJ93/DEFAULT
.
.
.
INFO - Image 'site_compute_timestamp' has been packaged into /var/opt/cray/imps/
boot_images/site_compute_timestamp.cpio.
```

## 7. If this is a netroot image, push the image root to the boot node.

**IMPORTANT:** Before pushing the image root, make sure that there is sufficient space on the boot node in `/var/opt/cray/imps/image_roots`.

```
smw# image sqpush site_compute_timestamp --destination boot
```

The `image sqpush` command puts a SquashFS compressed image root on the boot node. Cray recommends using this command instead of `image push` for better boot performance. For more information, see [About Image Pushes: push versus sqpush](#) on page 42.

### TEST IMAGE

## 8. Test the new boot image on a single node.

### a. Assign the boot image to a node with the NIMS `cnode` command.

The `cnode` and `cmap` commands replace the `nimscli` command, which was deprecated in CLE 6.0.UP04 and removed in CLE 6.0.UP05. Be sure to change any scripts that reference `nimscli`.

This example assigns the boot image file `site_compute_timestamp.cpio` (in the directory `/var/opt/cray/imps/boot_images/`) to the compute node with the `cname` `c0-0c0s15n3`.

- For a tmpfs image:

```
smw# cnode update -i \
/var/opt/cray/imps/boot_images/site_compute_timestamp.cpio c0-0c0s15n3
```

- For a netroot image:

```
smw# cnode update c0-0c0s15n3 \
--set-parameter netroot=site_compute_timestamp
```

- b. Warm-boot the node to test the boot image.

```
smw# xtcli shutdown c0-0c0s15n3
.
.
.
crayadm@smw> xtbootsys --reboot \
-r "testing new boot image site_compute_timestamp" c0-0c0s15n3
```

## ASSIGN IMAGE TO NODES

9. Change the NIMS map to assign the new image to the applicable nodes.

- a. Back up the current map before changing to the new image. First, identify the active map.

```
smw# cmap list | grep -i 'true'
```

The following steps use the active map name "*current-map*".

- b. Next, clone the current map.

```
smw# cmap create --clone current-map new-map
```

**Trouble?** The `cmap` command will first verify that the CLE config set associated with the NIMS map exists. If it does not exist, the command will fail with an error message to that effect.

- If the config set is expected to be missing (for example, during an installation when the CLE config set has not yet been created), then repeat the `cmap create` command with the `--no-verify` flag.
- If the config set is NOT expected to be missing, then create/locate the missing config set, set it as the default config set for the NIMS map, and repeat the `cmap create` command.

- c. Mark the new map as the active map.

```
smw# cmap setactive new-map
```

- d. Assign the new boot image to all applicable nodes. This example uses "`--group compute`" to assign the image to all compute nodes.

- For a tmpfs image:

```
smw# cnode update --group compute \
-i /var/opt/cray/imps/boot_images/site_compute_timestamp.cpio
```

- For a netroot image:

```
smw# cnode update --group compute \
--set-parameter netroot=site_compute_timestamp
```

**Trouble?** If problems occur, use this command to revert to the previous map (*current-map*):

```
smw# cmap setactive current-map
```

10. Choose when the nodes should switch to the new image.

- To immediately use the new image, warm-boot all applicable nodes with the new image. This example specifies the compute nodes as a comma-separated list of `cnames`; see the `xtcli(8)` man page for other ways of specifying multiple nodes.

```
smw# xtcli shutdown cname, cname, ... cname
.
.
.

smw# xtbootsys --reboot -r "Booting custom image on all compute nodes" \
cname, cname, ... cname
```

- To have the workload manager (WLM) reboot the node once the current user's job finishes, see "Apply Rolling Patches to Compute Nodes with `cnat`" in the *XC™ Series System Administration Guide (S-2393)*.
- Otherwise, wait until the next full system reboot. The nodes will boot with the new image.

After a recipe has been defined and tested, the `imgbuilder` command can be used to rebuild and package boot images.

## 8.9 Remove an Undesired RPM After Building From a Cray Recipe

### About this task

Sites can create their own new recipe and model it after a Cray recipe by including the Cray recipe as a subrecipe. Using a Cray-provided recipe to build an image root will have the set of RPMs Cray requires. If one or more of those RPMs are not needed for the site recipe, the site recipe can be augmented with `zypper` or `RPM` commands to explicitly erase an RPM by using the `POSTBUILD_CHROOT` method. One advantage of this method is that any Cray changes delivered in patches that require a rebuild of the recipe will be included in the recipe, while still using the site modifications in the site recipe to remove unneeded RPMs after the Cray subrecipe has been built.

### Procedure

1. Create a new recipe that will be similar to the login-large recipe.

```
smw# recipe create site-login-large_cle_6.0.up07_sles_12sp3_ari
```

2. Add the Cray login-large recipe as a subrecipe.

```
smw# recipe update -i login-large_cle_6.0.up07_sles_12sp3_ari \
site-login-large_cle_6.0.up07_sles_12sp3_ari
```

3. Edit this new site recipe to add the desired `POSTBUILD_CHROOT` steps.

This example calls `zypper` to remove three RPMs called `badrpm1`, `badrpm2`, and `badrpm3`. While editing the recipe, do not delete or change the version or metadata fields.

```
smw# vi /etc/opt/cray/imps/image_recipes.d/image_recipes.local.json
```

```
"site-login-large_cle_6.0.up07_sles_12sp3_ari": {
  "description": "Site login node image",
  "dist": "SLES12",
  "default-arch": "x86_64",
  "valid-arch": [
    "x86_64",
    "aarch64"
  ],
  "packages": [],
  "package_collections": [],
  "recipes": [
    {"name": "login-large_cle_6.0.up07_sles_12sp3_ari", "rationale": ""}
  ],
  "repositories": [],
  "postbuild_chroot": [
    "zypper remove badrpm1 badrpm2 badrpm3"
  ],
  "postbuild_copyfiles": [],
  "version": "2.0.0",
  "metadata": {
    "created": "2018-05-17T13:24:14"
    "history": [
      "2018-05-17T13:26:01: Extended recipes attribute with 1 Recipe."
      ...
    ]
  }
}
```

4. Build the new site recipe into a uniquely named image root using the `image` command to ensure a clean build.

Attempting to remove RPMs may determine that other RPMs have dependencies on the RPMs to be removed. Several iterations may be required to determine the proper set of RPMs to be removed. For example, if a dependency is discovered with `badrpm2`, that RPM will need to be removed from the `zypper` command of the previous step.

```
smw# image create -r site-login-large_cle_6.0.up07_sles_12sp3_ari \
site-login-large_cle_6.0.up07_sles_12sp3-x86_64_created20180517
```

5. Inspect the build image root to ensure that the desired content has been removed.

This example uses the Cray `image chroot` command, which can be used to `chroot` into any XC system image root, regardless of architecture.

```
smw# image chroot \
site-login-large_cle_6.0.up07_sles_12sp3-x86_64_created20180517
chroot-smw# exit
smw#
```

Things to note about `image chroot`:

- The `image chroot` command requires only the name of the image root, whereas the Linux `chroot` command requires the full path name of the image root.
- Using `image chroot` to `chroot` into a non-bootable image like PE or diags may result in a prompt like this because such images lack the content used to populate the prompt:  
[diags\_cle\_6.0.up07\_sles\_12sp3] I have no name!@smw:/"I have no name!"



6. To ensure that the new image will be built automatically by `imgbuilder` in the future, add the new image to the default image group of `cray_image_groups.yaml`.

The new image must be added as part of an image specification (stanza) like the one in the example. Substitute the correct recipe/image names and NIMS group for this system.

```
smw# vi /var/opt/cray/imps/config/sets/global/config/cray_image_groups.yaml
```

```
cray_image_groups:
  default:
    ...
    - recipe: "site-login-large_cle_{cle_release_lowercase}_sles_12sp3_ari"
      dest: "site-login-large_cle_{cle_release_lowercase}_sles_12sp3-x86_64-created{date}"
      arch: "x86_64"
      export_format: "cpio"
      nims_group: "login"
```

For more information, see [About Image Groups and How to Customize Them](#) on page 38.

## 8.10 Enable Multipath on an Installed XC System

### Prerequisites

This procedure assumes that the Cray XC system has already been installed and configured without multipath having been enabled. If performing a fresh install, this procedure is not necessary if multipath was already set up using [Prepare and Update the Global Config Set](#) on page 129 or [Update cray\\_multipath Worksheet](#) on page 191.

### About this task

This procedure describes how to enable multipath on a Cray XC system that has already been installed and configured. Note that multipath does NOT need to be fully cabled to be used. The multipath driver can handle using one path or many.

**IMPORTANT:** If this system has partitions, repeat any steps that modify 'p0' for each partition. Multipath must be enabled everywhere or nowhere; enabling it on only part of the system causes problems.

### Procedure

1. Start the multipath daemon now.

```
smw# systemctl start multipathd
```

Later in this procedure, the `cray-ansible` command will be used to enable the multipath daemon.

2. Obtain the host ID of the SMW and the cnames of any nodes in the system that are connected to the boot RAID with an HBA (host bus adapter).

The system should be bounced or booted for `xtcheckhss` to return a proper list.

```
smw# hostid
{8 digit hostid}
smw# xtcheckhss --detail=f --pci
```

Look for cnames with HBAs like 'QLogic\_ISP2532\_8Gb\_Fibre\_Channel\_HBA.'

————— UPDATE CRAY\_MULTIPATH IN GLOBAL CONFIG SET —————

3. Use the configurator to update `cray_multipath` in the global config set.

```
smw# cfgset update -s cray_multipath -m interactive -l advanced global
```

- a. Enable multipath.

Enter **E** at the configurator prompt to toggle the enable status of the multipath service, which is disabled by default.

```
Cray Multipath Configuration Service Menu [default: save & exit - Q] $ E
```

- b. Add the host ID and cnames obtained in an earlier step.

At the prompt, enter **1** to select the `node_list` setting, then enter **C** to configure it. At the prompt for that setting, enter values **+** to add `node_list` entries: add the host IDs and cnames obtained in an earlier step, one per line. When finished, press **Ctrl-d** and then **<cr>** to set the entries.

```
Cray Multipath Configuration Service Menu [default: save & exit - Q] $ 1
...
Cray Multipath Configuration Service Menu [default: configure - C] $ C
...
cray_multipath.settings.multipath.data.node_list
[<cr>=set 0 entries, +=add an entry, ?=help, @=less] $ +
Add node_list (Ctrl-d to exit) $
```

Do NOT save changes and exit the configurator yet.

4. Correct the values of three pre-populated multipath device settings.

Perform this step only if this system was updated from CLE 6.0.UP03 or an earlier release AND these values were not corrected during the update.

- a. View all enabled devices.

At the prompt, enter **33** to select the `enabled_devices` setting, then enter **C** to configure it.

```
Cray Multipath Configuration Service Menu [default: save & exit - Q] $ 33
Cray Multipath Configuration Service Menu [default: configure - C] $ C
```

At the prompt for this setting, enter **\*** to view all of the pre-populated device settings.

```
cray_multipath.settings.enabled_devices
[<cr>=set 11 entries, +=add an entry, ?=help, @=less] $ *
```

- b. Change the value of the path grouping policy field for the `DDN_EF3015` device.

Find the `DDN_EF3015` device in the list of enabled devices, and enter its number (5 in this example) followed by **'d'** and **'\*'** to select and edit the `path_grouping_policy` field.

```
cray_multipath.settings.enabled_devices
[<cr>=set 11 entries, +=add an entry, ?=help, @=less] $ 5d*
```

If this field is not already set to **group\_by\_prio**, set it to that value now.

```
cray_multipath.settings.enabled_devices.data.DDN_EF3015.path_grouping_policy
[<cr>=keep 'multibus', <new value>, ?=help, @=less] $ group_by_prio
```

- c. Change the value of the `product` field for the `DDN_SFA12K_20` device.

Find the `DDN_SFA12K_20` device in the list of enabled devices, and enter its number (10 in this example) followed by 'b' and '\*' to select and edit the `product` field.

```
cray_multipath.settings.enabled_devices
[<cr>=set 11 entries, +=add an entry, ?=help, @=less] $ 10b*
```

If this field is not already set to **SFA12K-20**, set it to that value now.

```
cray_multipath.settings.enabled_devices.data.DDN_SFA12K_20.product
[<cr>=keep 'SFA12K20', <new value>, ?=help, @=less] $ SFA12K-20
```

- d. Change the value of the `product` field for the `DDN_SFA12K_40` device.

Find the `DDN_SFA12K_40` device in the list of enabled devices, and enter its number (11 in this example) followed by 'b' and '\*' to select and edit the `product` field.

```
cray_multipath.settings.enabled_devices
[<cr>=set 11 entries, +=add an entry, ?=help, @=less] $ 11b*
```

If this field is not already set to **SFA12K-40 | SFA12KX\***, set it to that value now.

```
cray_multipath.settings.enabled_devices.data.DDN_SFA12K_40.product
[<cr>=keep 'SFA12K40', <new value>, ?=help, @=less] $ SFA12K-40 | SFA12KX*
```

Set the `enabled_devices` entries, but DO NOT save changes and exit the configurator yet.

```
cray_multipath.settings.enabled_devices
[<cr>=set 11 entries, +=add an entry, ?=help, @=less] $ <cr>
```

Do NOT save changes and exit the configurator yet.

5. Correct the syntax of the multipath blacklist devices setting.

Perform this step only if this system was updated from CLE 6.0.UP03 or an earlier release AND these values were not corrected during the update.

The multipath configuration contains syntax that works under SLES 12 but not under SLES 12 SP2 or SP3. That syntax must be corrected in three places (more if there is more than one CLE config set) on systems updating from CLE 6.0.UP03:

- the `/etc/multipath.conf` file in the new release snapshot
- multipath configuration service template in the global config set
- multipath configuration service template in every CLE config set in use

The `/etc/multipath.conf` file must be corrected manually because the corrections are needed for the init boot phase, and any changes to the multipath configuration service (the preferred approach) would not be reflected in `/etc/multipath.conf` until `cray-ansible` runs, which on the SMW occurs only in the multi-user boot phase. However, correcting only `/etc/multipath.conf` is not sufficient, because when `cray-ansible` runs in multi-user phase, that file is replaced with one that reflects the settings in the multipath configuration service. Therefore, the corrections must be made in the global and CLE config sets as well. Note that the corrected syntax works under both SLES 12 and SLES 12 SP2/SP3.

- a. Select the `blacklist_devices` setting.

At the configuration service menu prompt, enter **31** to select `blacklist_devices`, and then enter **c** to configure that setting. Both the vendor and product values will be changed from `*` to `.*`.

```
Cray Multipath Configuration Service Menu [default: save & exit - Q] $ 31
Cray Multipath Configuration Service Menu [default: configure - C] $ C
***** cray_multipath.settings.blacklist_devices *****
blacklist_devices
  Enter the devices which you would like to blacklist for multipath.
By default, all devices are blacklisted. Remove the 'all' key in this
setting to de-blacklist all devices.

Configured Values:
  1) 'all'
    a) vendor: *
    b) product: *

Inputs: menu commands (? for help)

|--- Information
| * Multiple 'blacklist_devices' entries can be added using this menu
|---

cray_multipath.settings.blacklist_devices
[<cr>=set 1 entries, +=add an entry, ?=help, @=less] $
```

- b. Enter **1a\*** to change the vendor value.

```
cray_multipath.settings.blacklist_devices
[<cr>=set 1 entries, +=add an entry, ?=help, @=less] $ 1a*
```

- c. Enter `.*` to update the current value to the correct value.

```
cray_multipath.settings.blacklist_devices.data.all.vendor
[<cr>=keep '.*', <new value>, ?=help, @=less] $ .*
```

- d. Enter **1b\*** to change the product value.

```
cray_multipath.settings.blacklist_devices
[<cr>=set 1 entries, +=add an entry, ?=help, @=less] $ 1b*
```

- e. Enter `.*` to update the current value to the correct value.

```
cray_multipath.settings.blacklist_devices.data.all.product
[<cr>=keep '.*', <new value>, ?=help, @=less] $ .*
```

- f. Set the changed `blacklist_devices` entry.

```
cray_multipath.settings.blacklist_devices
[<cr>=set 1 entries, +=add an entry, ?=help, @=less] $ <cr>
```

- g. Save changes and exit the configurator.

```
Cray Multipath Configuration Service Menu [default: save & exit - Q] $ Q
```

- h. Edit the multipath configuration file.

```
smw# vi /etc/multipath.conf
```

The following section in `/etc/multipath.conf` shows the incorrect `vendor` and `product` values of `"*"` and `"*"`:

```
blacklist {
    devnode "^(ram|raw|loop|fd|md|dm-|sr|scd|st) [0-9]*"
    devnode "^hd[a-z]"
    devnode "^cciss!c[0-9]d[0-9]*"
    device {
        vendor "*"
        product "*"
    }
}
```

The same section displayed with correct `vendor` and `product` values:

```
blacklist {
    devnode "^(ram|raw|loop|fd|md|dm-|sr|scd|st) [0-9]*"
    devnode "^hd[a-z]"
    devnode "^cciss!c[0-9]d[0-9]*"
    device {
        vendor ".*"
        product ".*"
    }
}
```

6. If still in the configurator, save changes and exit the configurator now.

If the previous step was skipped because the values had already been corrected during an update or this system had a fresh install of CLE 6.0.UP04 or a later release, then the configurator may still be running from an earlier step.

```
Cray Multipath Configuration Service Menu [default: save & exit - Q] $ Q
```

```
————— UPDATE CRAY_BOOTRAID IN GLOBAL CONFIG SET —————
```

7. Use the configurator to update `cray_bootraid` in the global config set.

```
smw# cfgset update -s cray_bootraid -m interactive global
```

- a. Select the storage sets setting to configure it.

```
Boot RAID Configuration Service Menu [default: save & exit - Q] $ 1
...
Boot RAID Configuration Service Menu [default: configure - C] $ C
```

- b. For each device in the `cledefault` and `smwdefault` storage sets, modify the path name from `scsi` to `dm-uuid-mpath`.

This example shows selecting the `cledefault` (1) volume group (a) `boot_node_vg` (1) devices (b) field. The `*` indicates that the selection is to be edited.

```
cray_bootraid.settings.storage_sets
[<cr>=set 2 entries, +=add an entry, ?=help, @=less] $ 1a1b*
```

Remove the `scsi` path name and replace it with the `dm-uuid-mpath` name.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.devices
[<cr>=set 1 entries, +=add an entry, ?=help, @=less] $ 1-

cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.devices
[<cr>=set 0 entries, +=add an entry, ?=help, @=less] $ +
Add devices (Ctrl-d to exit) $ /dev/disk/by-id/dm-uuid-mpath-3600a0980009ec0750000010a5762af70
Add devices (Ctrl-d to exit) $ <Ctrl-d>
```

Press **Enter** (<cr>) to set the entries for the `boot_node_vg` volume group.

```
cray_bootraid.settings.storage_sets.data.cledefault.volume_groups.boot_node_vg.devices
[<cr>=set 1 entries, +=add an entry, ?=help, @=less] $ <cr>
```

Repeat this substep for each device in the `cledefault` and `smwdefault` storage sets. Enter **\*** at the prompt to see all storage set entries.

- To select the next `cledefault` volume group device (`sdb_node_vg`), enter **1a2b\*** at the prompt. If there are more `cledefault` volume groups, increment the third character to select each one (**1a3b\***, **1a4b\***, and so forth).
- To select the first `smwdefault` volume group device (`smw_node_vg`), enter **2a1b\*** at the prompt. If there are more `smwdefault` volume groups, increment the third character to select each one (**2a2b\***, **2a3b\***, and so forth).

- c. Set the storage set entries, then save changes and exit the configurator.

```
cray_bootraid.settings.storage_sets
[<cr>=set 2 entries, +=add an entry, ?=help, @=less] $ <cr>
...
Boot RAID Configuration Service Menu [default: save & exit - Q] $ Q
```

————— UPDATE CRAY\_MULTIPATH IN CLE CONFIG SET(S) —————

8. Use the configurator to set up inheritance for multipath in the CLE config set.

This example uses 'p0' as the name of the CLE config set. Substitute the actual name used for this system.

```
smw# cfgset update -s cray_multipath -m interactive p0
```

Enter **I** at the configurator prompt to toggle the inherit status of the multipath service, which is disabled by default. This means that multipath settings in the global config set will be used instead of multipath settings in the CLE config set.

```
Cray Multipath Configuration Service Menu [default: save & exit - Q] $ I
```

Repeat this step for each CLE config set.

————— VALIDATE CONFIG SETS AND APPLY CHANGES —————

9. Validate the config sets and run `cray-ansible` to apply the config set changes.

- a. Validate the config sets.

```
smw# cfgset validate global
```

```
smw# cfgset validate p0
```

- b. Run cray-ansible.

```
smw# /etc/init.d/cray-ansible start
```

————— FOR SYSTEMS USING DAL —————

10. For systems using direct-attached Lustre (DAL), update the `dal.fs_defs` file.

Repeat these steps for each partition.

- a. Locate the current `fs_defs` files (typically stored in `/home/crayadm`).

```
smw# find /home/crayadm -name "*fs_defs*"
```

- b. Find the `fs_defs` files that are currently installed and compare with the one found in `/home/crayadm`.

```
smw# cd /var/opt/cray/imps/config/sets
```

```
smw# find p0 -name "*fs_defs*"
```

```
smw# diff /home/crayadm/dal.fs_defs \
p0/lustre/.lctrl/dal.fs_defs.20160205.1454685527
```

- c. Edit the `dal.fs_defs` file to ensure that it has the proper `mpath` paths in it.

```
smw# cd /home/crayadm
```

```
smw# sed -i.nompath \
's/\dev\disk\by-id\scsi/\dev\disk\by-id\dm-uuid-mpath/g' \
dal.fs_defs
```

```
smw# cp -p dal.fs_defs dal.fs_defs.mpath
```

- d. Install the new `dal.fs_defs` file using `lustre_control`.

```
smw# lustre_control install -c p0 /home/crayadm/dal.fs_defs
```

————— SHUT DOWN AND REBOOT SYSTEM —————

11. Shut down all partitions of the Cray system.

12. Reboot the SMW.

13. Boot the Cray system.

## 8.11 Change the Time Zone

### Prerequisites

This procedure assumes that the XC system is booted.

### About this task

This procedure changes the time zone of an XC system by changing some configuration and then rebooting components. Most of these commands must be performed as root.

### Procedure

1. Check the current time zone.

- a. Check time zone on SMW.

```
smw# date
```

- b. Check time zone on cabinet and blade controllers.

```
smw# xtrsh -l root -s date
```

- c. Check time zone on boot node.

```
smw# ssh boot date
```

- d. Check time zone on SDB node.

This command works from the SMW if the SDB node is a tier1 node with an Ethernet connection to the SMW.

```
smw# ssh sdb date
```

- e. Check time zone on all service nodes.

```
smw# ssh sdb pcmd -r -n ALL_SERVICE_NOT_ME "date"
```

- f. Check time zone on all compute nodes.

```
smw# ssh sdb pcmd -r -n ALL_COMPUTE "date"
```

Continue to the next step only if the time zone needs to be changed.

2. Change the SMW local time zone, if needed.

The default time zone on the SMW is **America/Chicago**. To change it:

- a. Execute this command:

```
smw# yast2 timezone
```

yast2 opens a new window for changing the time zone, then a pop-up window appears with this message: "file /etc/ntp.conf has been changed manually. YaST might lose some of the changes."

- b. Select the **Do not show this message anymore** checkbox, then select **Continue**.



- c. Choose the time zone either by selecting a region on the map or by using the drop-down menus for **Region** and **Time Zone**.
- d. Select **Other Settings** if the time is incorrect, then select the **Manually** radio button and enter **Current Time** and **Current Date**. Select **Accept** when done.
- e. Select **OK** when done with time zone settings.

The change on the SMW is immediate, but any users on the system need to log out and then log in again to get the new environment. This does not change the time zone for the CLE nodes or the cabinet and blade controllers. Continue to step 3 to make those changes.

### 3. Change the time zone in the global config set.

- a. Set `cray_time.settings.service.data.timezone` to the desired time zone.

A list of possible time zones is available on the SMW in `/usr/share/zoneinfo/zone1970.tab`.

```
smw# cfgset update -s cray_time -m interactive global
```

- b. Validate the config set.

```
smw# cfgset validate global
```

### 4. Change the time zone in the CLE config set.

If the CLE config set has `cray_time.inherit` set to true, then the time zone and other time settings from the global config set will be inherited by the CLE config set. If the CLE config set has `cray_time.inherit` set to false, then use these commands to change the setting and validate the config set.

- a. Set `cray_time.settings.service.data.timezone` to the desired time zone.

A list of possible time zones is available on the SMW in `/usr/share/zoneinfo/zone1970.tab`.

```
smw# cfgset update -s cray_time -m interactive p0
```

- b. Validate the config set.

```
smw# cfgset validate p0
```

### 5. Put the SMW time zone setting where the cabinet and blade controllers can access it.

```
smw# cp /etc/localtime /opt/tftpboot/localtime
```

### 6. Reboot to set the new time zone for all components.

- a. Shut down CLE.

```
smw# su - crayadm
crayadm@smw> xtbootsys -s last -a auto.hostname.stop
```

If this site does not have an `auto.hostname.stop` file, use the following command to shut down the system.

```
crayadm@smw> xtbootsys -s last -a auto.xtshutdown
```

- b. Reboot the SMW and verify that the time zone has been reset.

```
crayadm@adm> exit
smw# reboot
```

After the SMW reboots, check that the SMW has the desired time zone setting.

```
smw# date
```

- c. Reboot the cabinet controllers, then verify that all cabinet controllers are up.

```
smw# xtccreboot -c all
smw# sleep 120
smw# xtalive -a llsysd -l 11 s0
```

Repeat the `xtalive` command until all cabinet controllers are alive.

- d. Reboot the blade controllers, then verify that all blade controllers are up.

```
smw# xtccreboot -b all
smw# sleep 120
smw# xtalive s0
```

Repeat the `xtalive` command until all blade controllers are alive.

- e. Boot CLE nodes for the new time zone using the site boot automation file.

```
crayadm@smw> xtbootsys -a auto.hostname.start
```

## 7. Check the current time zone again.

- a. Check time zone on SMW.

```
smw# date
```

- b. Check time zone on cabinet and blade controllers.

```
smw# xtrsh -l root -s date
```

- c. Check time zone on boot node.

```
smw# ssh boot date
```

- d. Check time zone on SDB node.

This command works from the SMW if the SDB node is a tier1 node with an Ethernet connection to the SMW.

```
smw# ssh sdb date
```

- e. Check time zone on all service nodes.

```
smw# ssh sdb pcmd -r -n ALL_SERVICE_NOT_ME "date"
```

- f. Check time zone on all compute nodes.

```
smw# ssh sdb pcmd -r -n ALL_COMPUTE "date"
```

If these checks show the correct time zone, then the time zone has been successfully changed.

## 8.12 Initialize zsh for Non-interactive Jobs

Each shell available to users has its own rules for run-time setup. This affects which files are read in different contexts; interactive, batch, and `ssh` commands use different files depending on the shell being used. For example, `zsh` (Z shell) does not initialize like `bash`, and when `ssh` launches a shell, the environment for `bash` users and `zsh` users is different. For `bash` users, this means that the `modules` command is available for remotely executed commands, and for `zsh` users it is not available.

Users of `zsh` may find that `bash` scripts executed as part of batch scheduled jobs do not have the same environment as the login node they are submitted from because they are invoked on non-local MOM nodes. As a result, some commands that work interactively outside of the job script may fail when the batch scheduler tries to invoke the job script. In the context of `ssh` commands under `zsh`, only one type of file from two possible locations is read: the `zshenv` file. To enable the job script to operate, users with accounts on the remote machine can make changes to `~/ .zshenv` on the remote machine. If the system administrator believes that all users should automatically have a change made, the contents of `/etc/zshenv` can be modified by a site-local Ansible play.

There are two methods of affecting the environment for the `bash` scripts that comprise the batch scheduler jobs. If `zsh` makes environment changes before the `bash` script is invoked, they can be inherited by the script. Alternatively, `zsh` can nominate a file that provides the environment that `bash` processes, including scripts, should use in the environment variable `BASH_ENV`. Remember that the contents of `zshenv` are executed by every `zsh` process, so if `ssh` is heavily used for purposes other than batch scheduler jobs, it might be more efficient to use `zshenv` to nominate `BASH_ENV` and only 'pay' the cost for batch scheduler jobs.

**method 1** Export an environment variable that will enable `bash` scripts (like the job script) to include the `module` command when the script starts. To enable `bash` scripts to execute with modules for `zsh` users as they would for `bash` users, append the following line to the `zshenv` file:

```
export BASH_ENV=/etc/bash.bashrc.local
```

**method 2** Modify `zsh` initialization to always execute interactive startup code every time `zsh` is launched. To force every `zsh` shell to perform the startup that an interactive `zsh` shell does, append the following line to the `zshrc` file:

```
source /etc/zshrc
```

These two methods can be used by individual `zsh` users or by system administrators who wish to make system-wide changes.

## 8.13 Prepare Site and Software Revision Information Reporting using `xtgetrev` and `xtshowrev`

### Prerequisites

To run `xtgetrev`, the boot node must be booted and accessible.

## About this task

System administrators use the `xtgetrev` and `xtshowrev` commands to gather and display machine, software revision, Field Notice (FN), and patch set information. The `xtgetrev` command collects information from the administrator and from the SMW and boot node. The `xtshowrev` command displays that information, even when CLE is not running. These tools are useful for gathering information to send to Cray after installing a software upgrade, FN, or patch set and for help with troubleshooting.

This procedure describes how to use these two tools on a Cray XC Series system. These steps (except for running `xtshowrev`) must be executed as root.

**ATTENTION:** Any information that is submitted to `site_install_data@cray.com` will be used only within Cray, Inc. and will not be made public. The `xtshowrev` command does not submit any information to Cray automatically.

## Procedure

1. Load the module to enable use of the tools.

```
smw# module load xtshowrev
```

2. Run `xtgetrev` to create and populate the initial files.

Only root can run this command. The first time `xtgetrev` is executed, when there are no files populated, the tool will prompt for site information. If the boot node does not have passwordless ssh, then the tool will prompt for the password.

This example uses `CRAY/INTERNAL` as the site name and `9999` as the serial number of the machine. Substitute the actual values for this site.

```
smw# xtgetrev
xtgetrev: No site information has been defined.

Site name: CRAY/INTERNAL
Serial Number: 9999
System Name [panda1]:
System Type [XC40]:

<snip>
```

**Trouble?** If `xtgetrev` does not allow entry of those values, it may be because the initial configuration files have been created already. In that case, manually edit `/etc/opt/cray/release/pkginfo/site_config` and modify 'site name:' and 'serial number:' values.

```
smw# vi /etc/opt/cray/release/pkginfo/site_config
```

3. Run `xtshowrev` to see the formatted information.

Note the prompt, which indicates that any user can run this command.

```
user@smw> xtshowrev
Site:                CRAY/INTERNAL
S/N:                 9999
System Type:         XC40
Install Date:        2016-06-01
```

```
<snip>  
user@smw>
```

## 8.14 Shut Down the CLE System

### About this task

To shut down the CLE system, first determine whether it is booted, then use the shutdown automation file to shut it down gracefully.

### Procedure

1. Check whether the boot node is up.

```
smw# ping -c3 boot
```

2. If the boot node is up, then shut down the CLE system.

```
smw# su - crayadm  
crayadm@smw> xtbootsys -s last -a auto.hostname.stop  
crayadm@smw> exit  
smw#
```

If this site does not have an `auto.hostname.stop` file, use the following command to shut down the system.

```
crayadm@smw> xtbootsys -s last -a auto.xtshutdown
```

## 9 Checklists for XC™ Series Software Installation

The process of installing and configuration software for a Cray XC Series system is not necessarily sequential: some steps are optional, and some can be performed in parallel by different people. Cray recommends using the provided checklists to track progress through the installation/configuration process.

*Master Checklist: Install and Configure New SMW/CLE Software* on page 440 lists all of the high-level tasks needed for initial installation and configuration of software on an XC system. It includes links to more detailed checklists for some tasks.

### 9.1 Master Checklist: Install and Configure New SMW/CLE Software

Table 19. Master Checklist: Install and Configure New SMW/CLE Software

	Task	Notes
	Prepare for an SMW/CLE Fresh Install	
	<a href="#">Installation Checklist 1: Install the Base Operating System on the SMW</a> on page 440	
	<a href="#">Installation Checklist 2: Install the SMW and CLE Software</a> on page 441	
	<a href="#">Installation Checklist 3: Configure SMW for CLE Hardware during a Fresh Install</a> on page 442	
	<a href="#">Installation Checklist 4: Configure CLE</a> on page 443 (includes <a href="#">Installation Checklist 5: Update CLE Configuration Services</a> on page 444)	
	<a href="#">Installation Checklist 6: Prepare Boot Images and Boot the CLE System during a Fresh Install</a> on page 445	
	<a href="#">Installation Checklist 7: Configure Other Features and Services</a> on page 445	
	<a href="#">Installation Checklist 8: Install Additional Software</a> on page 447	

## 9.2 Installation Checklist 1: Install the Base Operating System on the SMW

Table 20. Installation Checklist 1: Install the Base Operating System on the SMW

	Task	Notes
Prepare to Install the Base Linux Distribution		
	(only for Dell R815 SMW) Dell R815 SMW: Change the BIOS and iDRAC Settings	
	(only for Dell R630 SMW) Dell R630 SMW: Configure the RAID Virtual Disks	
	(only for Dell R630 SMW) Dell R630 SMW: Change the BIOS and iDRAC Settings	
Install the base OS		
	Install the SLES 12 SP3 Base Linux Distribution on the SMW	
Configure Boot RAID Devices		
	Install SANtricity Storage Manager for NetApp, Inc. Devices	
	(only for systems using DAL) Set Up Boot RAID Space for Direct-attached Lustre	
	Create Boot RAID Volume Group and Volumes for NetApp, Inc. Devices	
	Zone the SAS (Serial Attached SCSI) or FC (Fibre Channel) switch using one of these procedures: <ul style="list-style-type: none"> <li>• Zone the QLogic FC Switch</li> <li>• Zone the Brocade FC Switch</li> <li>• Zone the LSI SAS Switch</li> </ul>	
	Reboot the SMW and Verify LUNs are Recognized	
Make the First Snapshot		
	Make a Snapshot Manually	

## 9.3 Installation Checklist 2: Install the SMW and CLE Software

Table 21. Installation Checklist 2: Install the SMW and CLE Software

	Task	Notes
	Start a Typescript File	
	Prepare to install the SMW and CLE software	
	Collect Software Media	
	Mount Software Media and Prepare <code>install.cle.conf</code> during a Fresh Install	
	Determine the Persistent Device Name for a LUN	
	Bootstrap and install the SMW and CLE software	
	Bootstrap the SMW Installation	
	Provision SMW Storage	
	Run the Installer for an Initial Installation	
	Set Default Snapshot and Boot the SMW	

## 9.4 Installation Checklist 3: Configure SMW for CLE Hardware during a Fresh Install

Table 22. Installation Checklist 3: Configure SMW for CLE Hardware during a Fresh Install

	Task	Notes
	Prepare to configure the SMW for CLE hardware	
	Set or Change the HSS Data Store (MariaDB) Root Password	
	Start a Typescript File	
	Make a Post-install Snapshot using <code>snaputil</code>	
	Make a Post-install Backup of Current Global and CLE Config Sets	
	Update <code>install.cle.conf</code> for Software Updates	
	Prepare the global config set and the CLE configuration worksheets	
	Prepare and Update the Global Config Set	
	Prepare the CLE Configuration Worksheets	



	Task	Notes
Discover hardware		
	Bootstrap Hardware Discovery	
	Discover Hardware and HSN Routing, Prepare STONITH	
	Update Firmware	
	(optional) Configure Partitions	
	(if needed) Repurpose a Compute or Service Node	
	Finish Configuring the SMW for the CLE System Hardware	

## 9.5 Installation Checklist 4: Configure CLE

Table 23. Installation Checklist 4: Configure CLE

	Task	Notes
	<a href="#">Installation Checklist 5: Update CLE Configuration Services</a> on page 444	
	Create New CLE Config Set from Worksheets	
	Update Unconfigured CLE Configuration Services	
	Change Group Ownership for Log Files and Directories	
	Add or Migrate Site Data to <code>/etc/hosts</code> File	
	Update CLE Config Set after a Fresh Install	
Perform post-configuration activities		
	Check CLE Host Names in <code>/etc/hosts</code> File	
	(if needed) Set Up Advanced RSIP Configuration Before Booting the System	
	Update <code>/etc/motd</code> for Nodes	
	(if needed) Copy Files for External Lustre Fine-grained Routing	
	(if needed) Configure Files for Cray Simple Sync Service	
	Display and Capture all Config Set Information	
	Validate Config Sets	

	Task	Notes
	Make a Post-config Snapshot using snaputil	
	Make a Post-config Backup of Current Global and CLE Config Sets	

## 9.6 Installation Checklist 5: Update CLE Configuration Services

Edit only the configuration worksheets listed here. Leave all other configuration worksheets unchanged—their associated config services will be updated in a later procedure after all worksheets (both updated and unchanged) are used to create a CLE config set.

Table 24. Installation Checklist 5: Update CLE Configuration Worksheets

	Task	Notes
	Update cray_node_groups Worksheet	
	Update cray_net Worksheet	
	Update cray_alps Worksheet	
	Update cray_auth Worksheet	
	Update cray_boot Worksheet	
	Update cray_drc Worksheet	
	Update cray_dvs Worksheet	
	Update cray_image_binding Worksheet	
	Update cray_lnet Worksheet	
	Update cray_local_users Worksheet	
	Update cray_login Worksheet	
	Update cray_lustre_client Worksheet	
	Update cray_lustre_server Worksheet	
	Update cray_multipath Worksheet	
	Update cray_netroot_preload Worksheet	
	Update cray_opa Worksheet	
	Update cray_persistent_data Worksheet	
	Update cray_rsis Worksheet	
	Update cray_scalable_services Worksheet	
	Update cray_sdb Worksheet	

	Task	Notes
	Update cray_simple_services Worksheet	
	Update cray_simple_shares Worksheet	
	Update cray_ssh Worksheet	
	Update cray_storage Worksheet	
	Update cray_sysconfig Worksheet	

## 9.7 Installation Checklist 6: Prepare Boot Images and Boot the CLE System during a Fresh Install

Table 25. Installation Checklist 6: Prepare Boot Images and Boot the CLE System during a Fresh Install

	Task	Notes
	Create a NIMS Map	
	Build Boot Images for a Fresh Install	
	(if needed) Configure Boot and/or SDB Node Failover	
	(if needed) Set the Turbo Boost Limit	
	Check NIMS Information during a Fresh Install	
	Boot the System using a Boot Automation File	
	(if system is air cooled) Check Cabinet Cooling Parameters for an Air-Cooled XC System	
	Run Tests after Boot is Complete	
	Prepare Site and Software Revision Information Reporting using <code>xtgetrev</code> and <code>xtshowrev</code>	
	Test <code>xtdumpsys</code> and <code>cdump</code>	
	Make a Post-Boot Snapshot using <code>snaputil</code>	
	Make a Post-Boot Backup of Current Global and CLE Config Sets	

## 9.8 Installation Checklist 7: Configure Other Features and Services

Table 26. Installation Checklist 7: Configure Other Features and Services

	Task	Notes
	(required) Configure Power Management	
	(required if using diags) Push Diag Image to Boot Node and Update the Diags Bind Mount Profile	
	(required if using netroot) Configure Netroot	
	(required if using SEC) Configure the Simple Event Correlator (SEC) and <code>check_xt</code> Monitoring and Notification Utilities	
	(required if using DAL) Configure Direct-attached Lustre (DAL)	
	(optional if using DAL) LMT Configuration for DAL (Lustre Monitoring Tool for direct-attached Lustre)	
	(required if using Docker) Enable and configure Docker, if needed for this system.  Use <i>XC™ Series Docker Administration Guide</i> (S-2586).	
	(required if using Shifter) Enable and configure Shifter, if needed for this system.  Use <i>XC™ Series Shifter Configuration Guide</i> (S-2572).	
	(recommended) Reduce Impact of Btrfs Periodic Maintenance on SMW Performance	
	(optional) Configure Cray NAT Masquerading Service	
	(optional) Prevent Unintentional Re-creation of Mail Configuration Files	
	(optional) Configure custom Node Health Checker (NHC) plugins, if needed for this system.  See "Configure Node Health Checker Tests" under "Modify an Installed System" in <i>XC™ Series System Administration Guide (CLE 6.0.UP07)</i> S-2393.	
	(optional) Configure custom Resource Utilization Reporting (RUR) data plugins or output plugins, if needed for this system.  see "Resource Utilization Reporting" under "Monitor the System" in <i>XC™ Series System Administration Guide (CLE 6.0.UP07)</i> S-2393.	

	Task	Notes
	(optional) Set up advanced RSIP configuration, if needed for this system and not already configured prior to system boot.  See "Set Up Advanced RSIP Configuration on a Booted System" under "Modify an Installed System" in <i>XC™ Series System Administration Guide (CLE 6.0.UP07) S-2393</i> .	
	(informational) About System Environmental Data Collections (SEDC)	

## 9.9 Installation Checklist 8: Install Additional Software

Table 27. Installation Checklist 8: Install Additional Software

	Task	Notes
	Install the Dell Systems Management Tools and Documentation DVD	
	Install and Configure DataWarp	
	Install Cray Programming Environment (PE) Software	
	Install and Configure a Workload Manager (WLM)	
	Install and Configure eLogin	

## 9.10 Installation Checklist 9: Customize Preinstalled SMW/CLE Software

Table 28. Installation Checklist 9: Customize Preinstalled SMW/CLE Software

	Task	Notes
	Update Site Information and Install Needed Patches	
	Change the Default System Management Workstation (SMW) Passwords	
	Change the Time Zone	
	(optional) Configure the SMW Firewall	

□	Task	Notes
	Configure LAN on the SMW	
	Change Networks, IP Addresses in Global Config Set	
	Change Networks and IP Addresses in CLE Config Set	
	Configure iDRAC network information. <ul style="list-style-type: none"> <li>• For a Dell R630 SMW: Set Up iDRAC for a Dell R630 SMW</li> <li>• For a Dell R815 SMW: Set Up iDRAC for a Dell R815 SMW</li> </ul>	
	Change the Default iDRAC Password	
	Enable Write Cache on SMW Boot RAID Volume	
	(optional) Configure the Simple Event Correlator (SEC) and check_xt Monitoring and Notification Utilities	
	(optional) Configure Site Lightweight Log Manager (LLM)	
	(optional) Prevent Unintentional Re-creation of Mail Configuration Files	
	(if system is air cooled) Check Cabinet Cooling Parameters for an Air-Cooled XC System	
	Make a Post-customize Snapshot using snaputil	
	Make a Post-customize Backup of Current Global and CLE Config Sets	