Hewlett Packard
Enterprise

HPE Cray Operating System Installation Guide for HPE Performance Cluster Manager (2.3.101) (S-8026)

Part Number: S-8026
Published: July 2022

HPE Cray Operating System Installation Guide for HPE Performance Cluster Manager

Contents

1 Copyright and Version

COS: 2.3.101-53

Doc git hash: 994ac7dc96c64495399412a2504f39378873d6b3

Generated: Wed Jul 13 2022

2 Overview

2.1 Installation Overview

This document describes how to install, upgrade, configure and boot the HPE Cray Operating System (COS) on a system managed by HPE Performance Cluster Manager (HPCM) software.

The installation process does not automatically affect the state of any other existing COS software on the system, for example any COS software that runs on compute nodes.

Multiple releases of COS can be installed on a system at the same time. The administrator can decide which release to use on compute nodes (or use multiple releases).

The same instructions are followed whether the administrator is installing COS for the first time or upgrading COS on a previously installed system.

2.2 COS Version Information

This document is specific to the COS 2.3 release. All commands and sample output refer to the release as 2.3.XX, where "XX" should be replaced with the version number in the release distribution file being installed. For example, if your release distribution file is cos-2.3.51-sle15sp3-x86_64.iso, you should substitute 2.3.51 for 2.3.XX when following the documentation.

2.3 Differences from the Previous Release

Significant changes from the previous release of COS are described in the following subsections.

2.3.1 New Features

- General security improvements and RPM signing
- DVS now has support for the `kkfilnd` Lustre Network Driver (LND) for running on the following hardware:
  - HPE Slingshot switch and HPE Slingshot SA210S Ethernet 200GB 1-port PCIe NIC (air-cooled)
  - HPE Slingshot switch and HPE Slingshot SA220M Ethernet 200GB NIC mezzanine card (NMC) (liquid-cooled)
- GPU support for Nvidia SDK 22.3 has been added.
- GPU support for CUDA driver 510.47.03 has been added.
- GPU support for AMD ROCm 5.0.2 has been added.
- GPU support for AMD driver 21.50.2 has been added.

2.3.2 Deprecated Features

- None

2.3.3 Removed Features

- GPU support for Nvidia SDK 21.9 has been removed.
- GPU support for CUDA driver 470.103.01 has been removed.
- GPU support for AMD ROCm 4.5.2 has been removed.
- GPU support for AMD driver 21.40.2 has been removed.

2.3.4 Other Changes

- None

2.4 Prerequisites

- The slingshot-host-software product is installed. slingshot-host-software provides Slingshot kernel modules and related software compiled to match the COS kernel environment. The version of slingshot-host-software installed should correspond to the version of COS being installed.

3 Introduction to COS Image Creation

This is a brief introduction to how the COS ISO file is used to produce a bootable HPCM COS image.

3.1 Procedure

This is the general workflow to produce a base image. Administrators can then add more software products to that base image. Consult the ***HPE Performance Cluster Manager Administration Guide*** (1.7 release, April 2022) for specific details of each of the steps below.

1. Obtain and download the COS ISO onto the HPCM system admin node
2. Import the content of the COS ISO into a repo
3. Create a repo group (collection of repos) that contains the Cluster Manager, COS, and SLES repos
4. Build an RPM list by combining the provided Cluster Manager and COS RPM lists
5. Create a COS image using the RPM list and repo group you have created
6. Test boot the image
7. Add Slingshot Host Software to COS image
    1. Obtain media
    2. Create Slingshot repo directory and extract media to it
    3. Create Slingshot repos
    4. Add Slingshot repos to COS image repo group
    5. Create a cos-hpcm-ss.rpmlist that builds on the COS RPM list and add the packages needed for Slingshot
    6. Create an Slingshot enabled image
8. Customize the image
9. Test boot the Slingshot-enabled COS image
10. Add more low-level software or software products to image (LNet, DVS, ROCM, PE, Slurm, etc.)

4 Install COS 2.3

See the "COS Version Information" section of this document for details on how COS version numbers are referenced in the commands and output below.

4.1 Basic Installation

HPCM manages the installation process for COS similar to how it manages installation of other operating systems. Refer to the "Creating a COS image" section of the "HPE Performance Cluster Manager Administration Guide" for instructions on using cluster manager commands to

- import the COS ISO to a repository,
- create a repository group,
- build a RPM list, and
- create a COS compute image

5 Configure COS 2.3

5.1 Boot Parameters

COS includes a set of Linux kernel boot parameters that should be specified to ensure proper behavior. The boot parameters are included in the `cray-boot-parameters-mss-compute` RPM which can be found in the COS repository. The following steps describe how to query and specify the boot parameters.

1. Extract the `./boot/parameters-mss_c` file from the `cray-boot-parameters-mss-compute` RPM and examine its contents to determine the boot parameters to be set. For example:

   ```
   admin# cd /tmp
   admin# cp /opt/clmgr/repos/other/COS-2.3.XX-sle15sp3-x86_64/RPMS/cray-boot-parameters-mss-compute-* .
   admin# rpm2cpio cray-boot-parameters-mss-compute-* | cpio -id
   2 blocks
   admin# cat ./boot/parameters-mss_c
   bad_page=panic
   hugepagelist=2m-2g
   intel_iommu=off
   intel_pstate=disable
   iommu=pt
   numa_interleave_omit=headless
   oops=panic
   pageblock_order=14
   pcie_ports=native
   quiet
   turbo_boost_limit=999
   biosdevname=0
   ```

2. Use the following cluster manager command to associate the COS boot parameters from the previous step with the COS compute image. Substitute the appropriate COS image name for `IMAGE_NAME` and include the remaining parameters in place of `...` in this example.

   ```
   admin# cm image set -i IMAGE_NAME --kernel-extra-params "bad_page=panic hugepagelist=2m-2g ..."
   ```

3. Ensure the `crashkernel` boot parameter is set to at least 1024M. Substitute the appropriate COS image name for `IMAGE_NAME` in this example. Additional information about querying and setting `crashkernel` values is provided in the “HPE Performance Cluster Manager Administration Guide”.

   ```
   admin# cm image set -i IMAGE_NAME --crashkernel 1024M
   ```

5.2 COS RPM List

The COS RPM list contains references to some RPMs that are not provided by the COS ISO. These RPMs are provided by the SUSE Linux Enterprise Server (SLES) distribution. In order to successfully create a COS compute image from the COS RPM list, SLES content must be installed and part of the repository group used to create the COS compute image.

5.3 Related Products

COS compute image content is usually created in conjunction with other HPE products and third party software to provide a fully functional compute image. The products most likely to be of potential interest are

- HPE Slingshot
- HPE LNet
- HPE Cray Data Virtualization Service (DVS)
- HPE Cray Programming Environment (CPE), including Workload Manager (WLM) software such as Slurm and PBS Pro
- AMD ROCm, See Install AMD Driver and ROCm for GPU Support
- Nvidia SDK, See Install Nvidia GPU driver and HPC SDK support for COS on HPCM
- SLES

Details on how to incorporate Slingshot and CPE product content in compute images can be found in the HPE documentation for those products. SLES content is incorporated into COS compute images as described in the “COS RPM list” section of this document.

6 LNet Installation for HPCM Compute Nodes

6.1 Required Material

A COS image containing the Slingshot Host Software appropriate for the HSN hardware installed in the system.

6.1.1 LNet Repository

The LNet RPMs are part of the COS product stream and are contained within the COS repo.

6.2 Installation

1. Create a net LNet image or skip to next step if adding LNet to an existing image.

   Note: This step is not required if you intend to add LNet to an existing image.

   Create a new LNet image, cloned from the working Slingshot image:

   ```
   IMAGE_NAME=cos-hpcm-ss-lnet
   cm image copy -o cos-hpcm-ss -i ${IMAGE_NAME} -g cos-hpcm-ss
   ```

2. Install LNet into image.

   The following RPMs should be added to the image based on Slingshot version:

   - On Industry Standard NIC systems:

   ```
   cm image zypper -i ${IMAGE_NAME} --repo-group cos-hpcm-ss install cray-lustre-client \
   cray-lustre-client-devel cray-lustre-client-kmp-cray_shasta_c
   ```

   - On HPE Slingshot NIC systems:

   ```
   cm image zypper -i ${IMAGE_NAME} --repo-group cos-hpcm-ss install cray-kfabric-kmp-cray_shasta_c \
   cray-lustre-client-ofed cray-lustre-client-ofed-devel cray-lustre-client-ofed-kmp-cray_shasta_c
   ```

   Note: Since the Slingshot Host Software is installed as a prerequisite to this procedure, then look for the cray-kfabric-kmp RPM. If it is installed, the system has HPE Slingshot NICs. Install the ofed version of the lustre-client RPMs. Otherwise install the lustre-client RPMs for Industry Standard NICs.

3. Change directory into the working copy of the image.

   ```
   cd /opt/clmgr/image/images/${IMAGE_NAME}
   ```

4. Create the lnet.conf file.

   The following `/etc/lnet.conf` file should be added to the image based on Slingshot version:

   - On Industry Standard NIC systems:

   ```
   vi etc/lnet.conf
   cat etc/lnet.conf

   net:
       - net type: o2ib
         local NI(s):
           - interfaces:
                 0: hsn0
   ```

   - On HPE Slingshot NIC systems:

   ```
   vi etc/lnet.conf
   cat etc/lnet.conf

   net:
       - net type: kfi
         local NI(s):
           - interfaces:
   ```

```
            0: cxi0
      - interfaces:
            0: cxi1
```

5. Create the modprobe conf file.

   ```
   vi etc/modprobe.d/lnet.conf
   cat etc/modprobe.d/lnet.conf

   options lnet lnet_transaction_timeout=120
   options ko2iblnd map_on_demand=1
   ```

6. Schedule LNet kernel module to be loaded at boot time.

   The following `/etc/modules-load.d/lnet.conf` file should be added to the image based on Slingshot version:

   1. On Industry Standard NIC systems:

   ```
   echo -e "ko2iblnd" > etc/modules-load.d/lnet.conf
   ```

   2. On HPE Slingshot NIC systems:

   ```
   echo -e "kkfilnd" > etc/modules-load.d/lnet.conf
   ```

7. Enable unsupported kernel modules in newly created image directory.

   ```
   sed -i 's/allow_unsupported_modules 0/allow_unsupported_modules 1/' \
   /opt/clmgr/image/images/${IMAGE_NAME}/etc/modprobe.d/10-unsupported-modules.conf
   ```

8. If a tmpfs image, there are no additional steps. If not a tmpfs image, contact HPCM support for instructions on how to recompress/rebuild the image to ensure the lnet and modprobe files addition persists into the booted image.

9. Boot new image when ready.

7 Install or Upgrade DVS

7.1 Determine COS image to be used for DVS Install

There are two approaches for including DVS into the COS image. The first is to use a single image for both DVS server and DVS client. In this approach the /etc/dvs_server_list.conf file will determine which nodes will be configured as a DVS server. The single image approach is suitable for systems that do not need separate DVS parameters for DVS server and client. The second approach is to configure separate images for the DVS server and for the DVS client. This approach is appropriate for systems that wish to have different DVS tuning parameters for server and client. The DVS server image will have entries in the `/etc/dvs_exports.yml` file for the directories to be projected to clients. The DVS client image will contain an empty /etc/dvs_server_list.conf file. The server list file must exist, otherwise the DVS service file will configure DVS as a server. The LNet and DVS RPMS will be installed in each of the separate images if that approach is chosen. The Add LNet and DVS configuration files to image section describes how the configuration files differ between DVS server and client.

The command examples in this procedure will follow the single image for DVS server and client.

1. List images available

   ```
   system_name-adm# cm image show
   my_cos-2.3_image_dvs
   my_cos-2.3_image_dvs-client
   my_cos-2.3_image_dvs-server
   my_cos-2.3_image_dvs-gateway
   fmn_sles15sp3v2
   fmn_sles15sp3v2_A
   ldap_recipe_8.0.0_ga
   login_image
   sles15sp2
   su-sles15sp3-new
   ```

2. List installed cray-lustre-client RPMs from the selected COS image

   For Mellanox hardware you will use the cray-lustre-client RPMs and for Cassini hardware you will use the cray-lustre-client-ofed RPMs.

Use the `cm image zypper` command to examine the image and confirm lustre-client RPMs installed match your HSN network hardware. All packages in the repo that match the search pattern are listed. The `i+` in the first column designates that the package is installed in the image.

```
system_name-adm# cm image zypper -i my_cos-2.3_image_dvs --duk search -s cray-lustre-client
. . .
S  | Name                                              | Type    | Version | Arch   | Repository
---+---------------------------------------------------+---------+---------+--------+-----------
   | cray-lustre-client                                | package | 2.12.4.5 . . .
   | cray-lustre-client-cray_shasta_c-lnet-devel       | package | 2.12.4.5 . . .
   | cray-lustre-client-devel                          | package | 2.12.4.5 . . .
   | cray-lustre-client-kmp-cray_shasta_c              | package | 2.12.4.5 . . .
   | cray-lustre-client-lnet-headers                   | package | 2.12.4.5 . . .
i+ | cray-lustre-client-ofed                           | package | 2.12.4.5 . . .
   | cray-lustre-client-ofed-cray_shasta_c-lnet-devel  | package | 2.12.4.5 . . .
i+ | cray-lustre-client-ofed-devel                     | package | 2.12.4.5 . . .
i+ | cray-lustre-client-ofed-kmp-cray_shasta_c         | package | 2.12.4.5 . . .
   | cray-lustre-client-ofed-lnet-headers              | package | 2.12.4.5 . . .
```

Note: The output was truncated to fit the page.

Make adjustments as required using the 'cm image zypper' command See section on LNet Installation for HPCM Compute Nodes

3. For Cassini Hardware you need the cray-kfabric-udev and cray-cxi-driver-udev packages installed

```
system_name-adm# cm image zypper -i my_cos-2.3_image_dvs --repo-group my_cos-2.3_image \
install cray-kfabric-udev
system_name-adm# cm image zypper -i my_cos-2.3_image_dvs --repo-group my_cos-2.3_image \
install cray-cxi-driver-udev
system_name-adm# echo -e "kkfilnd" >/etc/modules-load.d/lnet.conf
```

7.2 Install the DVS RPMs into the COS Image

```
system_name-adm# cm image zypper -i my_cos-2.3_image_dvs --repo-group my_cos-2.3_image \
install cray-dvs-kmp-cray_shasta_c cray-dvs-devel cray-dvs-common cray-dvs-hpcm
```

7.3 Add LNet and DVS configuration files to image

You can copy, create, or edit files directly to the image directories on the hpcm-adm file system. But for examining the rpm database, enabling or disabling systemd services, you will need to chroot to the image directory.

Change directory to the working copy of the image, then create or make adjustments to the following files: /etc/lnet.conf, /etc/dvs_exports.yml, /etc/dvs_node_map.conf, /etc/modprobe.d/lnet.conf, /etc/modprobe.d/dvs.conf, and /etc/dvs_server_list.conf.

```
system_name-adm# cd /opt/clmgr/image/images/my_cos-2.3_image_dvs
```

7.3.1 The /etc/lnet.conf and /etc/modprobe.d/lnet.conf files

The LNet Installation for HPCM Compute Nodes describes the LNet configuration file contents for Cassini and commercial network hardware. The configuration examples here represent Cassini hardware using kfi.

```
system_name-adm# cd /opt/clmgr/image/images/my_cos-2.3_image_dvs
system_name-adm# cat etc/lnet.conf

net:
   - net type: kfi
     local NI(s):
       - interfaces:
             0: cxi0
       - interfaces:
             0: cxi1
       - interfaces:
```

```
            0: cxi2
      - interfaces:
            0: cxi3

system_name-adm# cat etc/modprobe.d/lnet.conf
options lnet lnet_transaction_timeout=120
options libcfs cpu_npartitions=8
```

7.3.2 The /etc/modprobe.d/dvs.conf file

The cray-dvs-common RPM installs a default /etc/modprobe.d/dvs.conf file. Edit the file and add a options line for the dvsipc_lnet kernel module. This sets the LNet LND type. Selections are `tcp`, `o2ib`, and `kfi`. The selection must match the "net type:" in the /etc/lnet.conf file and the "lnet" parameter in the /etc/dvs_node_map.conf file. An example line, `options dvsipc_lnet lnd_name=kfi`, using the kfi LND type is shown below without the comment lines present in the file.

```
system_name-adm# cd /opt/clmgr/image/images/my_cos-2.3_image_dvs
system_name-adm# cat etc/modprobe.d/dvs.conf
options dvsproc dvs_debug_mask=0x0
options dvsipc_lnet lnd_name=kfi
```

7.3.3 The /etc/dvs_exports.yml and /etc/dvs_server_list.conf files

The /etc/dvs_exports.yml file is needed on the nodes that are acting as a dvs servers. This file allows you to control which directory paths are made available to the DVS clients. You may want to add this file with a post install script that only runs on the nodes designated as DVS servers. You then need to execute the /opt/cray/dvs/default/sbin/dvs_exportfs -a command to make DVS aware of the change. If installed in the image and a /etc/dvs_server_list.conf file is not provided, then every node booting the image will be configured as a DVS server. If the dvs_exports.yml and dvs_server_list.conf files are provided in the image, then the nodes not listed in the dvs_server_list.conf file will have their dvs_exports.yml file changed to remove any exported directory. The file will have the single line `exports: []`.

An example of the DVS client /etc/dvs_exports.yml file

```
system_name-adm# cd /opt/clmgr/image/images/my_cos-2.3_image_dvs
system_name-adm# cat etc/dvs_exports.yml
exports: []
```

An example of a DVS server /etc/dvs_exports.yml file.

```
system_name-adm# cd /opt/clmgr/image/images/my_cos-2.3_image_dvs
system_name-adm# cat etc/dvs_exports.yml
exports:
- mode: rw
  path: /
```

If a /etc/dvs_server_list.conf file is provided in the image, the DVS service file will configure the nodes specified in this file as DVS servers, and the unspecified nodes as DVS clients.

An example of a /etc/dvs_server_list.conf file.

```
system_name-adm# cd /opt/clmgr/image/images/my_cos-2.3_image_dvs
system_name-adm# cat etc/dvs_server_list.conf
dvo01
dvs02
gw01
gw02
```

7.3.4 The /etc/dvs_node_map.conf file

The /etc/dvs_node_map.conf file is necessary for creating the /etc/dvs_node_map file. For HPCM systems, we will be pre-building the node map. We then add the node map into the COS node images and boot them. The node map is generated on a node that has the DVS RPMs installed and all the DVS capable nodes are booted to a COS image. Once the /etc/dvs_node_map file is generated it is then included into the image. See the How to Populate node maps section of the "HPE Cray Operating System Administration Guide for HPCM" for details on how the DVS node map is generated for HPCM systems.

An example dvs_node_map.conf file.

```
system_name-adm# cd /opt/clmgr/image/images/my_cos-2.3_image_dvs
system_name-adm# cat etc/dvs_node_map.conf

{
   "config" : [
      {
         "lnet" : "kfi",
         "nid" : 0,
         "xname" : "h0"
      }
   ]
}
```

If you are using the separate image approach for DVS server and client, then perform the previous steps for both DVS server and client images.

7.4 Apply work-around for Slingshot service start order issue

Currently there is an issue with the lnet.service file being executed before the slingshot-ama.service file is run. This is expected to be resolved in the future. This applies to kfilnd configured HSN systems. The work-around involves editing the /usr/lib/systemd/system/lnet.service file and changing the network-online target name to the slingshot target name in the `Requires and After` directives. Change the multi-user target name to slingshot for the `WantedBy` directive. In addition, insert this line `ExecStart=/opt/cray/dvs/default/sbin/dvs_wait_for_mac` after the `RemainAfterExit=true` line. And finally insert this line `ExecStart=/sbin/modprobe kkfilnd` after the `ExecStart=/sbin/modprobe lnet` line.

Note: You will need to be either in the `chroot` environment for the image or in the image working directory to apply this work-around.

```
system_name-adm# cat fix-up
#!/bin/bash

cp /usr/lib/systemd/system/lnet.service /usr/lib/systemd/system/lnet.bak
sed -i s/network-online/slingshot/g /usr/lib/systemd/system/lnet.service
sed -i '/RemainAfterExit=true/a ExecStart=/opt/cray/dvs/default/sbin/dvs_wait_for_mac' \
/usr/lib/systemd/system/lnet.service
sed -i '/ExecStart=\/sbin\/modprobe lnet/a ExecStart=/sbin/modprobe kkfilnd' \
/usr/lib/systemd/system/lnet.service
sed -i s/multi-user/cray-dvs-lnet/ /usr/lib/systemd/system/lnet.service
```

A diff to illustrate the changes the script performs.

```
 $ diff lnet.bak  lnet.service
4,5c4,5
< Requires=network-online.target
< After=network-online.target openibd.service rdma.service opa.service
---
> Requires=slingshot.target
> After=slingshot.target openibd.service rdma.service opa.service
11a12
> ExecStart=/opt/cray/dvs/default/sbin/dvs_wait_for_mac
12a14
> ExecStart=/sbin/modprobe kkfilnd
20c22
< WantedBy=multi-user.target
---
> WantedBy=cray-dvs-lnet.target
```

7.5 Enable unsupported kernel modules in image

```
system_name-adm# sed -i 's/allow_unsupported_modules 0/allow_unsupported_modules 1/' \
/opt/clmgr/image/images/my_cos-2.3_image_dvs/etc/modprobe.d/10-unsupported-modules.conf
```

7.6 Enable LNet service

It is necessary for the slingshot-ama.service file to run to completion and the lldpd service restart complete before the lnet.service file starts.

```
system_name-adm:/ # chroot /opt/clmgr/image/images/my_cos-2.3_image_dvs
system_name-adm:/ # systemctl enable lnet
Created symlink /etc/systemd/system/slingshot.target.wants/lnet.service → /usr/lib/systemd/
system/lnet.service.
```

7.7 Enable cray-dvs service

```
system_name-adm:/ # systemctl enable cray-dvs
Created symlink /etc/systemd/system/slingshot.target.wants/cray-dvs.service → /usr/lib/systemd/
system/cray-dvs.service.
system_name-adm:/ # systemctl enable cray-dvs-lnet.target
Created symlink /etc/systemd/system/default.target.wants/cray-dvs-lnet.target → /usr/lib/systemd/
system/cray-dvs-lnet.target.
system_name-adm:/ # exit
exit
```

Confirm /etc/lnet.conf in image is correct for LNet network transport selected.

7.8 Now push the changes to the leader nodes so they will be available for boot

If you modified an existing image and an existing version of this image is running on nodes at this time, then power off any nodes that use the image you updated before you run this command.

```
system_name-adm:~/ # cm image activate -i my_cos-2.3_image_dvs --force
```

7.9 Specify the image to boot

```
system_name-adm: / # cm node set --image my_cos-2.3_image_dvs -n x9000c1s2b0n0
```

7.10 Reset the node to start the boot

```
system_name-adm: / # cm power reset -t node x9000c1s2b0n0
```

7.11 Monitor the boot (optional)

```
system_name-adm: / # console x9000c1s2b0n0 (**Ctl-e c .** will exit the console).
```

7.12 Create a DVS Node Map and include it in the image

In the HPCM environment, user file systems may be DVS-projected from any node that is booting the COS image with the DVS RPMs installed and DVS kernel modules configured and loaded. DVS uses a node map to index/correlate information about each of the DVS capable nodes in a system. This node map is constructed by the dvs_generate_map script. Once generated, the node map is placed in the /etc/dvs_node_map file. After the dvsproc kernel module has been loaded, the dvs_node_map file is loaded into kernel memory with the command `cat /etc/dvs_node_map > /proc/fs/dvs/node_map`. The cray-dvs service file is responsible for loading the DVS kernel modules and the in-memory DVS node map.

For HPCM systems, we will be pre-building the node maps. We then add the node map into the COS node images and boot them.

Use the following steps to generate the `dvs_node_map` file and include it in the image. The “How to Populate Node Maps” section in the “HPE Cray Operating System Administration Guide for HPE Performance Cluster Manager” document describes how to generate the /etc/dvs_node_map file in more detail.

7.12.1 All nodes to utilize DVS are booted with a COS image

The dvs_generate_map script needs to identify all nodes that are to use DVS. The script utilized the `cm node show -I` command to identify the booted COS image nodes. Therefore these nodes must be booted with a COS image. That image does not need to have DVS RPMs installed, but must have the Slingshot HSN configured.

7.12.2 Login to the node that you booted in the previous steps of this procedure

This will be the node that you booted with the image that has the DVS RPMs installed, which gives access to the dvs_generate_map script to create the DVS node map.

7.12.3 Run the `dvs_generate_map` script

The `dvs_generate_map` script is located in the `/opt/cray/dvs/default/sbin` directory. Execute the script. The script will utilize the `/etc/dvs_node_map.conf` file that you customized in previous steps of this procedure by default. There should be only one `/etc/dvs_node_map.conf` file utilized for your system. If your system employs multiple images for the various node types in your system (gateway, compute, or login) then all of these images must use the same LNet network transport and `dvs_node_map.conf` file.

```
dvs01# /opt/cray/dvs/default/sbin/dvs_generate_map
```

7.12.4 Include the `dvs_node_map` file in the image

Copy the `dvs_node_map` file to the HPCM system admin node and include it in the image you installed DVS RPMs. Repeat the following command for each of the DVS images utilized in your system.

```
dvs01# scp /etc/dvs_node_map root@system_name-adm:/opt/clmgr/image/images/<DVS_IMAGE_NAME>/etc/
```

7.12.5 Now push the changes to the leader nodes so they will be available for boot

Perform the following command for each of the DVS images that you included the dvs_node_map in. If you modified an existing image and an existing version of this image is running on nodes at this time, then power off any nodes that use the image you updated before you run this command.

```
system_name-adm:~/ # cm image activate -i <DVS_IMAGE_NAME> --force
```

7.12.6 Boot nodes with updated DVS image that includes dvs_node_map file

```
system_name-adm: / # cm power reset -t node <NODE_LIST>
```

8 Install GPU Support for COS on HPCM

The sections below detail installing GPU support for COS on HPCM.

8.1 Install AMD Driver and ROCm for GPU Support

1. Create new AMD repos.

   a. Create the ROCm repo directory.

      ```
      mkdir -p /opt/clmgr/repos/other/rocm-5.0.2
      ```

   b. Download the AMD ROCm content.

      ```
      cd /opt/clmgr/repos/other/rocm-5.0.2/
      wget -r -l1 --no-parent -A.rpm https://repo.radeon.com/rocm/zyp/5.0.2/
      ```

   c. Create the ROCm repo.

      ```
      cm repo add --custom rocm-5.0.2 /opt/clmgr/repos/other/rocm-5.0.2
      cm repo refresh rocm-5.0.2
      ```

   d. Create the AMD GPU driver directory.

      ```
      mkdir -p /opt/clmgr/repos/other/amdgpu-21.50.2
      ```

e. Download the AMD GPU driver content.

```
cd /opt/clmgr/repos/other/amdgpu-21.50.2/
wget -r -l1 --no-parent -A.rpm https://repo.radeon.com/amdgpu/21.50.2/sle/15/main/x86_64/
```

f. Create the AMD GPU driver repo.

```
cm repo add --custom amdgpu-21.50.2 /opt/clmgr/repos/other/amdgpu-21.50.2
cm repo refresh amdgpu-21.50.2
```

g. Create the DKMS plus repo directory.

```
mkdir /opt/clmgr/repos/other/dkms-plus/
```

h. Download DKMS plus other content.

```
cd /opt/clmgr/repos/other/dkms-plus
DOWNLOAD=https://download.opensuse.org/repositories/
wget $DOWNLOAD/home:/Ximi1970:/Dkms/openSUSE_Leap_15.3/noarch/dkms-2.5-lp153.5.1.noarch.rpm
wget $DOWNLOAD/devel:/languages:/perl/SLE_15_SP3/noarch/perl-File-BaseDir-0.09-21.2.noarch.rpm
wget $DOWNLOAD/devel:/languages:/perl/SLE_15_SP3/noarch/perl-URI-Encode-1.1.1-1.1.noarch.rpm
```

i. Create DKMS plus repo

```
cm repo add --custom dkms-plus /opt/clmgr/repos/other/dkms-plus
cm repo refresh dkms-plus
```

2. Build a compute image with GPU support.

a. Create the ROCM repo group.

Create a repo group with everything you need for your image including the newly added repos for AMD support: rocm-5.0.2, amdgpu-21.50.2, and dkms-plus. An example of all the repos you might need is below with the newly created repos at the top. The repos and versions you need will likely be different.

```
cm repo group add amd_gpu_compute --repos \
  rocm-5.0.2\
  amdgpu-21.50.2\
  dkms-plus\
  aocc-compiler-3.1.0\
  Cluster-Manager-1.6-sles15sp3-x86_64\
  COS-2.3-sles15sp3-x86_64\
  cpe-21.12-sles15-sp3\
  mss-systemd-slurm\
  NETC_internal_combined_cn-SSHOT1.6.1-20220118\
  patch11707-pre\
  SLE-15-SP3-Full-x86_64\
  slingshot-host-software-1.7.0-21\
  slurm-21.08.2_cray_sles15sp3
```

b. Create a rpm list for generating your image. Consult other documentation as necessary to create the complete rpm list.

```
touch amd_gpu_compute.rpmlist
```

c. Add the following package to the rpm list for GPU support.

```
echo cray-rocm-meta >> amd_gpu_compute.rpmlist
```

d. Create ROCM Image using the rpmlist generated above.

```
cm image create -i amd_gpu_compute --repo-group amd_gpu_compute --rpmlist amd_gpu_compute.rpmlist
```

e. Associate the repo group with the image.

```
cm image set -i amd_gpu_compute --repo-group amd_gpu_compute
```

3. Test the image by booting a node.

a. Assign the image to a node.

```
cm node set --image amd_gpu_compute -n <xname>
```

b. Verify assignment is correct.

```
cm node show -I
```

c. Reboot the node.

```
cm power reset -t node <xname>
```

d. Monitor the boot.

Console can be monitored here:

```
tail -f /var/log/consoles/<xname>
```

or via the console command:

```
console <xname>
```

e. Verify the ROCm install on a booted node.

SSH to the node and perform:

```
lsmod | grep amd
```

In the above output following modules should be shown as loaded: amdgpu, amdkcl, amd-sched, and amdttm.

Run the following command and verify the GPUs are found:

```
/opt/rocm/bin/rocm-smi
```

8.2 Install Nvidia GPU driver and HPC SDK support for COS on HPCM

All content to support Nvidia GPUs with COS can be installed into an existing COS image as created through the normal process in the COS Installation Guide. The COS ISO also includes Nvidia build support packages for the COS kernel, COS kernel specific builds of gdrcopy and nv-peer-mem, and a package that provides user environment module files that can be used by end-users and Cray PE to enable Nvidia CUDA builds with the supported Nvidia HPC SDK. Those packages are available in the standard COS repo along with the rest of the COS content.

1. Download the supported Nvidia GPU driver.

```
NVIDIA=https://us.download.nvidia.com/tesla/
wget $NVIDIA/510.47.03/nvidia-driver-local-repo-sles15-510.47.03-1.0-1.x86_64.rpm
```

2. Unpack the driver container RPM.

```
rpm2cpio nvidia-driver-local-repo-sles15-510.47.03-1.0-1.x86_64.rpm | cpio -idmv
```

3. Copy the repo directory from the driver container RPM into the admin nodes system repo location.

```
cp -r var/nvidia-driver-local-repo-sles15-510.47.03 /opt/clmgr/repos/other/
```

4. Add and refresh the new repo.

```
cm repo add --custom nvidia-driver-local-repo-sles15-510.47.03 \
    /opt/clmgr/repos/other/nvidia-driver-local-repo-sles15-510.47.03
cm repo refresh nvidia-driver-local-repo-sles15-510.47.03
```

5. Create a new repo target for the Nvidia HPC SDK and download it.

```
mkdir /opt/clmgr/repos/other/nvhpc-22-3
cd /opt/clmgr/repos/other/nvhpc-22-3
SDK=https://developer.download.nvidia.com/hpc-sdk/sles/x86_64
wget $SDK/nvhpc-22-3-22.3-1.suse.x86_64.rpm \
     $SDK/nvhpc-2022-22.3-1.suse.x86_64.rpm \
     $SDK/nvhpc-22-3-cuda-multi-22.3-1.suse.x86_64.rpm
```

6. Add and refresh the new repo.

```
cm repo add --custom nvhpc-22-3 /opt/clmgr/repos/other/nvhpc-22-3
cm repo refresh nvhpc-22-3
```

7. Create a new repo group that adds the new Nvidia repos to an existing set of repos used to generate a COS compute image.

```
cm repo group add cos-2.3-nvidia-repo-group --repos nvhpc-22-3 \
    nvidia-driver-local-repo-sles15-510.47.03 \
    [ LIST OF COS REPOS ]
```

8. Prep the existing COS boot image for installation.

   NOTE: Building the Nvidia GPU driver against the COS kernel requires setting up the proper source linking. Depending on what is installed into a compute image that linking can be "polluted" by standard SLES kernel-devel packages being pulled in as a dependency. For the Nvidia driver to correctly build for the COS kernel target the default link at `/usr/src/linux-obj/x86_64/default` in the COS compute base image must be removed if it exists prior to attempting to install the Nvidia packages.

```
chroot /opt/clmgr/images/images/[ COS IMAGE NAME ]
rm /usr/src/linux-obj/x86-64/default
exit
```

9. Install the Nvidia content into an existing COS image.

   NOTE: cuda-drivers and nvhpc-22-3-cuda-multi are both 'meta' packages that will pull in all required Nvidia driver and HPC SDK packages and dependencies. The cuda-drivers install will build the Nvidia driver for the COS kernel as part of its rpm post-install.

```
cm image zypper -i [ EXISTING COS IMAGE NAME ] --repo-group cos-2.3-nvidia-repo-group install \
    nvidia-gpu-build \
    cuda-drivers \
    nvidia_peer_memory \
    gdrcopy-kmod-shasta-kmp-510.47.03 \
    gdrcopy gdrcopy-devel \
    nvhpc-22-3-cuda-multi \
    cray_sdk_cudatoolkit_module
```

10. Nvidia device driver parameters can be edited in the compute image after driver installation. It is recommended to update the Nvidia device permissions to allow normal system users GPU access without having to maintain default "video" user group permissions. Optionally enabling perf counter access to all users by adding `NVreg_RestrictProfilingToAdminUsers=0`.

```
chroot /opt/clmgr/images/images/[ COS IMAGE NAME ]
sed -i 's/NVreg_DeviceFileMode=0660/NVreg_DeviceFileMode=0666/' /etc/modprobe.d/50-nvidia-default.conf
exit
```

11. Post-Boot Configuration Commands

    To fully enable and configure all Nvidia GPU-related COS integration features there is a short list of commands that need to be ran on the Nvidia GPU compute nodes following boot. There are multiple methods to automate this on HPCM including via image post-install scripts and editing the compute boot image to run them automatically. Note that the commands need to be ran after the Nvidia GPU drivers are loaded which happens automatically during compute boot but after any post-install image scripts would be ran.

    Commands:

```
# enable nvidia persistence mode to keep the gpu drivers loaded across job launches
nvidia-persistenced --persistence-mode

# may be needed if not ran previously as part of image creation
depmod -a

# set the GPUs to exclusive process mode ( may not be wanted for all applications )
nvidia-smi -c 3

# load the nvidia_peer_memory driver
modprobe nv_peer_mem

# load the gdrcopy driver
modprobe gdrdrv

# create the gdrcopy device file
```

```
MAJOR=`fgrep gdrdrv /proc/devices | cut -b 1-4`
mknod -m 666 /dev/gdrdrv c $MAJOR 0
```

12. Health Checks for Nvidia GPU

a. Verify Nvidia kernel drivers are loaded as expected. There should be 4 Nvidia device drivers, an nv-peer-mem driver, and a gdrcopy driver running.

```
lsmod | grep nv
nvidia
nvidia_drm
nvidia_modeset
nvidia_uvm
nv_peer_mem
```

```
lsmod | grep gdrdrv
gdrdrv
```

b. Verify that all GPUs are detected, the expected Nvidia driver version is being used, persistence mode is enabled, and that exclusive process mode has been set.

```
nvidia-smi
```

c. Verify Nvidia NVLink health as necessary.

```
nvidia-smi nvlink --status
```

d. Run the basic sanity tests.

```
sanity
...
copybw
...
apiperf
...
```

e. Verify the cudatoolkit environment module files are available

```
module avail
cudatoolkit/22.3_10.2
cudatoolkit/22.3_11.0
cudatoolkit/22.3_11.6(default)
```

9 Documentation Conventions

Several conventions have been used in the preparation of this documentation.

- Markdown Format
- File Formats
- Typographic Conventions
- Annotations for how we identify sections of the documentation that do not apply to all systems
- Command Prompt Conventions which describe the context for user, host, directory, chroot environment, or container environment

9.1 Markdown Format

This documentation is in Markdown format. Although much of it can be viewed with any text editor, a richer experience will come from using a tool which can render the Markdown to show different font sizes, the use of bold and italics formatting, inclusion of diagrams and screen shots as image files, and to follow navigational links within a topic file and to other files.

There are many tools which can render the Markdown format to get these advantages. Any Internet search for Markdown tools will provide a long list of these tools. Some of the tools are better than others at displaying the images and allowing you to follow the navigational links.

9.2 File Formats

Some of the installation instructions require updating files in JSON, YAML, or TOML format. These files should be updated with care since some file formats do not accept tab characters for indentation of lines. Only space characters are supported. Refer to online documentation to learn more about the syntax of JSON, YAML, and TOML files.

9.3 Typographic Conventions

`This style` indicates program code, reserved words, library functions, command-line prompts, screen output, file/path names, and other software constructs.

(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).

9.4 Annotations

This repository may change annotations, for now, under the MarkDown governance these are the available annotations.

You must use these to denote the right steps to the right audience.

These are context clues for steps, if they contain these, and you are not in that context you ought to skip them.

```
EXTERNAL USE
```

This tag should be used to highlight anything that an HPE Cray internal user should ignore or skip.

```
INTERNAL USE
```

This tag is used before any block of instruction or text that is only usable or recommended for internal HPE Cray systems.

External (GitHub or customer) should disregard these annotated blocks - they maybe contain useful information as an example but are not intended for their use.

9.5 Command Prompt Conventions

9.5.0.1 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 non-privileged account is referred to as user.

9.5.0.2 Node abbreviations

The following list contains abbreviations for nodes used below

- CN - compute Node
- NCN - Non Compute Node
- AN - Application Node (special type of NCN)
- UAN - User Access Node (special type of AN)
- PIT - Pre-Install Toolkit (initial node used as the inception node during software installation booted from the LiveCD)

Prompt	Description
ncn#	Run the command as root on any NCN, except an NCN which is functioning as an Application Node (AN), such as a UAN.
ncn-m#	Run the command as root on any NCN-M (NCN which is a Kubernetes master node).
ncn-m002#	Run the command as root on the specific NCN-M (NCN which is a Kubernetes master node) which has this hostname (ncn-m002).
ncn-w#	Run the command as root on any NCN-W (NCN which is a Kubernetes worker node).
ncn-w001#	Run the command as root on the specific NCN-W (NCN which is a Kubernetes master node) which has this hostname (ncn-w001).
ncn-s#	Run the command as root on any NCN-S (NCN which is a Utility Storage node).
ncn-s003#	Run the command as root on the specific NCN-S (NCN which is a Utility Storage node) which has this hostname (ncn-s003).

Prompt	Description
pit#	Run the command as root on the PIT node.
linux#	Run the command as root on a linux host.
uan#	Run the command as root on any UAN.
uan01#	Run the command as root on hostname uan01.
user@uan>	Run the command as any non-root user on any UAN.
cn#	Run the command as root on any CN. Note that a CN will have a hostname of the form nid124356, that is “nid” and a six digit, zero padded number.
hostname#	Run the command as root on the specified hostname.
user@hostname>	Run the command as any non-root user son the specified hostname.

9.5.0.3 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.

```
hostname# chroot /path/to/chroot
chroot-hostname#
```

9.5.0.4 Command prompt inside Kubernetes pod

If executing a shell inside a container of a Kubernetes pod where the pod name is $podName, the prompt changes to indicate that it is inside the pod. Not all shells are available within every pod, this is an example using a commonly available shell.

```
ncn# kubectl exec -it $podName /bin/sh
pod#
```

9.5.0.5 Command prompt inside image customization session

If using ssh during an image customization session, the prompt changes to indicate that it is inside the image customization environment (pod). This example uses $PORT and $HOST as environment variables with specific settings. When using chroot in this context the prompt will be different than the above chroot example.

```
hostname# ssh -p $PORT root@$HOST
root@POD# chroot /mnt/image/image-root
:/#
```

9.5.0.6 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

Examples of prompts as they appear on the system:

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

Examples of prompts as they appear in this publication:

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

9.5.0.7 Command prompts for network switch configuration

The prompts when doing network switch configuration can vary widely depending on which vendor switch is being configured and the context of the item being configured on that switch. There may be two levels of user privilege which have different commands available and a special command to enter configuration mode.

Example of prompts as they appear in this publication:

Enter “setup” mode for the switch make and model, for example:

```
remote# ssh admin@sw-leaf-001
sw-leaf-001>  enable
sw-leaf-001#  configure terminal
sw-leaf-001(conf)#
```

Refer to the switch vendor OEM documentation for more information about configuring a specific switch.