Hewlett Packard
Enterprise

HPE Cray Operating System Administration Guide for HPE Performance Cluster Manager (2.3.101) (S-8041)

Part Number: S-8041
Published: July 2022

HPE Cray Operating System Administration Guide for HPE Performance Cluster Manager

Contents

1 Copyright and Version **4**

2 Data Virtualization Service **5**

2.1 Introduction to DVS . . . 5
2.1.1 DVS Use Cases . . . 5
2.1.2 DVS Startup . . . 5
2.1.3 DVS Projection of External Filesystems . . . 5
2.2 Procedure To Change DVS Network Transport . . . 5
2.2.1 Change DVS Network Transport on DVS server and client Nodes . . . 7
2.3 Tune and Optimize DVS . . . 12
2.3.1 Configure Read-Only Client Mounts for Optimal Performance . . . 13
2.3.2 Improve Performance and Scalability of Spectrum Scale (GPFS) Mounts . . . 14
2.3.3 Universally Disable Fairness of Service . . . 15
2.4 DVS Statistics . . . 15
2.4.1 Collect and Analyze DVS Statistics . . . 15
2.5 Control and Format Options for DVS Statistics . . . 16
2.5.1 Caveats for Interpreting DVS Statistics . . . 18
2.5.2 Files That Contain DVS Statistics . . . 19
2.5.3 DVS Statistics Collected . . . 20
2.6 Choose and Configure a DVS Mode . . . 24
2.7 Manage and Customize DVS . . . 26
2.7.1 Procedures to Manage and Optimize DVS Use . . . 26
2.7.2 Allowing Client Access to DVS Server File Systems . . . 27
2.7.3 Select DVS image from VCS . . . 28
2.7.4 Commit DVS image changes to VCS . . . 29
2.7.5 Enable DVS Failover and Failback . . . 30
2.7.6 Project an External Filesystem to Compute Nodes or Login Nodes . . . 31
2.7.7 Quiesce a DVS-Projected File System . . . 34
2.7.8 Quiesce All Directories on a DVS Server . . . 35
2.7.9 Quiesce a Single Directory on a Single DVS Server . . . 36
2.7.10 List Outstanding DVS Client Requests . . . 36
2.7.11 Force a Cache Revalidation on a DVS Mount Point . . . 36
2.7.12 Change the Name of the DVS LNet Network . . . 37
2.7.13 Change LNet or DVS Parameters . . . 37
2.7.14 Configure DVS Using `dvs.conf` . . . 38
2.8 DVS Log Files . . . 39
2.8.1 DVS Request Logs . . . 39
2.8.2 DVS File System (FS) Call Logs . . . 39
2.8.3 Disable or Re-enable DVS Logging . . . 39
2.8.4 Reset the DVS Log . . . 40
2.8.5 Read the DVS Log . . . 40
2.8.6 Tune DVS Logging . . . 41
2.9 Troubleshoot DVS . . . 42
2.9.1 Verify All Necessary DVS Kernel Modules for the Running Kernel Are Installed . . . 42

2.9.2 Confirm All Necessary DVS Kernel Modules Are Loaded . . . 42
2.9.3 Confirm the LNet network is active . . . 42
2.9.4 Print Out the DVS Node Map . . . 43
2.9.5 Search for DVS-Related Kernel Messages . . . 43
2.9.6 List All Running DVS Processes . . . 44
2.9.7 Troubleshoot Node Map IP Change Issues . . . 44
2.10 DVS Reference Information . . . 44
2.10.1 DVS Modes . . . 44
2.10.2 DVS Failover and Failback . . . 49
2.10.3 About the Close-to-Open Coherency Model . . . 49
2.10.4 DVS Fairness of Service . . . 50
2.10.5 HPE Cray EX DVS Configuration Overview . . . 50
2.10.6 When to Use Temporary, Client-Side, Per-File Read Caching . . . 52
2.10.7 DVS Environment Variables . . . 52
2.10.8 DVS Configuration Settings . . . 53
2.10.9 Periodic Sync Promotes Data and Application Resiliency . . . 58
2.11 How to Populate Node Maps . . . 58
2.11.1 Overview . . . 58
2.11.2 Environment . . . 59
2.11.3 Pre-Conditions . . . 60
2.11.4 Operations . . . 60
2.11.5 Usage . . . 62
2.12 DVS COS 2.3 Failover Story . . . 62
2.12.1 Overview . . . 62
2.12.2 Environment . . . 62
2.12.3 Pre-Conditions . . . 62
2.12.4 How DVS Failover works. . . . 64
2.12.5 Manually Failover or Fail-back a DVS Projected File System . . . 64
2.12.6 Use Cases . . . 64
2.12.7 Usage . . . 67
2.13 Projecting Shared Libraries with DVS . . . 67
2.13.1 Prerequisites . . . 67
2.13.2 Procedure . . . 67
2.13.3 Alternative Procedure - Add mount to live system . . . 69

3 Install Spectrum Scale **70**
3.1 Install Spectrum Scale on a Gateway Node . . . 70
3.1.1 Create Spectrum Scale Gateway Node Image . . . 71
3.1.2 Deploy and Test Spectrum Scale Gateway Image . . . 79

4 Configure Lustre **80**
4.1 Mount a Lustre File System on GWs, CNs, and Login Nodes . . . 80
4.1.1 Procedure . . . 81
4.1.2 Alternative Procedure - Add Lustre mount to live system . . . 85
4.2 Set Up LNet Routers . . . 89

5 Power Management **93**
5.1 Overview . . . 93
5.2 PM Counters . . . 93

6 Set the Turbo Boost Limit **93**
6.1 Set or Change the Turbo Boost Limit Parameter . . . 93

7 User Access to Compute Node Power Data **94**
7.1 pm_counters . . . 94

8 Enable Low Noise Mode **94**
8.1 Kernel Parameters . . . 94
8.2 The LNM Configuration File . . . 95

8.2.1 CPU Section 95
8.2.2 Tunables Section 96
8.2.3 Processes Section 96
8.2.4 IRQs Section 96
8.2.5 Interaction With Workload Manager Software 97

1 Copyright and Version

COS: 2.3.101-53

Doc git hash: 994ac7dc96c64495399412a2504f39378873d6b3

Generated: Wed Jul 13 2022

2 Data Virtualization Service

2.1 Introduction to DVS

The Data Virtualization Service (DVS) distributes images and external file systems data to diskless client nodes in an HPE Cray EX system.

The Data Virtualization Service (DVS) is a distributed network service that projects file systems mounted on non-compute nodes to other nodes within the HPE Cray EX system. Projecting is simply the process of making a file system available on nodes where it does not physically reside. DVS-specific configuration settings enable clients to access a file system projected by DVS servers, providing I/O that is typically of higher-performance and greater scalability than using native file system client software. These DVS clients include compute nodes, Login nodes, and gateway nodes. Thus DVS, while not a file system, represents a software layer that provides scalable transport for file system services. DVS, in turn, uses Lustre™ Networking (LNet) to communicate over the network.

2.1.1 DVS Use Cases

DVS plays an important role in the HPE Cray EX system:

- DVS can be used to project external file systems to compute nodes, login nodes, and gateway nodes within the HPCM managed HPE Cray EX system.
- In the HPCM environment, external file systems may be DVS projected from any node that is booting the COS image. This means that any system that boots the COS image could serve as either a DVS server or a DVS client. A node can function as a DVS server for a DVS client's mount path. That same node can also simultaneously function as a DVS client with a mount path that has one or more DVS servers.
- DVS servers are typically allocated to dedicated nodes for performance and for bridging networks for access to high performance file systems (i.e. a gateway node).

System administrators must install the DVS RPMs in the COS image to configure DVS and make it available on HPCM managed HPE Cray EX systems. Installing DVS RPMs is normally done as part of the HPE Cray EX software installation. See "Install or Upgrade DVS" in the "HPE Cray Operating System Installation Guide for HPE Performance Cluster Manager" documentation.

2.1.2 DVS Startup

DVS is loaded and started by a systemd service. DVS is then used for projecting read/write file systems or external file systems such as Lustre or IBM Spectrum Scale to CNs and other DVS clients.

DVS needs further configuration only for performance tuning or if a user must project external file systems to nodes within the HPE Cray EX system. DVS configuration parameters enable system administrators to provide their users with client mounts. These parameters can then be tuned for high performance in a variety of use cases.

2.1.3 DVS Projection of External Filesystems

When projecting external file systems, DVS provides I/O performance and scalability to many nodes, far beyond the typical number of clients supported by a single NFS server. Operating system noise and impact on compute node memory resources are both minimized in the HPE DVS configuration. DVS uses the Linux-supplied virtual file system (VFS) interface to process file system access operations. This allows DVS to project any POSIX-compliant file system. When DVS is used to project external file systems, DVS operates as illustrated in the following figure. DVS can project file systems such as Spectrum Scale (formerly GPFS), NFS, and Lustre. Refer to Project an External Filesystem to Compute Nodes or Login Nodes for instructions on how to project external file systems with DVS.

DVS projection of external file systems is usually read-write data projected from gateway or dedicated DVS server nodes.

In COS 2.3, automatic DVS fail-over in not supported. However, a procedure and a helper script are provided to aid with performing a manual DVS fail-over.

2.2 Procedure To Change DVS Network Transport

This procedure documents the steps to change the network transport that DVS uses to talk between nodes. DVS can be configured to use the Node Management Network (NMN) or the High Speed Network (HSN). It can also be configured to use any of a number of different Lustre Network Drivers (LNDs). DVS also supports both Industry Standard and HPE Slingshot NICs.

This procedure will cause service outage for all DVS servers, gateways, CNs, and login nodes. Scheduling downtime is recommended when carrying out this procedure.

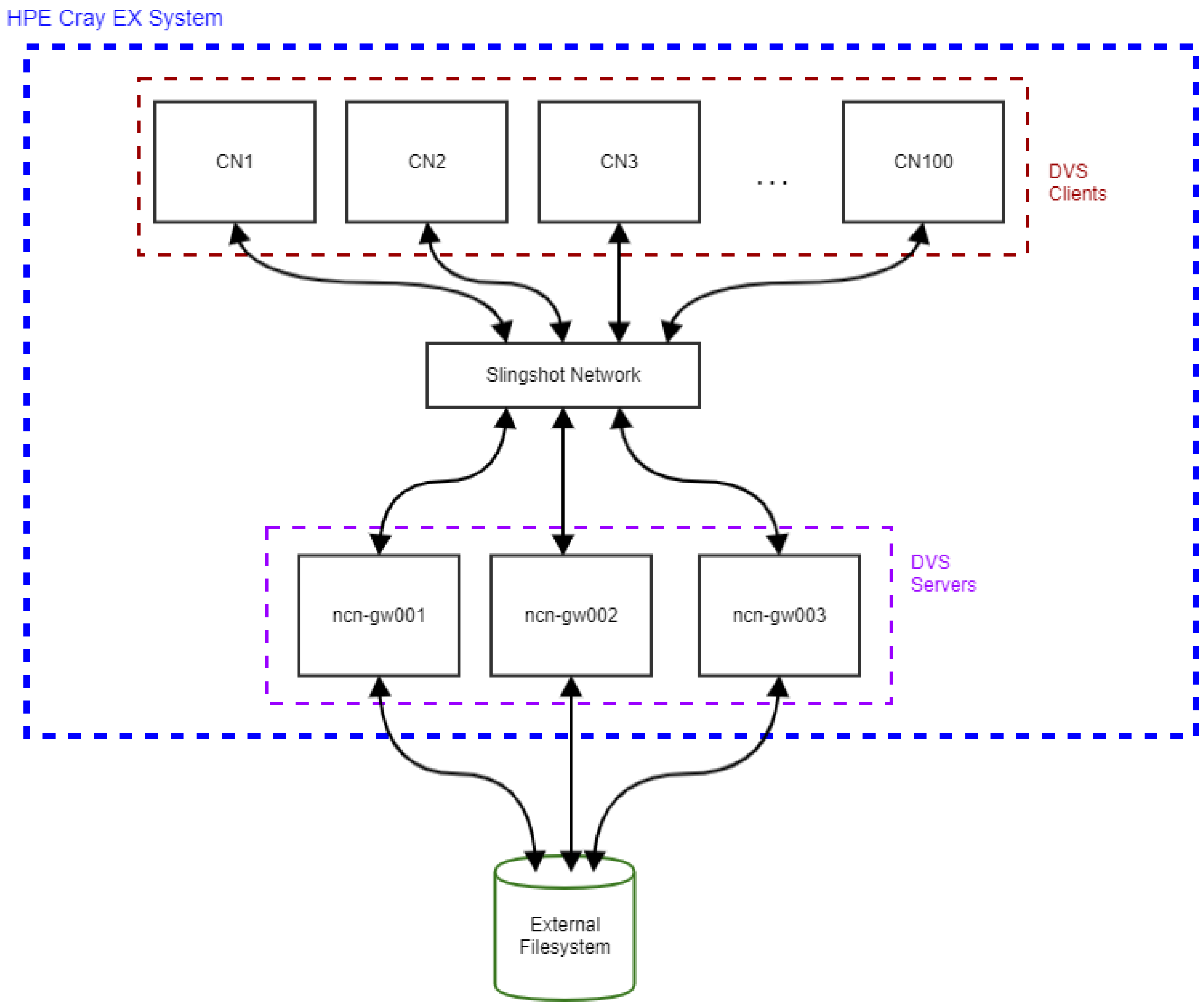

Figure 1: “DVS Projecting External Filesystems”

A Gateway node is a type of DVS server node used to project external file systems to the CNs and login nodes (DVS clients). Refer to Project an External File system to Compute Nodes or Login Nodes section for more details on the gateway nodes.

2.2.1 Change DVS Network Transport on DVS server and client Nodes

To change the DVS network transport on DVS server and client nodes, the following tasks are required:

- Make changes to a few configuration files for DVS and LNet in the COS image configuration.
- Generate a new dvs_node_map for the change in network transport.
- Update the Leader nodes with the new COS image if the system contains Leader nodes.
- Assign the new COS image configuration to the nodes running DVS.
- Reboot the DVS servers and clients that were running over the old transport.

Perform the steps below to accomplish these tasks.

1. Start a typescript to capture the commands and output from the deployment for troubleshooting if needed.

   ```
   system_name-adm# script -af dvs-over-hsn.$(date +%Y-%m-%d).txt
   system_name-adm# export PS1='\u@\H \D{%Y-%m-%d} \t \w # '
   ```

2. Copy the current working COS image.

   It is recommended to make the DVS network transport change to a copy of the working COS image. This preserves a working copy of the old DVS network transport configuration in case it is needed again.

   ```
   system_name-adm # cm image copy -o <CURRENT_WORKING_IMAGE> \
   -i <NEW_IMAGE_NAME> -g <CURRENT_WORKING_IMAGE_REPO_GROUP>
   ```

   There are two approaches for utilizing DVS in the COS image. The first is to use a single image for both DVS server and client. The second is to use separate images for the DVS server and client. The changes made to either a single image or separate DVS server and client images will be the same to change the DVS network transport. It is also possible to have different DVS client images for CNs, login, or other client node configurations. Regardless of the number of images used on a system to support DVS functionality, each of the images must have the same DVS network transport changes applied to them using this procedure on each of the images involved. The generate new node map step only needs to be performed once and the new node map can then be included in each of the DVS image employed.

3. Remove and install the LNet RPMs appropriate to the LNet transport change.

Industry Standard NICs fit best with the cray-lustre-client set of RPMs and HPE Slingshot NICs fit best with the cray-lustre-client-ofed set of RPMs. If your system has a mix of the two, it's best to use the cray-lustre-client-ofed RPMs. For the Industry Standard Mellanox NIC you will use the cray-lustre-client RPMs and for HPE Slingshot Cassini NIC 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 pachages 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

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

1. On Industry Standard NIC systems:

```
cm image zypper -i <IMAGE_NAME> --repo-group <IMAGE_NAME_REPO_GROUP> install cray-lustre-client \
cray-lustre-client-devel cray-lustre-client-kmp-cray_shasta_c
```

2. On HPE Slingshot NIC systems:

```
cm image zypper -i <IMAGE_NAME> --repo-group <IMAGE_NAME_REPO_GROUP> install cray-kfabric-kmp-cray_shasta_c \
cray-lustre-client-ofed cray-lustre-client-ofed-devel cray-lustre-client-ofed-kmp-cray_shasta_c
```

Example command to remove the industry standard NIC Lustre client RPMs

```
system_name-adm# cm image zypper -i <IMAGE_NAME> --repo-group <IMAGE_NAME_REPO_GROUP> remove \
cray-lustre-client cray-lustre-client-devel cray-lustre-client-kmp-cray_shasta_c
```

4. Make the appropriate changes to the LNet and DVS configuration files to configure the new network transport.

   DVS and LNet are configured by changing variables in files in the /etc and /etc/modprobe.d directories of the COS images. The relevant files are:

   - **/etc/lnet.conf**: It's the primary way to configure LNet. It specifies which Lustre Network Drivers (LNDs) and which network interfaces will be used.
   - **/etc/modprobe.d/lnet.conf**: This file contains the modprobe variables for the LNet modules.
   - **/etc/modules-load.d/lnet.conf**: This file controls which LNet LNDs to load with the lnet kernel module. It should contain the LNDs which are configured referenced in the lnet.conf file. The possible LNet LND kernel modules are: ksocklnd, ko2iblnd, kkfilnd.
   - **/etc/modprobe.d/dvs.conf**: This file contains the modprobe variables for the DVS modules.
   - **/etc/dvs_node_map.conf**: This file contains the JSON config that controls how the DVS node map is created. There should be only one instance of this file for the whole system, since it needs to be consistent for all nodes. On HPCM systems the dvs_node_map is generated and added to the image after a change in the LNet transport.

   The changes to the LNet and DVS configuration files may be made directly to the image directories on the hpcm-adm file system. But for examining the rpm database and systemd enable/disable of services you will need to chroot to the image directory.

   Some recipes for common network transports are shown below.

   These recipes just describe the values of the variables needed for DVS. If Lustre is involved, there may need to be additional values in the lnet-related files. See the Lustre documentation and the ***Mount a Lustre File System on GWs, CNs, and Login Nodes*** document for more information.

   a. DVS over the Node Management Network (NMN) with ksocklnd

      Which of the cray-lustre-client RPMs does not matter for this recipe. However, it will make later migration easier if it matches your hardware type. Industry Standard NICs fit best with the cray-lustre-client set of RPMs and HPE Slingshot NICs fit best with the cray-lustre-client-ofed set of RPMs. If your system has a mix of the two, it's best to use the cray-lustre-client-ofed RPMs.

      The file `/etc/lnet.conf` should look like what's below.

```
# cat /etc/lnet.conf
ip2nets:
  - net-spec: tcp99
    interfaces:
      0: bond0
```

      The file `/etc/modprobe.d/lnet.conf` should look like what's below.

```
# cat /etc/modprobe.d/lnet.conf
  options lnet lnet_transaction_timeout=120
```

      Tell systemd which LNet LNDs to load on system boot.

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

      The file `/etc/modprobe.d/dvs.conf` should look like what's below.

```
# cat /etc/modprobe.d/dvs.conf
  options dvsipc_lnet lnd_name=tcp99
```

The file `/etc/dvs_node_map.conf` should look like what's below.

```
# cat /etc/dvs_node_map.conf
{
   "config" : [
      {
         "lnet" : "tcp99",
         "nid" : 0,
         "xname" : ""
      }
   ]
}
```

b. DVS over the High-Speed Network (HSN) with ksocklnd

   Which of the cray-lustre-client RPMs does not matter for this recipe. However, it will make later migration easier if it matches your hardware type. Industry Standard NICs fit best with the cray-lustre-client set of RPMs and HPE Slingshot NICs fit best with the cray-lustre-client-ofed set of RPMs. If your system has a mix of the two, it's best to use the cray-lustre-client-ofed RPMs.

   If your system has or will have Lustre clients which will also be using ksocklnd, make the "tcp" network name match the same LNet network which Lustre will use (e.g. tcp, tcp1, etc) in the four spots below.

   The file `/etc/lnet.conf` should look like what's below.

```
  ip2nets:
    - net-spec: tcp
      interfaces:
        0: hsn0
        1: hsn1
        2: hsn2
        3: hsn3
```

   The file `/etc/modprobe.d/lnet.conf` should look like what's below.

```
  options lnet lnet_transaction_timeout=120
```

   Tell systemd which LNet LNDs to load on system boot.

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

   The file `/etc/modprobe.d/dvs.conf` should look like what's below.

```
# cat /etc/modprobe.d/dvs.conf
  options dvsipc_lnet lnd_name=tcp
```

   The file `/etc/dvs_node_map.conf` should look like what's below.

```
# cat /etc/dvs_node_map.conf
{
   "config" : [
      {
         "lnet" : "tcp",
         "nid" : 0,
         "xname" : "h0"
      }
   ]
}
```

c. DVS over the High-Speed Network (HSN) with ko2iblnd

   This is the preferred recipe to be used with Industry Standard (non-HPE Slingshot) NICs.

   The selection of Lustre RPMs **does** matter for this recipe. The Industry Standard NIC version of the cray-lnet-client RPMs should be installed.

If your system has or will have Lustre clients which will also be using ko2iblnd, make the "o2ib" network name match the same LNet network which Lustre will use (e.g. o2ib1, o2ib2, etc) in the four spots below.

The file `/etc/lnet.conf` should look like what's below.

```
ip2nets:
  - net-spec: o2ib
    interfaces:
      0: hsn0
      1: hsn1
      2: hsn2
      3: hsn3
```

The file `/etc/modprobe.d/lnet.conf` should look like what's below.

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

Tell systemd which LNet LNDs to load on system boot.

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

The file `/etc/modprobe.d/dvs.conf` should look like what's below.

```
# cat /etc/modprobe.d/dvs.conf
  options dvsipc_lnet lnd_name=o2ib
```

The file `/etc/dvs_node_map.conf` should look like what's below.

```
# cat /etc/dvs_node_map.conf
{
   "config" : [
      {
         "lnet" : "o2ib",
         "nid" : 0,
         "xname" : "h0"
      }
   ]
}
```

d. DVS over the High-Speed Network (HSN) with kkfilnd

This is the preferred recipe to be used with HPE Slingshot NICs.

The selection of Lustre RPMs **does** matter for this recipe. The HPE Slingshot NIC version of the cray-lustre-client-ofed RPMs should be installed.

If your system has or will have Lustre clients which will also be using kkfilnd, make the "kfi" network name match the same LNet network which Lustre will use (e.g. kfi1, kfi2, etc) in the four spots below.

The file `/etc/lnet.conf` should look like what's below.

```
net:
  - net type: kfi
    local NI(s):
      - interfaces:
          0: cxi0
      - interfaces:
          0: cxi1
      - interfaces:
          0: cxi2
      - interfaces:
          0: cxi3
```

The file `/etc/modprobe.d/lnet.conf` should look like what's below.

```
options lnet lnet_transaction_timeout=120
```

Tell systemd which LNet LNDs to load on system boot.

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

The file `/etc/modprobe.d/dvs.conf` should look like what's below.

```
# cat /etc/modprobe.d/dvs.conf
  options dvsipc_lnet lnd_name=kfi
```

The file `/etc/dvs_node_map.conf` should look like what's below.

```
# cat /etc/dvs_node_map.conf
{
   "config" : [
      {
         "lnet" : "kfi",
         "nid" : 0,
         "xname" : "h0"
      }
   ]
}
```

Note: It's important that each node has all of their cxiX interfaces configured. They should all be in the "/etc/lnet.conf" file for each node. This makes sure that the LNet peer discovery code is able to allow nodes to talk to each other. If only some of the interfaces are configured, some nodes may have problems communicating via DVS.

This means that if your system has different classes of nodes with different numbers of HPE Slingshot NICs, you want to create different node images. Each node image should have a "/etc/lnet.conf" file that lists the correct number of cxiX interfaces for that node class.

5. Generate a new dvs_node_map for the change in network transport. See How to Populate Node Maps

   If your system employs multiple images to support DVS functionality, then the generate new node map step only needs to be performed once and the new node map can then be included in each of the DVS images employed.

6. If your HPCM system contains Leader nodes, then copy the updated image to the Leader nodes using the `cm activate -i <UPDATED_IMAGE>`. Otherwise skip this step and continue with the next.

7. Assign the updated image to the desired set of DVS running nodes. Replace NODE_LIST in the following command with comma separated list of gateway nodes that will use the new gateway image.

   ```
   system_name-adm # cm node set --image <UPDATED_IMAGE> -n <NODE_LIST>
   ```

8. Reboot the DVS servers and clients to run over the new transport. This step can also be done on multiple DVS servers and clients in parallel. Replace NODE_LIST in the following command with comma separated list of DVS running nodes that will use the COS image with the new transport configured.

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

9. Verify that DVS and LNet services are running over the new transport.

   In our example below, the HSN uses ksocklnd, where DVS is supposed to use LNet network "tcp" on interface "hsn0" for the DVS running nodes:

   ```
   system_name-adm # pdsh -w dvs[001-NNN] lsmod | grep -P '\bdvs\b' | dshbak -c
   ----------------
   dvs[001-003]
   ----------------
   dvs                    425984  0
   dvsipc                 188416  2 dvs
   dvsproc                151552  3 dvsipc,dvsipc_lnet,dvs
   craytrace               20480  5 dvsipc,dvsproc,dvsipc_lnet,dvs
   dvskatlas               24576  4 dvsipc,dvsproc,dvsipc_lnet,dvs
   system_name-adm# pdsh -w dvs[001-NNN],x1000c0s0b[0,1]n0 lnetctl net show | grep -B 5 'hsn0' | dshbak -c
   ----------------
   dvs001
   ```

```
----------------
    - net type: tcp
       local NI(s):
        - nid: 10.253.0.6@tcp
          status: up
          interfaces:
              0: hsn0
----------------
dvs002
----------------
    - net type: tcp
      local NI(s):
        - nid: 10.253.0.11@tcp
          status: up
          interfaces:
              0: hsn0
----------------
dvs003
----------------
    - net type: tcp
      local NI(s):
        - nid: 10.253.0.24@tcp
          status: up
          interfaces:
              0: hsn0
----------------
x1000c0s0b0n0
----------------
    - net type: tcp
      local NI(s):
        - nid: 10.253.0.14@tcp
          status: up
          interfaces:
              0: hsn0
----------------
x1000c0s0b1n0
----------------
    - net type: tcp
      local NI(s):
        - nid: 10.253.0.23@tcp
          status: up
          interfaces:
              0: hsn0
```

10. Finish the typescript file started at the beginning of this procedure.

~~~
```bash
system_name-adm# exit
```
~~~

2.3 Tune and Optimize DVS

An overview of the different ways HPE Cray EX administrators can optimize the performance of DVS.

This section contains procedures administrators can use to increase the performance of DVS on HPE Cray EX systems. Consult the following table to find common DVS optimization procedures:

If:	Then try:
The I/O pattern of an application exhibits a "read many, write once" file I/O pattern	Enabling `ro_cache` for the DVS client mounts of that application. See When to Use Temporary, Client-Side, Per-File Read Caching
DVS client nodes will not write to a DVS mount	Configure Read-Only Client Mounts for Optimal Performance
Optimal I/O performance of a single application is wanted over uniform performance of all DVS clients	Universally Disable Fairness of Service
DVS client nodes will mount an external IBM Spectrum Scale (formerly GPFS) file system	Improve Performance and Scalability of Spectrum Scale (GPFS) Mounts

Gathering and comparing DVS performance statistics is useful for DVS tuning. See Collect and Analyze DVS Statistics for more details on how to collect this information.

2.3.1 Configure Read-Only Client Mounts for Optimal Performance

HPE Cray EX administrators can increase the I/O performance of read-only DVS client mounts by enabling `loadbalance` mode and by leaving the `attrcache_timeout` at the default of `14400`. The procedure for setting these options for external file systems differs from the procedure for internal file systems.

- Confirm that DVS client (compute) nodes will not write to the DVS-projected file system.
- Confirm that the data in the projected file system rarely changes, if ever.

When DVS client (compute) nodes require read-only access to a file system, and the data in the source file system never or infrequently changes, HPE Cray EX administrators can increase DVS I/O performance. This procedure describes how to configure client mounts with the `loadbalance` option enabled and an `attrcache_timeout` set to the default value of `14400` seconds (four hours).

This procedure applies to both external file systems (for example, Spectrum Scale, NFS) as well as internal mounted file systems.

1. Determine if the file system to be mounted read-only is internal or external to the HPE Cray EX system.

2. Perform one of the following procedures:

   - If the read-only file system is external, follow the procedure in Project an External Filesystem to Compute Nodes or Login Nodes and follow the guidance for a read-only file system.
   - If the read-only file system is internal, follow the procedure in Mount a Internal Read-Only Filesystem Optimally to mount the file system manually. File systems mounted using this method do not persist between node reboots.
   - For projecting a persistent internal read-only file system to clients use the Mount a Internal Read-Only Filesystem Optimally procedure, but instead of working on the booted nodes, make the changes to the image on the system admin node, then activate and boot the nodes with the modified images.

2.3.1.1 Mount an Internal Read-Only Filesystem Optimally

Mount internal read-only file systems in a way that optimizes DVS I/O performance. If DVS client nodes need read-only access to rarely or never changing data stored on an internal file system, use this procedure.

- **OBJECTIVE**

  Quickly and manually use DVS to project a file system to a client.

- **LIMITATIONS**

  Client mounts created this way do not persist across reboots of the DVS servers or CNs.

HPE Cray EX system administrators can configure client nodes to take advantage of the performance benefits available for DVS-projected read-only file systems. This procedure is for situations where such file systems:

- Are not the root file system images for compute nodes.
- Are not stored externally to the HPE Cray EX system. For external (such as IBM Spectrum Scale or NFS) read-only file systems, refer to Project an External Filesystem to Compute Nodes or Login Nodes.

Mounting a directory containing software shared libraries is a typical use case.

1. On the booted image of the DVS server or Gateway nodes, create a directory in which to copy the read-only content. Modify the `/etc/dvs_exports.yml` file to add the content directory to be exported read-only.

   ```
   dvs-01# mkdir /opt/my_shared_library
   dvs-01# cd /opt/my_shared_library; tar -xf /tmp/my_shared_library.tgz
   dvs-01# vi /etc/dvs_exports.yml
   dvs-01# cat /etc/dvs_exports.ynl
     exports:
     - mode: ro
       path: /opt/my_shared_library
   ```

2. Mount the directory on the DVS clients using the `mount` command. The `pdsh` command may be used from the system admin node for any more than a few nodes to add the mount. Replace *SOURCE_PATH* in the following command with the directory created in the previous step, and *MOUNT_POINT* with the wanted location on the client node file system.

   ```
   cn-01# mkdir MOUNT_POINT; mount -t dvs -o path=SOURCE_PATH,nodename=dvs-01:dvs-02,noclusterfs,
   maxnodes=1,attrcache_timeout=3,loadbalance SOURCE_PATH MOUNT_POINT
   ```

   Or using `/etc/fstab` for mounting.

   ```
   cn-01# mkdir MOUNT_POINT
   cn-01# echo "SOURCE_PATH MOUNT_POINT dvs path=SOURCE_PATH,nodename=dvs-01:dvs-02,noclusterfs,
   maxnodes=1,attrcache_timeout=3,loadbalance 0 0" >> /etc/fstab
   cn-01# mount -a
   ```

   Or using `pdsh`

   ```
   system_name-adm# pdsh -w NODE_LIST 'mkdir MOUNT_POINT; mount -t dvs -o path=SOURCE_PATH,
   nodename=dvs-01:dvs-02,noclusterfs,maxnodes=1,attrcache_timeout=3,loadbalance SOURCE_PATH MOUNT_POINT'
   ```

   NODE_LIST is the comma separated list of DVS client host names that the file system mount is desired on.

2.3.2 Improve Performance and Scalability of Spectrum Scale (GPFS) Mounts

How to configure both read-only and read/write Spectrum Scale (formerly known as GPFS) DVS client mounts to improve performance and scalability.

2.3.2.1 Optimize DVS Read/Write Mounts of External IBM Spectrum Scale Filesystems

To improve the performance of readable and writable external IBM Spectrum Scale file systems projected by DVS, ensure that:

- The client mount option `blksize` is equal to, or a multiple of, the Spectrum Scale file system blocksize. Otherwise, degraded performance will result.
- If more than one external Spectrum Scale file system will be projected by DVS, and those file systems have different blocksizes, configure a separate DVS client mount (with an appropriate `blksize`) for each Spectrum Scale file system that has a different blocksize.
- All other guidance for optimizing DVS for file system performance in this section is followed as appropriate for the specific site and IBM Spectrum Scale file systems.

2.3.2.2 Optimize DVS Read-Only Mounts of External IBM Spectrum Scale Filesystems

When HPE Cray EX compute nodes require highly scalable read-only access to static (or rarely changing) data stored in an external IBM Spectrum Scale file system, such as shared libraries and input files, Hewlett Packard Enterprise recommends enabling loadbalance mode and specifying `attrcache_timeout=14400` in the mount options. Following the procedure in Project an External Filesystem to Compute Nodes or Login Nodes to do so.

2.3.2.3 How to Handle Multiple IBM Spectrum Scale Mounts with Different Settings

Having separate mounts configured with different options has the advantage of providing maximum performance and scalability of access. However, having more than one mount has the disadvantage of requiring users, jobs, and applications to know which mount (path) to use when. This disadvantage could be mitigated by setting environment variables for each path (for example, setting `LD_LIBRARY_PATH` to use the read-only/read-write path).

2.3.3 Universally Disable Fairness of Service

Disable DVS fairness of service within an HPE Cray EX system by editing the DVS server settings for `dvs.conf` files within the HPCM configuration management repository.

Read DVS Fairness of Service to understand the benefits of leaving Fairness of Service enabled.

- **OBJECTIVE**

  Disable DVS fairness of service for all Gateways within an HPE Cray EX system.

DVS fairness of service is a server-only setting. Changing either `single_msg_queue` or the sixth field in `dvs_instance_info` on DVS clients has no effect.

1. Perform Select DVS image from VCS for the image utilized by the Gateway nodes in the system.

   Disable fairness of service in dvs.conf

2. Inspect the `dvs.conf` file in all the gateway images found in the `/opt/clmgr/image/images` directory. Repeat the next two steps for all the instances of the DVS Gateway node images found in this directory.

   The default configuration is one `dvs.conf` file for all nodes, but site-specific needs may require a custom configuration and thus separate images for DVS servers, CNs, login, and Gateway nodes.

3. Open the file `dvs.conf` in an editor. For example:

   ```
   system_name-adm# vi /opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>/etc/modprobe.d/dvs.conf
   ```

4. Inspect the file for any lines containing `options dvsipc dvs_instance_info=`.

   Since the `dvs_instance_info` kernel module parameter values override any separately set parameters (such as `single_msg_queue`), it must be used to disable fairness of service if it is present.

5. Edit the `dvs.conf` file using either of the following methods:

   - If `options dvsipc dvs_instance_info` is present, then set the `single_msg_queue` field (the sixth number) to 1.
   - If `options dvsipc dvs_instance_info` is absent, then add the line `options dvsipc dvsipc_single_msg_queue=1`.

   For example:

   ```
   system_name-adm # cd /opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>
   system_name-adm # cat etc/modprobe.d/dvs.conf
     options dvsipc dvsipc_single_msg_queue=1
   ```

6. Perform Commit DVS image changes to VCS

   Force HPE Cray EX system to immediately start using the updated configuration

7. Reboot the Gateway nodes.

2.4 DVS Statistics

DVS can collect and report a large amount of detailed statistics about DVS performance and errors. HPE Cray EX administrators can use these statistics for system monitoring, performance tuning, and troubleshooting.

2.4.1 Collect and Analyze DVS Statistics

Aggregate and per-mount DVS statistics are available for client and server nodes. DVS users and admins can use this data for performance tuning and root-cause analysis of issues.

This procedure walks administrators through the process of obtaining and using DVS statistics.

1. Determine the DVS statistics that report the data of interest and decide if aggregate or per-mount statistics must be collected.

   Refer to DVS Statistics Collected for a comprehensive listing of the statistics DVS collects.

2. Determine the files which contain that statistics identified in the previous step and the specific nodes those files must be read from.

   Refer to Files That Contain DVS Statistics for guidance on selecting the correct `/sys/kernel/debug/` file on the correct node to obtain the information wanted.

3. Decide on an output format for the DVS statistics and determine the format and control options which produce that wanted format. Decide if these options should be set at boot time or after boot.

   See Control and Format Options for DVS Statistics for a list of all the control and format options available, descriptions for each of them, and how to set these options during boot or after a node is booted.

4. Set the control and format options as decided in the previous step.

5. Collect the statistics.

   - Manually read out the contents of appropriate stats file.
   - Use a script to read out one or more stats files. For per-mount statistics, this can be done by parsing the output of `mount -t` dvs and obtaining the absolute path to the right of `statsfile=`. That path is the stats file to read from for that mount. In future releases, HPE may change the order of lines and may deprecate (or remove as obsolete) individual statistics. Therefore, commands and scripts may sometimes require modification after system software upgrades.

6. Use Caveats for Interpreting DVS Statistics to help analyze and interpret the results.

   All output is readable (ASCII) and in a self-describing format.

2.5 Control and Format Options for DVS Statistics

DVS administrators can disable or enable DVS statistics collection, reset the statistics counters, and as well as control the output format that data is reported in.

2.5.0.1 Control and Format Options

Use the options and values described in the following table to control the collection and reporting of statistics.

- Multiple options can be specified, separated by commas, semicolons, spaces, or line-feeds.
- If incompatible options within the same category are specified (for example, "`verbose,brief`"), the last option specified will govern.
- If incompatible options from different categories are specified (for example, "`disable,json`"), the option in a higher-level category will take precedence. The order of precedence (decreasing) for option categories is: `control`, `format`, `flags`. For example, control options such as `disable` take precedence over format options such as `json`.
- Options not specified will continue to use their existing values.

Option Category	Option Value	Result
control	disable	Disables statistics collection and reporting.
control	enable	Enables statistics collection and reporting.
control	reset	Resets statistics.
format	help	Displays current information about the statistics values collected and reported.
format	flat	Displays statistics in flattened text format.
format	json	Displays statistics in JSON structured format.

Option Category	Option Value	Result
flags	brief | verbose	`brief` omits statistics that have not been collected. `verbose` displays all statistics, even those that have not been collected. Applies to `flat` and `json` formats only.
flags	plain | pretty	`plain` uses a more machine-readable presentation. `pretty` uses a more human-readable presentation. Applies to `flat` and `json` formats only.
flags	test | notest	`test` overrides data with a test pattern, which means the statistics data is replaced with fixed test data designed to be used for regression testing of the formatting alone. `notest` removes the test pattern override and restores normal operation.

These combinations of format and flag options are common:

- **flat, plain**

  Structured data is displayed as key/value pairs, one pair per line, key and value separated by a single space. The format is:

  ```
  [container.[container.]...]key value[,value[,value...]]
  ```

  Multiple comma-separated values represent an array of values.

- **flat, pretty**

  Same as flat, plain except that the single space is expanded to produce columns to enhance readability.

- **json, plain**

  Structured data is displayed in JSON format. It consists of a single, unnamed JSON object that contains key/value pairs. There are no spaces or line-feed characters to enhance readability.

- **json, pretty**

  Same as json, plain except that spaces and line-feeds are added to enhance readability.

The incompatible options `list flat,pretty,disable` resolves to `disable` and turns off stats.

2.5.0.2 How to use the DVS statistics control and format options

By default, DVS statistics are enabled, collected, and reported in a simple text format. Statistics can be controlled by:

- Specifying a module parameter at module load time
- Writing text values into the corresponding stats file (see Files That Contain DVS Statistics for a list of these)

Use the `dvsdebug_stat_defaults` parameter to enable and control DVS statistics.

- **dvsdebug_stat_defaults**

  Sets the DVS statistics reporting options to default values at module load time. Every new stats file will take these defaults, which can be changed later at any time for each stats file. Use this parameter to disable, enable, and format DVS statistics.

  - Default value: enable
  - To view: `cat /sys/module/dvsproc/parameters/dvsdebug_stat_defaults`
  - To change prior to boot, add this line to `/etc/modprobe.d/dvs-local.conf` and replace *`options_list`* with a comma-separated list of options:

    ```
    # Disable/enable and format DVS statistics
    options dvsdebug dvsdebug_stat_defaults="options_list"
    ```

    Note: See HPE Cray EX DVS Configuration Overview for more details.

  - To change dynamically:

    This is root writable at `/sys/module/dvsproc/parameters/dvsdebug_stat_defaults`, but changes should be made only through the `/sys/kernel/debug/dvs/stats` interface, as shown in this example.

    ```
    hostname# echo disable > /sys/kernel/debug/dvs/stats
    hostname# echo enable > /sys/kernel/debug/dvs/stats
    hostname# echo reset > /sys/kernel/debug/dvs/stats
    hostname# echo json,pretty > /sys/kernel/debug/dvs/stats
    ```

2.5.1 Caveats for Interpreting DVS Statistics

DVS users and admins who collect and use DVS performance statistics must be aware of a few caveats when interpreting that data.

2.5.1.1 Statistics and Caching

DVS statistics measure I/O that passes through DVS. In general, all `read`, `aio_read`, `write`, and `aio_write` operations specified by the user application result in a DVS `read`, `aio_read`, `write`, or `aio_write` operation. DVS records the total bytes reported to the application for each read or write, and the maximum file offset accessed. These statistics are recorded regardless of whether the file system is cache-enabled.

If client-side caching is disabled, each read or write operation is forwarded to one or more servers. This results in a file system read or write on the server and a transfer of data between client and server. The accumulated totals in the statistics represent actual load on the DVS transport.

If client-side caching is enabled, the read or write operation is not forwarded to one or more servers. Instead, it is diverted to a Linux routine that attempts to satisfy the read or write using the Linux file system cache.

- If the data is already in the cache, Linux completes this operation entirely in memory. The accumulated totals in the statistics represent no load on the DVS transport.
- If the data is not already in the cache, Linux issues a page read or write operation to DVS, which results in that read or write sent to one or more servers. DVS statistics record the total bytes reported for this cache read or write.

The effectiveness of the cache is represented by the ratio of user read or write bytes to cache read or write bytes.

- **ratio > 1.0**

  Indicates that the cache is being used effectively, with more cache hits than misses.

- **ratio = 1.0**

  Represents something like sequential reads of a large file: reads force continual cache fills, but the reads then consume all the bytes in the cache before forcing a new cache fill.

- **ratio < 1.0**

  Indicates that cache is thrashing, such as would be caused by making small reads from many open files, such that only a tiny fraction of each cache page is ever consumed.

It is not feasible to determine whether any specific user application read or write "hit" or "missed" the cache, without modifying the Linux kernel.

2.5.1.2 Statistics and the `mmap()` Functions

The `mmap()` functions use the same mechanisms as the Linux cache: mmap can be thought of as a named, "private" Linux cache that the application sets up. The `mmap()` functions cannot be used unless the DVS file system is cache-enabled. As a result, any attempt to read from or write to the file is diverted to the same Linux routine as would be used for any caching. The Linux routine determines that this is a `mmap()` file and sends a page request to DVS, then satisfies the read or write from memory. As in the case of caching, DVS statistics track the reads and writes of the user application and the Linux page requests separately. As with caching, it cannot be determined whether a specific read or write "hit" or "missed" the mmap.

Because it is difficult to link a user application read or write with a corresponding page read or write, which is usually a many-to-one linkage, it is difficult to distinguish between a Linux file system cache operation and an `mmap()` cache operation.

2.5.1.3 Rates and Averaging

Performance rate measurements are simply counts that are automatically zeroed when sampled by reading the stats file. The resulting total is displayed, divided by the time since the last stats file read. Therefore it is possible to create real-time performance data with one-second resolution by reading the stats file every second. Because the DVS code normalizes the value by dividing by the time elapsed since the last sample, variations in the sampling rate should not be reflected in the averages. For example, if the actual data transfer rate is a steady 1GB/sec, a rate of 1GB/sec will be displayed every time the stats file is sampled, even if it is sampled manually at random intervals. If the actual data transfer rate is fluctuating, all the fluctuating rates will be averaged together (unweighted) over the entire sampling window, however long that may be.

2.5.1.4 Races

To keep DVS performance high, statistics reporting is not atomic: atomic values are sampled as each line of output is rendered, while DVS is running. As a result, counts that might be expected to have an exact mathematical relationship may not. For example, if all applications are reading in fixed-size blocks, one might expect an exact relationship between IOPS and bytes read. However, that relationship will not generally be exact, because values are sampled as the statistics output is rendered, and values rendered later in time will contain new counts not found in values rendered earlier.

2.5.2 Files That Contain DVS Statistics

DVS records data about its performance (both aggregate and per mount point) in several files in `/sys/kernel/debug/`. To collect and report DVS statistics, read from these files.

DVS provides both aggregate and per-mount statistics to enable performance and root-cause analysis. It reports statistics for client and server nodes in the following files. Each stats file is readable and writable.

- **`/sys/kernel/debug/dvs/statistics/`**

  Contains two files with statistics for the average timing of messages between the DVS client and the DVS server. Including:

  - Time spent queued on the server.
  - Time spent being processed by the server.
  - Time spent in the underlying file system
  - Time spent on the network and in network transport software
  - **`client_message_timings`**

    The `client_message_timings` file includes timing information for all requests and replies that the node sends or receives as a client. On servers, the `client_message_timings` file will likely be mostly zeroes.
  - **`server_message_timings`**

    The `server_message_timings` file includes timing information for all messages that a node receives as a server. On clients, the `server_message_timings` file will likely be mostly zeroes.

- `/sys/kernel/debug/dvs/stats`

  Contains aggregate statistics for the node. These reflect system operations that cannot be correlated to a specific DVS mount point and are therefore most interesting on DVS servers.

- `/sys/kernel/debug/dvs/mounts/NNN/stats`

  Contains per-mount statistics for the node, where *NNN* is a decimal integer value global to DVS on that node. Every invocation of the mount command creates the numbered mount directory and increments *NNN*. Every invocation of the umount command deletes the numbered mount directory but does not decrement *NNN*. The value of *NNN* appears in the `statsfile=/sys/kernel/debug/dvs/mounts/NNN/stats` parameter in the mount options for the mounted DVS file system. This can be obtained using the `mount -t dvs` command. More information about each of these mount points can be obtained by viewing the mount file that resides in the same directory (`/sys/kernel/debug/dvs/mounts/NNN/mount`).

- `/sys/kernel/debug/dvsipc/stats`

  Contains DVS interprocess communication (IPC) statistics, including bytes transferred and received, and NAK counts. It also contains message counts by type and size.

2.5.3 DVS Statistics Collected

Lists all the statistics collected by DVS.

The following table shows the full keys in flat format and describes whether the statistic is aggregate across all mounts (`/sys/kernel/debug/dvs/sta`
or per-mount (`/sys/kernel/debug/dvs/mounts/NNN/stats`), or both.

2.5.3.1 dvs statistics table

Key	AGG	MNT	Meaning
STATS.version	*	*	Current version of statistics. Anytime this number changes, the format or content of the statistics have changed. Because DVS uses a self-describing format (key/value or JSON), analysis code may be able to run with new versions without code changes; however, there is no guarantee.
STATS.flags	*	*	Current option flags.
RQ.req.req.ok	*	*	Count of requests sent by this host that resulted in successful response. See the table Valid req Values for valid values for the RQ keys.
RQ.req.req.err	*	*	Count of requests sent by this host that resulted in a failed send or a failure response.
RQ.req.reqp.ok	*	*	Count of requests received by this host that resulted in successful action on this host.
RQ.req.reqp.err	*	*	Count of requests received by this host that resulted in failed action on this host.
RQ.req.reqp.dur.prv	*	*	Duration (seconds) of the most recent request received and processed by this host.
RQ.req.reqp.dur.max	*	*	Maximum duration (seconds) of any requests received and processed by this host.
OP.fileop.ok	*	*	Count of successful file operations called by Linux on this host. See the table Valid fileop Values for valid values for the OP keys.
OP.fileop.err	*	*	Count of failed file operations called by Linux on this host.
OP.fileop.dur.prv	*	*	Duration (seconds) of the most recent file operation on this host.
OP.fileop.dur.max	*	*	Maximum duration (seconds) of any file operations on this host.

Key	AGG	MNT	Meaning
IPC.requests.ok, IPC.requests.err, IPC.async_requests.ok, IPC.async_requests.err, IPC.replies.ok, IPC.replies.err	*		These have the same meanings as the preceding RQ.req.req.ok and RQ.req.req.err keys, except for these differences: 1) IPC keys show aggregated messages across all mounts (RQ keys show individual messages) 2) IPC keys grouped by whether messages sent synchronously, asynchronously, or as a reply message.
PERF.user.read.min_len		*	Low-water mark of all nonzero-length user read operations. Includes both read() and aio_read() calls.
PERF.user.read.max_len		*	High-water mark of all nonzero-length user read operations.
PERF.user.read.max_off		*	High-water mark of all file byte offsets read by user calls.
PERF.user.read.total.iops		*	Accumulated I/O operation count of all user read operations since module load.
PERF.user.read.total.bytes		*	Accumulated byte transfer count of all user read operations since module load.
PERF.user.read.rate.iops		*	Accumulated I/O operation count of all user read operations since the last read of stats, divided by the number of seconds since the last read of stats.
PERF.user.read.rate.bytes		*	Accumulated byte transfer count of all user read operations since the last read of stats, divided by the number of seconds since the last read of stats.
PERF.user.write.min_len		*	Low-water mark of all nonzero-length user write operations. Includes both write() and aio_write() calls.
PERF.user.write.max_len		*	High-water mark of all nonzero-length user write operations.
PERF.user.write.max_off		*	High-water mark of all file byte offsets written by user calls.
PERF.user.write.total.iops		*	Accumulated I/O operation count of all user write operations since module load.
PERF.user.write.total.bytes		*	Accumulated byte transfer count of all user write operations since module load.
PERF.user.write.rate.iops		*	Accumulated I/O operation count of all user write operations since the last read of stats, divided by the number of seconds since the last read of stats.
PERF.user.write.rate.bytes		*	Accumulated byte transfer count of all user write operations since the last read of stats, divided by the number of seconds since the last read of stats.
PERF.cache.read.min_len		*	Low-water mark of all nonzero-length Linux cache read operations.
PERF.cache.read.max_len		*	High-water mark of all nonzero-length Linux cache read operations.
PERF.cache.read.max_off		*	High-water mark of all file byte offsets read by Linux cache calls.
PERF.cache.read.total.iops		*	Accumulated I/O operation count of all Linux cache read operations since module load.
PERF.cache.read.total.bytes		*	Accumulated byte transfer count of all Linux cache read operations since module load.
PERF.cache.read.rate.iops		*	Accumulated I/O operation count of all Linux cache read operations since last read of stats, divided by the number of seconds since the last read of stats.
PERF.cache.read.rate.bytes		*	Accumulated byte transfer count of all Linux cache read operations since last read of stats, divided by the number of seconds since the last read of stats.

Key	AGG	MNT	Meaning
PERF.cache.write.min_len		*	Low-water mark of all nonzero-length Linux cache write operations.
PERF.cache.write.max_len		*	High-water mark of all nonzero-length Linux cache write operations.
PERF.cache.write.max_off		*	High-water mark of all file byte offsets written by Linux cache calls.
PERF.cache.write.total.iops		*	Accumulated I/O operation count of all Linux cache write operations since module load.
PERF.cache.write.total.bytes		*	Accumulated byte transfer count of all Linux cache write operations since module load.
PERF.cache.write.rate.iops		*	Accumulated I/O operation count of all Linux cache write operations since the last read of stats, divided by the number of seconds since the last read of stats.
PERF.cache.write.rate.bytes		*	Accumulated byte transfer count of all Linux cache write operations since the last read of stats, divided by the number of seconds since the last read of stats.
PERF.legacy.inodes.created		*	Accumulated count of inodes created on this host (includes mirrored inodes that represent existing files on the server).
PERF.legacy.inodes.deleted		*	Accumulated count of inodes deleted on this host.
PERF.files.created		*	Accumulated count of regular files created on server file systems.
PERF.files.deleted		*	Accumulated count of regular files deleted on server file systems.
PERF.files.open		*	Current count of DVS files open.
PERF.symlinks.created		*	Accumulated count of symlinks created on server file systems.
PERF.symlinks.deleted		*	Accumulated count of symlinks deleted on server file systems.
PERF.directories.created		*	Accumulated count of directories created on server file systems.
PERF.directories.deleted		*	Accumulated count of directories deleted on server file systems.

2.5.3.2 DVS Messages

The following table shows the valid req values to be used in the RQ statistics, as described in the table dvs statistics table. They represent messages passed between the DVS client and server.

2.5.3.3 Valid req values table

req	Meaning
RQ_CLOSE	Close open file on the server.
RQ_CREATE	Create a regular file on the server.
RQ_FASYNC	Manage asynchronous notifications of a file descriptor on the server.
RQ_FLUSH	Close an instance of an open file on the server.
RQ_FSYNC	Flush a file to backend storage on the server.
RQ_GETATTR	Get file attributes on the server.
RQ_GETEOI	Return size of file on the server.
RQ_GETXATTR	Get file extended attributes on the server.
RQ_IOCTL	Perform a general ioctl on the server.
RQ_LINK	Create a link from one path to another on the server.
RQ_LISTXATTR	List extended attributes on the server.
RQ_LOCK	Lock a file on the server.
RQ_LOOKUP	Determine if a path exists on the server.

req	Meaning
RQ_MKDIR	Create a directory on the server.
RQ_MKNOD	Create a special device file on the server.
RQ_OPEN	Open a file on the server.
RQ_PARALLEL_READ	Read from an open file on the server.
RQ_PARALLEL_WRITE	Write to an open file on the server.
RQ_PERMISSION	Determine if user has permission to an object on the server.
RQ_READDIR	Read the contents of a directory on the server.
RQ_READLINK	Read the contents of a symlink on the server.
RQ_READPAGES_RP	Client only - data for page cache file read has been returned.
RQ_READPAGES_RQ	Read from an open file on the server that resides in the client page cache.
RQ_REMOVEXATTR	Remove extended attributes on the server.
RQ_RENAME	Rename a file on the server.
RQ_RMDIR	Remove a directory on the server.
RQ_RO_CACHE_DISABLE	Disable the DVS ro_cache option for an open file on the server.
RQ_SETATTR	Set basic attributes of an object on the server.
RQ_SETXATTR	Set extended attributes of an object on the server.
RQ_STATFS	Perform a file stat on the server.
RQ_SYMLINK	Create a symlink on the server.
RQ_SYNC_UPDATE	Return list of inodes that have been synced to backend storage on server recently.
RQ_TRUNCATE	Truncate an open file on the server.
RQ_UNLINK	Unlink (delete) a file object on the server.
RQ_VERIFYFS	Verify that a path exists on the server so that it can be mounted on a client.
RQ_WRITEPAGES_RP	Client only - data for page cache file write has been written.
RQ_WRITEPAGES_RQ	Write from an open file residing in the client page cache to the server.

2.5.3.4 Virtual File System Operations

The following list contains the valid fileop values to be used in IPC statistics, as described in the table dvs statistics table. They are virtual file system (VFS) operations requested by the Linux VFS framework, and they represent file system actions to be performed by the kernel on behalf of a user-space application. For more details on these VFS operations, refer to https://www.kernel.org/doc/html/latest/filesystems/vfs.html or to the corresponding page for the kernel version used on the CNs and NCNs.

For example, a call to `open()` in a user application results in a call to the open handler (fileop = open in the following list), requesting that the DVS file system open a file. VFS operations may operate locally (for example, out of local cache memory), or they may result in one or more request messages sent to the server. The `d_`, `f_`, and `l_` prefixes are vestigial references to the file system object on which the operation is performed. They will be removed in a future release.

2.5.3.4.1 Valid fileop values list

- `aio_read`
- `d_rmdir`
- `flock`
- `readdir`
- `aio_write`
- `d_setattr`
- `flush`
- `readpage`
- `d_create`
- `d_setxattr`
- `fsync`
- `readpages`
- `d_getattr`
- `d_symlink`
- `l_follow_link`
- `release`
- `d_getxattr`
- `d_unlink`

- `l_getattr`
- `show_options`
- `d_link`
- `direct_io`
- `l_put_link`
- `statfs`
- `d_listxattr`
- `evict_inode`
- `l_readlink`
- `unlocked_ioctl`
- `d_lookup`
- `f_getattr`
- `l_setattr`
- `write`
- `d_mkdir`
- `f_getxattr`
- `llseek`
- `write_begin`
- `d_mknod`
- `f_listxattr`
- `lock`
- `write_end`
- `d_permission`
- `f_removexattr`
- `mmap`
- `writepage`
- `d_removexattr`
- `f_setattr`
- `open`
- `writepages`
- `d_rename`
- `f_setxattr`
- `put_super`
- `d_revalidate`
- `fasync`
- `read`

2.6 Choose and Configure a DVS Mode

This procedure helps administrators of HPE Cray EX supercomputers select and configure the best DVS mode for projecting a new file system. The best mode for projecting a particular file system depends on:

- The amount of load distribution wanted.
- The performance and other requirements of the codes that use the data in that file system.
- The type and properties of the projected file system.

1. Determine the requirements and properties of the file system to be projected.

    - Is POSIX read/write atomicity required?
    - Is the file system a cluster (that is, shared) file system?
    - Must the file system provide high I/O performance for reads, writes, or both?
    - Will DVS clients mount the file system read-only?
    - How large is the file system?

2. Select the most appropriate DVS mode for the file system by using the following list and DVS Modes.

    - **Use Serial Mode** to project a non-cluster file system that requires POSIX read/write atomicity from a single DVS server node.
    - **Use Atomic Stripe Parallel Mode** to project from multiple DVS servers a ***non-cluster*** file system that requires both POSIX read/write atomicity and I/O parallelism.

- **Use Cluster Parallel Mode** to project from multiple DVS servers a ***cluster*** (shared) file system, such as IBM Spectrum Scale, that requires both POSIX read/write atomicity and I/O parallelism.
- **Stripe Parallel Mode** to project from multiple DVS servers a read/write file system that is not NFS and does not require POSIX read/write atomicity.
- **Use Loadbalance Mode** to project a read-only file system with I/O parallelism and failover across multiple DVS servers.

3. Perform Select DVS image from VCS.
4. Configure the new external or internal file system on the DVS servers.

   Create an systemd mount unit file in the `/etc/systemd/system` directory in the working copy of the gateway image in the VCS repository which describes how to mount the external file system.

   1. Create the `/opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>/etc/systemd/system/<MOUNT_NAME>.mount` file in an editor.
   2. Configure the new file system according the documentation for the external file system.

5. Update the DVS exports.

   Refer to Allowing Client Access to DVS Server File Systems

6. Configure the file system mount on the DVS clients.

   1. Check the `/opt/clmgr/image/images/<IMAGE_NAME>/etc/systemd/system` directory in the working copy of the COS compute image for any files with the .mount extension that contain `Type=dvs` similar to:

      ```
      system_name-adm# cd /opt/clmgr/image/images/<IMAGE_NAME>/etc/systemd/system
      system_name-adm# cat mnt-dvs_testing.mount
      [Unit]
      Description=dvs internal mount
      After=cray-dvs.service
      [Mount]
      What=/tmp/testing
      Where=/mnt/dvs_testing
      Type=dvs
      Options=noauto,path=/tmp/testing,nodename=dvs-01:dvs-02
      [Install]
      WantedBy=cray-dvs-lnet.target
      ```

   2. Edit the `mnt-dvs_testing.mount` file shown in the previous step to set the correct mount options. If the `/etc/systemd/system` directory does not have a mount unit file for your mount, add one with the correct options. Add the appropriate options as follows:

      - Serial Mode: `nodename=DVS_SERVER_1,maxnodes=1,path=PATH`
      - Cluster Parallel Mode: `nodename=DVS_SERVER_1,DVS_SERVER_2 . . .,maxnodes=1,path=PATH`
      - Stripe Parallel Mode: `nodename=DVS_SERVER_1,DVS_SERVER_2 . . .,maxnodes=X,path=PATH`
      - Atomic Stripe Parallel Mode: `nodename=DVS_SERVER_1,DVS_SERVER_2 . . .,maxnodes=X,atomic,path=PATH`
      - Loadbalance Mode: `nodename=DVS_SERVER_1,DVS_SERVER_2 . . .,maxnodes=1,loadbalance,path=PATH`

      Where:

      - `DVS_SERVER_1` and `DVS_SERVER_2` are the host names of two DVS servers
      - `X` is a value that is at least 2, but not greater than the number of nodes specified in the `nodename` list. All the modes, except for Serial Mode, support two or more DVS servers.
      - `. . .` are the host names of any additional DVS servers.
      - `PATH` is the mount point on the clients for the projected file system (the `path=` parameter in the options parameter of the systemd mount unit file `Options=` line entry). This is the same path that must be entered in the `Where=` line entry of the systemd mount unit file.

   3. Add any other client mount options, following the guidance in DVS Mount Options.
   4. Enter the rest of the required client mount information in the systemd mount unit file.

      The "Enable DVS Failover and Failback" subsection of the ***Manage and Customize DVS*** section contains two client configuration examples: one of a read-only Serial Mode file system and one of read-only file system projected using Loadbalance mode.

- An example `systemd mount unit` file for Cluster Parallel Mode:

  ```
  system_name-adm# cat cluster_parallel_mode.mount
  [Unit]
  Description=dvs Cluster Parallel Mode mount
  After=cray-dvs.service
  [Mount]
  What=/nfs_fs
  Where=/cluster_parallel_mode
  Type=dvs
  Options=rw,noauto,path=/cluster_parallel_mode,
  nodename=DVS_SERVER_1:DVS_SERVER_2,maxnodes=1
  [Install]
  WantedBy=cray-dvs-lnet.target
  ```

- A Stripe Parallel Mode example:

  ```
  system_name-adm# cat stripe_parallel_mode.mount
  [Unit]
  Description=dvs Stripe Parallel Mode mount
  After=cray-dvs.service
  [Mount]
  What=/gpfs_fs
  Where=/stripe_parallel_mode
  Type=dvs
  Options=rw,noauto,path=/stripe_parallel_mode,
  nodename=DVS_SERVER_1:DVS_SERVER_2,maxnodes=2
  [Install]
  WantedBy=cray-dvs-lnet.target
  ```

- An example Atomic Stripe Parallel Mode:

  ```
  system_name-adm# cat atomic_stripe_parallel_mode.mount
  [Unit]
  Description=dvs Atomic Stripe Parallel Mode mount
  After=cray-dvs.service
  [Mount]
  What=/gpfs_fs
  Where=/atomic_stripe_parallel_mode
  Type=dvs
  Options=rw,noauto,path=/atomic_stripe_parallel_mode,
  nodename=DVS_SERVER_1:DVS_SERVER_2,maxnodes=2,atomic
  [Install]
  WantedBy=cray-dvs-lnet.target
  ```

7. Perform Commit DVS image changes to VCS

2.7 Manage and Customize DVS

A collection of procedures for managing DVS and enabling additional functionality. This topic does not include performance optimization.

2.7.1 Procedures to Manage and Optimize DVS Use

DVS is able to perform its core duty within an HPE Cray EX system without user configuration after installation. DVS can be modified, however, to serve site-specific needs. System administrators may also need to occasionally perform maintenance tasks. Use the following procedures, as needed, during operation or for further configuration:

Site-specific need:	Procedure:
Nodes on the HPE Cray EX system need access to an external file system (such as NFS or IBM Spectrum Scale)	Project an External Filesystem to Compute Nodes or Login Nodes

Site-specific need:	Procedure:
Administrators must safely perform operations on a DVS-projected directory (or file system) without any possibility of interference by a DVS client. Examples include repairing a projected file system or taking a DVS server node out of service safely.	Quiesce a DVS-Projected File System
Allow DVS clients to access recent changes on a DVS-projected file system without waiting for the attribute cache timeout period to expire.	Force a Cache Revalidation on a DVS Mount Point

2.7.2 Allowing Client Access to DVS Server File Systems

Starting in COS 2.1, DVS switched from projecting all file system content on a node to requiring directories to be explicitly exported for projection. This prevents DVS nodes from accidentally projecting sensitive data to other nodes in the system. That could lead to compromised client nodes reading or altering sensitive data they shouldn't have access to.

Each node in the system is assigned a list of directory paths to export to other nodes in the system. Each path is exported in a Read-Write or Read-Only mode. A Read-Write directory path allows any client to read or write anything in that directory (or the directory's sub-directories), while a Read-Only directory only allows clients to read. Any directory which is not explicitly exported (or which is not a subdirectory of an explicitly exported directory) is unaccessible by any client. Read-Write and Read-only access applies to not only files, but also directories. Trying to create or remove a file or directory in a Read-Only-exported directory results in an EROFS error.

A node can have an empty exported directory list. In this case, the node doesn't act as a DVS server at all. This is usually the case for compute nodes.

It is allowed to export directories which are subdirectories of other directories. This is most often used to change the manner in which the subdirectory is exported. For example, directory `/foo` could be exported Read-Write while directory `/foo/bar` is exported Read-Only. A client would be able to change, read, and write files and directories in /foo and its subdirectories except for /foo/bar. /foo/bar and its subdirectories would be able to be read, but not changed.

Directories may also be exported with a "None" mode which specifically disallows access to a directory. This is useful in the case where the admin wants to prevent access to a smaller part of a bigger exported directory. Using the above example, /foo/bar could be exported "None". In that case, /foo/bar would be unaccessible, while the rest of /foo was read and writeable. Any attempt to access /foo/bar would result in an EPERM error.

The exports for a node are controlled by changing the `/etc/dvs_exports.yml` file for that node image. The image must contain a `/etc/dvs_server_list.conf` file with a list of DVS servers, one per line. If the `/etc/dvs_server_list.conf` is empty or not present in the image, the cray-dvs service file will configure the node on boot as a DVS client and replace the `/etc/dvs_exports.yml` file with the single line `exports: []`.

As an example, the default `/etc/dvs_exports.yml` for DVS servers is seen below. The file contains an `exports` dictionary entry which contains a list of exports. Each element in the list is a dictionary which two entries: `mode` and `path`. The path is the path of the directory to export and mode is one of `ro`, `rw`, or `none`.

~~~
```yaml
  exports:
  - mode: rw
    path: /
```
~~~

The definitions for the other node types all default to empty lists, such as the one below. For Compute and login nodes, there isn't much need for changing the default, as these node groups should probably not ever project DVS.

~~~
```yaml
  exports: []
```
~~~

Since the Gateway node type is explicitly for projection of customer file systems, the `/etc/dvs_exports.yml` file in that image should be customized for the correct paths for the site. For example:

~~~
```bash
system_name-adm # cat /etc/dvs_exports.yml
  exports:
  - mode: rw
    path: /gpfs1
~~~

```
  - mode: rw
    path: /nfs2
```
```` ``` ````

The system administrator populates the `/etc/dvs_exports.yml` file in each node image. The contents of `/etc/dvs_exports.yml` file is loaded into the DVS kernel driver by a systemd unit file which calls the `dvs_exportfs` command line tool.

See the ***Project an External Filesystem to Compute Nodes or User Access Nodes*** section for more information on how to configure exports on a Gateway Node.

2.7.3 Select DVS image from VCS

While it is possible to manage images using the HPCM image cloning method, the preferred method is to use the version control system (VCS) that HPCM provides. See the "Using the version control system (VCS)" Section in the "HPE Performance Cluster Manager Administration Guide". Using VCS to manage an image is similar to the cloning method. The difference is that when you use VCS, you do not need to employ a second name for the changed image. VCS does the implicit cloning and lets you change the clone. The changed image retains the same name as the original, but VCS assigns different version numbers to the images.

Perform this procedure to accomplish all the preparation steps for modifying the DVS configuration for CNs, Gateways, and DVS servers. After completing this procedure, the administrator will be able to change the DVS configuration persistently by modifying the local working copy of the COS image files.

Perform this procedure as directed by the other procedures in this guide. All commands must be run from a HPCM system admin node. In this method, you update the working copy of the image in the `/opt/clmgr/image/images` image environment on the admin node. This procedure assumes that you have the DVS related images under VCS control. This procedure also assumes that the COS images that contain DVS configurations were created in an earlier time at system install or upgrade.

1. Log into the admin node as the `root` user.
2. Obtain a list of the DVS server COS image in the VCS repository on the HPCM system admin node and determine which image listed in the output is used to configure the system.

   ```
   system_name-adm # cm image show
   my_recipe_cos-2.3
   my_recipe_cos-2.3_dvs
   my_recipe_cos-2.3_dvs-server
   my_recipe_cos-2.3_dvs-gateway
   my_recipe_cos-2.3_dvs-compute
   fmn_sles15sp3v2
   fmn_sles15sp3v2_A
   login_image
   rhel85-test-1
   sles15sp2
   su-sles15sp3-new
   ```

3. Confirm that the last image revision and the working copy are the same. Resolve any differences.

   ```
   system_name-adm # cm image revision diff-contents -i <IMAGE_NAME>
   ```

4. Compare the image on a running node with the image on the admin node. This check is performed if you are unsure about untracked changes to the image on the running node.

   Check that working copy of image is the same as image booted. Resolve any differences.

   ```
   system_name-adm #  cm image revision diff -i my_recipe_cos-2.3_dvs -n system_name0035
   Image my_recipe_cos-2.3_dvs using repo group: my_recipe_cos-2.3. Use cadmin to unset.
   Repo group my_recipe_cos-2.3 specified, using repos: amdgpu-21.50.2-sle15-main-x86_64
   aocc-compiler-3.2.0 Cluster-Manager-1.7-sles15sp3-x86_64 COS-2.3-addons-20220323
   COS-2.3.83-sles15sp3-x86_64 cpe-22.04-sles15-sp3 dkms-plus
   HPE-SLE-Updates-Module-Basesystem-15-SP3-x86_64-22.02.1
   HPE-SLE-Updates-Module-Development-Tools-15-SP3-x86_64-22.02.1
   HPE-SLE-Updates-Module-Python2-15-SP3-x86_64-22.02.1
   HPE-SLE-Updates-Module-Server-Applications-15-SP3-x86_64-22.03.0
   ROCm-5.1.1-sle15-20211214 ROCm-5.0.2-sle15 SLE-15-SP3-Full-x86_64
   ```
````

```
slingshot-host-software-1.7.3-47 slurm-21.08.5_cray_sles15sp3

CM config check and RPM verification output of node:system_name0035
------------------------------------------------
CM config check on node:system_name
Running: ssh system_name0035 /usr/bin/sgi_config_changed
        rpmsave  /etc/modprobe.d/dvs.conf
rpmnew           /etc/ssh/ssh_config
1       1        Total:1482 Pattern:* Verdict:NOT Clean

RPM verification on node:system_name0035
Running: ssh system_name0035 rpm -Vva --nofiles --nodigest

List RPM differences between node:system_name0035 and image:my_recipe_cos-2.3_dvs
------------------------------------------------
*** RPMs different or not in node:system_name0035 ***

*** RPMs different or not in image:my_recipe_cos-2.3_dvs ***

RPM difference check completed

File differences: Using selection criteria in /etc/opt/sgi/diff-selection-criteria.conf
------------------------------------------------
Running command: rsync -avHirnc --delete --checksum postoak0035:/etc/
/opt/clmgr/image/images/my_recipe_cos-2.3_dvs/etc/ | sed -r 's/deleting/extraneous file/g'
receiving incremental file list

...
Log file created in /var/log/cinstallman-diff.log
```

Examine log file.

5. Repeat Steps 2-4 to confirm the gateway image.

   Examine the list of images in VCS on the system admin node. Find the image that represents the version of the gateway image you want to use.

6. Repeat Steps 2-4 to confirm the compute image.

   This time, examine the list of images in VCS on the system admin node for the image that represents the version of the compute node image you want to use.

Configure DVS as wanted by resuming the procedure that referred to this one.

### 2.7.4 Commit DVS image changes to VCS

After you have made the DVS configuration changes to the working copy of the image, display the changes between the latest image checked into VCS and the working copy of the image to review changes.

```
system_name-adm # cm image revision diff-contents -i <IMAGE_NAME>
```

Use the `cm image revision commit` command to commit your changes to VCS. The working copy of the image resides in the `/opt/clmgr/image/images/<IMAGE_NAME>` directory. The commit requires you to enter a log message. You can specify the log message with the -m parameter or you can enter the message from the terminal during the running of the command.

```
system_name-adm # cm image revision commit -i <IMAGE_NAME> -m "changing DVS parameters for client
mount options"
```

Confirm commit by reviewing revision history.

```
system_name-adm # cm image revision history -i <IMAGE_NAME>
```

### 2.7.5 Enable DVS Failover and Failback

Automatic DVS failover is not in the HPCM COS 2.3 release. A manual failover procedure has been provide for the COS 2.3 release. Automatic DVS failover is being provided in the COS 2.4 release.

Perform this procedure to enable failover and failback resiliency when projecting a file system with DVS. System administrators must update the configuration management repository, create new client images, then boot the clients with those new images.

Before performing this procedure, perform Select DVS image from VCS

This procedure assumes:

- One node running standalone DVS in Serial Mode is already projecting the file system to clients that will have failover and failback enabled.
- The necessary configuration for the new client mount exists in a `/etc/fstab` or systemd mount unit file within the image working copy directory of the HPCM configuration management repositories. Refer to Project an External Filesystem to Compute Nodes or Login Nodes procedure in the HPCM version of the COS documentation for examples.

To enable failover and failback resiliency for a DVS-projected file system, HPE Cray EX administrators must:

1. Edit the files within the working copy of the image that configure DVS on the clients.
2. Reboot the CNs and other DVS clients with the new images.

To save time, enable failover and failback when initially configuring standalone DVS to project the content. In that scenario, only perform Step 3 of this procedure. All the other steps will be performed as part of configuring and starting standalone DVS projection.

1. Determine where in the `image working copy` directory the client mount configuration for the projected file system resides.

   The following example systemd mount unit file is for a read-only file system. This configuration information may reside in either the `/etc/systemd/system/mydvsmount.mount` file or in `/usr/lib/systemd/system/mydvsmount.mount` file. This is the preferred method for specifying mounts.

   ```
   system_name-adm: / # cat /opt/clmgr/image/images/<IMAGE_NAME>/etc/systemd/system/mydvsmount.mount
   [Unit]
   Description=My new file system
   After=cray-dvs.service

   [Mount]
   What=/example_projected_fs
   Where=/example_projected_fs_mount_point
   Type=dvs
   Options=attrcache_timeout=14400,ro,path=/example_projected_fs_mount_point,
   nodename=x3000c0s25b0n0

   [Install]
   WantedBy=cray-dvs-lnet.target
   ```

   The following example 'fstab' file is for a read-only file system.

   ```
   system_name-adm: / # cat /opt/clmgr/image/images/<IMAGE_NAME>/etc/fstab
   . . .
   /example_projected_fs /example_projected_fs_mount_point dvs attrcache_timeout=14400,ro,
   path=/example_projected_fs_mount_point,nodename=x3000c0s25b0n0 0 0
   ```

2. Open the client mount configuration file for editing.

3. Configure the client mount for failover and failback.

   1. Ensure that the xnames of at least two DVS servers or DVS gateways are specified for the nodename argument on the client mount.

   2. Specify one of the three parallel DVS modes.

   3. Add the failover option unless loadbalance mode is chosen.

The DVS server list for the nodename argument is colon-separated. The following excerpt is identical to the previous one except that the file system is projected in DVS Loadbalance Mode using two separate DVS server nodes. The unnecessary `ro` option is removed as `loadbalance` sets the mount as read-only.

Specify the mount with the `systemd` mount unit file method.

```
system_name-adm: / # cat /opt/clmgr/image/images/<IMAGE_NAME>/etc/systemd/system/mydvsmount.mount
[Unit]
Description=My new file system
After=cray-dvs.service

[Mount]
What=/example_projected_fs
Where=/example_projected_fs_mount_point
Type=dvs
Options=attrcache_timeout=14400,loadbalance,path=/example_projected_fs_mount_point,
nodename=x3000c0s25b0n0:x3000c0s23b0n0

[Install]
WantedBy=cray-dvs-lnet.target
```

Specify the DVS mount with the `fstab` file method.

```
system_name-adm: / # cat /opt/clmgr/image/images/<IMAGE_NAME>/etc/fstab
. . .
/example_projected_fs /example_projected_fs_mount_point dvs attrcache_timeout=14400,loadbalance,
path=/example_projected_fs_mount_point,nodename=x3000c0s25b0n0:x3000c0s23b0n0 0 0
```

Refer to DVS Modes and Choose and Configure a DVS Mode for guidance on choosing a parallel DVS mode.

4. Update the configuration used by the HPE Cray EX by performing Commit DVS image changes to VCS.

5. If your system contains Leader nodes, push the image 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 <IMAGE_NAME> --force
```

6. Reboot the DVS clients using the new client image.

Replace *NODE_LIST* in the following command with the comma separated list of CNs.

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

### 2.7.6 Project an External Filesystem to Compute Nodes or Login Nodes

Perform this procedure to project either a read-only or read/write external file system to CNs and login nodes with DVS.

Satisfy these prerequisites before performing this procedure:

- Verify that an external file system exists which can be physically attached to the HPE Cray EX system.
- To project an IBM Spectrum Scale file system over DVS, first install the Spectrum Scale client software on the gateway nodes according to the instructions at www.ibm.com.

External file systems can contain data such as user home directories and be NFS, IBM Spectrum Scale (formerly GPFS), Lustre, or others. HPE Cray EX Gateway Nodes connect to the external file system and then use DVS to project it to the CNs and login nodes.

1. Configure the network interfaces on the gateway node which will connect to the external file system.

   Refer to the site network documentation, this guide, and other HPE Cray EX documentation for details.

2. Perform Select DVS image from VCS for the gateway image.

   Note that these instructions require changes to the COS image.

**Configure the external file system mount on both DVS servers and clients**

3. Configure the mount of the file system to be projected.

   If the external file system in question requires mounting on the Gateway node, follow the instructions in this step. An example of this type of mount might be a Lustre file system or NFS file system that requires IP routes to be added after boot during node personalization.

   If the file system mount is accomplished by another means, you can skip this step. For example, if the file system is already mounted by a `systemd mount unit` file or an `/etc/fstab` entry previously installed in the image during image customization, this step can be skipped.

   Create a systemd mount unit file in the working copy of the gateway image on the HPCH admin node which describes how to mount the external file system.

   ```
   system_name-adm: / # cd /opt/clmgr/image/images/<IMAGE_NAME>/etc/systemd/system
   system_name-adm: / # vi home.mount
   system_name-adm: / # cat home.mount
   [Unit]
   Description=Mount for /home
   After=network.target
   [Mount]
   What=nfs.server.example.com:/home
   Where=/home
   Type=nfs
   Options=_netdev,auto

   [Install]
   WantedBy=multi-user.target
   ```

4. Update the `dvs_exports.yml` file to allow access to the file system(s)

   Edit the working copy of the image and add the file systems to be projected to the `dvs_exports.yml` file. You can set the mode to “ro” or “rw” depending on whether or not you want the file system to be exported Read-Only or Read-Write. See the ***Allowing Client Access to DVS Server File Systems*** section for more information.

   ```
   system_name-adm: / # cd /opt/clmgr/image/images/<IMAGE_NAME>/etc
   system_name-adm: / # vi dvs_exports.yml
   system_name-adm: / # cat dvs_exports.yml
   ...
   dvs_exports_yml:
     exports:
     - mode: rw
       path: /home
     - mode: rw
       path: /gpfs1
     - mode: ro
       path: /static-data
   ```

5. Perform Commit DVS image changes to VCS.

   This is for the working copy of the Gateway image in the HPCM VCS repository on the HPCM system admin node.

6. Create the systemd mount unit file in the working directory for the CN image to mount the file system on CNs. Skip this step if the new file system will not be projected to CNs.

   1. Create the systemd mount unit file so that it contains `external file systems` entries similar to the example in next sub-step.

   2. Add the client mount configuration for the external file system to the systemd mount unit file.

      In the `Options=` line, specify the host names of the Gateway nodes (separated by colons) that will project the external file system as part of the `nodename` argument. The following example `home.mount` file is for a CN mount of a remote `/home` directory, that is projected using the settings in Step 3.

      ```
      system_name-adm # cat /opt/clmgr/image/images/<CN_IMAGE_NAME>/etc/systemd/system/home.mount
      [Unit]
      ```

```
Description=dvs external mount
After=cray-dvs.service

[Mount]
What=/home
Where=/home
Type=dvs
Options=path=/home,nodename=px3000c0s25b0n0:x3000c0s23b0n0

[Install]
WantedBy=cray-dvs-lnet.target
```

This example mounts a remote read-only file system *`/READ_ONLY_FS`* on the client at *`/READ_ONLY_FS_MOUNT_POINT`*. Note the `attrcache_timeout=14400` and `loadbalance` mount options in the `Options=` field, since this example is a read-only file system:

```
system_name-adm # cat /opt/clmgr/image/images/<CN_IMAGE_NAME>/etc/systemd/system/READ_ONLY_FS_MOUNT_
[Unit]
Description=dvs external mount
After=cray-dvs.service

[Mount]
What=/READ_ONLY_FS
Where=/READ_ONLY_FS_MOUNT_POINT
Type=dvs
Options=attrcache_timeout=14400,loadbalance,
path=/READ_ONLY_FS_MOUNT_POINT,nodename=px3000c0s25b0n0:x3000c0s23b0n0

[Install]
WantedBy=cray-dvs-lnet.target
```

**Update the HPE Cray EX configuration with the changes**

7. Perform Commit DVS image changes to VCS.
8. Create the systemd mount unit file to mount the file system on login nodes. Skip this step if the new file system will not be projected to login nodes.
    1. Perform Select DVS image from VCS for the login image.
    2. Create the systemd mount unit file so that it contains `external file systems` entries similar to the example in Step 6.
    3. Add the client mount configuration for the external file system to the systemd mount unit file.

       In the `options` field, specify the host names of the Gateway nodes (separated by colons) that will project the external file system as part of the `nodename` argument. The following example `home.mount` file is for a mount of a remote `/home` directory.

```
system_name-adm # cat /opt/clmgr/image/images/<CN_IMAGE_NAME>/etc/systemd/system/home.mount
[Unit]
Description=dvs external mount
After=cray-dvs.service

[Mount]
What=/home
Where=/home
Type=dvs
Options=path=/home,nodename=px3000c0s25b0n0:x3000c0s23b0n0

[Install]
WantedBy=cray-dvs-lnet.target
```

This example mounts a remote read-only file system *`/READ_ONLY_FS_`* ***on the client at*** `/READ_ONLY_FS_MOUNT_POINT_`. Note the `attrcache_timeout=14400` and `loadbalance` mount options in the `Options=` field, since this example is a read-only file system:

```
system_name-adm # cat /opt/clmgr/image/images/<CN_IMAGE_NAME>/etc/systemd/system/READ_ONLY_FS_MOUNT_
[Unit]
Description=dvs external mount
After=cray-dvs.service

[Mount]
What=/READ_ONLY_FS
Where=/READ_ONLY_FS_MOUNT_POINT
Type=dvs
Options=attrcache_timeout=14400,loadbalance,
path=/READ_ONLY_FS_MOUNT_POINT,nodename=px3000c0s25b0n0:x3000c0s23b0n0

[Install]
WantedBy=cray-dvs-lnet.target
```

4. Perform Commit DVS image changes to VCS.

**Force the system to immediately use the new configuration**

9. If your system contains Leader nodes, push the image changes for Gateway, CNs, and login nodes to the leader nodes so they will be available for boot. You will perform the `cm activate image` for each image modified in the previous steps. 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 <IMAGE_NAME> --force
   ```

10. Reboot the Gateways to cause them to mount the external file system.

    If you changed the image name from the previous boot, specify the image to boot.

    ```
    system_name-adm: / # cm node set --image <NEW_GATEWAY_IMAGE_NAME> -n <NODE_LIST>
    ```

    Reset the node to start the boot.

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

    Note: contains the list of Gateway nodes.

11. Reboot the compute nodes, login nodes, or both, depending on what types of nodes will be mounting the external file system.

    If you changed the image name from the previous boot, specify the image to boot.

    ```
    system_name-adm: / # cm node set --image <NEW_IMAGE_NAME> -n <NODE_LIST>
    ```

    Reset the node to start the boot.

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

    Note: contains the list of compute nodes, login nodes, or both.

    Once the nodes finish rebooting, the DVS-projected external file system mount will persist across future CN, login, and Gateway reboots.

12. If mounting the projected external file system to the DVS servers is needed, repeat the “Add the client mount configuration for the external file system” steps earlier in this procedure and apply them to the DVS server image.

### 2.7.7 Quiesce a DVS-Projected File System

DVS quiesce is used to temporarily suspend traffic to a DVS-projected file system on any number of DVS servers. When a directory is quiesced, the DVS server completes any outstanding requests, but does not honor any new requests for that directory. When the quiesce is finished, administrators know it is safe to perform operations on the quiesced directory without any possibility of interference by a DVS client. DVS quiesce can be used when a file system needs to be repaired or to safely take a DVS server node out of service.

**CAUTION:** Because it may cause data corruption, do not use DVS quiesce to quiesce a directory on every DVS server.

An administrator must write into `/sys/fs/dvs/quiesce` to use DVS quiesce. Refer to the following use cases for more information:

- Quiesce a Single Directory on a Single DVS Server
- Quiesce All Directories on a DVS Server

#### 2.7.7.1 How Quiesce Works: the User space Application View

User space applications have no visibility into any specific quiesce information. A quiesced directory will present in one of two ways:

- Entirely normally, if the directory is quiesced only on servers that the application is not using
- Useable but with degraded performance, if the application finds that its main server is quiesced and must query other servers

#### 2.7.7.2 How Quiesce Works: the Server View

To provide the quiesce capability, DVS servers keep a list of quiesced directories and the current outstanding requests on each quiesced directory. When an admin requests that DVS quiesce a directory on a server, DVS does the following:

- Adds that directory to the list of quiesced directories on the server
- Iterates over all open files, closing any file that resides in the quiesced directory and setting a flag to indicate that the file was closed due to the directory being quiesced

When a DVS server receives a request from a client, DVS checks the request path against the list of quiesced directories. The comparison between the pathname in the request and the quiesced directory is a simple string compare to avoid any access of the underlying file system that has been quiesced. If DVS finds that the request is for a quiesced file system, it sends a reply indicating that the request could not be completed due to a quiesce and noting which directory is quiesced. If the client request is for a file that has been closed due to quiesce, the server returns a reply to the client indicating that the request could not be completed due to a quiesce.

When an admin unquiesces a directory on a DVS server, DVS simply removes that directory from the list of quiesced directories on the server and clears all quiesce-related flags for that directory.

#### 2.7.7.3 How Quiesce Works: the Client View

When making a request of a server, a client may get back reply indicating that the request was for a file in a quiesced directory. The client then retries the operation on the next server in its server list. If it makes the request of every server in its server list and gets the same reply from each of them, then one of two things happens, depending on the type of request:

- **Pathname request**

  If the request is a pathname request (`lookup`, `stat`, file open, and so forth), then DVS reattempts the operation on a different server in a round-robin fashion until it finds a server that allows the operation to complete successfully.

- **Open file**

  If the request is for an open file (`read`, `write`, `lseek`, and so on), then DVS attempts the operation on a different server. If the file is not open on any other servers, DVS attempts to open on the file on a server in a round robin fashion until it gets a successful open. DVS will then attempt to perform the operation.

If a client receives a reply indicating a quiesced directory, the client adds that directory to a list of quiesced directories held on the DVS superblock. This is intended to reduce network traffic by avoiding requests that target quiesced directories. The list of quiesced directories on the client expires about every 60 seconds, thereby allowing clients to try those directories again in case one or more have been unquiesced during that time. This mechanism enables DVS to strike a balance between the timely unquiescing of a file system and a large reduction in network traffic and requests coming into the server. It also naturally staggers clients when they start to use a server.

### 2.7.8 Quiesce All Directories on a DVS Server

Remove a DVS server from service and let any outstanding requests complete first.

#### 2.7.8.1 Prerequisites

- The administrator is logged into a DVS server
- Verify existing requests with List Outstanding DVS Client Requests

#### 2.7.8.2 Procedure

1. Quiesce all directories on the DVS server.

   ```
   dvs1# echo quiesce / > /sys/fs/dvs/quiesce
   ```

   When the quiesce has completed, the shell prompt will return to user control. There is no command output.

2. Unquiesce all of the projected directories to allow traffic to the server to resume.

   ```
   dvs1# echo unquiesce / > /sys/fs/dvs/quiesce
   ```

### 2.7.9 Quiesce a Single Directory on a Single DVS Server

Quiesce a single directory on a DVS server. This is an unlikely use case, but it is an illustrative example. The example directory used in this procedure is `/gpfs/test/foo`.

#### 2.7.9.1 Prerequisites

- The administrator is logged into a DVS server

#### 2.7.9.2 Procedure

1. Quiesce the directory.

   ```
   dvs1# echo quiesce /gpfs/test/foo > /sys/fs/dvs/quiesce
   ```

   When the directory quiesce has completed, the shell prompt will return to user control. There is no command output.

2. When finished with the directory, unquiesce it.

   ```
   dvs1# echo unquiesce /gpfs/test/foo > /sys/fs/dvs/quiesce
   ```

3. Ensure that the directory is no longer on the quiesced list.

   ```
   dvs1# cat /sys/fs/dvs/quiesce
   ```

### 2.7.10 List Outstanding DVS Client Requests

Perform this procedure to list detailed information about any pending DVS client requests from client nodes in `/sys/kernel/debug/dvsipc/requests`.

This file lists the following:

- The DVS server node
- The type of request
- The DVS file system path
- The `uid` of the user initiating the request
- The amount of time that the request has been waiting for a response

1. Print out the list of pending DVS requests.

   ```
   ncn-w002# cat /sys/kernel/debug/dvsipc/requests
   server: x3000c0s25b0n0 request: RQ_LOOKUP path: /cray_home user: 12795 time: 0.000 sec
   ```

### 2.7.11 Force a Cache Revalidation on a DVS Mount Point

Administrators can force a DVS cache revalidation at any time without having to wait for the attribute cache timeout to expire. If any content in a DVS-projected read-only file system changes (such as HPE Cray Programming Environment content), this procedure will enable DVS clients to see and use the new content immediately.

This procedure enables a system administrator with root privilege to force a cache revalidation at any time without having to wait for the timeout to expire—a process which can take up to four hours (14,400 seconds). This procedure is useful for shortening the time between when changes are made on a DVS-mounted file system (such as HPE Cray Programming Environment content) and when the new content can be used by the DVS client nodes.

1. Force a cache revalidation on a DVS client using one of the following methods:

- To revalidate all cached attributes on a single DVS mount point, `echo 1` into the `drop_caches/sys` file of that mount point. The following example uses the second mount point on the client and uses the `cat` command first to confirm that is the wanted mount point. To specify a different mount point, replace the 2 with some other integer (0-n).

  ```
  cn# cat /sys/kernel/debug/dvs/mounts/2/mount
  cn# echo 1 > /sys/kernel/debug/dvs/mounts/2/drop_caches
  ```

- To revalidate all attributes across all DVS mounts, `echo 1` into the universal DVS `drop_caches/sys` file. For example:

  ```
  cn# echo 1 > /sys/fs/dvs/drop_caches
  ```

2. **(Optional)** Clear out other caches on the DVS client to ensure that all data is revalidated.

   ```
   cn# echo 3 > /sys/fs/dvs/drop_caches
   ```

3. **(Optional)** Clear out other caches on the DVS ***server*** to ensure that all data is revalidated.

   If underlying file system is NFS, it is also likely that the same procedure will be required on the DVS servers to allow all the changes on the NFS file system to properly propagate out to the DVS clients. This is due to NFS caching behavior.

   ```
   ncn# echo 3 > /sys/fs/dvs/drop_caches
   ```

### 2.7.12 Change the Name of the DVS LNet Network

Avoid collisions between the LNet network used by DVS and the one used by an attached Lustre file system by changing the name of the DVS network.

- **PREREQUISITES**
  - Select DVS image from VCS
- **OBJECTIVE**

  If the HPE Cray EX system is connected to a Lustre file system, use this procedure to change the name of the LNet network that DVS uses. Changing the name of the DVS LNet network will prevent conflicts with the LNet network used by Lustre.
- **LIMITATIONS**

  This requires downtime because the reboot step will cause those nodes to stop working until they are rebooted with the new image.

Perform Change LNet or DVS Parameters. While executing that procedure, update the `etc/lnet.conf`, `etc/dvs_node_map.conf`, and `etc/modprobe.d/dvs.conf` files in the `/opt/clmgr/image/images/<IMAGE_NAME>/` working copy of the image as part of Step 2. The DVS LNet Network name **must** be the same across these three files.

This must be done in the images of the HPCM VCS repository for each of the DVS servers, Gateways, CNs, and login nodes used on the system.

1. Edit the lnet.conf and dvs_node_map.conf files in `/opt/clmgr/image/images/<IMAGE_NAME>/etc/` directory so that `tcp99` is replaced by `tcpXX` where `XX` is an integer other than 99.
2. Edit the dvs.conf file in `/opt/clmgr/image/images/<IMAGE_NAME>/etc/modprobe.d/` directory so that `tcp99` is replaced by `tcpXX` where `XX` is an integer other than 99. The LNet network name must be the same one used in the lnet.conf and dvs_node_map.conf files.
3. Generate a new dvs_node_map file using the updated dvs_node_map.conf file with the new DVS LNet Network name and add it to the images that support DVS on this system. There may be separate images for each of DVS servers, Gateways, CNs, and login nodes. See How to Populate node maps

### 2.7.13 Change LNet or DVS Parameters

To change the kernel module parameters used by DVS and LNet, determine the appropriate images in VCS and modify the appropriate files. Then commit the updated images, reboot DVS servers, and finally reboot the Gateway, CNs, and login nodes.

- **OBJECTIVE**

  Perform this procedure to modify the kernel module parameters used by DVS or LNet on an HPE Cray EX system.

- **ADMIN IMPACT**

  This procedure must be performed to persistently modify any LNet and DVS kernel module parameters. This includes changing DVS modes, configuring modes for projecting new content, and DVS performance tuning.

1. Perform the procedure Select DVS image from VCS to update the LNet and/or DVS kernel parameters in HPE Cray EX system configuration.
2. Modify the files that control the LNet or DVS configuration in the image working copy. Knowing which files to changes depends on what you're trying to accomplish. Often the section that refers you to this one will tell you what needs changing.
3. Perform the procedure Commit DVS image changes to VCS to update the LNet and/or DVS kernel parameters in HPE Cray EX system configuration.
4. Reboot the DVS server, Gateway, CNs, and login nodes (in that order).

   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 <IMAGE_NAME> --force
   ```

   Specify the image to boot.

   ```
   system_name-adm: / # cm node set --image <IMAGE_NAME> -n <NODE_LIST>
   ```

   Reset the node to start the boot.

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

### 2.7.14 Configure DVS Using `dvs.conf`

Use this procedure to apply different DVS configurations to different groups of nodes within an HPE Cray EX system.

**Prerequisites** - Refer to the HPCM documentation for information on VCS. - Perform Select DVS image from VCS

The contents of each node's `/etc/modprobe.d/dvs.conf` file can vary by node type (DVS server, Gateway, and CNs). You can set the DVS parameter for each type of node by setting it in `dvs.conf` file in the appropriate image working copy directory in VCS. For example, you can change the `dvs.conf` file for the CNs by editing the working copy for the CN image in the `/etc/modprobe.d/dvs.conf` file, like so:

~~~
```bash
system_name-adm# cat /opt/clmgr/image/images/<IMAGE_NAME>/etc/modprobe.d/dvs.conf
  options dvsipc_lnet lnd_name=tcp
  options dvsipc dvs_debug_mask=0xffffffff
```
~~~

This procedure demonstrates how to use HPCM VCS to deploy different versions of the `dvs.conf` file to DVS servers, CNs, and gateways. These types of nodes can require different content in this file.

Perform Change LNet or DVS Parameters. While executing that procedure, make the changes below to the VCS repositories as part of Step 2.

This must be done in the working copies of the DVS server, Gateway, CN, and login images in HPCM VCS repository.

1. Navigate into the `/opt/clmgr/image/images/<IMAGE_NAME>/etc/modprobe.d/` directory.

   ```
   system_name-adm # cd /opt/clmgr/image/images/<IMAGE_NAME>/etc/modprobe.d
   ```

   Alternatively you may `chroot` to the working copy of the image directory then change directories to the `/etc/modprobe.d` directory.

   ```
   system_name-adm # chroot /opt/clmgr/image/images/<IMAGE_NAME>
   system_name-adm # cd /etc/modprobe.d
   ```

   After making your changes, exit the `chroot` environment.

   ```
   system_name-adm # exit
   exit
   ```

2. Edit the `dvs.conf` file to include the wanted DVS kernel module options. Each line must follow the syntax demonstrated in the following example as a guide. Replace `FOO` with a valid DVS kernel module parameter and `bar` with a valid value for that parameter.

   ```
   system_name-adm# vi dvs.conf
   ...
   system_name-adm# cat dvs.conf
     options dvs FOO=bar
   ```

3. Repeat the previous two steps for all the node types in the system.

## 2.8 DVS Log Files

This section describes how to enable, disable, and reset DVS request logging and file system (FS) call logging. It also includes instructions on how to tune logging for optimized results.

### 2.8.1 DVS Request Logs

To aid in debugging problems that may arise, DVS request logging is enabled by default. To reduce the overhead to each client request, DVS logs only those requests that take more than a certain number of seconds to complete. For each request that exceeds the threshold, DVS writes a single line to `/sys/kernel/debug/dvs/request_log`. That minimum threshold can be changed (default is 15 seconds) to see more or fewer requests in the log.

There are two ways to make changes to DVS request logging. When the system is running, an administrator can `ssh` to a node and make changes dynamically by writing values to certain `/sys` files. Alternatively, to set the behavior from the moment the DVS kernel modules load, administrators can set kernel module parameters prior to boot.

### 2.8.2 DVS File System (FS) Call Logs

DVS can log details about the calls that the DVS server node makes into the underlying file system. The log files are created in `/sys/fs/dvs/fs_log` and can provide the administrator with useful information, such as:

- The pid and name of the thread making the file system call
- The operation being executed (open, close, read, write, getattr, ...)
- The uid of the thread making the file system call, which DVS sets to the uid of the process on the DVS client
- The xname of the client node that sent the request to the server
- The path corresponding to the operation, if any

The file system operations logged are a larger set than the list of DVS requests. This is because it is sometimes necessary to make multiple and different file system calls to complete a single DVS request. For these secondary file system operations, the operation logged contains op[func] where op is the file system operation performed and func is the DVS function that contained the call. This lets the reader distinguish the primary from the secondary file system calls.

### 2.8.3 Disable or Re-enable DVS Logging

Disable or enable DVS request logging or FS call logging using kernel module parameters. When enabled, logging is useful for debugging problems with DVS.

Request and FS call logging are enabled by default.

Replace the `PARAMETER_NAME` value in the following commands with `dvs_request_log_enabled` for request logs, or `dvs_fs_log_enabled` for FS call logs.

1. Check if the logging type is enabled or disabled.

   To view the kernel module parameter in read-only mode:

   ```
   hostname# cat /sys/module/dvsproc/parameters/PARAMETER_NAME
   ```

2. Use the appropriate kernel module parameter to enable or disable DVS request or FS call logging.

   Refer to the Configure DVS Using dvs.conf section of the HPE Cray Operating System Administration Guide: CSM on HPE Cray EX Systems for instructions on how to add this change on all nodes.

   Use one of the following options to enable/disable DVS logging dynamically or prior to boot.

- Edit the `/etc/modprobe.d/dvs-local.conf` file and add one of the following lines prior to boot:

  To disable DVS request logging:

  ```
  # Disable DVS request log
  options dvsproc PARAMETER_NAME=0
  ```

  To enable DVS request logging:

  ```
  # Enable DVS request log
  options dvsproc PARAMETER_NAME=1
  ```

- Dynamically:

  Replace `LOG_TYPE` with `request_log` or `fs_log` depending on the type of log in use.

  0 disables the log:

  ```
  hostname# echo 0 > /sys/kernel/debug/dvs/LOG_TYPE
  ```

  1 enables the log:

  ```
  hostname# echo 1 > /sys/kernel/debug/dvs/LOG_TYPE
  ```

### 2.8.4 Reset the DVS Log

Reset the DVS log to remove logs that are no longer needed and to make it easier to search when debugging current issues.

Dynamically reset the DVS request log or FS call log with the following command:

Replace `LOG_TYPE` with `request_log` or `fs_log` depending on the type of log in use.

```
hostname# echo 2 > /sys/kernel/debug/dvs/LOG_TYPE
```

### 2.8.5 Read the DVS Log

Use the following command to view the DVS request log or FS call log:

Replace `LOG_TYPE` with `request_log` or `fs_log` depending on the type of log in use.

```
hostname# cat /sys/kernel/debug/dvs/LOG_TYPE
```

The following is example output from the DVS request log:

```
2022-1-31 16:22:01-UTC: pid=2302223 cmd=sadc path=/var/lib/cps-local/boot-images/
a4d33fad-0107-4c77-8f26-b0715b25ce7a type=dir req=RQ_STATFS count=1 node=x3000c0s9b0n0 time=60.064
2022-1-31 16:23:01-UTC: pid=2302223 cmd=sadc path=/var/
lib/cps-local/boot-images/PE type=dir req=RQ_STATFS count=1 node=x3000c0s9b0n0 time=60.016
2022-1-31 16:40:01-UTC: pid=2564090 cmd=sadc path=/var/lib/cps-local/boot-images/
a4d33fad-0107-4c77-8f26-b0715b25ce7a type=dir req=RQ_STATFS count=1 node=x3000c0s9b0n0 time=60.036
2022-1-31 16:41:01-UTC: pid=2564090 cmd=sadc path=/var/lib/cps-local/boot-images/PE type=dir req=RQ_STATFS
count=1 node=x3000c0s9b0n0 time=60.016
2022-2-5 11:52:14-UTC: pid=217823 cmd=sadc path=/var/lib/cps-local/boot-images/
a4d33fad-0107-4c77-8f26-b0715b25ce7a type=dir req=RQ_STATFS count=1 node=x3000c0s9b0n0 time=252.824

[...]
```

Depending on the log type, a combination of the following information will be included in each line of the log:

- date/time

  Date and time (in UTC) of request.

- pid

  User process ID of the process initiating the request.

- cmd

  Command being executed by the user process ID.

- op

  The operation being executed (open, close, read, write, getattr, ...).

- path

  Path on the server node that is being referenced.

- type

  Type of path (file, directory, link, etc.).

- req

  Type of DVS request.

- count

  Number of servers the request was sent to. Some DVS client operations send a request to a single server, and others send a request to multiple servers. The former will log a line with count=1, and the latter will log a line with count=N, where N is the number of servers the request was sent to. When the request has been sent to multiple servers, the single line in the log is a summary of the requests.

- node

  Node is set to the node that the DVS request was sent to. If multiple nodes were targeted (i.e., the count field is > 1), then the value displayed is [multiple].

- time

  Time is set to the amount of time in seconds it took for the request to be sent to the server, execute, and for the reply to be received by the client. Granularity is milliseconds, so 0.000 is displayed for requests that take less than a millisecond.

### 2.8.6 Tune DVS Logging

Alter how DVS request logs and FS call logs are captured with kernel module parameters.

#### 2.8.6.1 Buffer Size

The default buffer size for both log types is 16384 KB (16384 * 1024 bytes).

Replace the `PARAMETER_NAME` value in the following commands with `request_log_size_kb` for request logs, or `fs_log_size_kb` for FS call logs.

1. Determine the size (KB) of the log buffer.

   To determine the current buffer size, `cat` the file. For example:

   ```
   hostname# cat /sys/kernel/debug/dvs/PARAMETER_NAME
   16384
   ```

   Alternatively, the following command can be used to view in read-only mode:

   ```
   hostname# cat /sys/module/dvsproc/parameters/dvs_PARAMETER_NAME
   ```

2. Update the size of the request log with the appropriate kernel module parameter.

   Use one of the following options to update the size of the request log dynamically or prior to boot.

   - To change prior to boot, add these lines to `/etc/modprobe.d/dvs-local.conf`:

     ```
     # Set size (in kb) of the request log buffer
     options dvsproc dvs_PARAMETER_NAME=17000
     ```

   - To change dynamically:

     ```
     hostname# echo 17000 > /sys/kernel/debug/dvs/PARAMETER_NAME
     ```

#### 2.8.6.2 Log Time

DVS only logs requests that take more than a certain number of seconds to complete. For each request that exceeds the threshold, DVS writes a single line to `/sys/kernel/debug/dvs/request_log` or `/sys/kernel/debug/dvs/fs_log`. Use this procedure to set the minimum threshold to see more or fewer requests in the log.

The default threshold for both log types is 15 seconds.

Replace the `PARAMETER_NAME` value in the following commands with `request_log_time_min_secs` for request logs, or `fs_log_time_min_secs` for FS call logs.

1. View the current threshold of time for data to be logged.

   ```
   cat /sys/module/dvsproc/parameters/dvs_PARAMETER_NAME
   ```

2. Update the time threshold with the appropriate kernel module parameter.

   Use one of the following options to update the time threshold dynamically or prior to boot.

   - To change prior to boot, add these lines to `/etc/modprobe.d/dvs-local.conf`:

     ```
     # Set threshold (in seconds) for time a DVS request
     # takes before logged. Requests taking fewer seconds
     # will not be logged.
     options dvsproc dvs_PARAMETER_NAME
     ```

   - To change dynamically:

     ```
     hostname# echo 15 > /sys/kernel/debug/dvs/PARAMETER_NAME
     ```

## 2.9 Troubleshoot DVS

This topic contains several commands for gathering useful information for troubleshooting DVS.

In addition to the specific methods listed here, use DVS statistics for troubleshooting and root cause analysis. See Collect and Analyze DVS Statistics for more details.

### 2.9.1 Verify All Necessary DVS Kernel Modules for the Running Kernel Are Installed

Use the following command to confirm that all four DVS kernel modules are installed in the `/lib/modules/` directory for the running kernel.

The five DVS kernel modules are: - `dvsipc.ko` - `dvs.ko` - `dvskatlas.ko` - `dvsproc.ko` - `dvsipc_lnet.ko`

```
ncn-w001# find /lib/modules/`uname -r` -name dvs* -ls
```

### 2.9.2 Confirm All Necessary DVS Kernel Modules Are Loaded

After confirming that all the DVS kernel modules are installed, run the following command to verify that they and the `lnet` kernel module are loaded.

```
ncn-w001# lsmod | grep dvs
dvs                    405504  0
dvsipc                 172032  2 dvs
dvsipc_lnet             57344  1 dvsipc
dvsproc                143360  3 dvsipc,dvsipc_lnet,dvs
craytrace               20480  5 dvsipc,dvsproc,dvsipc_lnet,dvs
dvskatlas               20480  4 dvsipc,dvsproc,dvsipc_lnet,dvs
lnet                   634880  3 ksocklnd,dvsipc_lnet
libcfs                 512000  3 ksocklnd,lnet,dvsipc_lnet
```

### 2.9.3 Confirm the LNet network is active

```
x9000c1s1b0n0# lnetctl net show
net:
    - net type: lo
      local NI(s):
```

```
        - nid: 0@lo
          status: up
    - net type: tcp
      local NI(s):
        - nid: 10.150.0.18@tcp
          status: up
          interfaces:
              0: hsn0
        - nid: 10.150.0.20@tcp
          status: up
          interfaces:
              0: hsn1
        - nid: 10.150.0.19@tcp
          status: up
          interfaces:
              0: hsn2
        - nid: 10.150.0.17@tcp
          status: up
          interfaces:
              0: hsn3
```

### 2.9.4 Print Out the DVS Node Map

Print out the file DVS uses as the list of all nodes on the HPE Cray EX system. This list contains the mapping of xnames to IP addresses used by DVS to route traffic.

```
ncn-w001# cat /proc/fs/dvs/node_map
...
800024 x3000c0s10b0n0    10.252.0.12@tcp99
800032 x3000c0s12b0n0    10.252.0.4@tcp99
800040 x3000c0s14b0n0    10.252.0.5@tcp99
800048 x3000c0s16b0n0    10.252.0.6@tcp99
800064 x3000c0s20b0n0    10.252.0.8@tcp99
800072 x3000c0s22b0n0    10.252.0.9@tcp99
     8 x3000c0s24b1n0    10.252.0.28@tcp99
    16 x3000c0s24b2n0    10.252.0.33@tcp99
    24 x3000c0s24b3n0    10.252.0.27@tcp99
    32 x3000c0s24b4n0    10.252.0.26@tcp99
800008 x3000c0s6b0n0     10.252.0.10@tcp99
800016 x3000c0s8b0n0     10.252.0.11@tcp99
```

### 2.9.5 Search for DVS-Related Kernel Messages

Search for log messages from DVS in `/var/log/messages`.

```
ncn-w001# grep -Pi 'dvs|dvs-exports' /var/log/messages
. . .
2022-04-27T13:40:04.496968+00:00 postoak0035 dvs_wait_for_mac[6440]:  Waiting for valid mac
2022-04-27T13:40:09.502414+00:00 postoak0035 dvs_wait_for_mac[6440]: found mac  02
2022-04-27T13:40:09.519661+00:00 postoak0035 systemd[1]: Starting Cray DVS CLI...
2022-04-27T13:40:09.532420+00:00 postoak0035 kernel: katlas: init_module: katlas loaded, currently disabled
2022-04-27T13:40:09.558931+00:00 postoak0035 kernel: DVS: Revision: 148b11f4 Built: Apr 20 2022 @ 03:01:04 ag
2022-04-27T13:40:09.558937+00:00 postoak0035 kernel: DVS debugfs: Revision: 148b11f4 Built: Apr 20 2022 @ 03:
2022-04-27T13:40:09.567320+00:00 postoak0035 kernel: DVS: successfully added 10 new nodes into map.
2022-04-27T13:40:09.598931+00:00 postoak0035 kernel: DVS: message size checks complete.
2022-04-27T13:40:09.696768+00:00 postoak0035 systemd[1]: cray-dvs.service: Succeeded.
2022-04-27T13:40:09.696970+00:00 postoak0035 systemd[1]: Finished Cray DVS CLI.
2022-04-27T13:40:09.697097+00:00 postoak0035 systemd[1]: Startup finished in 2min 51.678s (firmware) + 7.398s
2022-04-27T13:40:09.698585+00:00 postoak0035 systemd[1]: Starting Cray DVS Exports CLI...
```

```
2022-04-27T13:40:09.751215+00:00 postoak0035 systemd[1]: cray-dvs-exports.service: Succeeded.
2022-04-27T13:40:09.751428+00:00 postoak0035 systemd[1]: Finished Cray DVS Exports CLI.
```

### 2.9.6 List All Running DVS Processes

Run the following command to list all running DVS processes:

```
ncn-w001# ps ax | grep -i dvs | grep -v 'grep'
```

### 2.9.7 Troubleshoot Node Map IP Change Issues

DVS but does not process changes in existing node map entries. For example, if the IP address of a node changes, already configured nodes will not see that the new address of that node until DVS is reloaded.

If the compute node IP addresses change, the /var/log/messages file fill have entries that will look like:

```
[Tue Apr 26 21:54:26 2022] DVS: merge_one#376: New node map entry does not match the existing entry
[Tue Apr 26 21:54:26 2022] DVS: merge_one#378:  nid: 7 -> 7
[Tue Apr 26 21:54:26 2022] DVS: merge_one#380:  name: 'postoak0041' -> 'postoak0041'
[Tue Apr 26 21:54:26 2022] DVS: merge_one#382:  address: '2160@kfi' -> '2288@kfi'
[Tue Apr 26 21:54:26 2022] DVS: merge_one#383:  Ignoring.
```

The NCN dmesg output will contain messages like:

```
2020-07-21T18:01:56.400796+00:00 ncn-w002 kernel: [101229.684364] DVS: merge_one#351: New node map entry
does not match the existing entry
2020-07-21T18:01:56.400818+00:00 ncn-w002 kernel: [101229.684365] DVS: merge_one#353: nid: 24 -> 24
2020-07-21T18:01:56.400818+00:00 ncn-w002 kernel: [101229.684365] DVS: merge_one#355: name: 'x3000c0s19b3n0'
-> 'x3000c0s19b3n0'
2020-07-21T18:01:56.400819+00:00 ncn-w002 kernel: [101229.684366] DVS: merge_one#357: address:
'10.252.0.29@tcp99' -> '10.252.0.31@tcp99'
2020-07-21T18:01:56.400819+00:00 ncn-w002 kernel: [101229.684366] DVS: merge_one#358: Ignoring.
```

If there are just a few nodes with changed addresses, fix this issue without causing systemwide DVS service disruption:

1. Generate a new node map and include it in the DVS server and compute images. Activate the image if your system has Leader nodes. Perform a reboot on each DVS server, one at a time, to restart DVS on those nodes. Perform the manual DVS failover and failback procedures after each dvs server reboot. This unloads the map on each worker node and reloads the new one. Space out the restarts, so that the running compute nodes will fail over from the restarting node to a running one. The compute node will not ever suffer complete disruption.
2. Reboot the nonresponsive compute nodes. At this point, the DVS servers and problem compute nodes will have matching values in their map and DVS will operate normally.

## 2.10 DVS Reference Information

This section contains content that explains DVS behavior and details DVS configuration options.

### 2.10.1 DVS Modes

Some DVS mount option combinations are common enough to be characterized as "modes" of use. There are several parallel DVS modes and one serial mode.

#### 2.10.1.1 Serial vs parallel DVS modes

A DVS ***mode*** is a name given to a combination of mount options which satisfies a common use case. These modes allow customers to tune DVS behavior as wanted to increase performance, ensure POSIX read/write atomicity, or both. There are two general types of DVS modes:

- **Serial:** One DVS node projects a file system to multiple client nodes. There is only one serial mode. See DVS Serial Mode.
- **Parallel:** multiple DVS nodes project a file system to client nodes. There are several ways to configure DVS servers to work in parallel to project a file system and thus multiple parallel modes. For more information on each of these parallel modes, see following sections:
  - DVS Loadbalance Mode
  - DVS Cluster Parallel Mode
  - DVS Stripe Parallel Mode

- DVS Atomic Stripe Parallel Mode

#### 2.10.1.2 A single projected file system can use multiple modes

The HPE Cray EX system administrator can set the DVS mount options for projected file systems during initial system configuration and when the system is configured to project additional (for example, external) file systems over DVS. The administrator can make several DVS modes for the same file system available on the same client node. Administrators can accomplish this by creating multiple systemd mount unit files for the same underlying file system in the `/etc/systemd/system` directory, with different mount options in each unit file. This will cause the affected DVS clients to mount the same file system with different mount options on different mount points.

Non-root users cannot choose among DVS modes unless the system administrator has configured the system to make more than one mode available.

- **DVS Serial Mode**
  Serial mode is the simplest implementation of DVS and the only option if no cluster or shared file system available. In this mode, a single DVS server node handles all I/O traffic for the projected file system.
- **DVS Cluster Parallel Mode**
  DVS Cluster Parallel mode distributes I/O among multiple servers by assigning each projected file to a single server. All file data and metadata I/O for a given file is handled by the DVS server assigned to that file.
- **DVS Stripe Parallel Mode**
  DVS Stripe Parallel Mode can provide greater I/O performance than Cluster Parallel Mode. Stripe Parallel Mode, however, has restrictions that Cluster Parallel Mode does not.
- **DVS Atomic Stripe Parallel Mode**
  DVS Atomic Stripe Parallel Mode provides both parallel file data I/O and POSIX read/write atomicity compliance.
- **DVS Loadbalance Mode**
  DVS Loadbalance Mode combines caching with the load distribution. This mode can only be used on read-only file systems.

#### 2.10.1.3 DVS Serial Mode

##### 2.10.1.3.1 How Serial Mode Operates

In this mode, a single DVS server node handles all I/O traffic from all clients (usually compute nodes) for the projected file system. DVS can project multiple file systems from the same server by using the `maxnodes=1` option for each client mount.

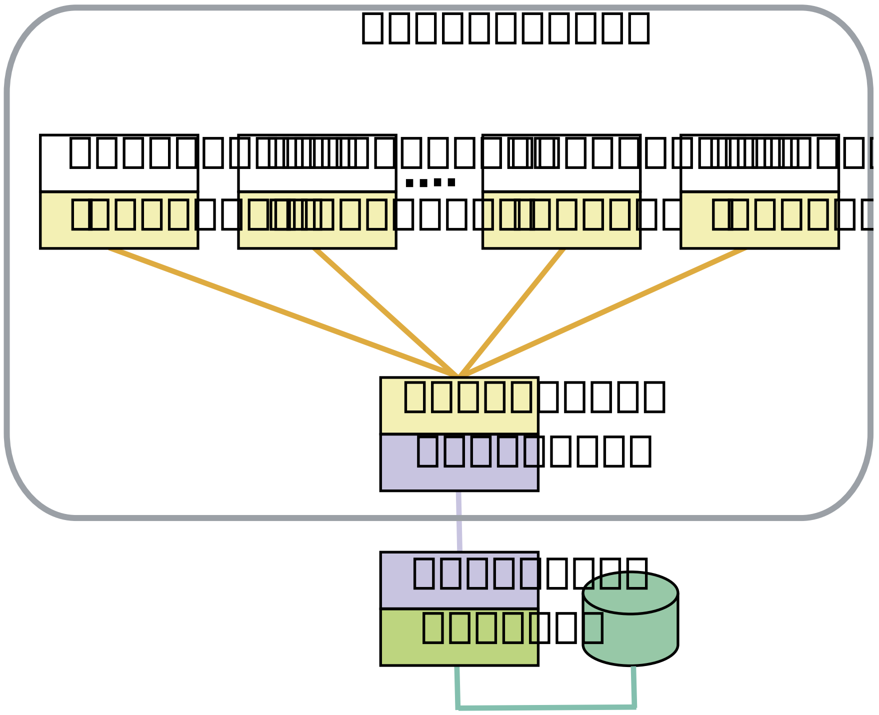
Figure 2: diagram showing a single DVS server using Serial Mode to project to clients

#### 2.10.1.3.2 Advantages of Serial Mode

Serial Mode is the only DVS mode that can be used if there is no underlying cluster or shared file system available. This mode is the simplest implementation of DVS, since it requires only one DVS server node to project a file system.

This mode guarantees that all bytes associated with a read or write are not interleaved with bytes from other read or write operations. In other words, DVS Serial mode adheres to POSIX read/write atomicity rules.

#### 2.10.1.3.3 Limitations of Serial Mode

DVS Serial Mode does not provide any failover or failback resiliency, and I/O performance is limited to what a single DVS server node can provide.

### 2.10.1.4 DVS Cluster Parallel Mode

#### 2.10.1.4.1 How Cluster Parallel Mode Operates

DVS Cluster Parallel mode distributes file I/O and metadata operations among multiple servers by assigning each projected file to a single server. All file data and metadata I/O for a given file is handled by the DVS server assigned to that file. I/O for a single file goes only to the chosen server. Therefore, each DVS client interacts with multiple servers to use the entire projected file system. This mode is often used for large file systems.

The following figure illustrates how Cluster Parallel Mode works. DVS is mounted on `/foo` on the DVS client, and three different files—`bar1`, `bar2`, and `bar3`—are handled by three different DVS servers (nodes), thus distributing the load. The server used to perform I/O or metadata operations on a file is chosen by an internal hash on the underlying file or directory inode number. Once a server has been selected for a file, Cluster Parallel Mode looks like Serial Mode: All I/O and metadata operations for that file from all clients use the selected server.

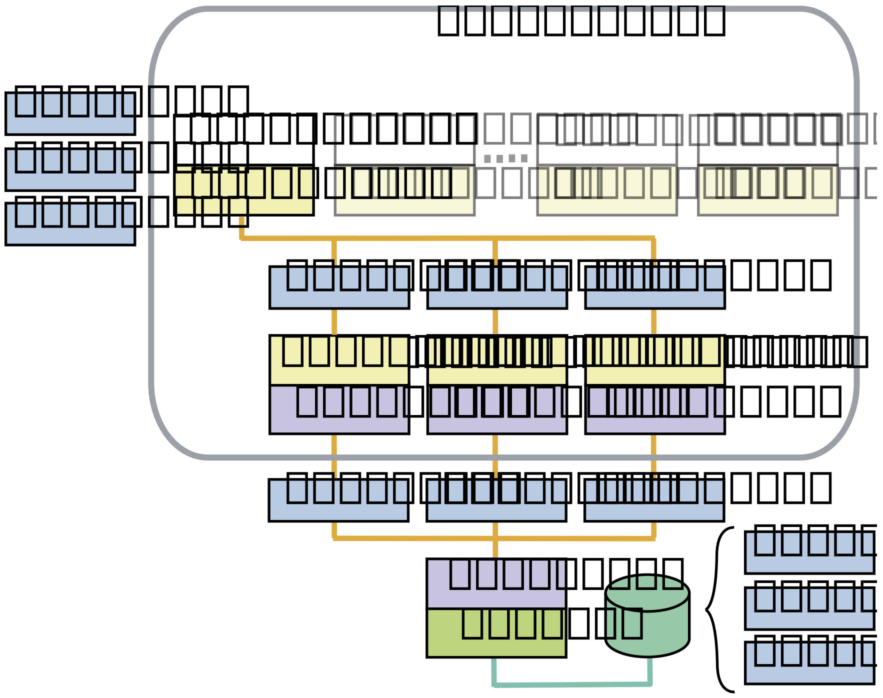

Figure 3: Diagram showing multiple DVS servers each projecting a different file from the same file system

#### 2.10.1.4.2 Advantages of Cluster Parallel Mode

DVS Cluster Parallel Mode adheres to POSIX read/write atomicity rules. That is, all bytes associated with a read or write are not interleaved with bytes from other read or write operations.

This mode improves performance while preventing any one server from becoming overloaded. Also, the way Cluster Parallel Mode distributes I/O prevents file system coherency thrash.

#### 2.10.1.4.3 Limitations of Cluster Parallel Mode

Only a shared file system such as IBM Spectrum Scale (formerly "GPFS") or Lustre can be projected through DVS Cluster Parallel Mode. Therefore, Cluster Parallel Mode cannot be used to project an NFS file system.

### 2.10.1.5 DVS Stripe Parallel Mode

#### 2.10.1.5.1 How Stripe Parallel Mode Operates

Stripe Parallel Mode builds upon Cluster Parallel Mode to provide an extra level of I/O forwarding parallelization. Each DVS server (node) can serve all files. DVS servers are selected based on the file inode and offsets of data within the file relative to the DVS block size value (`blksize`).

For example, in the following figure, DVS is mounted to /foo on the DVS client. The I/O for three different blocks (or segments) of data within file `bar`—seg1, seg2, and seg3—is handled by three different DVS servers. Thus Stripe Parallel Mode distributes the load at a more granular level than that achieved by cluster parallel mode.

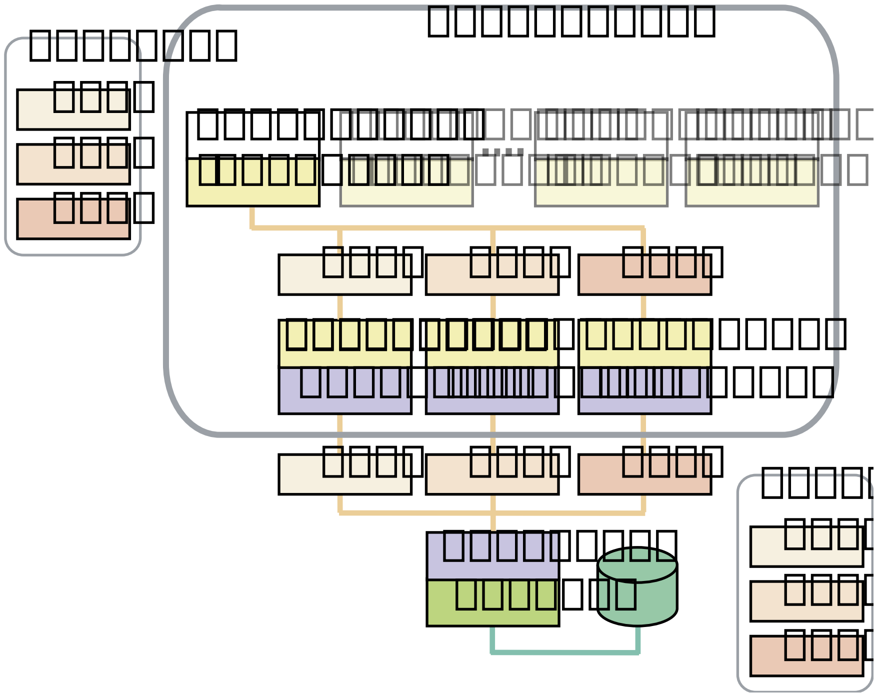

Figure 4: Diagram showing different file segments of a file and each one is projected from a different DVS server

#### 2.10.1.5.2 Advantages of Stripe Parallel Mode

Stripe Parallel Mode can provide greater aggregate I/O bandwidth than Cluster Parallel Mode when forwarding I/O from a coherent cluster file system. IBM Spectrum Scale has been tested extensively using this mode.

DVS routes all data I/O for each block of file data for the same file from all clients to the same server. Since only one DVS server reads or writes to a given block, file system coherency thrash is prevented. While file I/O is distributed at the block level, however, file metadata operations are distributed as in Cluster Parallel Mode: the metadata operations of a given file are always handled by the same DVS server.

#### 2.10.1.5.3 Limitations

NFS cannot be used in Stripe Parallel Mode because NFS implements close-to-open cache consistency. Therefore, striping data across the NFS clients could compromise data integrity. Also, Stripe Parallel Mode **does not** adhere to POSIX read and write atomicity rules. Therefore applications which require such atomic reads and writes will have to implement their own file locking.

### 2.10.1.6 DVS Atomic Stripe Parallel Mode

DVS Atomic Stripe Parallel Mode provides both parallel file data I/O and POSIX read/write atomicity compliance.

#### 2.10.1.6.1 How Atomic Stripe Parallel Mode operates

Atomic Stripe Parallel Mode uses the same method as Stripe Parallel Mode to select a server. Atomic Stripe Parallel Mode, however, uses only that selected server for the entire read or write operation. Thus, all I/O operations are atomic. Atomic Stripe Parallel Mode allows DVS clients to access different servers for subsequent I/O requests if they have different starting offsets within the file.

Whereas Stripe Parallel Mode provides parallelism within a file at the granularity of the DVS block size, Atomic Stripe Parallel Mode provides I/O granularity based on the size of the application writes.

#### 2.10.1.6.2 Advantages of Atomic Stripe Parallel Mode

DVS Atomic Stripe Parallel mode adheres to POSIX read and write atomicity rules while still allowing for possible parallel file data I/O. This mode allows applications which do not implement their own file locking to enjoy the performance benefit of parallel file data I/O.

#### 2.10.1.6.3 Disadvantages of Atomic Stripe Parallel Mode

NFS cannot be projected using Atomic Stripe Parallel Mode.

### 2.10.1.7 DVS Loadbalance Mode

#### 2.10.1.7.1 How Loadbalance Mode operates

The client nodes automatically select the server based on a DVS-internal node ID (NID) from the list of available server nodes.

DVS automatically enables the `cache` mount option in Loadbalance Mode because using cache on a read-only mount can improve performance. The first time a DVS client requests file data from a DVS server, the server retrieves that data. The client then stores it in the page cache. While the application is running, all future references to that data are local to the memory of the client, and DVS will not be involved at all. However, if the node runs low on memory, the Linux kernel may remove these pages. Then the client must fetch the data from the DVS server on the next reference to repopulate the page cache of the client.

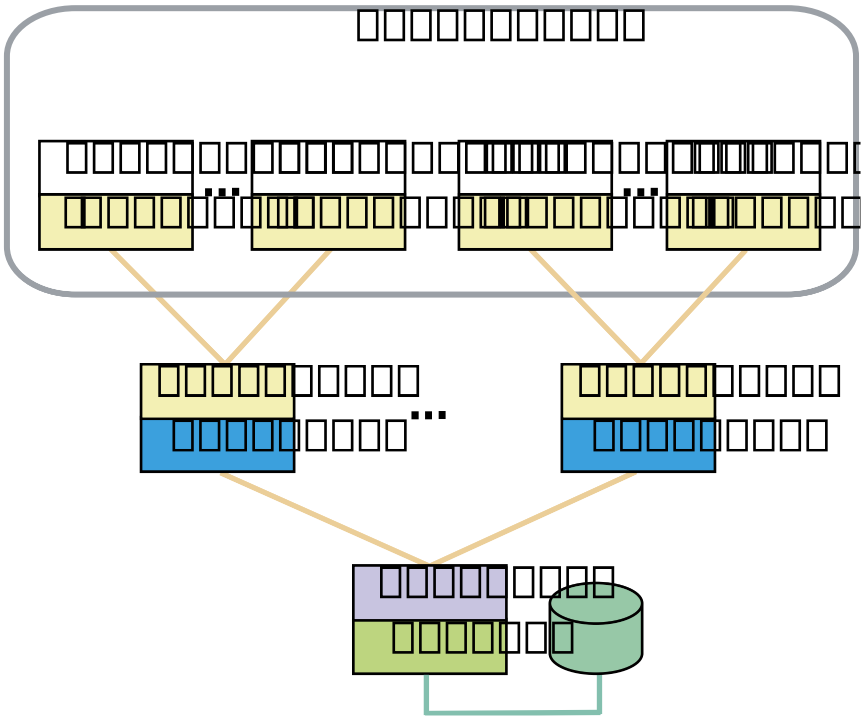

Figure 5: diagram showing multiple DVS servers projecting the same file system to different groups of clients

#### 2.10.1.7.2 Advantages of Loadbalance Mode

DVS Loadbalance Mode combines fast, cached access to read-only file systems with load distribution across multiple DVS servers. Loadbalance Mode evenly distributes loads across servers.

#### 2.10.1.7.3 Limitations of Loadbalance Mode

DVS Loadbalance Mode can only project read-only file systems.

### 2.10.2 DVS Failover and Failback

Automatic DVS failover is not available in the HPCM environment for the COS 2.3 release. A manual failover capability is provide and documented in DVS COS 2.3 Failover Story DVS automatic failover will be provided the COS 2.4 release.

#### 2.10.2.1 General System Requirements for Failover

In order for automatic failover to work, the DVS nodes in the system need notifications about the changes of availability of the other DVS nodes. These State Change Notifications (SCNs) need to be provided by the cluster infrastructure.

There are two basic types of SCNs that DVS needs:

1. DVS is available on a node. That is the DVS modules are loaded and ready to receive RPCs from other nodes.
2. DVS is unavailable on a node. The modules are unavailable for RPCs from other nodes. This type of SCN has two causes: A) The DVS modules were unloaded on the node (perhaps because they're being removed as part of an upgrade) and B) The whole node crashed or rebooted or was made unavailable in some way.

It's interesting to note that while DVS needs to know when a node is shutdown, DVS does not care about a node booting. It's not until the DVS modules are loaded that an SCN should be sent to the other nodes in the system.

**Note** that the speed of detecting these events and the distribution of the resulting SCNs is important. Particularly, the SCN about servers becoming unavailable determines how big of a performance hiccup that the clients see during a failure event.

In general, every DVS node should receive SCNs for every other node. That's the ideal. The bare minimum is that the clients get SCNs about the availability of the servers. So, the possible solutions (in order of increasing desirability) are:

1. Clients receive SCNs about the Servers. This the bare minimum.
2. Clients receive SCNs about the Servers and the Servers receive SCNs about the Clients. Servers knowing about Clients becoming unavailable allows them to clean up messages stuck in transit. Those messages are automatically cleaned up if the client remounts, but the SCN allows the process to happen more quickly.
3. All DVS nodes receive SCNs about all the other DVS nodes. In general DVS clients don't talk to other clients and servers don't talk to other servers. However, sending SCNs everywhere allows for the possibility of DVS being improved to take advantage of those communication patterns.

#### 2.10.2.2 DVS failover behavior is dependent on the DVS mode used:

- **Loadbalance Mode:** Any mount point using loadbalance mode automatically recalibrates the existing client-to-server routes to ensure that the clients are evenly distributed across the remaining servers. When failback occurs, this process is repeated.
- **Cluster Parallel Mode:** Any mount point using cluster parallel mode automatically redirects I/O to one of the remaining DVS servers for any file that previously routed to the now-down server. When failback occurs, files are rerouted to their original server.
- **Stripe Parallel or Atomic Stripe Parallel Modes:** Any mount point using stripe parallel mode or atomic stripe parallel mode automatically re-stripes I/O across the remaining DVS servers in an even manner. When failback occurs, files are re-striped to their original pattern.

If all servers fail, I/O is retried as described by the retry client mount option. See DVS Mount Options for more information about the DVS retry mount option.

### 2.10.3 About the Close-to-Open Coherency Model

The only times the client cached copy of a file is guaranteed to match the copy stored by the DVS server are when the file is closed and when it is opened. This aspect of DVS read/write caching is important to understand. Also, kernel page granularity affects coherency.

The DVS read/write cache capability provides a close-to-open coherency model, which means File read operations are only guaranteed to see file data that was available on the server at the time the file was opened.

This does not imply that server data newer than file open time cannot be read by the client or that some amount of client write data will not be written to the server prior to file close. Rather, that file open and close are the only times this can be **guaranteed**. To preserve the close-to-open coherency, any remaining 'dirty' (that is, not yet written back) cached client data for a file is written back to the DVS server at file close time, and any cached client data is invalidated at file open if the file on the server is newer than the cached data. This coherency model is similar to that provided by NFS.

### 2.10.4 DVS Fairness of Service

By default, DVS services incoming I/O requests in a way which prevents demanding users or clients from degrading the I/O performance of others. Administrators can disable DVS Fairness of Service if absolute application performance is wanted.

DVS creates user- or job-specific request queues for clients. Originally, DVS used one queue to handle requests in a First In, First Out (FIFO) fashion. Since all clients shared one queue, a demanding job could tax the server disproportionately and the other clients would have to wait until the requests of the demanding client are completed. DVS Fairness of Service solves that problem by creating a ***list*** of queues—one queue for each client and job. The list of queues is processed in a circular fashion. When a message thread is available, it fetches the first queue on the list, moves that queue to the end of the list, and processes the first message in that queue. This helps to distribute the workload and potentially helps contending applications perform better.

Fairness of Service is enabled by default in DVS, but administrators can disable it. Disabling Fairness of Service reverts DVS to using a single message queue for all incoming requests, causing all requests to be processed in the order they were received. For demanding applications which issue a disproportionately large number of requests, this could increase their performance as all their requests can be processed in order.

### 2.10.5 HPE Cray EX DVS Configuration Overview

In HPE Cray EX supercomputers, DVS is configured mainly by editing six configuration files within the COS image.

There are three approaches for including DVS into the COS image.

- The first is to use a single image for both DVS server and DVS client and the /etc/dvs_server_list.conf file will determine which nodes will be configured as a DVS server. This is suitable for smaller systems or systems that do not use different DVS tuning parameters between the DVS servers and DVS clients.
- The second is to configure separate images for the DVS server, DVS server running on a gateway node, and for the DVS client (Compute Node). This is nearly identical to the approach used in CSM, but the images are managed with HPCM. The DVS client image will contain and 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 third is to use a hybrid of the first and second approaches. An example would be where the DVS servers and gateway nodes share the same image and the compute nodes use a separate image.

The system administrator creates or modifies the DVS, LNet, and systemd configuration files in a COS based image to load and start LNet and DVS services. See the “HPE Performance Cluster Manager Administration Guide”, Section “Using commands to provision nodes and manage software images”.

It is recommended that the COS base images used with DVS be manitained with the HPCM Verson Control System (VCS). See the “Using the version control system (VCS)” Section in the “HPE Performance Cluster Manager Administration Guide”.

#### 2.10.5.1 How DVS Configuration is Applied in HPCM

The DVS configuration process (managed by the system administrator) is the same for DVS servers and DVS clients (compute nodes).

Most kernel module parameters which affect DVS and LNet are typically changed by adding lines to a configuration file inside the `/etc/modprobe.d/` directory residing on the nodes that run the DVS servers and DVS clients. Configuration changes to these files are made prior to booting the affected nodes, the changes take effect at boot, and the kernel module parameters changed will persist across boots. Refer to Change LNet or DVS Parameters for directions on how to change these parameters.

For DVS servers, gateways, and CNs, LNet and DVS are configured through editing the LNet and DVS configuration files in the image directory on the HPCM system admin node local file system. These are `/etc/lnet.conf`, `/etc/modprobe.d/lnet.conf`, `/etc/modprobe.d/dvs.conf`, `/etc/dvs_exports.yml`, and `/etc/dvs_server_list.conf`.

The `/etc/dvs_node_map` file is included in the image as well, but it is generated prior to production image boot. See How to Populate Node Maps

You can copy files directly to the image directories on the hpcm-adm file system. But for examining the rpm database and systemd enable/disable of services you will need to chroot to the image directory.

#### 2.10.5.2 DVS Configuration Files

Brief descriptions and locations of the DVS configuration files in HPE Cray EX supercomputers managed by HPCM.

#### 2.10.5.2.1 `/etc/dvs_node_map.conf` and `/etc/modprobe.d/dvs.conf`

The `/etc/modprobe.d/dvs.conf` file configures the DVS kernel modules with different parameters and affects the way DVS works internally. In the HPCM environment, the contents of this file is defined by the system administrator.

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.

The `/etc/dvs_node_map.conf` file configures how the DVS node map is generated. The node map contains a list of all the xnames and LNet addresses which the DVS cluster will use. In the HPCM environment, the contents of this file is defined by the system administrator.

The cray-dvs-hpcm RPM installs a default /etc/dvs_node_map.conf file.

Sometimes, different sets of options must be specified on different nodes. For example, the options needed on CNs may be different from what are needed on DVS servers or gateway nodes. To accommodate such a scenario, users can use different, specific images to specify different values in the configuration files for different types of hosts. For example, the `/etc/modprobe.d/dvs.conf` file for DVS servers can be specified in a DVS server image and the version for CNs can be specified in another CN specific image. For instructions on how to apply different `dvs.conf` files to different nodes, see Configure DVS Using dvs.conf.

Note that the `/etc/dvs_node_map.conf` must be the same for all nodes in the system.

#### 2.10.5.2.2 `/etc/modprobe.d/lnet.conf` and `/etc/lnet.conf`

The `/etc/modprobe.d/lnet.conf` file configures the LNet kernel modules with different parameters and affects the way LNet works. In the HPCM environment, the contents of this file is defined by the system administrator.

The `/etc/lnet.conf` file contains more robust configuration options than the `/etc/modprobe.d/lnet.conf` file. In the HPCM environment, the contents of this file is defined by the system administrator.

#### 2.10.5.2.3 The /etc/dvs_server_list.conf file

The `/etc/dvs_server_list.conf` file provides a list of the DVS servers within a HPCM managed system, one server per line. If a /etc/dvs_server_list.conf file is provided in the image then the DVS service file will set the DVS clients (those not in the file).

#### 2.10.5.2.4 The /etc/dvs_exports.yml file

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 executing the /opt/cray/dvs/default/sbin/dvs_exports -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 could be 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 file

```
system_name-adm#  cat /etc/dvs_exports.yml
exports: []
```

An example of a DVS Server /etc/dvs_exports file.

```
system_name-adm#  cat /etc/dvs_exports.yml
exports:
- mode: rw
  path: /
```

#### 2.10.5.2.5 The /etc/dvs_node_map file.

In the HPCM environment, a dvs_node_map file will need to be generated that includes all the COS image nodes that utilize DVS projected file systems. In HPCM, we will be pre-building the node maps. We add the node map into the COS node images and boot them. Since we are not generating the node map at boot time, there will be no DNS lookups to slow boot time and hence there’s no reason to have smaller maps for the compute nodes in this case. Since there’s no downside to having the full map everywhere, all the nodes have the same node map.

### 2.10.6 When to Use Temporary, Client-Side, Per-File Read Caching

Applications which have a “read many, write once” I/O pattern can benefit greatly from the DVS `ro_cache` mount option.

The `ro_cache` mount option is still useful for situations where DVS client nodes perform several reads of a file before performing a single write (if any at all). The `ro_cache` mount option enables per-file, client-side, read caching until the first write operation on all files in mounted file system.

When the `ro_cache` mount option is used on a read/write file system, DVS enables caching for all reads on every file in that file system until a DVS client node opens one of the files with write access. When that first client attempts to write to the file, the DVS server notifies every client node to drop their locally cached version of the file. That file is never cached again and DVS handles all subsequent file I/O as it would on a normal read/write file system. All other files in that file system, however, will still be cached as long as every node opens them only for reads.

This caching mode enables codes which have a “read many, write once” I/O pattern to enjoy the benefits of client-side caching and coherency across multiple DVS client nodes at the same time for as long as possible.

### 2.10.7 DVS Environment Variables

By default, user environment variables allow client override of mount options specified during configuration.

By default, user environment variables allow client override of options specified during configuration and are evaluated whenever a file is opened by DVS. However, if the `nouserenv` option is included in the `options` setting of the `client_mount` configuration setting, then user environment variables are disabled for that client mount.

The following environment variables are for use in the default case:

**Cray DVS User Environment Variables**

| Variable Name | Options | Purpose |
|---|---|---|
| `DVS_ATOMIC` | `on\|off` | Overrides the `atomic` or `noatomic` mount options. |
| `DVS_BLOCKSIZE` | `n` | A nonzero number, `n` overrides the `blksize` mount option. **Note:** Unlike most other mount option and environment variable pairs, `DVS_BLOCKSIZE` and `blksize` have subtly different spellings. |
| `DVS_CACHE` | `on\|off` | Overrides the `cache` or `nocache` mount options. Exercise caution if using this variable. |
| `DVS_CACHE_READ_SZ` | `n` | A positive integer, n overrides the `cache_read_sz` mount option. |
| `DVS_CLOSESYNC` | `on\|off` | Overrides the `closesync` or `noclosesync` mount options. **Note:** Periodic sync functions similarly to the DVS `closesync` mount option, but it is more efficient and is enabled by default. Hewlett Packard Enterprise recommends not using `closesync` or this associated environment variable. |

| Variable Name | Options | Purpose |
|---|---|---|
| `DVS_DATASYNC` | `on\|off` | Overrides the `datasync` or `nodatasync` mount options.**Note:** Setting `DVS_DATASYNC` to `on` can slow down an application considerably. The periodic sync feature, enabled by default, is a better way to synchronize data. See Periodic Sync Promotes Data and Application Resiliency for more information. |
| `DVS_DEFEROPENS` | `on\|off` | Overrides the `deferopens` or `nodeferopens` mount options. |
| `DVS_KILLPROCESS` | `on\|off` | Overrides the `killprocess` or `nokillprocess` mount options. |
| `DVS_MAXNODES` | `n` | A nonzero number, n overrides the `maxnodes` mount option. The specified value of `maxnodes` must be greater than zero and less than or equal to the number of server nodes specified on the mount, otherwise the variable has no effect. |

#### 2.10.8 DVS Configuration Settings

An overview of the three ways DVS users and administrators can configure DVS and enhance its performance.

Administrators and users can configure DVS (Data Virtualization Service) through:

- Writing values to the DVS-related `/sys` files on DVS clients and servers using the echo command.
- Adding mount options in the systemd mount unit file (using the `Options=` directive) used by DVS clients to mount DVS projected content other than compute node root file systems.
- Setting DVS kernel module parameters and mount options by creating or editing the `dvs.conf` and `lnet.conf` files (both reside in the `/etc/modprobe.d` directory) in the COS image.

**Important:** To prevent mount failure, only use settings that are compatible, in accordance with the instructions in this guide.

##### 2.10.8.1 DVS Mount Options

DVS clients can use a long list of options when mounting a file system. This topic explains what they all do.

- `loadbalance`

  Used to specify loadbalance mode, which more evenly distributes loads across DVS servers, as all servers are used to access the same file. The server is chosen based on the compute node doing the access. This maximizes the scalability of both reads and meta operations like `open()`, `stat()`, and `close()` because all servers can be used for every file. Loadbalance mode is valid only for read-only mounts.

  - Default value: not enabled
  - Associated environment variable: none
  - Related settings/options: When `loadbalance` is enabled, the underlying DVS implementation automatically sets the `readonly` setting to `true` and sets these additional options: `cache=1`, `failover=1`, `maxnodes=1`, and `hash_on_nid=1`. Hewlett Packard Enterprise recommends setting `attrcache_timeout=14400` as well to take advantage of the mount being read-only.

- `attrcache_timeout`

Enables and sets client-side attribute caching, which can significantly increase performance, most notably in pathname lookup situations. File attributes and `dentries` for getattr requests, pathname lookups, and so forth, are read from DVS servers and cached on the DVS client for the number of seconds specified. For example, `attrcache_timeout=5` caches attributes for five seconds. Subsequent lookups or getattr requests use the cached attributes until the timeout expires, at which point they are read and cached again on first reference. When attribute caching is disabled, DVS clients must send a lookup request to a DVS server for every level of a pathname, and repeat this for every pathname operation. When it is enabled, it sends a lookup request to a DVS server for every level of a pathname once per the number of specified seconds.

**Note:** An administrator with root privilege can force a cache revalidation at any time, not just when the timeout has expired. See Force a Cache Revalidation on a DVS Mount Point .

- Default value: 3 seconds.

  **Important:** This default is intended to maximize the safety of read/write mounts. This means that to enhance system performance for read-only mounts, configure this setting by entering a larger value (Hewlett Packard Enterprise recommends `attrcache_timeout=14400`). Large values (for example, `14400`) for this mount option should not be used for read/write file systems due to the risk of file system corruption. Not specifying this mount option will result in the underlying safe default being used.

  This setting must be kept at a low value for read/write mounts to prevent application segfaults and other problems. The default setting allows the caching to be effective during the start-up of an application. DVS will detect a stale or deleted existing file on every open, so `attrcache_timeout` is needed primarily to prevent a previous failed access (due to ENOENT) from masking the subsequent creation of that file for too long.
- Associated environment variable: none

• **`atomic` / `noatomic`**

`atomic` enables atomic stripe parallel mode. This ensures that stripe parallel requests adhere to POSIX read/write atomicity rules. DVS clients send each I/O request to a single DVS server to ensure that the bytes are not interleaved with other requests from DVS clients. The DVS server used to perform the read, write, or metadata operation is selected using an internal hash involving the underlying file or directory inode number and the offset of data into the file relative to the DVS block size.

`noatomic` disables atomic stripe parallel mode. If there are multiple DVS servers and neither loadbalance nor cluster parallel mode is specified, DVS stripes I/O requests across multiple servers and does not necessarily adhere to POSIX read/write atomicity rules if file locking is not used.

- Default value: `noatomic`
- Associated environment variable: `DVS_ATOMIC`
- Related settings/options: none

• **`blksize=n`**

`blksize=n` sets the DVS block size to n bytes. Used in striping.

- Default value: 524288 (512 KB)
- Associated environment variable: `DVS_BLOCKSIZE`
- Related settings/options: none

• **`cache` / `nocache`**

`cache` enables client-side caching of data. DVS on HPE Cray EX systems only support client-side read caching. The client node caches reads from the DVS server node and provides data to user applications from the page cache if possible, instead of performing a data transfer from the DVS server node. DVS is not a clustered file system; no coherency is maintained among multiple DVS client nodes reading the same file. If cache is enabled and data consistency is required, applications must take care to synchronize their accesses to the shared file.

`nocache` disables client-side read caching.

- Default value: `nocache`
- Associated environment variable: `DVS_CACHE` (use with caution)
- Related settings/options: When `loadbalance` is enabled, DVS automatically enables `cache`. If `loadbalance` is not used, both `readonly` and `cache` option must be specified for the client node to cache read data.

• **`cache_read_sz`**

`cache_read_sz` is a limit that can be specified to prevent reads or writes over this size from being cached in the Linux page cache.

  - Default value: 0
  - Associated environment variable: DVS_CACHE_READ_SZ
  - Related settings/options: If the `cache` mount option is not used, DVS ignores `cache_read_sz`.

- **`closesync` / `noclosesync`**

  `closesync` enables data synchronization on last close of a file. When a process performs the final close of a file descriptor, and forwards the close to the DVS server, the DVS server node waits until data has been written to the underlying media before indicating that the close has completed. Because DVS does not cache data on client nodes (unless the `cache` option is used) and has no replay capabilities, this ensures that data is not lost if a server node crashes after an application has exited.

  `noclosesync` causes DVS to return a `close()` request immediately.

  - Default value: `noclosesync`
  - Associated environment variable: DVS_CLOSESYNC
  - Related settings/options: The `closesync` option is redundant with periodic sync, which is enabled by default. Because periodic sync is more efficient than `closesync`, Hewlett Packard Enterprise recommends letting periodic sync take care of data synchronization instead of using this mount option.

- **`datasync` / `nodatasync`**

  `datasync` enables data synchronization. The DVS server node waits until data has been written to the underlying media before indicating that the write has completed. Can significantly impact performance.

  `nodatasync` causes a DVS server node to return from a write request as soon as the user data has been written into the page cache on the server node.

  - Default value: `nodatasync`
  - Associated environment variable: DVS_DATASYNC
  - Related settings/options: none

- **`deferopens` / `nodeferopens`**

  `deferopens` defers DVS client `open()` requests to DVS servers for a given set of conditions. When a file is opened in stripe parallel mode or atomic stripe parallel mode, DVS clients send the `open()` request to a single DVS server only. More `open()` requests are sent as necessary when the DVS client performs a read or write to a different server for the first time. The `deferopens` option deviates from POSIX specifications. For example, if a file was removed after the initial `open()` succeeded but before deferred opens were initiated by a read or write operation to a new server, the read or write operation would fail with `errno` set to ENOENT because the `open()` was unable to open the file.

  `nodeferopens` disables the deferral of DVS client `open()` requests to DVS servers. When a file is open in stripe parallel mode or atomic stripe parallel mode, DVS clients send `open()` requests to all DVS servers denoted by `nodename` or `nodefile`.

  - Default value: `nodeferopens`
  - Associated environment variable: DVS_DEFEROPENS

- **`distribute_create_ops` / `nodistribute_create_ops`**

  `distribute_create_ops` causes DVS to change its hashing algorithm so that create and lookup requests are distributed across all the servers, as opposed to being distributed to a single server. This applies to `create()`s, `mkdir()`s, `lookup()`s, `mknod()`s, `link()`s, and `symlink()`s.

  `nodistribute_create_ops` causes DVS to use its normal algorithm of using just one target server.

  - Default value: `nodistribute_create_ops`
  - Associated environment variable: none
  - Related settings/options: none

- **`failover` / `nofailover`**

  `failover` enables failover and failback of DVS servers. If all servers fail, operations for the mount point behave as described by the `retry` option until at least one server is rebooted and has loaded DVS. If multiple DVS servers are listed for a client mount and one or more of the servers fails, operations for that mount continue by using the subset of servers still available. When the downed servers are rebooted and start DVS, any client mounts that had performed failover operations failback to once again include the servers as valid nodes for I/O forwarding operations.

`nofailover` disables failover and failback of DVS servers. If one or more servers for a given client mount fail, operations for that mount behave as described by the `retry` or `noretry` option specified for the client mount.

- Default value: `failover`
- Associated environment variable: none
- Related settings/options: When the `failover` option is enabled (occurs automatically when `loadbalance` is enabled), the `noretry` option cannot be enabled.

- **`hash`**

  Except in cases of advanced administrators or specific advice from DVS developers, do not use the `hash` mount option. The best course of action is to let DVS use its default value. The `hash` option has three possible values:

  - **`fnv-1a`**

    `hash=fnv-1a` offers the best overall performance with little variation due to differing numbers of servers.

  - **`jenkins`**

    `hash=jenkins` is the hash that DVS previously used. It is included in the unlikely case of edge-case pathological issues with the `fnv-1a` hash, but it has worse overall performance.

  - **`modulo`**

    `hash=modulo` does not do any hash at all, but rather takes the `modulo` of the seed that it is given. This option can potentially have high load-balancing characteristics, but is vulnerable to pathological cases such as file systems that only allocate even-numbered inodes or a prime number of servers.

  - Default value: `fnv-1a`

  - Associated environment variable: none

  - Related settings/options: none

- **`hash_on_nid`/`nohash_on_nid`**

  When the `hash_on_nid` mount option is used, DVS uses the nid of the client as the hash seed instead of using the file inode number. This effectively causes all request traffic for the compute node to go to a single server. This can help metadata operation performance by avoiding lock thrashing in the underlying file system when each process on a set of DVS clients is using a separate file. The `nohash_on_nid` option disables this behavior, causing DVS to use the file inode number as the hash seed.

  - Default value: `nohash_on_nid`
  - Associated environment variable: none
  - Related settings/options: When `hash_on_nid` is enabled, DVS automatically sets the `hash` option to `modulo`. When `loadbalance` is enabled, DVS automatically sets `hash_on_nid=1`.

- **`killprocess` / `nokillprocess`**

  `killprocess` enables killing processes that have one or more file descriptors with data that has not yet been written to the backing store. DVS provides this option to minimize the risk of silent data loss, such as when data still resides in the kernel or file system page cache on the DVS server after a write has completed.

  `nokillprocess` disables the killing of processes that have written data to a DVS server when a server fails. When a server fails, processes that have written data to the server are not killed. If a process continues to perform operations with an open file descriptor that had been used to write data to the server, the operations fail (with `errno` set to `EHOSTDOWN`). A new open of the file is allowed, and subsequent operations with the corresponding file descriptor function normally.

  - Default value: `killprocess`
  - Associated environment variable: `DVS_KILLPROCESS`
  - Related settings/options: With the periodic sync feature (enabled by default), DVS servers attempt to fsync dirty files to minimize the number of processes that are killed and will also fsync a dirty file's data when the file is closed. If periodic sync is disabled (not recommended), the `killprocess` option alone cannot fully guarantee prevention of silent data loss (though it is highly unlikely) because a close() does not guarantee that data has been transferred to the underlying media (see the `closesync` option).

- **`magic`**

`magic` defines what the expected file system magic value for the projected file system on the DVS servers should be. When a DVS client attempts to mount the file system from a server, it verifies that the underlying file system has a magic value that matches the specified value. If not, the DVS client excludes that DVS server from the list of servers it uses for the mount point and prints a message to the system console. Once the configuration issue on the DVS server has been addressed and the client mounts the correct file system, DVS can be restarted on the server. All clients subsequently verify that the server is configured correctly and include the server for that mount point. Many file system magic values are defined in the `/usr/include/linux/magic.h` file. Commonly used magic values are:

- `0x6969` for NFS
- `0x47504653` for IBM Spectrum Scale (GPFS)
- `0x9123683E` for BTRFS
- `0x01021994` for TMPFS

The default value is the underlying file system's magic value. There are no associated environment variables or related settings or options for `magic`.

- **`maxnodes`**

  `maxnodes` is used in configuring DVS modes.

  - Default value: number of nodes specified in node name list or contained in nodefile.
  - Associated environment variable: `DVS_MAXNODES`
  - Related settings/options: When `loadbalance` is enabled, DVS automatically sets `maxnodes=1`.

- **nodefile**

  nodefile is the file name of a file with a list of server nodes specified as xnames separated by a colon (:) and no spaces.

- **nodename**

  nodename is a list of server nodes specified as xnames separated by a colon (:) and no spaces.

- **`retry` / `noretry`**

  `retry` enables the retry option, which affects how a DVS client node behaves in the event of a DVS server node going down. If `retry` is specified, any user I/O request is retried until it succeeds, receives an error other than a "node down" indication, or receives a signal to interrupt the I/O operation.

  `noretry` disables retries of user I/O requests when the DVS server receiving the request is down.

  - Default value: `retry`
  - Associated environment variable: none
  - Related settings/options: When the `failover` option is enabled, the `noretry` option cannot be enabled.

- **`ro_cache` / `no_ro_cache`**

  `ro_cache` enables read-only caching for files on writable client mounts. Files opened with read-only permissions in `ro_cache` mode are treated as if they were on a DVS read-only cached client mount. If the file has any concurrent open that has write permissions, all instances of that file revert to the default `no_ro_cache` mode for the current and subsequent reads. For more information, see When to Use Temporary, Client-Side, Per-File Read Caching.

  `no_ro_cache` disables read-only caching for files on writable client mounts.

  - Default value: `no_ro_cache`
  - Associated environment variable: none
  - Related settings/options: none

- **`userenv` / `nouserenv`**

  `userenv` specifies that DVS must honor end-user environment variable overrides for DVS mount options.

  `nouserenv` allows the administrator to block end-user environment variable overrides for DVS mount options.

  - Default value: `userenv`
  - Associated environment variable: none
  - Related settings/options: none

- `clusterfs/noclusterfs`

  `clusterfs` specifies that the multiple DVS servers (if more than one are used) projecting the file system to be mounted are backed by a coherent cluster file system. As a result, DVS performs consistency checks on that file system, including verifying that the inode numbers are the same across all duplicated file systems. In previous versions of DVS, this option was enabled by default, but not available for users to enable or disable. Previous versions of DVS assumed that when multiple servers are used, they all mounted the same underlying file system, and thus the inode information for a given directory is the same, regardless of the specific server node used by a client.

  Starting with HPE Cray EX release 1.2, this assumption is no longer true. When DVS is run in conjunction with local file systems and multiple servers are used, each of those servers is projecting multiple copies of the same content from separate DVS Server local file systems. This results in inode number mismatches between the servers for the same content, causing DVS to return errors when `clusterfs` is enabled (which it is by default).

  `noclusterfs` was introduced in release 1.2 to relax the consistency checks that DVS performs on file system metadata, thus allowing multiple DVS servers to be used with local file system copies of the same content. As a result, load balancing and failover configurations are possible with DVS. When the `noclusterfs` mount option is specified, DVS enforces read-only access.

  **Note:** Like `clusterfs`, the `noclusterfs` mount option assumes that the multiple DVS servers are projecting a coherent, parallel file system.

  - Default value: `clusterfs`
  - Associated environment variable: none
  - Related settings/options: none

#### 2.10.9 Periodic Sync Promotes Data and Application Resiliency

DVS periodic sync improves data and application resiliency and is more efficient than `closesync`.

DVS periodic sync improves data resiliency and application resiliency so that applications may continue executing in the event of a stalled file system or DVS server failure. Without periodic sync, such an event would result in DVS clients killing any processes with open files that were written through the failed server. Any data written through that server that was only in the server's page cache and not written to disk would be lost, and processes using the file would see data corruption.

Periodic sync works by periodically performing an `fsync` on individual files with written data on the DVS servers, to ensure that those files are written to disk. For each file, the DVS client tracks when a DVS server performs a file sync and when processes on DVS clients write to it, and then notifies the DVS server when `fsync` is needed. Periodic sync functions similarly to the DVS `closesync` mount option, but it is more efficient because it is aware of which files may have written data. Unlike `closesync` and `datasync`, periodic sync can sync data over time asynchronously so that the client does not need to wait for the sync operation to complete.

DVS periodic sync is effectively enabled by default because the `sync_period_secs` parameter, which affects the amount of time between syncs on the server, is non-zero by default. The only way to effectively disable periodic sync is to set that parameter to 0. HPE recommends keeping periodic sync enabled instead of using the `closesync` mount option.

## 2.11 How to Populate Node Maps

Generate a dvs_node_map file to be included in the COS image on HPE HPCM managed systems.

#### 2.11.1 Overview

The Data Virtualization Service (DVS) is a distributed network service that projects file systems mounted on non-compute nodes (NCN) to other nodes within the HPE Cray EX system. Projecting is simply the process of making a file system available on nodes where it does not physically reside. DVS-specific configuration settings enable clients to access a file system projected by DVS servers. These clients include compute nodes, login nodes, and gateway nodes. Thus DVS, while not a file system, represents a software layer that provides scalable transport for file system services.

DVS, in turn, uses Lustre™ Networking (LNet) to communicate over the network.

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.

The DVS node map consists of a line for each DVS capable node in a system with each line consisting of the Node ID (NID), the node name, the LNet address.

A JSON configuration file is used by the dvs_generate_map script. It specifies among other things the LNet network name, which is used to determine how to generate the LNet addresses in the node map.

### 2.11.2 Environment

File systems are projected from DVS servers to DVS clients. DVS clients typically run on a system's compute nodes. Each DVS client manages one or more DVS-projected file systems, each of which is mounted on a unique mount point.

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. This means that any system that boots the COS image could serve as either a DVS server or a DVS client. A node can function as a DVS server for a DVS client's mount path. That same node can also simultaneously function as a DVS client with a mount path that has one or more DVS servers. DVS servers are typically allocated to dedicated nodes for performance and for bridging networks for access to high performance file systems (i.e., a gateway node).

A dvs_node_map file will need to be generated that includes all the COS image nodes that utilize DVS projected file systems. In HPCM, we will be pre-building the node maps. We add the node map into the COS node images and boot them. Since we are not generating the node map at boot time, there will be no DNS lookups to slow boot time and hence there's no reason to have smaller maps for the compute nodes in this case. Since there's no downside to having the full map everywhere, all the nodes have the same node map.

The dvs_generate_map script should populate the /etc/dvs_node_map with the topology of the cluster. The file is composed of a bunch of lines, one per node, which contain the Node ID (NID) of the node, the name of the node, and the LNet address of the node. The only restriction on the NID values is that they must be unique across the system.

For example, a map might look like:

```
1 x0c0s01b0n0 10.2.0.19@tcp
2 x0c0s02b0n0 10.2.0.18@tcp
3 x0c0s04b0n0 10.2.0.17@tcp
4 x0c0s06b0n0 10.2.0.16@tcp
5 login1 10.2.0.15@tcp
6 dvs01 10.2.0.14@tcp
7 dvs02 10.2.0.12@tcp
```

Or for a "kfi" lnet type, a map might look like:

```
1 login1 13@kfi
2 login2 12@kfi
3 postoak0033 2162@kfi
4 postoak0035 2290@kfi
5 postoak0037 2146@kfi
6 postoak0039 2274@kfi
7 postoak0041 2160@kfi
8 postoak0043 2288@kfi
9 postoak0049 2129@kfi
10 postoak0051 2257@kfi
```

The dvs_node_map.conf file is used to specify the LNet network type and which of the NMN or HSN networks to use. The file uses JSON formatting. Possible `lnet' values are tcp and kfi. The possible "xname" values are "" for using the NMN network and "h0" for using the HSN network. Under CSM the xname parameter is typically "h0". This parameter is appended to the xname in the dvs_generate_map script and a DNS lookup is performed with the concatenated name. But in the HPCM environment, the DNS is typically not populated with the xname concatenated with h0. Instead in the HPCM environment we just use the hostname for DNS look up of the NMN network IPs and thecm node show -N hsn0`` command to list the hsn0 IP in HPCM. The HSN IPs are added to the HPCM configuration data base when the node is added to the system. The HPCM version of the dvs_generate_map script uses the "xname" parameter as a "switch" between specifying either NMN or hsn0 IPs in the node map.

```
dvs01# cat /etc/dvs_node_map.conf
{
   "config" : [
      {
         "lnet" : "tcp",
         "nid" : 0,
         "xname" : "h0"
      }
   ]
}
```

### 2.11.3 Pre-Conditions

#### 2.11.3.1 Booted COS image with the Slingshot HSN configured and DVS RPMs installed.

The dvs_generate_map script will need to be run on this node.

#### 2.11.3.2 All nodes to utilize DVS are booted with 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. Therefor these nodes must be booted with a COS image. That image need not have DVS RPMs installed, but must have the Slingshot HSN configured.

#### 2.11.3.3 need a dvs_node_map.conf file.

If a configuration file isn't specified, the dvs_generate_map script looks for one in /etc/dvs_node_map.conf. The cray-dvs-hpcm RPM provides a default version of the configuration file in the /etc directory. The Default value for the lnet parameter is "tcp". You may need to update the LNet network name to match your LNet environment. The -f option can be used to select a configuration file other than the default and the -d option is to use a dummy configuration of `'{"config":[{"lnet":"tcp99","nid":0,"xname":""}]}'`

### 2.11.4 Operations

#### 2.11.4.1 Boot nodes with COS image at least one of those must have the DVS RPMs installed

#### 2.11.4.2 Log into a COS imaged node that has DVS RPMs installed

Verify that DVS RPMs are installed:

```
dvs01# rpm -qa | grep dvs
cray-dvs-hpcm-2.12_4.2.56-2.2_9.18__g868d8f02.x86_64
cray-dvs-common-2.12_4.2.56-2.2_9.18__g868d8f02.x86_64
cray-dvs-kmp-cray_shasta_c-2.12_4.2.56_k5.3.18_59.16_11.0.40-2.2_9.18__g868d8f02.x86_64
cray-dvs-devel-2.12_4.2.56-2.2_9.18__g868d8f02.x86_64
```

#### 2.11.4.3 Edit and verify the dvs_node_map.conf configuration file

Default is in /etc/dvs_node_map.conf or specify an alternative configuration file with -f option to the dvs_generate_map script.

```
dvs01# cat /etc/dvs_node_map.conf
{
   "config" : [
      {
         "lnet" : "tcp",
         "nid" : 0,
         "xname" : ""
      }
   ]
}
```

#### 2.11.4.4 Select location to output generated dvs_node_map

The default output file is /etc/dvs_node_map. You may specify the output DVS node map name and location with -o option to dvs_generate_map script.

#### 2.11.4.5 Invoke the dvs_generate_map script with the desired options.

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

#### 2.11.4.6 Transfer the output dvs_node_map file to the HPCM admin node so that it may be included in the COS image that includes the DVS RPMs

```
dvs01# scp /etc/dvs_node_map root@systen_name-adm:/root
```

#### 2.11.4.7 Add the dvs_node_map to the desired COS image

Login to the HPCM cluster manager admin node and list available images.

```
system_name-adm# cm image show
ap_recipe_03
DESIRED_COS_IMAGE
fmn_sles15sp3v2
fmn_sles15sp3v2_A
ldap_recipe_8.0.0_ga
login_image
```

Copy node map to desired image /etc directory and name it dvs_node_map if not already that name.

```
system_name-adm# cp MY_NODE_MAP_NAME /opt/clmgr/image/images/DESIRED_COS_IMAGE/etc/dvs_node_map
```

Systems with Leader nodes require the image to be staged to the Leader nodes before the image is booted. The staging is accomplished through the `cm activate` command. 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 DESIRED_COS_IMAGE
```

#### 2.11.4.8 Boot COS image to verify that DVS loads

Assign the DESIRED_COS_IMAGE to nodes to be booted with that image.

```
system_name-adm# cm node set --image DESIRED_COS_IMAGE -n NODE_LIST

Please choose one of the following kernels

1) 5.3.18-57-default
2) 5.3.18-59.16_11.0.40-cray_shasta_c

[2]: 2
```

Tell the nodes to boot.

```
system_name-adm# cm power reset -t node NODE_LIST
```

For large systems, please consult the ***HPE Performance Cluster Manager Administration Guide*** section “Rebooting, halting, powering on, and powering off” for alternate ways of specifying the nodes to be booted.

Verify the DVS kernel modules are loaded.

```
dvs01# lsmod | grep dvs
dvs                    405504  0
dvsipc                 172032  2 dvs
dvsipc_lnet             57344  1 dvsipc
dvsproc                143360  3 dvsipc,dvsipc_lnet,dvs
craytrace               20480  5 dvsipc,dvsproc,dvsipc_lnet,dvs
dvskatlas               20480  4 dvsipc,dvsproc,dvsipc_lnet,dvs
lnet                   634880  3 ksocklnd,dvsipc_lnet
libcfs                 512000  3 ksocklnd,lnet,dvsipc_lnet
```

The in-memory dvs_node_map may be listed with:

```
dvs01# cat /proc/fs/dvs/node_map
```

#### 2.11.5 Usage

```
system_name-adm# ./dvs_generate_map.hpcm -h
Usage: dvs_generate_map.hpcm [options]

  -D|--debug              Turn on debugging mode
  -d|--dummy-config       Use a dummy config file
  -f|--config <filename>  JSON config file
  -h|--help               Print this help message and exit
  -m|--me                 Create map with just me
  -o|--output <filename>  Path to new map (default: /etc/dvs_node_map)
  -s|--servers            Create servers-only map
  -T|--timeout <seconds>  Timeout for waiting
```

## 2.12 DVS COS 2.3 Failover Story

Manual failover/fail-back of DVS projected file systems on HPE HPCM managed systems.

### 2.12.1 Overview

DVS failover or fail-back involves notifying DVS clients of a DVS server change of status.

Manual steps involved:

- Notify DVS clients that a DVS server is no longer available (Failover).
- DVS server corrective action (reboot or DVS reload).
- Notify DVS clients that a DVS server is now available (Fail-back).

DVS clients are notified for each DVS server failure separately.

### 2.12.2 Environment

File systems are projected from DVS servers to DVS clients. DVS clients typically run on a system's compute nodes. Each DVS client manages one or more DVS-projected file systems, each of which is mounted on a unique mount point.

When a DVS client locally mounts a DVS projected file system, it specifies a list of one or more DVS servers that will handle all the metadata and data operations to/from that file system for that DVS client.

In the HPCM environment, user file systems may be DVS-projected from any node that is booting the COS image. This means that any every system that boots the COS image could serve as either a DVS server or a DVS client. A node can function as a DVS server for a DVS client's mount path. That same node can also simultaneously function as a DVS client with a mount path that has one or more DVS servers. DVS servers are typically allocated to dedicated nodes for performance and for bridging networks for access to high performance file systems (i.e. a gateway node).

### 2.12.3 Pre-Conditions

For DVS failover to be successful, a number of system pre-conditions must be met. Those pre-conditions define a system configuration & state where DVS is properly installed & configured, and where system services on which DVS failover depends have been verified to be sufficiently functional. Do NOT exercise DVS Failover when the manual system checks indicate that DVS failover system pre-conditions are not met.

#### 2.12.3.1 DVS Server Failover Pre-Conditions

##### 2.12.3.1.1 Each DVS server node must have all required DVS RPMs installed.

```
cray-dvs-devel
cray-dvs-kmp-cray_shasta_c
cray-dvs-common
cray-dvs-hpcm
```

##### 2.12.3.1.2 Must not have multiple versions of any required DVS RPMs installed.

#### 2.12.3.1.3 Must have all required DVS kernel modules loaded.

```
dvs
dvsipc
dvsipc_lnet
dvsproc
craytrace
dvskatlas
lnet
libcfs
```

#### 2.12.3.1.4 Must have enough valid DVS server nodes.

At least 'N+1' DVS server nodes must meet all DVS server pre-conditions, where 'N' is the number of DVS server nodes that will be taken down during DVS failover.

NOTE: The necessary DVS Server nodes must remain to service each remaining DVS client mounts on the compute nodes. Prior to DVS failover there must be at least 2 active DVS servers if performing a failover test and one active DVS server if the other DVS servers are no longer active (failed). If that is no longer the case, then we cannot proceed with DVS failover.

### 2.12.3.2 DVS Client Failover Pre-Conditions

#### 2.12.3.2.1 Each DVS client Compute node must have all required DVS RPMs installed.

```
cray-dvs-devel
cray-dvs-kmp-cray_shasta_c
cray-dvs-common
cray-dvs-hpcm
```

#### 2.12.3.2.2 Must not have multiple versions of any required DVS RPMs installed.

#### 2.12.3.2.3 Must have all required DVS kernel modules loaded.

```
dvs
dvsipc
dvsipc_lnet
dvsproc
craytrace
dvskatlas
lnet
libcfs
```

#### 2.12.3.2.4 All user file system DVS mounts must have enough functioning DVS servers

On each DVS client node, each user file system mount must specify an active server list that includes at least 'N+1' DVS server nodes, where 'N' is the number of DVS servers that will be taken down during DVS failover. For example, the DVS client nid000002 user file system mount shown below can successfully failover when either 1 or 2 of the 3 listed DVS servers is taken down, but not all 3:

~~~
```bash
system-adm# ssh nid000002
nid000002# mount | grep users
... /home/users type dvs (...nodefile=/sys/kernel/debug/dvs/mounts/0/nodenames,...)
nid000002# grep '^active' /sys/kernel/debug/dvs/mounts/0/mount
active_nodes x3000c0s13b0n0 x3000c0s21b0n0 x3000c0s23b0n0
nid000002#
```
~~~

NOTE: There must be at least one active DVS server for DVS clients to failover to. Also, the mount command in the example above was used to determine which DVS mount instance was being used in order to query the specified mount, since multiple DVS mounts may be present on the DVS clients. In this case, the /home/users file system corresponds to the DVS mount instance '0'.

#### 2.12.3.3 Time Service Failover Pre-Conditions

For DVS failover to work, the timestamps that are attached to all messages & events across all DVS clients & DVS servers must be reasonably-consistent with each other. This means that the system time service must not be compromised.

To verify that the time service is working sufficiently for DVS failover, run the command date +'%s' on each DVS client and DVS server. Note the minimum and the maximum times to determine the longest time difference between any 2 nodes in seconds. If the maximum time difference is >60 seconds then it is likely that DVS failover will not function properly.

### 2.12.4 How DVS Failover works.

DVS failover or fail-back is managed by the DVS client. During a failover or a fail-back event, a user space application on the DVS client interprets these events and notifies the dvs kernel code on the DVS client by writing to the kernel space /sys/fs/dvsipc/failover file specifying the dvs server involved and followed by either "up" or "down" key word (i.e. echo x1003c1s0b0n1 down > /sys/fs/dvsipc/failover).

```
For DVS server node down:
echo <DVS_SERVER> down > /sys/fs/dvsipc/failover
For DVS server node up:
echo <DVS_SERVER> up 0 > /sys/fs/dvsipc/failover
```

We need to provide the identity in the up case. It can be 0 or it can be the actual identity. When it is the actual identity, it can be useful in some of the code path for recovering the connection between DVS server and client. The identity may be obtained from the "/sys/kernel/debug/dvsipc/node_identity" file on the affected DVS server. Providing the node identity in the up message may help avoid the unload/reload of the DVS server for some cases.

**Note:** Without the identity field provided in the up case, the process that does the echo will wedge and became a D state process.

### 2.12.5 Manually Failover or Fail-back a DVS Projected File System

For COS 2.3 under HPCM, a script was developed that will perform manual failover until automatic failover can be provided for, by writing these "up/down" messages on the DVS client. The script will be called dvs_set_node_status and will utilize a parallel shell (such as pdsh) to be performant on large systems. The script will accept command line arguments for available (-a) or unavailable (-u) and the DVS server affected (-n failed_dvs_server) to determine what to write to the /sys/fs/dvsipc/failover file. The script contains additional command line arguments for pdsh fanout threads (-f #threads), pdsh connect timeout (-t seconds), and a file listing the DVS clients to contact (-c filename). The file of client nodes will be used for testing on a subset of dvs enabled nodes and for rerunning the failover command on nodes that erred out on the first pass of using the script. If the -c option is not used the script will use the HPCM `cm node show -I` command to determine the list of servers that have a COS image running on them. The command line specified DVS server will be removed from the list of COS nodes in either case (-c or default option). The debug option (-D) works like that in other DVS scripts in that when specified it does everything but send the pdsh command but prints the command that would have been sent.

The client list file consists of COS imaged DVS client nodes, one per line.

### 2.12.6 Use Cases

- Perform manual DVS failover
- Sending failover or fail-back messages to individual DVS clients
- Perform manual DVS fail-back

#### 2.12.6.1 Perform manual DVS Failover

A DVS server has stopped projecting an external file system and a Manual failover is needed for the DVS clients to select one of the remaining active DVS servers. Notify DVS clients of a DVS server change of status.

##### 2.12.6.1.1 Prerequisites

- The administrator is logged into a DVS server.
- Alternatively, the administrator is logged into the HPCM admin node and has copied the dvs_set_node_status script to the HPCM admin node.
- Verify that a DVS server is active for the DVS clients to failover to. It will be one of the DVS server nodes specified in the mount command on the DVS client. All user file system DVS mounts must have enough functioning DVS servers

#### 2.12.6.1.2 Procedure

1. Verify that there is an active DVS server to failover to.
2. Notify DVS clients that a DVS server in no longer available.

   ```
   /opt/cray/dvs/default/sbin/dvs_set_node_status -u -n DVS_SERVER_NAME
   ```

   Record any DVS clients that reported an error so that they may be investigated, and the notification can be delivered in a second invocation of the dvs_set_node_status script.

3. Verify that the DVS client shows the specified DVS server is now in the inactive_nodes list. For large systems, a sampling of DVS clients may be more practical.

   ```
   system-adm# ssh nid000002
   nid000002# mount | grep users
   ... /home/users type dvs (...nodefile=/sys/kernel/debug/dvs/mounts/0/nodenames,...)
   nid000002# grep '^inactive_nodes' /sys/kernel/debug/dvs/mounts/0/mount
   inactive_nodes DVS_SERVER_NAME
   nid000002#
   ```

### 2.12.6.2 Sending failover or fail-back messages to individual DVS clients

If errors were recorded during the execution of either the manual failover or fail-back procedures, then send "down" (failover) or "up" (fail-back) messages to individual DVS clients that did not receive them on the first invocation of the dvs_set_node_status script.

#### 2.12.6.2.1 Prerequisites

- The administrator is logged into a DVS server.
- Alternatively, the administrator is logged into the HPCM admin node and has copied the dvs_set_node_status script to the HPCM admin node.

#### 2.12.6.2.2 Procedure

1. Useful information available at DVS client
   1. List DVS mounts on a client and show list of DVS servers supporting the mount

      The DVS servers are identified by the "nodename=" parameter in the following command output. A colon separates the DVS server names.

      ```
      nid001101# mount | grep "type dvs"
      /var/lib/cps-local/b59d7ebd319fa4df7f81b8bc2b799c17 on /var/opt/cray/gpu/nvidia-squashfs-21.9 type d
      /var/lib/cps-local/16761123d0687e12ecbf5cb106ae8cda on /var/opt/cray/analytics/Cray-Analytics.x86_64
      /var/lib/cps-local/acd1a0710ea8f50379a31ff1a98ce4cf on /var/opt/cray/pe/base-22.03 type dvs (ro,rela
      ```

   2. List all the mounts the DVS client is tracking

      ```
      nid001101# ls /sys/kernel/debug/dvs/mounts/
      0  1  2  3
      ```

   3. Obtain details of the DVS mount

      The active_nodes are the DVS servers that are available to support this mount. The inactive_nodes are DVS servers not able to support this mount, normally blank. After failover, the failed DVS server supplied from the manual failover message will be moved from the active_nodes list to the inactive_nodes list. The loadbalance_node is the DVS server supporting this mount. There is only one and it is taken from the list of DVS servers in active_nodes. This will change in a failover if the load balance node was the DVS server that was supplied to the unavailable message sent to the DVS client. It will become one of the remaining DVS servers in the active_nodes list.

      ```
      nid001101# cat /sys/kernel/debug/dvs/mounts//0/mount
      local-mount /tmp/cps
      remote-path /var/lib/cps-local/f603538ba01995bdfeb10616536b2db4
      options (ro,blksize=524288,statsfile=/sys/kernel/debug/dvs/mounts/0/stats,attrcache_timeout=14400,ca
      active_nodes x3000c0s8b0n0 x3000c0s5b0n0 x3000c0s7b0n0
      inactive_nodes
      ```

```
loadbalance_node x3000c0s5b0n0
remote-magic 0x794c7630
```

2. Determine list of DVS clients to receive the unavailable notification. Add the list of DVS clients to a file (CLIENT_LIST_FILE), one per line.

3. If this is cleanup for sending the unavailable message to the DVS client(s), then perform the following otherwise skip to the next step to send the available message.

   ```
   /opt/cray/dvs/default/sbin/dvs_set_node_status -u -n DVS_SERVER_NAME -c CLIENT_LIST_FILE
   ```

   Record any DVS clients that reported an error so that they may be investigated, and the notification can be delivered in any subsequent invocations of the dvs_set_node_status script.

Verify that the DVS client now shows the specified DVS server is now in the inactive_nodes list. For large systems, a sampling of DVS clients may be more practical.

```
system-adm# ssh nid000002
nid000002# mount | grep users
... /home/users type dvs (...nodefile=/sys/kernel/debug/dvs/mounts/0/nodenames,...)
nid000002# grep '^inactive_nodes' /sys/kernel/debug/dvs/mounts/0/mount
inactive_nodes DVS_SERVER_NAME
nid000002#
```

4. If this is cleanup for sending the available message to the DVS client(s), then perform the following.

   ```
   /opt/cray/dvs/default/sbin/dvs_set_node_status -a -n DVS_SERVER_NAME -c CLIENT_LIST_FILE
   ```

   Record any DVS clients that reported an error so that they may be investigated, and the notification can be delivered in any subsequent invocations of the dvs_set_node_status script.

Verify that the DVS client now shows the specified DVS server is now in the active_nodes list. For large systems, a sampling of DVS clients may be more practical.

```
system-adm# ssh nid000002
nid000002# mount | grep users
... /home/users type dvs (...nodefile=/sys/kernel/debug/dvs/mounts/0/nodenames,...)
nid000002# grep '^active_nodes' /sys/kernel/debug/dvs/mounts/0/mount
active_nodes DVS_SERVER_NAME OTHER_ACTIVE_DVS_SERVERS
nid000002#
```

#### 2.12.6.3 Perform manual DVS fail-back

A previously unavailable DVS server has become available and a manual fail-back is needed for the DVS clients to select their original load balance DVS server. Notifying DVS clients of a DVS server change of status.

##### 2.12.6.3.1 Prerequisites

- The administrator is logged into a DVS server.
- Alternatively, the administrator is logged into the HPCM admin node and has copied the dvs_set_node_status script to the HPCM admin node.
- Verify that the failed DVS server is now active for the DVS clients to fail-back to. This requires at a minimum of reloading DVS kernel modules on the failed DVS server.

##### 2.12.6.3.2 Procedure

1. Verify that the failed DVS server has been recovered before initiating the fail-back. If the DVS server is not active and the fail-back is initiated, then the DVS client may lose access to the DVS projected file system.

2. Notify DVS clients that a DVS server in now available.

   ```
   /opt/cray/dvs/default/sbin/dvs_set_node_status -a -n DVS_SERVER_NAME
   ```

   Record any DVS clients that reported an error so that they may be investigated, and the notification can be delivered in a second invocation of the dvs_set_node_status script.

3. Verify that the DVS client now shows the specified dvs server is now in the active_nodes list. For large systems, a sampling of DVS clients may be more practical.

   ```
   system-adm# ssh nid000002
   nid000002# mount | grep users
   ... /home/users type dvs (...nodefile=/sys/kernel/debug/dvs/mounts/0/nodenames,...)
   nid000002# grep '^active_nodes' /sys/kernel/debug/dvs/mounts/0/mount
   active_nodes DVS_SERVER_NAME OTHER_ACTIVE_DVS_SERVERS
   nid000002#
   ```

### 2.12.7 Usage

~~~
```bash
system-adm: # /opt/cray/dvs/default/sbin/dvs_set_node_status -h
Usage: dvs_set_node_status [options]

  -a            Set DVS state to available
  -c file       File to use for client list
  -D            Turn on debugging mode
  -f threads    Specify MAX number of threads for pdsh
  -h            Print this help message and exit
  -n DVS_svr    Specify DVS server nodename
  -T seconds    Timeout for waiting for the pdsh connect
  -u            Set DVS state to unavailable
```
~~~

## 2.13 Projecting Shared Libraries with DVS

This section describes procedures for using DVS to mount shared libraries to the CNs in the HPCM environment. There are two methods available. The first will modify the gateway and CN node images to make the mounts persistent across reboots. And the second is to add the mounts to the live gateway and CN nodes to test the mount and/or make the mount available immediately without waiting for a reboot of the gateway node. The persistent procedure could then be performed except for the reboot step to stage the persistent change the next time the gateway and CNs are rebooted.

### 2.13.1 Prerequisites

Satisfy these prerequisites before performing this procedure:

- The file system containing the shared libraries is to be mounted read-only.
- It is assumed that the shared libraries are made available (mounted) on the gateway node by the system administrator. The system administrator has a number of options to accomplish this but not limited to mounting the shared libraries using NFS, GPFS, or copying the shared libraries to disk on the gateway node. The system administrator can then use DVS to project the file system containing the shared libraries to the CNs and other DVS clients in the system.

### 2.13.2 Procedure

This procedure describes the steps to project shared libraries to the CNs and login nodes using DVS, in a way that is persistent across node reboots.

1. Verify that the shared libraries have been mounted to the gateway node.
2. Perform Prepare to Configure DVS with VCS for the gateway and compute node images.

   Note that these instructions require changes to the COS image.
3. Update the `dvs_exports.yml` file to allow access to the file system(s)

   Edit the working copy of the gateway image and add the shared library file system to be projected to the `dvs_exports.yml` file. Set the mode to "ro" so that the file system will be exported Read-Only. See the Allowing Client Access to DVS Server File Systems section for more information.

```
system_name-adm: / # cd /opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>/etc
system_name-adm: / # vi dvs_exports.yml
system_name-adm: / # cat dvs_exports.yml
...
dvs_exports.yml:
  exports:
  - mode: rw
    path: /home
  - mode: rw
    path: /gpfs1
  - mode: ro
    path: /SHARED_LIBRARIES
```

4. Perform Commit DVS image changes to VCS for the working copy of the the gateway image, using the HPCM VCS repository.
5. Edit the `/etc/fstab` file in the working copy for the CN image to mount the shared library file system on the CNs. Skip this step if the new file system will not be projected to the CNs.
    1. Make the `/READ_ONLY_FS_MOUNT_POINT` directory and edit the `fstab` file so that it contains a shared library file system entry similar to the example shown in the next sub-step.

       ```
       system_name-adm # mkdir /opt/clmgr/image/images/<CN_IMAGE_NAME>/READ_ONLY_FS_MOUNT_POINT
       ```

    2. Add the client mount configuration for the shared library file system to the `fstab` file.

       In the `options` field, specify the host names of the Gateway nodes (separated by colons) that will project the shared libraries file system as part of the `nodename` argument. This example mounts a remote read-only file system */SHARED_LIBRARIES* on the client at */READ_ONLY_FS_MOUNT_POINT*. Note the use of `attrcache_timeout=14400` and `loadbalance` mount options in the `options` field, since this example is a read-only file system:

       ```
       system_name-adm # cat /opt/clmgr/image/images/<CN_IMAGE_NAME>/etc/fstab
       . . .
       /SHARED_LIBRARIES /READ_ONLY_FS_MOUNT_POINT dvs attrcache_timeout=14400,loadbalance,
             path=/READ_ONLY_FS_MOUNT_POINT,nodename=gw01:gw01
       ```

6. Perform Commit DVS image changes to VCS for the working copy of the the CN image, using the HPCM VCS repository.
7. Edit the `/etc/fstab` file in the working copy of the login node image to mount the file system on login nodes. Skip this step if the shared libraries file system will not be projected to login nodes.
    1. Perform Prepare to Configure DVS with VCS for the login image.
    2. Make the `/READ_ONLY_FS_MOUNT_POINT` directory

       ```
       system_name-adm # mkdir /opt/clmgr/image/images/<LOGIN_IMAGE_NAME>/READ_ONLY_FS_MOUNT_POINT
       ```

    3. Add the client mount configuration for the shared libraries file system to the `/etc/fstab` file entry similar to the example in Step 5.

       In the `options` field, specify the host names of the Gateway nodes (separated by colons) that will project the external file system as part of the `nodename` argument. This example mounts a remote read-only file system */SHARED_LIBRARIES* ***on the client at*** `/READ_ONLY_FS_MOUNT_POINT_`. Note the use of `attrcache_timeout=14400` and `loadbalance` mount options in the `options` field, since this example is a read-only file system:

       ```
       system_name-adm # cat /opt/clmgr/image/images/<LOGIN_IMAGE_NAME>/etc/fstab
       . . .
       /SHARED_LIBRARIES /READ_ONLY_FS_MOUNT_POINT dvs attrcache_timeout=14400,loadbalance,
             path=/READ_ONLY_FS_MOUNT_POINT,nodename=gw01:gw02
       ```

    4. Perform Commit DVS image changes to VCS for the login image.

**Force the system to immediately use the new configuration**

8. If your system contains Leader nodes, push the image changes for Gateway, CNs, and login nodes to the leader nodes so they will be available for boot. You will perform the `cm activate image` for each image modified in the previous steps. 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 <IMAGE_NAME> --force
```

9. Reboot the Gateways to mount the new external file system.

   If you changed the image name from the previous boot, specify the image to boot.

   ```
   system_name-adm: / # cm node set --image <NEW_GATEWAY_IMAGE_NAME> -n <NODE_LIST>
   ```

   Reset the node to start the boot.

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

   Note: contains the list of Gateway nodes.

10. Reboot the compute nodes, login nodes, or both, depending on what types of nodes will be mounting the external file system.

    If you changed the image name from the previous boot, specify the image to boot.

    ```
    system_name-adm: / # cm node set --image <NEW_IMAGE_NAME> -n <NODE_LIST>
    ```

    Reset the node to start the boot.

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

    Note: contains the list of compute nodes, login nodes, or both.

    Once the nodes finish rebooting, the DVS-projected shared library file system mount will persist across future CN, login, and Gateway reboots.

### 2.13.3 Alternative Procedure - Add mount to live system

This is the alternate procedure that can be used to project file systems without rebooting the gateway nodes at runtime but is not persistent across gateway reboots. The system administrator may want to perform this procedure on a live gateway node to test the projected file system and to make the file system available without requiring a reboot of the gateway node. The persistent procedure could then be performed (except for the reboot step) to persist the change, the next time the gateway is rebooted. Likewise this procedure can be performed on CNs to test the mount, but a script and `pdsh` may be needed to replicate the mount across all CNs for larger systems. See the "Automate DVS Mount using `pdsh`" step later in this procedure.

1. Log into the gateway node to verify that the shared libraries have been mounted to the gateway node.

2. Update the `/etc/dvs_exports.yml` file to allow access to the file system(s)

   Edit the live file system of the gateway node and add the shared library file system to be projected to the `dvs_exports.yml` file. Set the mode to "ro" so that the file system will be exported Read-Only. See the Allowing Client Access to DVS Server File Systems section for more information.

   ```
   gw01: / # cd /etc
   gw01: / # vi dvs_exports.yml
   gw01: / # cat dvs_exports.yml
   ...
   dvs_exports.yml:
     exports:
     - mode: rw
       path: /home
     - mode: rw
       path: /gpfs1
     - mode: ro
       path: /SHARED_LIBRARIES
   ```

3. Run the `dvs_exportfs` command on each gateway node to activate the DVS export

   ```
   gw01: / # /opt/cray/dvs/default/sbin/dvs_exportfs -r
   ```

4. Login to a compute node to perform the DVS mount of the projected file system containing the shared libraries.

5. Edit the `/etc/fstab` file on the booted CN node to mount the shared library file system on the CNs. Skip this step if the new file system will not be projected to the CNs. This step is intended to test the DVS mount on a live system but may not be practical for making the mount available across a system. See the "Automate DVS Mount using pdsh" step bellow.

1. Edit the `/etc/fstab` file so that it contains a shared library file system entries similar to the example in next sub-step.
2. Add the client mount configuration for the shared library file system to the `fstab` file.

   In the `options` field, specify the host names of the Gateway nodes (separated by colons) that will project the shared libraries file system as part of the `nodename` argument. This example mounts a remote read-only file system */SHARED_LIBRARIES* on the client at */READ_ONLY_FS_MOUNT_POINT*. Note the `attrcache_timeout=14400` and `loadbalance` mount options in the `options` field, since this example is a read-only file system:

   ```
   x1000c0s0b0n0# mkdir /READ_ONLY_FS_MOUNT_POINT
   x1000c0s0b0n0# cat /opt/clmgr/image/images/<CN_IMAGE_NAME>/etc/fstab
   . . .
   /SHARED_LIBRARIES /READ_ONLY_FS_MOUNT_POINT dvs attrcache_timeout=14400,loadbalance,
         path=/READ_ONLY_FS_MOUNT_POINT,nodename=gw01:gw02
   ```

3. Use the `mount` command to perform the DVS mount.

   ```
   x1000c0s0b0n0# mount /SHARED_LIBRARIES
   ```

6. Login to a login node to perform the DVS mount of the projected file system containing the shared libraries.
7. Edit the `/etc/fstab` file to mount the file system on the login nodes in the same manner that was done for the CN step. Skip this step if the shared libraries file system will not be projected to the login nodes. This step is intended to test the DVS mount on a live system but may not be practical for making the mount available across a system. See the “Automate DVS Mount using pdsh” step bellow.
8. Automate DVS Mount using `pdsh`.

   After testing the DVS mount of the shared libraries file system in the previous steps, you can use the `pdsh` utility to perform the mount across the system's CNs and login nodes. The preferred method is to issue a mount command using `pdsh` and specifying the DVS mount parameters.

   ```
   system_name-adm# pdsh -w <NODE_LIST> mkdir /READ_ONLY_FS_MOUNT_POINT
   system_name-adm# pdsh -w <NODE_LIST> mount -t dvs -o path=/SHARED_LIBRARIES,nodename=gw01:gw02,
       attrcache_timeout=14400,loadbalance /SHARED-LIBRARIES /READ_ONLY_FS_MOUNT_POINT
   ```

   NOTE: is a comma separated list of host names.

# 3 Install Spectrum Scale

## 3.1 Install Spectrum Scale on a Gateway Node

Keep in mind the following three things before performing Create Spectrum Scale Gateway Node Image:

1. **There are many 3rd-party dependencies that must be satisfied to successfully build and install the Spectrum Scale client software**. The guidance below suggests a procedure that is flexible enough for you to obtain and install all required dependencies. However, your available specific 3rd-party dependencies, and the sources for those 3rd-party dependencies, may vary from what is described. Please refer to documentation at www.ibm.com as needed.
2. **Per IBM documentation, Spectrum Scale servers and Spectrum Scale clients need to maintain password-less SSH access to each other across reboots.** For nodes that do not have local storage, the contents of a Gateway authorized keys file is not automatically preserved across a reboot. This means that, to re-establish the authorized keys for the external Spectrum Scale servers, the guidance below hard-codes those key values into the Gateway customized boot process. Therefore, if the SSH keys of the external Spectrum Scale servers change, the customized boot process must be performed to re-establish appropriate authorized keys of those Spectrum Scale servers on the Gateway-GW nodes that have Spectrum Scale clients. Customers are free to explore alternative approaches to address this issue.
3. **Nodes without persistent storage will not automatically rejoin the Spectrum Scale cluster after a reboot.** Nodes that do not have local or network-mounted storage, for storing dynamically updated Spectrum Scale data across reboots will lose cluster information after a reboot. This prevents that such nodes from automatically re-joining the Spectrum Scale cluster. The guidance below describes updating the customized boot process to establish and auto-run an ansible script to run the Spectrum Scale `mmstartup` and `mmsdrrestore` commands to recover and re-establish the necessary Spectrum Scale cluster membership on reboot. Customers are free to explore alternative approaches to address this issue.

### 3.1.1 Create Spectrum Scale Gateway Node Image

**PREREQUISITES**

Identify a Gateway node that will support a DVS server and Spectrum Scale client, that can be reserved for the entire duration of this procedure.

**OBJECTIVE**

This procedure guides HPE Cray EX administrators through the phases of creating a Spectrum Scale-enabled Gateway image:

1. Installing prerequisite software
2. Copying the Spectrum Scale software to a Gateway node.
3. Extracting and installing the Spectrum Scale software on the Gateway node.
4. Building the Spectrum Scale client kernel module (portability layer) RPM
5. Installing the portability layer and IBM-provided RPMs into a base Gateway image to create a Spectrum Scale-enabled Gateway image.

**LIMITATIONS**

This procedure provides HPE Cray EX-specific instructions and guidance for manually installing Spectrum Scale on HPE Cray OS. These steps are not intended to replace the instructions and guidance provided on the IBM website (www.ibm.com).

**PROCEDURE**

The command examples in this procedure use `gw01` as the hostname of the Gateway node.

**Connect HPE Cray EX system to Spectrum Scale cluster**

1. Connect the Spectrum Scale servers to the Gateway, following local administrative procedures.
2. Verify that password-less SSH can be successfully configured between Spectrum Scale servers and Gateway nodes.

**Backup the image used to boot the Gateway node**

3. Log into `System_Name-adm` of the HPE Cray EX system as root through SSH.
4. Use the following command to save a copy of the Gateway image you want to preserve.

   ```
   System_Name-adm# cm image revision commit -i <GATEWAY_IMAGE_NAME> \
   -m "Gateway image backup before Spectrum Scale install"
   ```

5. Verify the revision history.

   ```
   System_Name-adm# cm image revision history -i <GATEWAY_IMAGE_NAME>
   ```

**Download the Spectrum Scale software to a Gateway node**

6. From `System_Name-adm` log into the Gateway node, for example `gw01`, of the HPE Cray EX system as root through SSH.
7. Create and navigate into a directory on that Gateway node to contain the Spectrum Scale installation software that will be downloaded in the next step.

   ```
   gw01# mkdir -p /tmp/GPFS5
   gw01# cd /tmp/GPFS5
   ```

8. Download and copy the install file of the Spectrum Scale Data Management release version 5.1.3.0 installation software for Linux from the IBM website to the `/tmp/GPFS5` directory.

   Only the 5.1.3.0 release version has been qualified for this HPE Cray OS release.

**Extract and install the Spectrum Scale software**

9. Run the self-extracting Spectrum Scale software archive.

   ```
   gw01# chmod +x Spectrum_Scale_Data_Management-*-x86_64-Linux-install
   gw01# ./Spectrum_Scale_Data_Management-*-x86_64-Linux-install
   ```

   Within 2 minutes, the self-extracting archive will print out several lines of output during the installation process. Then it will ask for acceptance of the Spectrum Scale license.

10. Enter 1 to accept the license agreement when prompted.

```
. . .
LICENSE INFORMATION

The Programs listed below are licensed under the following
License Information terms and conditions in addition to the
Program license terms previously agreed to by Client and
IBM. If Client does not have previously agreed to license
terms in effect for the Program, the International Program
License Agreement (Z125-3301-14) applies.

Program Name (Program Number):
IBM Spectrum Scale Data Management Edition 5.1.3.0 (5737-
F34)
IBM Spectrum Scale Data Management Edition 5.1.3.0 (5641-
DM1)

Press Enter to continue viewing the license agreement, or
enter "1" to accept the agreement, "2" to decline it, "3"
to print it, "4" to read non-IBM terms, or "99" to go back
to the previous screen.
'1'

License Agreement Terms accepted.
...
```

The installer will then finish and display the message `Product packages successfully extracted to /usr/lpp/mmfs/5.1.3.0`.

11. Navigate to the directory containing the Spectrum Scale packages and install them.

```
gw01# cd /usr/lpp/mmfs/5.1.3.0/gpfs_rpms
gw01# rpm -ivh gpfs.base*.rpm gpfs.gpl*rpm gpfs.license*rpm gpfs.gskit*rpm \
gpfs.msg*rpm gpfs.compression*rpm gpfs.adv*rpm gpfs.crypto*rpm gpfs.docs*.rpm gpfs.afm*.rpm
warning: gpfs.base-5.1.3-0.x86_64.rpm: Header V3 RSA/SHA256 Signature, key ID efeb6eeb: NOKEY
Preparing...                          ################################# [100%]
Updating / installing...
1:gpfs.base-5.1.3-0                ################################# [ 10%]
Created symlink /etc/systemd/system/multi-user.target.wants/mmautoload.service →
/usr/lib/systemd/system/mmautoload.service.
Created symlink /etc/systemd/system/multi-user.target.wants/mmccrmonitor.service →
/usr/lib/systemd/system/mmccrmonitor.service.
 2:gpfs.adv-5.1.3-0                ################################# [ 20%]
 3:gpfs.license.dm-5.1.3-0         ################################# [ 30%]
 4:gpfs.gpl-5.1.3-0                ################################# [ 40%]
 5:gpfs.compression-5.1.3-0        ################################# [ 50%]
 6:gpfs.crypto-5.1.3-0             ################################# [ 60%]
 7:gpfs.afm.cos-1.0.0-5            ################################# [ 70%]
 8:gpfs.docs-5.1.3-0               ################################# [ 80%]
 9:gpfs.msg.en_US-5.1.3-0          ################################# [ 90%]
10:gpfs.gskit-8.0.55-19            ################################# [100%]
```

**Build the portability layer RPM**

12. Add the directory containing the Spectrum Scale programs to the PATH on the Gateway node.

```
gw01# export PATH=$PATH:/usr/lpp/mmfs/bin
```

If this directory is not manually added, one or more Spectrum Scale scripts will fail.

13. Install the `rpm-build` package and its dependencies. This step is performed from the HPCM system admin node since that is where the package repos are maintained.

This utility must be installed to package the Spectrum Scale kernel module (portability layer) into an RPM.

```
System_Name-adm# cm node zypper -n gw01 --repo-group <GATEWAY_IMAGE_REPO_GROUP> install rpm-build
.  .  .
gw01: Loading repository data...
gw01: Reading installed packages...
gw01: Resolving package dependencies...
gw01:
gw01: The following NEW package is going to be installed:
gw01:   rpm-build
gw01:
gw01: The following package has no support information from its vendor:
gw01:   rpm-build
gw01:
gw01: 1 new package to install.
gw01: Overall download size: 35.0 KiB. Already cached: 0 B. After the operation, additional 29.1 KiB will
gw01: Continue? [y/n/v/...? shows all options] (y): y
gw01: Retrieving package rpm-build-4.14.3-150300.46.1.x86_64 (1/1),  35.0 KiB ( 29.1 KiB unpacked)
gw01: Retrieving: rpm-build-4.14.3-150300.46.1.x86_64.rpm [done]
gw01:
gw01: Checking for file conflicts: [...done]
gw01: (1/1) Installing: rpm-build-4.14.3-150300.46.1.x86_64 [.
gw01: warning: /var/cache/zypp/packages/other_HPE-SLE-Updates-Module-Development-Tools-15-SP3-x86_64-22.0
gw01: ...done]
```

14. On the Gateway node, gw01, apply a patch to /usr/lpp/mmfs/src/gpl-linux/verdep.h so that mmbuildgpl will complete successfully.

Create a patch file with the following contents:

```
--- src/gpl-linux/verdep.h.orig 2022-06-28 18:29:08.455387434 +0000
+++ src/gpl-linux/verdep.h  2022-06-28 18:35:10.363506450 +0000
@@ -61,6 +61,9 @@
 # include <Logger-gpl.h>
 # include <linux/moduleparam.h>

+#define HAVE_COMPAT_IOCTL 1
+#define HAVE_UNLOCKED_IOCTL 1
+
 /* Needed for DENTRY_OPEN predefined in verdep.h */
 /* path.h is newly added by 3.4 Linux kernel */
#if (LINUX_KERNEL_VERSION >= 30400000)
@@ -1864,7 +1867,8 @@
 #endif

 /* commit 021182e5 in kernel v4.8 enabled KASLR */
-#if defined(GPFS_ARCH_X86_64) && defined(CONFIG_RANDOMIZE_MEMORY)
+/* commit eedb92ab in kernel v4.17 changed __PAGE_OFFSET to support 5-level */
+#if defined(GPFS_ARCH_X86_64) && (defined(CONFIG_RANDOMIZE_MEMORY) || defined(CONFIG_DYNAMIC_MEMORY_LAYO
 #define KC_X86_64_RANDOM_PAGE_OFFSET_BASE
 #endif
```

Apply the patch to the file /usr/lpp/mmfs/src/gpl-linux/verdep.h:

```
gw01# patch -u /usr/lpp/mmfs/src/gpl-linux/verdep.h -i <patch_filename>
```

15. Build the Spectrum Scale kernel module RPM on the Gateway node.

This module serves as the portability layer between the Linux kernel and the Spectrum Scale daemon from IBM.

```
gw01# /usr/lpp/mmfs/bin/mmbuildgpl --build-package
--------------------------------------------------------
mmbuildgpl: Building GPL (5.1.0.3) module begins at Tue 28 Jun 2022 06:35:20 PM UTC.
--------------------------------------------------------
Verifying Kernel Header...
```

```
    kernel version = 50318999 (503189999059068, 5.3.18-150300.59.68_11.0.76-cray_shasta_c, 5.3.18-150300.59
    module include dir = /lib/modules/5.3.18-150300.59.68_11.0.76-cray_shasta_c/build/include
    module build dir   = /lib/modules/5.3.18-150300.59.68_11.0.76-cray_shasta_c/build
    kernel source dir  = /usr/src/linux-5.3.18-150300.59.68/include
    Found valid kernel header file under /lib/modules/5.3.18-150300.59.68_11.0.76-cray_shasta_c/build/inclu
  Getting Kernel Cipher mode...
    Will use blkcipher routines
  Verifying Compiler...
    make is present at /usr/bin/make
    cpp is present at /usr/bin/cpp
    gcc is present at /usr/bin/gcc
    g++ is present at /usr/bin/g++
    ld is present at /usr/bin/ld
  Verifying rpmbuild...
  Verifying Additional System Headers...
    Verifying linux-glibc-devel is installed ...
      Command: /bin/rpm -q linux-glibc-devel
      The required package linux-glibc-devel is installed
  make World ...
  make InstallImages ...
  make rpm ...
  Wrote: /usr/src/packages/RPMS/x86_64/gpfs.gplbin-5.3.18-150300.59.68_11.0.76-cray_shasta_c-5.1.0-3.x86_64
  --------------------------------------------------------
  mmbuildgpl: Building GPL module completed successfully at Tue 28 Jun 2022 06:35:31 PM UTC.
  --------------------------------------------------------
```

16. Confirm that the portability layer RPM was built.

```
gw01# cd /usr/src/packages/RPMS/x86_64
gw01# ls -l
total 1068
-rw-r--r-- 1 root root 1093196 Jun 28 18:35
 gpfs.gplbin-5.3.18-150300.59.68_11.0.76-cray_shasta_c-5.1.0-3.x86_64.rpm
```

**Prepare to customize a base Gateway image**

17. Copy the required Spectrum Scale software RPMs and portability layer RPM to a working directory.

```
gw01# cd /usr/lpp/mmfs/5.1.3.0/gpfs_rpms
gw01# cp gpfs.base*.rpm gpfs.gpl*rpm gpfs.license*rpm gpfs.gskit*rpm gpfs.msg*rpm \
gpfs.compression*rpm gpfs.adv*rpm gpfs.crypto*rpm gpfs.afm*rpm gpfs.docs*rpm \
/usr/src/packages/RPMS/x86_64
```

18. Login to `System_Name-adm` of the HPE Cray EX system as `root` through SSH.

19. Make a directory to hold the Spectrum Scale RPMs, and copy the RPMs from the Gateway node into that directory.

```
System_Name-adm# mkdir /tmp/GPFS5
System_Name-adm# scp root@gw01:/usr/src/packages/RPMS/x86_64/gpfs.*.rpm /tmp/GPFS5
```

20. Follow IBM instructions to add your Spectrum Scale gateway node as a client in your Spectrum Scale cluster, and verify proper functionality. The purpose of this step is to confirm that password-less ssh is working between the Gateway node and the Spectrum Scale servers. The process requires steps similar to the following:

    1. Login to a Spectrum Scale server node to obtain the Admin node name and Daemon node name for all Spectrum Scale server nodes

```
hi1# mmlscluster

GPFS cluster information
========================
  GPFS cluster name:         hi1
  GPFS cluster id:           17825395649093381303
  GPFS UID domain:           hi1
  Remote shell command:      /usr/bin/ssh
  Remote file copy command:  /usr/bin/scp
  Repository type:           CCR

 Node  Daemon node name  IP address    Admin node name  Designation
-------------------------------------------------------------------
   1   venom1-gpfs       10.253.100.1  hi1              quorum-manager
   2   venom2-gpfs       10.253.100.2  hi2              quorum-manager

hi1#
```

2. Add the following line to the `/etc/hosts` file of the Gateway node, making one entry for each Spectrum Scale server node:

   ```
   HSN_IP_ADDRESS  ADMIN_NODE_NAME DAEMON_NODE_NAME
   ```

   Where:

   - *HSN_IP_ADDRESS* is the IP address of the node on the HPE Cray EX HSN.
   - *ADMIN_NODE_NAME* is the Spectrum Scale `Admin node name` of the node.
   - *DAEMON_NODE_NAME* is the Spectrum Scale `Daemon node name` of the node.

3. On the Gateway node, the RSA SSH public key resides in `/root/.ssh/id_rsa.pub`

   ```
   gw01# ls /root/.ssh/id_rsa.pub
   /root/.ssh/id_rsa.pub
   ```

4. Copy the Gateway root user RSA SSH public key to every Spectrum Scale server node (*DAEMON_NODE_IP*) in the cluster by running the following command for each Spectrum Scale server node:

   ```
   gw01# ssh-copy-id GPFS_USER_NAME@DAEMON_NODE_IP
   ```

5. On each Spectrum Scale server node, add to the `/etc/hosts` file a line for each Gateway node:

   ```
   HSN_IP_ADDRESS  GW_HOSTNAME
   ```

   Where:

   - *HSN_IP_ADDRESS* is the IP address of the Gateway node on the HPE Cray EX HSN.
   - *GW_HOSTNAME* is the hostname of the Gateway node (`gw01`, for example).

6. Copy the Spectrum Scale server node root user RSA SSH public key to every Gateway node (*GW_HOSTNAME*) in the HPE Cray EX system by running the following command on each Spectrum Scale server node, repeating for each Gateway node:

   ```
   hi1# ssh-copy-id GW_HOSTNAME
   ```

7. Copy the root user RSA SSH public key for every Spectrum Scale server node to the HPCM admin node (*System_Name-adm*) /tmp/GPFS5 directory in the HPE Cray EX system. When copying the RSA SSH public key file to the HPCM admin node, be sure to append the ADMIN_NODE_NAME to the file to preserve each of the key files. Run the following command from each of the Spectrum Scale server nodes:

   ```
   hi1# scp ~/.ssh/id_rsa.pub root@System_Name-adm:/tmp/GPFS5/id_rsa.pub.hi1
   ```

21. Change directories to the working copy of .

```
System_Name-adm# cd /opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>
```

**Customize base Gateway image with Spectrum Scale software**

22. Copy the Spectrum Scale install files into the Gateway image working directory.

    ```
    System_Name-adm# cp /tmp/GPFS5/*.rpm /opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>/tmp
    ```

23. Create a HPCM post-install script to append the Spectrum Scale server RSA SSH public key file to the known_hosts file on the GW_HOSTNAME when the node boots. The post-install scripts reside in `/opt/clmgr/image/scripts/post-install` on the HPCM system admin node. The script will need to append the contents of each id_rsa.pub.ADMIN_NODE_NAME files to the ~/.ssh/authorized_keys file. Since the post-install scripts are copied to the /tmp/post-instal/ on the client's new file system you will need to embed the contents of each id_rsa.pub file in a `echo` statement within the script. See the /opt/clmgr/image/scripts/post-install/README quick start guide. For example:

```
System_Name-adm# vi /opt/clmgr/image/scripts/post-install/99all.add_spectrum_scale_pub_keys
System_Name-adm# cat /opt/clmgr/image/scripts/post-install/99all.add_spectrum_scale_pub_keys
# Spectrum Scale id_rsa.pub for hi1
cat << EOF >> /root/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC23nyxWQzSIOQSLaBxVOQrnFga7bqZKdqMK
l6FfoOusN7ht2Wua24H8rlGieTqv6OYBoaAkhgo0/xdFchUuhgzvZLHBCibrooCuLLkzDuH2o
ld1EZU4HsDY4lLLpw1FhOLXpP1lk9VVgquO5Dt69X/sOWTE74HelLLSylLEly3okFwyVZEeAv
2dv2AlmBOwCRqBMDYLztTdxETl6GqLtGYaaxBxRIRpmTmat+X5gMWE7ruryysh+DQzsOSiWBS
fxTK9/s960qnapgW2Vx8RfVLcFg8lI4wZ/Ot5v6IW0VwgerI77XfLySI6msqybjsEN8jQD70I
hphtT0nZLn6c+Fd3DrntQXfuiZgcUyzSZ62H/Q1tWN5DE0o5BunxarkAvN88/8NKVrwV6EHLA
pBPiA37cy69ww7tMz/r4/+1GUb29G5Vo2MzIJ7TkE1QHhh2ZD6QW2bQHbz+wa6NwbS+HND9jH
g9xBx0RFo7y8outfRRDciVCTicHrW8w0+gejqMSM=
EOF

# Spectrum Scale id_rsa.pub for hi2
cat << EOF >> /root/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC23nyxWQzSIOQSLaBxVOQrnFga7bqZKdqMK
l6FfoOusN7ht2Wua24H8rlGieTqv6OYBoaAkhgo0/xdFchUuhgzvZLHBCibrooCuLLkzDuH2o
ld1EZU4HsDY4lLLpw1FhOLXpP1lk9VVgquO5Dt69X/sOWTE74HelLLSylLEly3okFwyVZEeAv
2dv2AlmBOwCRqBMDYLztTdxETl6GqLtGYaaxBxRIRpmTmat+X5gMWE7ruryysh+DQzsOSiWBS
fxTK9/s960qnapgW2Vx8RfVLcFg8lI4wZ/Ot5v6IW0VwgerI77XfLySI6msqybjsEN8jQD70I
hphtT0nZLn6c+Fd3DrntQXfuiZgcUyzSZ62H/Q1tWN5DE0o5BunxarkAvN88/8NKVrwV6EHLA
pBPiA37cy69ww7tMz/r4/+1GUb29G5Vo2MzIJ7TkE1QHhh2ZD6QW2bQHbz+wa6NwbS+HND9jH
g9xBx0RFo7y8outfRRDciVCTicHrW8w0+gejqMSM=
EOF
```

**NOTE**: The RSA keys in the above example are each a single line, but they have been line-wrapped for readability.

24. Add the Spectrum Scale servers to the HPCM data base as unmanaged servers by creating a configuration file. Make one entry for each Spectrum Scale server node, then use the `fastdiscovery` command with the configuration file as an argument. Each entry in the file is one long line, beginning with `hostname1=`. The line entries in the following example have been wrapped to fit the page.

~~~
```bash
Systwn_Name-adm # cat spectrum-scale-servers.conf
[discover]
hostname1=ADMIN_NODE_NAME, internal_name=DAEMON_NODE_NAME, mgmt_bmc_net_name=head-bmc,
mgmt_bmc_net_ip=172.24.0.15, mgmt_net_name=head, mgmt_net_interfaces="eth0",
mgmt_net_interface_name="ADMIN_NODE_NAME", mgmt_net_ip=172.23.255.10, rootfs=disk,
transport=udpcast, conserver_logging=yes, conserver_ondemand=no,
predictable_net_names=yes, redundant_mgmt_network=yes, switch_mgmt_network=yes,
console_device=ttyS1, architecture=x86_64, card_type="none", baud_rate=default,
data1_net_name=hsn, data1_net_interfaces="hsn0",
data1_net_interface_name="ADMIN_NODE_NAME-cxi0", data1_net_ip="HSN_IP_ADDRESS", other
hostname1=ADMIN_NODE_NAME, internal_name=DAEMON_NODE_NAME, mgmt_bmc_net_name=head-bmc,
mgmt_bmc_net_ip=172.24.0.24, mgmt_net_name=head, mgmt_net_interfaces="eth0",
mgmt_net_interface_name="ADMIN_NODE_NAME", mgmt_net_ip=172.23.255.11, rootfs=disk,
transport=udpcast, conserver_logging=yes, conserver_ondemand=no,
predictable_net_names=yes, redundant_mgmt_network=yes, switch_mgmt_network=yes,
console_device=ttyS1, architecture=x86_64, card_type="none", baud_rate=default,
data1_net_name=hsn, data1_net_interfaces="hsn0",
data1_net_interface_name="ADMIN_NODE_NAME-cxi0", data1_net_ip="HSN_IP_ADDRES", other
```

Where:
- _`other`_ designates this as an unmanaged server to HPCM
- _`HSN_IP_ADDRESS`_ is the IP address of the node on the HPE Cray EX HSN.
- _`ADMIN_NODE_NAME`_ is the Spectrum Scale `Admin node name` of the node.
- _`DAEMON_NODE_NAME`_ is the Spectrum Scale `Daemon node name` of the node.
- _`BMC_IP_ADDRESS`_ is the BMC IP on the BMC management network.
~~~

~~~
- _`MGMT_IP_ADDRESS`_ is the HPCM management network IP address of the DAEMON_NODE_NAME.

Run the `fastdiscovery` command to add the Spectrum Scale servers to the HPCM
database and hence to the HPCM system DNS.

```bash
System_Name-adm# fastdiscovery spectrum-scale-servers.conf
```
~~~

25. Update the file `/opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>/etc/dvs_exports.yml` to contain the following entry, where `/gpfs1` is the mount point of your Spectrum Scale file system::

```
System_name-adm# cat /opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>/etc/dvs_exports.yml
exports:
-   mode: rw
    path: /gpfs1
```

26. Chroot into that image root.

```
  System_Name-adm# chroot /opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>
```

27. Get the kernel release of the image.

```
 :/ # uname -r
 5.3.18-150300.59.37-default
```

28. Create a symbolic link with that release name, but link it to the kernel release in the directory `/lib/modules`.

```
:/ # cd /lib/modules
:/ # ls
5.3.18-150300.59.54-default  5.3.18-150300.59.68_11.0.76-cray_shasta_c
:/ # ln -s 5.3.18-150300.59.68_11.0.76-cray_shasta_c 5.3.18-150300.59.37-default
```

29. Install Spectrum Scale RPMs, including the portability layer, on the Gateway node image.

```
:/ # cd /tmp
:/ # rpm -ivh ./gpfs*.rpm
warning: ./gpfs.adv-5.1.3-0.x86_64.rpm: Header V3 RSA/SHA256 Signature, key ID efeb6eeb: NOKEY
Preparing...                          ################################# [100%]
Updating / installing...
    1:gpfs.msg.en_US-5.1.3-0           ################################# [  9%]
    2:gpfs.gskit-8.0.55-19             ################################# [ 18%]
    3:gpfs.docs-5.1.3-0                ################################# [ 27%]
    4:gpfs.afm.cos-1.0.0-5             ################################# [ 36%]
Error, do this: mount -t proc proc /proc
    5:gpfs.base-5.1.3-0                ################################# [ 45%]
Failed to connect to bus: No such file or directory
Failed to connect to bus: No such file or directory
Failed to connect to bus: No such file or directory
Failed to connect to bus: No such file or directory
Failed to connect to bus: No such file or directory
Failed to connect to bus: No such file or directory
Failed to connect to bus: No such file or directory
Created symlink /etc/systemd/system/multi-user.target.wants/mmautoload.service ->
/usr/lib/systemd/system/mmautoload.service.
Created symlink /etc/systemd/system/multi-user.target.wants/mmccrmonitor.service ->
/usr/lib/systemd/system/mmccrmonitor.service.
    6:gpfs.adv-5.1.3-0                 ################################# [ 55%]
    7:gpfs.license.dm-5.1.3-0          ################################# [ 64%]
    8:gpfs.compression-5.1.3-0         ################################# [ 73%]
    9:gpfs.crypto-5.1.3-0              ################################# [ 82%]
   10:gpfs.gpl-5.1.3-0                 ################################# [ 91%]
   11:gpfs.gplbin-5.3.18-150300.59.43_1################################# [100%]
```

30. Remove the kernel release symlink

    ```
    :/ # rm /lib/modules/5.3.18-150300.59.37-default
    ```

31. Delete the Spectrum Scale RPMs in the Gateway image /tmp directory on the System_Name-adm node. Then finish the image customization session.

    1. Delete the Spectrum Scale RPMs in the /tmp directory of the Gateway image working directory.

       ```
       :/ # rm /tmp/gpfs*.rpm
       ```

    2. Escape the chroot environment of the customized Gateway image by pressing the **Ctrl** and **D** keys simultaneously.

    3. Delete the copy of the Spectrum Scale RPMs on System_Name-adm.

       ```
       System_Name-adm# rm -rf /tmp/GPFS5
       ```

32. Verify that the Gateway image has the gpfs RPMs installed. 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 <GATEWAY_IMAGE_NAME> --duk search -s gpfs
    . . .
    Loading repository data...
    Reading installed packages...

    S  | Name               | Type    | Version    | Arch   | Repository
    ---+--------------------+---------+------------+--------+--------------------
    i+ | gpfs.adv           | package | 5.1.0-3    | x86_64 | (System Packages)
    i+ | gpfs.afm.cos       | package | 1.0.0-1    | x86_64 | (System Packages)
    i+ | gpfs.base          | package | 5.1.0-3    | x86_64 | (System Packages)
    i+ | gpfs.compression   | package | 5.1.0-3    | x86_64 | (System Packages)
    i+ | gpfs.crypto        | package | 5.1.0-3    | x86_64 | (System Packages)
    i+ | gpfs.docs          | package | 5.1.0-3    | noarch | (System Packages)
    i+ | gpfs.gpl           | package | 5.1.0-3    | noarch | (System Packages)
    i+ | gpfs.gplbin-5.3.18| package | 5.1.0-3    | x86_64 | (System Packages)
    i+ | gpfs.gskit         | package | 8.0.55-12  | x86_64 | (System Packages)
    i+ | gpfs.license.dm    | package | 5.1.0-3    | x86_64 | (System Packages)
    i+ | gpfs.msg.en_US     | package | 5.1.0-3    | noarch | (System Packages)
       | nfs-ganesha-gpfs   | package | 3.5-9.fc36| x86_64 | Cluster-Manager-1.7
    ```

    Note: The output was truncated to fit the page.

**Create a new systemd service to recover the Spectrum Scale configuration and mount after node reboot**

33. Create a systemd service file in the working directory.
    1. Create a new file /opt/clmgr/image/images//usr/lib/systemd/system/spectrum-scale.service
       ```
       System_Name-adm# vi /opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>
       /usr/lib/systemd/system/spectrum-scale.service
       System_Name-adm# cat /opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>
       /usr/lib/systemd/system/spectrum-scale.service
       [unit]
       Description=Spectrum Scale Startup

       [Install]
       WantedBy=default.target

       [Service]
       Type=oneshot
       ExecStart=/usr/lpp/mmfs/bin/mmsdrrestore -p hi1
       ExecStart=/usr/lpp/mmfs/bin/mmstartup
       ```
    2. Enable the new Spectrum Scale service.
       ```
       System_Name-adm# chroot /opt/clmgr/image/images/<GATEWAY_IMAGE_NAME>
       :/ # systemctl enable spectrum-scale.service
       Created symlink /etc/systemd/system/default.target.wants/spectrum-scale.service
       ```

```
 → /etc/systemd/system/spectrum-scale.service
:/ # exit
System_Name-adm#
```

**Push the image changes to the Leader nodes so they will be available to boot the Gateway node(s) with the Spectrum Scale-enabled image**

34. 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 <GATEWAY_IMAGE_NAME> --force
    ```

### 3.1.2 Deploy and Test Spectrum Scale Gateway Image

**PREREQUISITES**

- Create Spectrum Scale Gateway Node Image

**OBJECTIVE**

This procedure guides system administrators and installers through deploying Spectrum Scale software on all Gateway nodes in an HPE Cray EX supercomputer.

**LIMITATIONS**

This procedure provides HPE Cray EX-specific instructions for deploying Spectrum Scale software on HPE Cray EX Gateway nodes. These steps are not intended to replace the instructions and guidance provided on the IBM website (www.ibm.com).

**PROCEDURE**

**Boot Gateway nodes with the Spectrum Scale-customized image**

1. Specify the new Spectrum Scale gateway image to boot. Replace *GATEWAY_IMAGE_NAME* in the following command with the name of the gateway image modified in the previous steps in Create Spectrum Scale Gateway Node Image.

    ```
    System_Name-adm# cm node set --image GATEWAY_IMAGE_NAME -n gw01,gw02,gw03

    Please choose one of the following kernels

    1) 5.3.18-150300.59.54-default
    2) 5.3.18-150300.59.68_11.0.76-cray_shasta_c

    [2]:
    Image bp_recipe_10.2.0.3_ss_dvs using repo group: bp_recipe_10.2.0.3_shs_2.0.0-393. Use cadmin to unset.
    Repo group GATEWAY_IMAGE_REPO_GROUP specified, using repos: amdgpu aocc-compiler
    Cluster-Manager-1.7-sles15sp3-x86_64 COS-2.3.83-sles15sp3-x86_64 cpe-22.06-sles15-sp3
    dkms-plus HPE-SLE-Updates-Module-Basesystem-15-SP3-x86_64-22.03.0
    HPE-SLE-Updates-Module-Development-Tools-15-SP3-x86_64-22.03.0
    HPE-SLE-Updates-Module-Python2-15-SP3-x86_64-22.03.0
    HPE-SLE-Updates-Module-Server-Applications-15-SP3-x86_64-22.03.0

    Configuration manager initiating node configuration.
    Populating Dataset...
    Populating Dataset complete: 0.330s
    1 of 1 nodes completed in 2.8 seconds, averaging 0.4s per node
    1 of 1 nodes completed in 2.8 seconds, averaging 0.4s per node
    Node configuration complete.
    ```

2. Reset the Gateway nodes to boot the new Spectrum Scale gateway image.

    ```
    System_Name-adm# cm power reset -t node gw01,gw02,gw03
    compute node gw01 power OFF
    compute node gw02 power OFF
    compute node gw03 power OFF
    compute node gw01 power ON
    ```

```
compute node gw02 power ON
compute node gw03 power ON
```

3. Watch the node console log during the reboot. Replace *GATEWAY_NODE_NAME* in the following command with the host name of a Gateway node that is rebooting.

   Administrators can run the following command with different Gateway host names to monitor the reboot process one each of the Gateway nodes.

   ```
   System_Name-adm# console GATEWAY_NODE_NAME
   ```

4. Verify that the Spectrum Scale client software is installed on all Gateway nodes. Replace *GW_SS_LIST* with a comma-separated list of the host names of all the Gateway nodes that were rebooted with the Spectrum Scale-enabled image.

   ```
   System_Name-adm# ALL_NCN_GWS=$(<list of Gateway Spectrum Scale client names>)
   System_Name-adm# pdsh -w ${ALL_NCN_GWS} "rpm -qa gpfs.gplbin"
   gw01: gpfs.gplbin-5.3.18-150300.59.43_11.0.48-cray_shasta_c-5.1.3-0
   gw02: gpfs.gplbin-5.3.18-150300.59.43_11.0.48-cray_shasta_c-5.1.3-0
   ...
   ```

**Configure and test Spectrum Scale environment**

5. Configure the site Spectrum Scale client and server environment. Refer to site documentation and www.ibm.com for further guidance.

6. Verify that the Spectrum Scale client software is functional. Refer to site documentation and www.ibm.com for further guidance.

7. Configure DVS server on the Gateway node to project the GPFS file system. The example shown is for a one-time mount that will not exist after a reboot of either the DVS server or the DVS client on the compute node. See Project an External Filesystem to Compute Nodes or User Access Nodes for in-depth instructions on making mounts persistent across reboots. See the Tune and Optimize DVS section for mount options. This example utilizes mount options for a read only file system projection.

```
gw01:~ # mount -t dvs -o loadbalance,attrcache_timeout=14400,noclusterfs,ro,
nodename=x3000c0s7b0n0:x3000c0s9b0n0:x3000c0s24b0n0,path=MOUNT_POINT
MY_EXPORT_PATH MOUNT_POINT
```

8. Configure the DVS client on the compute node to mount the DVS projected GPFS file system. The example shown is for a one-time mount that will not exist after a reboot of either the DVS server or the DVS client on the compute node. See Project an External Filesystem to Compute Nodes or User Access Nodes for more in-depth instructions on making mounts persistent across reboots.
   1. Create mount point directory for DVS projected GPFS file system.
      ```
      nid000003:~ # mkdir /tmp/my-gpfs-path
      ```
   2. Mount the DVS projected GPFS filesystem. mount -t dvs -o path,server_list,options remote_gpfs_path local_path
      ```
      nid000003:~ # mount -t dvs -o path=/tmp/my-gpfs-path,nodename=x3000c0s7b0n0:
      x3000c0s9b0n0:x3000c0s24b0n0,maxnodes=1,noclusterfs /tmp/remote-gpfs-path /tmp/my-gpfs-path
      ```

# 4 Configure Lustre

## 4.1 Mount a Lustre File System on GWs, CNs, and Login Nodes

This section describes procedures for mounting a Lustre file system on GWs, CNs, and Login nodes in the HPCM environment. There are two methods available. The first will modify the gateway, CN, and Login node images to make the mounts persistent across reboots. And the second is to add the mounts to the live gateway, CN, and Login nodes to test the mount and/or make the mount available immediately without waiting for a reboot of the gateway, CN, and Login nodes. The persistent procedure could then be performed except for the reboot step to stage the persistent change the next time the gateway, CN, and Logins are rebooted.

- **PREQUISITES**
- Familiarity with HPCM VCS workflows.
- General familiarity with using fstab entries or systemd mount unit file for mounting file systems.
- Determine the LNet network name used by the Lustre file system. If this name is the same as the default LNet network name used by DVS (tcp99), then perform Change the Name of the DVS LNet Network first before performing this procedure.
- **OBJECTIVE**

  Configure and enable a Lustre file system on gateway nodes, compute nodes or login nodes in a HPCM system.

**Important:** When setting up LNet in this procedure, ensure that the correct network is being configured. The LNet network must match the network name that is configured on the Lustre server. It is important to ensure that there are no LNet network conflicts in environments that consist of multiple LNet network configurations. In addition, a Lustre file system requires port policies to be configured on the Slingshot switch. Consult the Slingshot Operations Guide, Section 5.6 Port Policies, to determine the port policies need to be configured for the type of file system.

### 4.1.1 Procedure

This procedure describes the steps to mount a Lustre file system on the GWs, CNs, and login nodes in a way that is persistent across node reboots.

1. Log into the HPCM admin node as the `root` user.
2. Determine the HPCM HPC Cray OS (COS) images to update for mounting a Lustre file system

   While it is possible to manage images using the HPCM image cloning method, the prefferrd method is to use the version control system (VCS) that HPCM provides. See the "Using the version control system (VCS)" Section in the "HPE Performance Cluster Manager Administration Guide". Using VCS to manage an image is similar to the cloning method. The difference is that when you use VCS, you do not need to employ a second name for the changed image. VCS does the implicit cloning and lets you change the clone. The changed image retains the same name as the original, but VCS assigns different version numbers to the images.

   Typical systems maintain separate images for gateway, compute, and login nodes.

   1. List images available

      ```
      system_name-adm:~ # cm image show
      my_cos-2.3_image_dvs
      my_cos-2.3_image_dvs-login
      my_cos-2.3_image_dvs-compute
      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. Select images to be updated with the Lustre mount
   3. Confirm that the last image revision and the working copy are the same. Resolve any differences.

      ```
      system_name-adm # cm image revision diff-contents -i <IMAGE_NAME>
      ```

   4. Compare the image on a running node with the image on the admin node

      This check is performed if you are unsure about untracked changes to the image on the running node.

      Check that working copy of image is the same as image booted. Resolve any differences.

      ```
      system_name-adm #  cm image revision diff -i bp_recipe_10.1.1_dvs -n gw02
      ```

3. HPCM maintains the VCS working copies of the images in the '/opt/clmgr/image/images/' directory

   ```
   system_name-adm:~/ # ls /opt/clmgr/image/images
   my_cos-2.3_image_dvs          my_cos-2.3_image_dvs-compute  my_cos-2.3_image_dvs-login
   my_cos-2.3_image_dvs-gateway  fmn_sles15sp3v2_A             rhel85-test-1  su-sles15sp3-new
   fmn_sles15sp3v2               login_image                   sles15sp2
   ```

4. Change directory to the working copy of the image to be updated with the Lustre mount

   ```
   system_name-adm:~/ # cd /opt/clmgr/image/images/<IMAGE_NAME>
   ```

5. Edit the `/etc/lnet.conf` file in the working copy of the image to change the LNet configuration on the nodes to include the LNet network that Lustre will be using. If Lustre will be using a numbered network (e.g. tcp2, o2ib3, etc), you should use that network instead of plain "tcp" or "o2ib".

Add the appropriate stanza to the `/etc/lnet.conf` file. If configuration of Lustre on other node types is required, the `/etc/lnet.conf` file should also be similarly modified (in the working copy of the those images)

In the following example stanza, a `tcp` network type using the `hsn0` network is added after the vi editing session.

```
system_name-adm# cat etc/lnet.conf
  ip2nets:
    - net-spec: tcp99
      interfaces:
        0: nmn0
...
system_name-adm# vi etc/lnet.conf
. . .
system_name-adm# cat etc/lnet.conf
  ip2nets:
    - net-spec: tcp99
      interfaces:
        0: nmn0
    - net-spec: tcp
      interfaces:
        0: hsn0
        1: hsn1
        2: hsn2
        3: hsn3
...
```

This is suitable for accessing Lustre over the HSN using ksocklnd while DVS continues to operate over the NMN. If your Lustre filesystem uses ko2iblnd, instead of adding the tcp stanza above, you would want to add a stanza for o2ib, as follows:

```
system_name-adm# cat etc/lnet.conf
  ip2nets:
    - net-spec: tcp99
      interfaces:
        0: nmn0
...
system_name-adm# vi etc/lnet.conf
. . .
system_name-adm# cat etc/lnet.conf
  ip2nets:
    - net-spec: tcp99
      interfaces:
        0: nmn0
    - net-spec: o2ib
      interfaces:
        0: hsn0
        1: hsn1
        2: hsn2
        3: hsn3
...
```

**Note:** If you have configured DVS for HSN, you may not have the tcp99 stanza at all, and may already have the correct stanzas here for the HSN. At this time, we recommend using the same HSN configuration for Lustre and DVS if DVS is going to run on the HSN. Also, configuring DVS for management network should be considered a debugging tool. The preffered DVS configuration is to use the HSN. For more information on running DVS over the HSN, please see: DVS over HSN Deployment.

6. Define the Lustre file system.

   Create a systemd mount unit file, and add the mount information for the Lustre file system. The following example is for configuring gateway nodes, but applies to compute and login nodes as well.

   If using an external file system, refer to Project an External Filesystem to Compute Nodes or User Access Nodes for more information on other types of entries in systemd mount unit files.

Ensure that the correct network name is used for the LNet network. See the **OBJECTIVE** section for more information.

The mount point must be in a subdirectory and not a directory in the root directory.

```
. . .
system_name-adm# vi etc/systemd/system/lus-cls12345.mount
system_name-adm# cat etc/systemd/system/lus-cls12345.mount
[Unit]
Description=Mount Lustre cls12345
Requires=lnet.service slingshot-ama.service
After=lnet.service slingshot-ama.service

ConditionPathExists=/lus/cls12345

[Mount]
What=10.10.100.3@tcp:10.10.100.4@tcp:/cls12345
Where=/lus/cls12345
Type=lustre
Options=rw,noauto,flock,lazystatfs

[Install]
WantedBy=slingshot.target
```

**NOTE:** This example is for a ksocklnd-based Lustre file system. Hence the `@tcp` suffixes to the IP addresses in the – `src` line. If you are using ko2iblnd, the suffixes should be `@o2ib`.

7. Enable the systemd mount unit file. This must be performed in the chroot environment of the IMAGE_NAME you are working in.

```
system_name-adm:~/ # cd /opt/clmgr/image/images/<IMAGE_NAME>
system_name-adm:/ # systemctl enable lus-cls12345.mount
Created symlink /etc/systemd/system/slingshot.target.wants/lus-cls12345.mount
→ /etc/systemd/system/lus-cls12345.mount.
system_name-adm:/ # systemctl is-enabled lus-cls12345
enabled
system_name-adm:/ # exit
```

8. If you have an Arista switch, your nodes must be able to talk to the external Lustre cluster through that switch attached to HSN network. **Note:** Skip this section (10) if you do not have an Arista switch, e.g. if your Lustre cluster is DirectConnect.

   1. Edit the `/etc/sysconfig/network/ifroute-hsn0` file.

      The file is listed below for reference. The content of this file will be site-specific. The IP addresses in the file must be changed.

      Ensure that the correct network name is used for the LNet network. See the **OBJECTIVE** section for more information.

      ```
      system_name-adm# vi etc/sysconfig/network/ifroute-hsn0
      system_name-adm# chmod 0644 etc/sysconfig/network/ifroute-hsn0
      system_name-adm# cat etc/sysconfig/network/ifroute-hsn0
      10.10.0.0/16 10.253.254.250 - hsn0
      ```

   2. Add LNet route changes.

      ```
      system_name-adm# ifup hsn0
      ```

9. Determine if multi-rail Lustre is enabled.

   A multi-rail Lustre setup uses two interfaces on a single node to increase throughput or to connect to more than one network. For more information about multi-rail Lustre, refer to the Lustre documentation.

   a. View the `/etc/lnet.conf` file.

   b. Check for more than one HSN interface configured under the `interfaces:` specification.

   If there is more than one interface, the system uses a multi-rail Lustre configuration.

10. Disable peer discovery unless multi-rail Lustre is used.

LNet peer discovery is disabled by adding the line `options lnet lnet_peer_discovery_disabled=1` to the end of the `/etc/modprobe.d/lnet.conf` file.

Note that the value of lnet_transaction_timeout and lnet_retry_count depends on the types of Lustre network driver being used. For kkfilnd, use lnet_transaction_timeout=126 and lnet_retry_count=0. For ko2iblnd and ksocklnd, use lnet_transaction_timeout=31 and lnet_retry_count=2.

```
system_name-adm# cat etc/modprobe.d/lnet.conf
...
  options lnet lnet_transaction_timeout=31
  options lnet lnet_retry_count=2
  options ko2iblnd map_on_demand=1
system_name-adm# vi etc/modprobe.d/lnet.conf
...
system_name-adm# cat etc/modprobe.d/lnet.conf
...
  options lnet lnet_transaction_timeout=31
  options lnet lnet_retry_count=2
  options ko2iblnd map_on_demand=1
  options lnet lnet_peer_discovery_disabled=1
```

11. Create a systemd service to apply the Lustre tuning parameters after the filesystem is mounted. The Lustre file system must be mounted before the tuning parameters are applied

```
System_Name-adm # vi usr/lib/systemd/system/lustre-tuning.service
System_Name-adm # cat usr/lib/systemd/system/lustre-tuning.service
[Unit]
Description=Apply Lustre Tuning
After=slingshot-ama.service lnet.service lus-cfs12345.mount
Requires=lnet.service slingshot-ama.service lus-cfs12345.mount

[Service]
Type=oneshot
RemainAfterExit=true
ExecStart=-/usr/sbin/lctl set_param osc.*.max_rpcs_in_flight=64
ExecStart=-/usr/sbin/lctl set_param osc.*.max_pages_per_rpc=1024
ExecStart=-/usr/sbin/lctl set_param osc.*.max_dirty_mb=256
ExecStart=-/usr/sbin/lctl set_param llite.*.max_read_ahead_mb=512
ExecStart=-/usr/sbin/lctl set_param llite.*.max_read_ahead_per_file_mb=512

[Install]
WantedBy=slingshot.target
```

12. Enable lustre-tuning.service. This must be performed in the chroot environment of the IMAGE_NAME you are working in.

```
system_name-adm:~/ # cd /opt/clmgr/image/images/<IMAGE_NAME>
system_name-adm:/ # systemctl enable lustre-tuning.service
Created symlink /etc/systemd/system/slingshot.target.wants/
lustre-tuning.service → /etc/systemd/system/lustre-tuning.service.
system_name-adm:/ # systemctl is-enabled lustre-tuning.service
enabled
system_name-adm:/ # exit
```

13. Perform Save DVS Configuration with VCS for the working copy of the the node image, using the HPCM VCS repository.

14. If your system contains Leader nodes, push the image changes for Gateway, CNs, and login nodes to the leader nodes so they will be available for boot. You will perform the `cm activate image` for each image modified in the previous steps. 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 <IMAGE_NAME> --force
```

15. Reboot the gateway nodes, compute nodes, login nodes, or all, depending on what types of nodes will be mounting the Lustre file system.

    If you changed the image name from the previous boot, specify the image to boot.

    ```
    system_name-adm: / # cm node set --image <NEW_IMAGE_NAME> -n <NODE_LIST>
    ```

    Reset the node to start the boot.

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

    Note: contains the list of gateway, compute nodes, login nodes, or all.

    Once the nodes finish rebooting, the Lustre file system mount will persist across future CN, login, and Gateway reboots.

16. Repeat this entire process for the remaining node images to configure them to mount the Lustre file system. Make the same configuration changes in the CN and login image working copies that were made in the gateway image.

17. **Optional:** Confirm that peer discovery is disabled on the gateway, compute, and login nodes. Skip this step if LNet peer discovery was not disabled earlier.

    1. SSH to a rebooted compute node.

    ```
    system_name-adm# ssh nid000001-nmn
    ```

    2. Verify that peer discovery is disabled on that node.

    ```
    nid000001# lnetctl global show
    global:
     numa_range: 0
     max_intf: 200
     **discovery: 0**
     drop_asym_route: 0
     retry_count: 2
     transaction_timeout: 120
     health_sensitivity: 100
     recovery_interval: 1
     router_sensitivity: 100
    ```

    The output of this command will include discovery:0 when LNet peer discovery is disabled.

### 4.1.2 Alternative Procedure - Add Lustre mount to live system

This is the alternate procedure that can be used to mount Lustre file systems without rebooting the gateway, CNs, and Login nodes at runtime but is not persistent across node reboots. The system administrator may want to perform this procedure on a live gateway, CN, and Login node to test the mounted Lustre file system and to make the file system available without requiring a reboot of the gateway, CN, and Login nodes. The persistent procedure could then be performed (except for the reboot step) to persist the change, the next time the gateway is rebooted. Likewise this procedure can be performed on CNs to test the mount, but a script and `pdsh` may be needed to replicate the mount across all CNs for larger systems.

1. Login to a gateway, compute, or login node to mount of the Lustre file system to the node.

Refer to https://wiki.lustre.org/LNet_Router_Config_Guide for LNet dynamic configuration.

2. Edit the `/etc/fstab` file on the booted CN node to mount Lustre file system on the CNs. Skip this step if the new file system will not be projected to the CNs. This step is intended to test the DVS mount on a live system but may not be practical for making the mount available across a system.

    1. Edit the `/etc/fstab` file so that it contains a `shared library file system` entries similar to the example in next substep.

3. Edit the `/etc/lnet.conf` file in the working copy of the image to change the LNet configuration on the nodes to include the LNet network that Lustre will be using. If Lustre will be using a numbered network (e.g. tcp2, o2ib3, etc), you should use that network instead of plain “tcp” or “o2ib”.

Add the appropriate stanza to the `/etc/lnet.conf` file. If configuration of Lustre on login nodes is required, the `/etc/lnet.conf` file should also be similarly modified (in the working copy of the login image). Likewise, if Lustre is required on gateway nodes, the `/etc/lnet.conf` file should be modified in the working copy of the gateway image.

In the following example stanza, a `tcp` network type using the `hsn0` network is added after the vi editing session.

```
cn0001 # cat etc/lnet.conf
  ip2nets:
    - net-spec: tcp99
      interfaces:
        0: nmn0
...
cn0001 # vi etc/lnet.conf
. . .
cn0001 # cat etc/lnet.conf
  ip2nets:
    - net-spec: tcp99
      interfaces:
        0: nmn0
    - net-spec: tcp
      interfaces:
        0: hsn0
        1: hsn1
        2: hsn2
        3: hsn3
...
```

This is suitable for accessing Lustre over the HSN using ksocklnd while DVS continues to operate over the NMN. If your Lustre filesystem uses ko2iblnd, instead of adding the tcp stanza above, you would want to add a stanza for o2ib, as follows:

```
cn0001 # cat /etc/lnet.conf
  ip2nets:
    - net-spec: tcp99
      interfaces:
        0: nmn0
...
cn0001 # vi /etc/lnet.conf
. . .
cn0001 # cat /etc/lnet.conf
  ip2nets:
    - net-spec: tcp99
      interfaces:
      interfaces:
        0: nmn0
...
cn0001 # vi /etc/lnet.conf
. . .
cn0001 # cat /etc/lnet.conf
  ip2nets:
    - net-spec: tcp99
      interfaces:
        0: nmn0
    - net-spec: o2ib
      interfaces:
        0: hsn0
        1: hsn1
        2: hsn2
        3: hsn3
...
```

**Note:** If you have configured DVS for HSN, you may not have the tcp99 stanza at all, and may already have the correct stanzas here for the HSN. At this time, we recommend using the same HSN configuration for Lustre and DVS if DVS is going to run on the HSN. Also, configuring DVS for NMN should be considered a debugging tool. The prefered DVS configuration is to use the HSN. For more information on running DVS over the HSN, please see: DVS over HSN Deployment.

4. Define the Lustre file system.

   Edit the `/etc/fstab` file, adding the mount information for the Lustre file system. The following example is for configuring gateway nodes, but applies to compute and login nodes as well.

   If using an external file system, refer to Project an External Filesystem to Compute Nodes or User Access Nodes for more information on other types of entries in the `/etc/fstab` file.

   Ensure that the correct network name is used for the LNet network. See the **OBJECTIVE** section for more information.

   The mount point must be in a subdirectory and not a directory in the root directory.

   ```
   . . .
   cn0001 # vi /etc/fstab
   cn0001 # cat /etc/fstab
   10.10.100.3@tcp:10.10.100.4@tcp:/cls12345 /lus/cls12345
   lustre rw,noauto,flock,lazystatfs 0 0
   . . .
   ```

   Another option is to use a systemd mount unit file that will test what you will be committing to the image.

   ```
   . . .
   cn0001 # vi /etc/systemd/system/lus-cls12345.mount
   cn0001 # cat /etc/systemd/system/lus-cls12345.mount
   [Unit]
   Description=Mount Lustre cls12345
   Requires=lnet.service slingshot-ama.service
   After=lnet.service slingshot-ama.service

   ConditionPathExists=/lus/cls12345

   [Mount]
   What=10.10.100.3@tcp:10.10.100.4@tcp:/cls12345
   Where=/lus/cls12345
   Type=lustre
   Options=rw,noauto,flock,lazystatfs

   [Install]
   WantedBy=slingshot.target
   ```

   **NOTE:** This example is for a ksocklnd-based Lustre file system. Hence the `@tcp` suffixes to the IP addresses in the `- src` line. If you are using ko2iblnd, the suffixes should be `@o2ib`.

5. If you used a systemd mount unit file then enable it.

   ```
   cn0001 # systemctl enable lus-cls12345.mount
   Created symlink /etc/systemd/system/slingshot.target.wants/lus-cls12345.mount
   → /etc/systemd/system/lus-cls12345.mount.
   cn0001 # systemctl is-enabled lus-cls12345
   enabled
   ```

6. If you have an Arista switch, your nodes must be able to talk to the external Lustre cluster through that switch attached to HSN network. **Note:** Skip this section (6) if you do not have an Arista switch, e.g. if your Lustre cluster is DirectConnect.

   1. Edit the `/etc/sysconfig/network/ifroute-hsn0` file.

      The file is listed below for reference. The content of this file will be site-specific. The IP addresses in the file must be changed.

      Ensure that the correct network name is used for the LNet network. See the **OBJECTIVE** section for more information.

```
cn0001# vi /etc/sysconfig/network/ifroute-hsn0
cn0001# chmod 0644 /etc/sysconfig/network/ifroute-hsn0
cn0001# cat /etc/sysconfig/network/ifroute-hsn0
10.10.0.0/16 10.253.254.250 - hsn0
```

2. Add LNet route changes.

```
system_name-adm# ifup hsn0
```

7. Determine if multi-rail Lustre is enabled.

   A multi-rail Lustre setup uses two interfaces on a single node to increase throughput or to connect to more than one network. For more information about multi-rail Lustre, refer to the Lustre documentation.

   a. View the `/etc/lnet.conf` file.

   b. Check for more than one HSN interface configured under the `interfaces:` specification.

   If there is more than one interface, the system uses a multi-rail Lustre configuration.

8. Disable peer discovery unless multi-rail Lustre is used.

   LNet peer discovery is disabled by adding the line `options lnet  lnet_peer_discovery_disabled=1` to the end of the `/etc/modprobe.d/lnet.conf` file.

   Note that the value of lnet_transaction_timeout and lnet_retry_count depends on the types of Lustre network driver being used. For kkfilnd, use lnet_transaction_timeout=126 and lnet_retry_count=0. For ko2iblnd and ksocklnd, use lnet_transaction_timeout=31 and lnet_retry_count=2.

```
cn0001 # cat /etc/modprobe.d/lnet.conf
...
  options lnet lnet_transaction_timeout=31
  options lnet lnet_retry_count=2
  options ko2iblnd map_on_demand=1
cn0001 # vi /etc/modprobe.d/lnet.conf
...
cn0001 # cat /etc/modprobe.d/lnet.conf
...
  options lnet lnet_transaction_timeout=31
  options lnet lnet_retry_count=2
  options ko2iblnd map_on_demand=1
  options lnet lnet_peer_discovery_disabled=1
cn0001 # systemctl stop lnet
cn0001 # systemctl start lnet
```

   Verify LNet is loaded.

```
cn0001 #  lsmod | grep lnet
lnet                  688128  2 kkfilnd
libcfs                262144  2 kkfilnd,lnet
sunrpc                401408  8 lnet,lockd,nfsv3,nfs
```

9. **Optional:** Confirm that peer discovery is disabled on the gateway, compute, and login nodes. Skip this step if LNet peer discovery was not disabled earlier.

   Verify that peer discovery is disabled on that node.

```
  cn0001# lnetctl global show
  global:
   numa_range: 0
   max_intf: 200
   discovery: 0
   drop_asym_route: 0
   retry_count: 2
   transaction_timeout: 120
   health_sensitivity: 100
   recovery_interval: 1
   router_sensitivity: 100
```

The output of this command will include discovery:0 when LNet peer discovery is disabled.

10. Mount the Lustre file system.

    If you used the 'fstab' method:

    ```
    cn0001 # mount -a
    ```

    If you used the systemd mount unit file method

    ```
    cn0001 # systemctl start lus-cls12345.mount
    ```

11. **Optional:** Apply Lustre tuning parameters.

    ```
    cn0001 # lctl set_param osc.*.max_rpcs_in_flight=64
    cn0001 # lctl set_param osc.*.max_pages_per_rpc=1024
    cn0001 # lctl set_param osc.*.max_dirty_mb=256
    cn0001 # lctl set_param llite.*.max_read_ahead_mb=512
    cn0001 # lctl set_param llite.*.max_read_ahead_per_file_mb=512
    ```

## 4.2 Set Up LNet Routers

- **OBJECTIVE**

  Configure one or more Application nodes as LNet Routers.

- **ROLE**

  System administrator

- **Requirements**

  - A system domain expert (typically an admin) with the following background knowledge:
    * Hardware in the system, including how to configure the Lustre cluster for communication with routers. **Note:** Cluster configuration in particular is outside the scope of this document. Please see https://wiki.lustre.org/LNet_Router_Config_Guide and other Lustre documentation for further details.
    * HPCM image management, either through image cloning or using the Version Control System (VCS) that HPCM provides. See the "Using the version control system (VCS)" Section in the "HPE Performance Cluster Manager Administration Guide". Using VCS to manage an image is similar to the cloning method. The difference is that when you use VCS, you do not need to employ a second name for the changed image. VCS does the implicit cloning and lets you change the clone. The changed image retains the same name as the original, but VCS assigns different version numbers to the images.
    * Installing software into images. See the "Installing new software into new images" Section in the "HPE Performance Cluster Manager Administration Guide".
    * How to assign and boot images with HPCM. See HPCM documentation on Using the administrative interface for details.
  - Network interface prerequisites:
    * Network interfaces on the LNet router nodes are configured. Refer to the site network documentation for details.

- **Caveats**

  - This procedure is provided as guidance only. Hewlett Packard Enterprise has not thoroughly tested these steps and does not guarantee that they will work for all customers.

Perform this procedure to establish one or more Application nodes as LNet Routers.

1. Determine your LNet router image name:

   List nodes with their assigned images, booted image is displayed in second column:

   ```
   System_Name-adm # cm node show -I
   NODE                 IMAGE.NAME                  KERNEL
   fmn1                 fmn_sles15sp3v2_SS_2.0.0-393 5.3.18-59.37-default
   . . .
   ```

   List the images on your system and select the name of your LNet router image in the following query:

```
System_Name-adm # cm image show
my_cos-2.3_image_LNet-router
my_cos-2.3_image_dvs-compute
my_cos-2.3_image_application
my_cos-2.3_image_dvs-gateway
fmn_sles15sp3v2
fmn_sles15sp3v2_A ldap_recipe_8.0.0_ga
my_cos-2.3_login_image
sles15sp2
su-sles15sp3-new
```

If you do not find a LNet router image, you may need to clone an existing image and customize it for LNet router functionality.

```
System_Name-adm # cm image copy -o <EXISTING_IMAGE> -i <NEW_IMAGE> -g <REPO_GROUP>
```

When you create an image, it resides in the following directory: `/opt/clmgr/image/images/NEW_IMAGE`. In most cases, after the new image is created, the cluster manager sends a copy to the VCS repository and sets their revision number to 1.

2. Commit last set of changes to the selected image.

   Display the changes between the latest image checked into VCS and the working copy of the image to review changes.

```
System_Name-adm # cm image revision diff-contents -i <IMAGE_NAME>
```

   Use the `cm image revision commit` command to commit your changes to VCS. The working copy of the image resides in the /opt/clmgr/image/images/ directory. The commit requires you to enter a log message. You can specify the log message with the -m parameter or you can enter the message from the terminal during the running of the command.

```
System_Name-adm # cm image revision commit -i <IMAGE_NAME> \
-m "changing LNet parameters for client mount options"
```

   Confirm commit by reviewing revision history.

```
System_Name-adm # cm image revision history -i <IMAGE_NAME>
```

3. Configure LNet routing.

   Create or modify the lnet.conf file which describes the source and destination networks for LNet routers and enables routing.

```
System_Name-adm # cd /opt/clmgr/image/images/<LNETROUTER_IMAGE_NAME>/etc
System_Name-adm # vi lnet.conf
System_Name-adm # cat lnet.conf
net:
  - net type: <LC_NETWORK_NAME>
    local NI(s):
      - interfaces:
          0: <LC_INTERFACE_0>
      - interfaces:
          0: <LC_INTERFACE_1>
          ...
  - net type: <HSN_NETWORK_NAME>
    local NI(s):
      - interfaces:
          0: <HSN_INTERFACE_0>
      - interfaces:
          0: <HSN_INTERFACE_1>
          ...
routing:
  - enable: 1

System_Name-adm # cat modprobe.d/lnet.conf
options lnet lnet_transaction_timeout=<LTT>
options lnet lnet_retry_count=<LRC>
options ko2iblnd map_on_demand=1
```

where by LC we mean the network hosting the Lustre Cluster and by HSN we mean the network generally available to the rest of the HPC system.

The value of LTT and LRC depends on the types of networks being routed. For kfilnd, LTT=126 and LRC=0. For ko2iblnd and ksocklnd, LTT=31 and LRC=2. If both kfilnd and either ko2iblnd or ksocklnd are being used then the values for kfilnd should be specified.

The content in the `modprobe.d/lnet._conf` file may be different for your system, depending on the characteristics of your <LC_NETWORK> and <HSN_NETWORK>. We mainly include this section to demonstrate that this is how modprobe options should be specified for your routers.

4. Commit your new content.

```
System_Name-adm # cm image revision commit -i <LNETROUTER_IMAGE_NAME> \
-m "changing LNet routing parameters for client mount options"
```

5. If your system contains Leader nodes, push the image 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 <LNETROUTER_IMAGE_NAME> --force
```

6. To apply your changes, assign the <LNETROUTER_IMAGE_NAME to the LNet router node and boot the node:

   1. If you changed the image name previously assigned to the LNet router assign the new name to the LNet router node.

   ```
   System_Name-adm:~/ # cm node set --image <LNETROUTER_IMAGE_NAME> \
   -n <LNET-ROUTER_HOSTNAME>
   ```

   2. Reboot the LNet routers using the `cm power` command.

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

   NOTE: <NODE_LIST is a comma separated list of the LNet router nodes.

7. Validate that your LNet routers have booted with the expected configuration by running the following commands for each LNet router:

   1. Verify that local interfaces have been configured correctly:

   ```
   System_Name-adm # ssh <LNET_ROUTER> lnetctl net show
   net:
       - net type: lo
         local NI(s):
           - nid: 0@lo
             status: up
       - net type: <LC_NETWORK_NAME>
         local NI(s):
           - nid: <LC_ADDR_0>
             status: up
             interfaces:
                 0: <LC_INTERFACE_0>
   ...
       - net type: <HSN_NETWORK_NAME>
         local NI(s):
           - nid: <HSN_ADDR_0>
             status: up
             interfaces:
                 0: <HSN_INTERFACE_0>
   ...
   ```

   2. Verify that routing is enabled:

   ```
   System_Name-adm # ssh <LNET_ROUTER> lnetctl routing show
   routing:
       - cpt[0]:
             tiny:
                 npages: 0
   ```

```
                    nbuffers: 512
                    credits: 512
                    mincredits: 512
                small:
                    npages: 1
                    nbuffers: 4096
                    credits: 4096
                    mincredits: 4096
                large:
                    npages: 256
                    nbuffers: 256
                    credits: 256
                    mincredits: 256
          - enable: 1
    buffers:
        tiny: 512
        small: 4096
        large: 256
```

8. Capture LNet addresses for client configuration.

   Capture the <HSN_NETWORK> LNet addresses of each LNet router for use in client configuration by running the following command on each LNet router, and preserving the results (by e.g., appending to a file):

   ```
   System_Name-adm # ssh <LNET_ROUTER> lnetctl net show | grep nid: \
   > | grep <HSN_NETWORK_NAME> | tr -s ' ' | cut -d\  -f 4
   ```

9. Configure all desired client nodes in the system to route to your new LNet routers by updating their `lnet.conf` files in their respective image working copies, then deploy the updates.

   All node types will need a `lnet.conf` file equivalent to the following example based on the Compute node class:

   ```
   System_Name-adm # cd /opt/clmgr/image/images<COMPUTE_IMAGE_NAME>
   System_Name-adm # vi etc/lnet.conf
   System_Name-adm # cat etc/lnet.conf
   net:
       - net type: <NETWORK_NAME>
         local NI(s):
           - interfaces:
               0: <INTERFACE_0>
           - interfaces:
               0: <INTERFACE_1>
           ...
   route:
       - net: <NETWORK_NAME>
         gateway: <LNET_ADDR_0>
       - net: <NETWORK_NAME>
         gateway: <LNET_ADDR_1>
         ...
   ```

   where `<NETWORK_NAME>` here should be the same as the `<HSN_NETWORK_NAME>` of your router stanza(s), and a given `<LNET_ADDR>` should be one of the LNet addresses you recorded during the previous step: "Capture LNet addresses for client configuration".

   **Note:** There may already be configuration content in one or more of the lnet.conf files discussed below. Be sure to add content rather than replace it if that is the case.

   **Note**: Each of the client classes may have its own respective image working directory. Make the modifications to the `/etc/lnet.conf` file in each of the image working copies. Otherwise, the steps for generally managing the images (e.g. cloning; committing; pushing) are identical.

   Image management steps for client node types (Compute; Login; Gateway):

```
To set up routing on the desired client node types, you will need to
add configuration content based on the example client stanza above, but
then follow steps similar to those used when establishing LNet routers
above to actualize your configuration changes on the nodes, including:
- saving previous image changes
- modifying the /etc/lnet.conf file in the working copy of the image
- pushing the new images out to the Leader nodes
- assigning the modified images to the appropriate client nodes to use the new images
- and rebooting the nodes to use the modified images.
```

# 5 Power Management

Power management of HPE Cray EX systems.

## 5.1 Overview

Power management includes the following:

- Monitoring energy use
- Managing energy use at the system level and at the job level
- Managing the thermal health of the cluster

For power management information, see the following:

```
[HPE Performance Cluster Manager Power Mangement Guide](https://www.hpe.com/support/hpcm-power-010)
[HPE Performance Cluster Manager Administration Guide] (https://support.hpe.com/hpesc/public/docDisplay?docId
```

## 5.2 PM Counters

The blade-level and node-level accumulated energy telemetry is point-in-time power data. Blade accumulated energy data is collected out-of-band and is made available via workload managers. Users have access to the data in-band at the node-level via special sysfs files in /sys/cray/pm_counters on the node.

# 6 Set the Turbo Boost Limit

Set the turbo boost limit.

Turbo boost limiting is supported on the Intel® and AMD® processors. Because 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 values:

- 0 - Disable turbo boost
- 999 - (default) No limit is applied.

The following values are not supported by COS:

- 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

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.

## 6.1 Set or Change the Turbo Boost Limit Parameter

TBD

# 7 User Access to Compute Node Power Data

HPE Cray EX Liquid Cooled AMD EPYC compute node power management data is available to users.

HPE Cray EX Liquid Cooled compute blade power management counters (pm_counters) enable users access to energy usage over time for billing and job profiling.

The blade-level and node-level accumulated energy telemetry is point-in-time power data. Blade accumulated energy data is collected out-of-band and is made available via workload managers. Users have access to the data in-band at the node-level via special sysfs files in /sys/cray/pm_counters on the node.

Time-stamped energy data from each node can be captured for a specific job before, during, and after the job to generate a power profile about the job. This energy usage data can be used in conjunction with current energy costs to assign a monetary value to the job.

The node CPU vendor provides specific in-band and out-of-band interfaces for controlling power management. In-band interfaces are accessed from the node OS through special `sysfs` files in /sys/cray/pm_counters on the node. Out-of-band interfaces are accessed from a node BMC or Redfish API.

The combined power consumption of the CPU and the accelerator can never exceed this limit. Thus, power to either the CPU or the accelerator must be capped so it does not exceed the total amount of power available.

## 7.1 pm_counters

Access to compute node power and energy data is provided by a set of files located in /sys/cray/pm_counters/ on the node. All pm_counters are accompanied by a timestamp.

- `power`: Point-in-time power (Watts). When accelerators are present, includes `accel_power`. See the limitation on data collection from accelerators in the `accel_power` bullet point below.
- `energy`: Accumulated energy, in joules. When accelerators are present, includes `accel_energy`. See the limitation on data collection from accelerators in the `accel_energy` bullet point below.
- `cpu_power`: Point-in-time power (Watts) used by the CPU domain.
- `cpu_energy`: The total energy (Joules) used by the CPU domain.
- `cpu_temp`: Temperature reading (Celsius) of the CPU domain.
- `memory_power`: Point-in-time power (Watts) used by the memory domain.
- `memory_energy`: The total energy (Joules) used by the memory domain.
- `accel_energy`: Accumulated accelerator energy (Joules). The data is non-zero only when an accelerator is present on the node.
- `accel_power`: Accelerator point-in-time power (Watts). The data is non-zero only when an accelerator is present on the node.
- `generation`: A counter that increments each time a power cap value is changed.
- `startup`: Startup counter
- `freshness`: Free-running counter that increments at a rate of approximately 10Hz.
- `version`: Version number for power management counter support.
- `power_cap`: Current power cap limit in Watts; 0 indicates no capping. When accelerators are present, see `accel_power_cap` bullet point below.
- `accel_power_cap`: Accelerator power cap limit in Watts; 0 indicates no capping.
- `raw_scan_hz`: The power management scanning rate for all data in pm_counters.

# 8 Enable Low Noise Mode

Some application workloads show improved performance when compute node operating system tasks (sources of "OS noise") are migrated to one or more system CPUs that are excluded from application use. Low Noise Mode configures Linux features to achieve this configuration.

Two variations of Low Noise Mode (LNM) are available: full and lightweight. In full LNM, the Linux kernel is booted with parameters instructing it to reduce noise by moving system activities to CPU 0 (and potentially additional CPUs as well) and user space is configured to move any overhead processes to CPU 0. In lightweight mode, the kernel parameters are not used, but the user space configuration is still done.

## 8.1 Kernel Parameters

In full LNM mode, the Linux kernel must be booted with parameters to direct noise overhead to CPU 0:

```
nohz_full=1-255
rcu_nocbs=1-255
```

```
rcu_nocb_poll
```

The CPU range has to be specified for the number of CPUs on the node.

In addition, two parameters are specified on the kernel command line to guide the behavior of the user space configuration:

```
lnm={ full, lightweight }
[lnm.cpu=0]
```

The `lnm` parameter must be set to either `full` or `lightweight`. The `lnm.cpu` parameter is optional and is set to what CPU(s) to use to handle system overhead (see below for the full syntax).

## 8.2 The LNM Configuration File

At system boot, systemd runs lnmctl to configure the user space environment for low noise. Lnmctl reads default configuration from /etc/lnm-default.json and optional site configuration from /etc/lnm.json. The format of the two files is identical, however the site configuration file is not required to have all of the sections. The items in the site configuration file will override the ones in the defaults. This way individual items can be overridden without requiring that an entire section, or the whole file, be duplicated.

The following is an example of a LNM configuration file:

```
{
    "Tunables": {
        "sysctl": {
            "vm.stat_interval": 120
        },
        "files": {
            "/sys/bus/workqueue/devices/writeback/cpumask": 1,
            "/sys/kernel/mm/transparent_hugepage/enabled": "never"
        }
    },
    "Processes": [],
    "IRQs": {},
    "CPU": 0
}
```

### 8.2.1 CPU Section

The CPU section specifies which processors are used for system overhead CPUs. The system overhead CPUs are the CPUs that run all of the kernel services and user space processes that are not part of an application. These processors are not used to run user applications.

This parameter can be overridden by the `lnm.cpu=` kernel parameter.

The CPU field can list a single CPU, a range of CPUs, or `MAX` for lnmctl to find the highest numbered CPU on the node and use it.

```
{
    "CPU": 0
}
```

```
{
    "CPU": "16-32"
}
```

```
{
    "CPU": "MAX"
}
```

The same syntax is used for the `lnm.cpu` kernel parameter:

```
lnm.cpu=128
```

```
lnm.cpu=253-255
```

```
lnm.cpu=MAX
```

### 8.2.2 Tunables Section

The tunables section lists system tuning configuration, which is set automatically by LNM using `sysctl` or writing to files under `/sys`.

This section has two sub-sections: sysctl and files. The sysctl sub-section lists settings to be set with the sysctl utility, and their values. The files sub-section lists files to write to (typically under `/sys`) and the values to write.

These values can be overridden in the site configuration file.

This example sets the cpumask to `CPUMASK`, which will be calculated from the CPUs listed in the CPU section. It also removes setting `/sys/kernel/mm/transparent_hugepage/enabled` by setting it to an empty string.

```
{
    "Tunables": {
        "files": { '/sys/bus/workqueue/devices/writeback/cpumask': "CPUMASK",
            "/sys/kernel/mm/transparent_hugepage/enabled": ""
        }
    }
}
```

This example sets the cpumask to `0xffff`. The value is quoted so it looks like a string to json. Hexadecimal values are not valid json, so this must be in quotes, or converted to decimal.

```
{
    "Tunables": {
        "files": { '/sys/bus/workqueue/devices/writeback/cpumask': "0xffff",
        }
    }
}
```

### 8.2.3 Processes Section

This section contains a list of process names that should not be migrated to the system CPU(s). By default, all processes are migrated, with the exception of processes such as per-CPU threads that cannot be migrated.

This example lists two sets of processes to exclude from migration. The first is any process with `watchdog` in it's name. The second is any process with the exact name `spire_agent`.

```
{
    "Processes": [".*watchdog.*", "spire_agent" ]
}
```

### 8.2.4 IRQs Section

This section lists what hardware Interrupt Requests (IRQs) should be directed to CPUs handling system overhead.

IRQs can be listed two different ways:

- Migrate IRQs to the system overhead CPUs, listed in the CPU section, by providing a list of the IRQ numbers:

```
{
    "IRQs": [ 0, 1, 2 ]
}
```

- Migrate IRQs to specific CPUs using IRQ to CPU map. IRQs are listed by number, and there are two options for where IRQs can be directed: either CPU 0 (`cpu_0`) or the highest numbered CPU on the node (`cpu_last`).

This example directs IRQ 0 to CPU 0 and IRQ 1 to the highest numbered CPU.

```
{
    "IRQs": {
        "0": "cpu_0",
        "1": "cpu_last"
    }
}
```

### 8.2.5 Interaction With Workload Manager Software

This section provided information about enabling and customizing Low Noise Mode in COS. Workload manager software also needs to be configured to work properly with compute nodes booted with LNM enabled. See the “HPE Cray Programming Environment Installation Guide: HPCM on HPE Cray EX and HPE Cray Supercomputer Systems (S-8012)” for further details.