Multi-Rail Architecture in Cornelis™ Omni-Path™ Fabrics

Application Note

Rev. 1.0

June 2021

Doc. No: A00016, Rev.: 1.0

Legal Disclaimer

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Cornelis Networks products described herein. You agree to grant Cornelis Networks a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

All product plans and roadmaps are subject to change without notice.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cornelis Networks technologies may require enabled hardware, software or service activation.

No product or component can be absolutely secure.

Your costs and results may vary.

Cornelis Networks and the Cornelis Networks logo belong to Cornelis Networks, Inc. Other names and brands may be claimed as the property of others.

Contents

Figures

Tables

Revision History

Date	Revision	Description
June 2021	1.0	Initial release.

1 Introduction

This document provides guidance for configuring Multi-Rail systems using Cornelis Omni-Path clusters. It includes a high-level architecture description of the Multi-Rail capabilities, various fabric configurations, messaging paradigms, and options for varying rail counts on host nodes, followed by instructions for configuring the fabric and software.

Note: The concepts presented in this document for Dual-Rail configurations can be extended for configurations with more than two rails per host node.

1.1 Terminology

Table 1. Terminology

Term	Description
Multi-Rail	A configuration with more than one HFI per host (host/server/node are used interchangeably).
Dual-Rail	A special configuration of Multi-Rail with two HFIs per host. This can also be referred to as *dual injection* or *dual interfaces*.
Single-Fabric	All HFIs per host are connected to the same fabric.
Multi-Fabric	Each HFI in a host connects to a separate fabric.
Endpoint	An entity with a unique LID. An HFI can be assigned 1, 2, 4, or 8 LIDs based on LMC.
Fabric	A set of interconnected switches and hosts.
Lid Mask Control (LMC)	A parameter that determines the number of unique LIDs (2^LMC) assigned to an HFI for use with dispersive routing.
Local Identifier (LID)	A unique identifier or address given to an HFI or switch within its fabric domain to assist packet flow.
Subnet	A fabric management domain. This is distinct from IP subnet.

1.2 Reference Documents

The following table lists the relevant Cornelis Omni-Path publications.

Table 2. Reference Documents

Document Title	Location
Cornelis Omni-Path Fabric Host Software User Guide	https://customercenter.cornelisnetworks.com/#/customer/assets/download/10
Cornelis Omni-Path Fabric Performance Tuning User Guide	https://customercenter.cornelisnetworks.com/#/customer/assets/download/16
Cornelis Omni-Path Fabric Software Installation Guide	https://customercenter.cornelisnetworks.com/#/customer/assets/download/5
Cornelis Omni-Path Fabric Suite FastFabric User Guide	https://customercenter.cornelisnetworks.com/#/customer/assets/download/14
Cornelis Omni-Path IP and LNet Router Design Guide	https://customercenter.cornelisnetworks.com/#/customer/assets/download/15
Cornelis Performance Scaled Messaging 2 (PSM2) Programmer's Guide	https://customercenter.cornelisnetworks.com/#/customer/assets/download/17

1.3 Technical Support

Technical support for Cornelis Omni-Path products is available 24 hours a day, 365 days a year. Please contact Cornelis Networks Customer Support by visiting www.cornelisnetworks.com or send your requests directly to support@cornelisnetworks.com.

2 Overview

The Cornelis Omni-Path interconnect architecture provides support for using multiple Host Fabric Interface (HFI) cards/ports per host to achieve higher throughput.

In certain scenarios, configuring multiple HFIs can provide versatile communication capabilities through separation of traffic such as:

- Collectively supporting a single message stream
- Operating independently (and in parallel) on multiple message streams

Note: In this document, multiple interfaces are referred to as *multi-rail* regardless of the usage models described in various scenarios.

Not only can multi-rail architecture provide hosts with higher throughput but also reduced latency. The performance achieved will depend on the basic configuration of the nodes, the network topology, the communication needs of the application, and the potential tuning required for the configuration.

Higher performance can be achieved using multi-rail by following a few simple steps. Several decisions must be made at the time of cluster acquisition; other decisions are operational and may be made any time.

During the fabric design process, you will decide on the fabric configuration or topology. This is influenced by factors such as:

- Node count
- Fabric cost and power consumption
- Compute and storage performance

If you are planning to upgrade an existing cluster from single-rail to dual-rail, the ease of upgrade may be an additional consideration.

Cornelis Networks' *Fabric Design Generator* tool can be used to create different fabric designs with Multi-Rail capability.

Note: Your choice of topology, connections, and messaging will affect how the hosts can communicate with each other.

3 Fabric Configuration

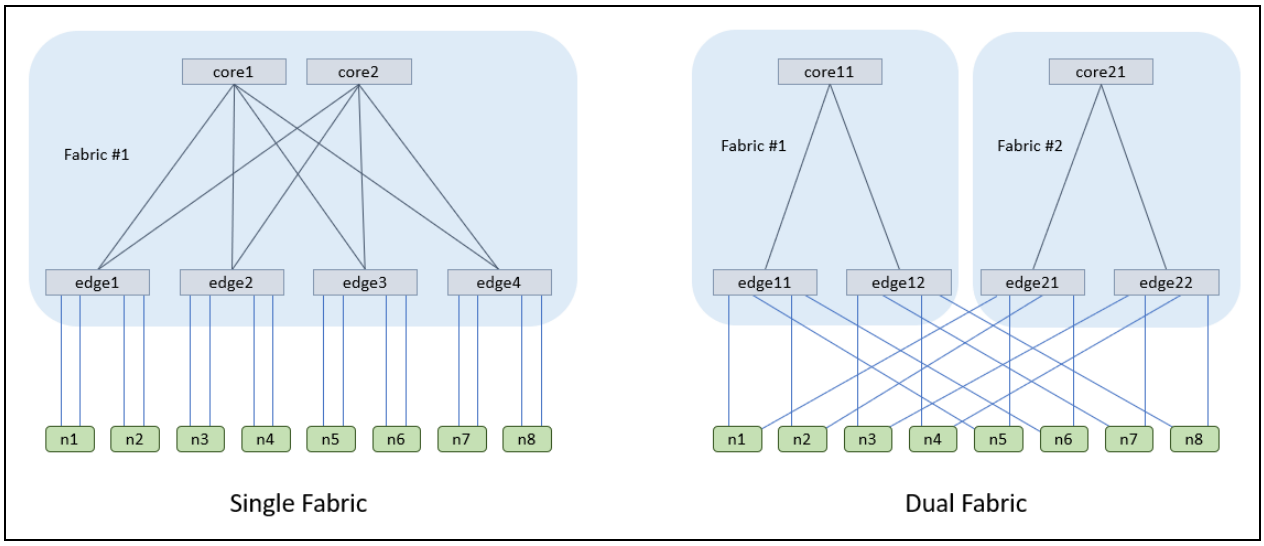
This section describes considerations for choosing the appropriate fabric with typical configuration and messaging use cases.

3.1 Configuration Choices—Single and Dual Fabrics

In general, you can choose a single or dual fabric implementation for dual-rail clusters.

When all HFIs on all servers connect to the same switching network, they form a single fabric (Figure 1, left). The two HFIs on a server may be connected to the same Cornelis Omni-Path Architecture (OPA) switch or to different switches in the fabric, providing 200 Gbps capability to the servers with dual rails.

Figure 1. Illustrative Difference Between Single and Dual Fabrics



In dual fabric installations (Figure 1, right), each HFI is connected to a separate, independent, switching fabric. Since the fabrics are independent, HFIs on one fabric cannot communicate with those on the other fabric. However, the hosts can leverage the dual-rail interfaces to target a combined 200 Gbps performance.

Variations to the above scenarios are possible. For example, when some servers in a dual fabric cluster have only one HFI, those servers can only communicate with other servers that are connected through its fabric.

Fabric Management

In a single fabric implementation, all HFIs and switches in the fabric are managed by a single instance of Cornelis Omni Path Fabric Suite Fabric Manager (FM). Consequently, a single-rail FM node is sufficient to perform this function.

A dual fabric implementation will require an instance of FM for each fabric. Typically, these run on the same dual-rail node, but are always independent entities.

The FM server can be configured to support a cluster with dual fabrics with just a few additional settings. Refer to Section 4, Software Configuration.

3.2 Configuration Use Cases

This section describes typical use cases as guidance for choosing between single and dual fabric configurations.

Table 3. Configuration Use Cases

Use Case	Guidance
Small Clusters: Less Than 576 Nodes	Single fabric is cost effective to achieve 200 Gbps performance since all HFIs can be cabled into a single Director Class Switch.
Medium Clusters: Between 576 and 1152 Nodes	Dual fabric has lower cost and lower power consumption since each fabric is a two-tier fat tree. A single fabric for the same number of adapters would require a three-tier fat tree, thus adding an entire layer of switches.
Large Clusters: More than 1152 Nodes but Less Than 48K Endpoints[1]	Cost and power considerations become immaterial as the cluster size increases beyond 1152 nodes since both single fabric and dual fabric configurations are three-tier topologies. The choice between single and dual fabric configuration moves to performance. If the applications running on the cluster would benefit from 200 Gbps speed for a single message stream, then choose single fabric.
Very Large Clusters: More Than 48K Endpoints[1]	Since the Cornelis Omni-Path architecture supports a linear forwarding table with 48K entries, dual fabric is the only option for clusters that need more than 48K entries. For example, this could be 48k nodes with no dispersive routing or 12k nodes with 4 LIDs (LMC=2) each.
Over-Subscribed Clusters	Analysis shows the above use case considerations apply to over-subscribed fabrics too, except for medium clusters where the advantage of dual fabrics diminishes at higher inter-switch link over-subscription ratios.

Use Case	Guidance
Storage Node Connectivity	On storage nodes, multiple interfaces are often used to obtain more storage network bandwidth to match the capabilities of dense storage platforms or to provide redundancy and failover capabilities. In such environments, a single fabric is the best choice and is supported by popular file systems such as Lustre* and IBM* Spectrum Scale. Note that if storage nodes are single-fabric and compute nodes are dual-rail dual-fabric, storage traffic occurs only over the fabric to which the storage interface is connected. In this case, the achievable bandwidth is only 100 Gbps.
Upgrading an Existing Single HFI Cluster	A dual fabric configuration will be attractive since no re-cabling of the existing connections will be necessary. A case-by-case analysis will be needed to ascertain the best choice.
NOTE: 1. Switch devices are also assigned LIDs and are therefore considered endpoints.	

3.3 Messaging Use Cases

This section describes typical messaging use cases as guidance for use with multi-rail configurations.

Table 4. Messaging Use Cases

Use Case	Guidance
MPI Communications Using PSM2	MPI over PSM2 library provides the best performance for MPI applications. Multi-rail is a feature that allows a process to use multiple HFIs to improve message bandwidth. The physical configuration of the servers as well as software settings influence the behavior of MPI communication. Two main aspects that affect application performance are *affinity* and *load balancing*—how tasks affinitize impacts load balancing. • When both rails are on the same CPU socket, they can share the PCI lanes and each task will affinitize to one of the two HFIs. • When each rail is on a different CPU socket, each task will affinitize to the local HFI. Because of task-to-HFI affinity, ranks cannot crosstalk with ranks on a different fabric in dual fabric configuration. The affinity of tasks to an HFI leads to certain considerations for load balancing. In a dual-rail, single fabric configuration, the HFI with primary affinity to the task is set as preferred to avoid cross-socket hops.

Use Case	Guidance
	At the MPI level, messages can be sent across both HFIs based on their MPI tag match information. A message below the eager limit is always sent out of an HFI selected based on tag information. Messages over the eager limit can be striped across the HFIs using rendezvous protocol. MULTIRAIL environment variables are provided to tune performance as explained in Section 4.4.1, General Information and Configuring for Non-CUDA* Applications.
Communication Over Verbs	An RDMA Verbs API provides support for communication based on Verbs over Omni Path. A typical use case is storage traffic. Both IBM Spectrum Scale and Lustre support multi-rail features. Guidance for their use is detailed under Section 4, Software Configuration. Use of Verbs for MPI communication is discouraged for performance reasons.
IP Over Fabric	IP over Fabric (IPoFabric) is a Cornelis Omni Path concept equivalent to IPoIB for sending IP traffic over InfiniBand. IPoFabric supports both datagram mode and connected mode of operation. The support for multi-queue network interface leverages existing OPA hardware features to accelerate parallel sends and receives. Cornelis Omni Path supports bonding across HFIs on a node. However, it is active-backup mode only. Dual rails do not participate in simultaneous data transfer; they only provide failover capability. Configuration guidance is provided under Section 4.5, IPoFabric Configuration.

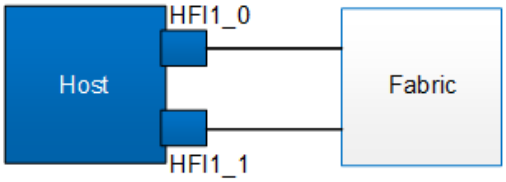
4 Software Configuration

4.1 Setting Up a Single Fabric

Once the HFIs are installed and the system is powered-on, the driver will detect whether there are one or two HFIs connected to the host on the same fabric. In most scenarios, no adjustments to `opafm.xml` file are required for this configuration.

Assumptions

- HFIs have been cabled as shown below:



No other configuration adjustments are required.

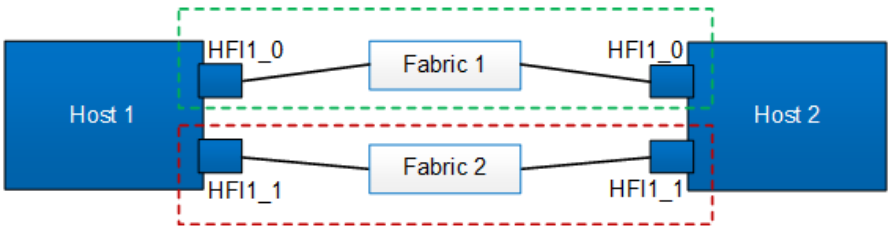
References

- *Cornelis Omni-Path Fabric Software Installation Guide*, "Setting Up Dual Rails for a Single Subnet"
- *Cornelis Omni-Path Fabric Setup Guide*, "Programming and Verifying Option ROM EEPROM Device"

4.2 Setting Up a Dual Fabric

Assumptions

- HFIs have been installed in the host servers.
- HFIs on all servers including the master and standby FMs have been cabled as shown in the figure below:



In multi-fabric implementations, applications need to know which fabric different HFIs are attached to. To accomplish this, you set a different `SubnetPrefix` for each fabric. An example for setting up dual fabrics is shown in the procedures below.

Procedures

Perform the following steps to set up the master and standby Fabric Managers:

1. Stop all standby FMs, then stop the master FM using `systemctl stop opafm`.
2. On the master FM host, open the `/etc/opa-fm/opafm.xml` file for editing.
3. Search for "`<Name>fm0`" to review the `fm0` settings.
4. Note the `<SubnetPrefix>` value for `fm0`. For example:

```
<SubnetPrefix>0xfe80000000000000</SubnetPrefix>
```

5. Search for "`<Name>fm1`" to review the `fm1` settings.
6. Note the `<SubnetPrefix>` value for `fm1`. For example:

```
<SubnetPrefix>0xfe80000000001001</SubnetPrefix>
```

7. Verify that `fm0` and `fm1` are configured with different `<SubnetPrefix>` values. In a multi-fabric environment, each FM instance must use a unique subnet prefix.
8. In the `fm1` settings section, configure the `fm1` instance of the Fabric Manager to start on the second HFI in the server:

```
<Start>1</Start>
<Name>fm1</Name>
<Hfi>2</Hfi>
<Port>1</Port>
```

9. Save the `opafm.xml` file.
10. Copy the `opafm.xml` file from the master FM host to the standby FM host.
11. Restart the master FM, then the standby FMs using `systemctl restart opafm`.
12. Run `systemctl status opafm` to verify that the FMs are running.

References

- *Cornelis Omni-Path Fabric Software Installation Guide*, "Configuring Dual Rails for Dual Subnets"
- *Cornelis Omni-Path Fabric Setup Guide*, "Programming and Verifying Option ROM EEPROM Device"

4.3 Modifying HFI1 Driver Module Settings

The hfi1 driver parameters `num_sdma` and `krcvqs` can be tuned to improve receive performance. For example, setting `krcvqs=2` on a dual-rail system will provide similar performance to `krcvqs=4` on a single-rail system, with four cores used for receiving in both scenarios. Likewise, when both HFIs are installed on the same NUMA, reducing the `num_sdma` from default 16 to 8 may reduce or even eliminate SDMA interrupt overlap based on the number of cores.

References

- *Cornelis Omni-Path Fabric Performance Tuning User Guide*, "MPI Performance"
- *Cornelis Omni-Path Fabric Host Software User Guide*
- *Cornelis Performance Scaled Messaging (PSM2) Programmer's Guide*

4.4 Configuring PSM2

4.4.1 General Information and Configuring for Non-CUDA* Applications

MPI and PSM2 are configured by default to take advantage of multiple HFIs to attempt to achieve the best performance. More advanced settings provide greater control over the way PSM2 uses the HFIs.

This section describes environment variables that can be used to provide finer control over how PSM2 can be configured to use multiple HFIs to achieve optimal performance.

The following table describes the environment variables that configure the PSM2 multi-rail behavior.

Table 5. PSM2 Environment Variables

Variable Name	Description
HFI_UNIT	This variable can be used to specify which HFI a PSM2 process uses to send and receive messages on the fabric. By default, HFI_UNIT is unset; MPI and PSM2 will try to utilize multiple HFIs to achieve the best performance. On a multi-rail node, when PSM2_MULTIRAIL is not enabled and HFI_UNIT is not set, PSM2 tries to select an HFI on the same NUMA node where the PSM2 process is running. If there are multiple PSM2 processes running on a node and multiple HFIs within a NUMA node, the PSM2 processes will try to use the MPI_LOCALRANKID environment variable to spread themselves across all available HFIs within the NUMA node. A user can also set HFI_UNIT to specify an HFI adapter to use, which will explicitly tell PSM2 which HFI adapter to use to send and receive data.

Variable Name	Description
	NOTE: When PSM2_MULTIRAIL is enabled, HFI_UNIT has no effect. In a dual fabric system, if PSM2_MULTIRAIL is not turned on, HFI_UNIT must be set to indicate which HFI to use. When context sharing is enabled on a system with multiple HFIs and the HFI_UNIT environment variable is set, the number of Cornelis Omni-Path contexts available to a PSM2 process (e.g., an MPI rank) is restricted to the number of contexts available on that unit.
PSM2_MULTIRAIL	PSM2 has the ability to split up messages across multiple HFI interfaces. This is referred to as message striping. When the PSM2_MULTIRAIL environment variable is set, PSM2 will stripe messages across the available HFIs. Using message striping can lead to an increase in aggregate bandwidth in many situations. By default, PSM2 does not stripe messages across multiple interfaces. It picks the first active HFI unit within the socket, regardless of the subnet the HFI is on. This behavior can be changed using the PSM2_MULTIRAIL environment variable according to the following options: • 0 PSM2 multi-rail striping capability disabled (default). • 1 Enable multi-rail capability and use all available HFI(s) in the system. • 2 Enable multi-rail within a single NUMA socket capability. More advanced settings can give a user greater control over the way PSM2 executes message striping. This is done by configuring the PSM2 Rendezvous protocol with respect to message sizes. This can be accomplished by modifying the PSM2_MQ_RNDV_HFI_WINDOW and PSM2_MQ_RNDV_HFI_THRESH environment variables, which are described further in later sections of this table. **NOTE:** In a dual fabric system, if PSM2_MULTIRAIL is not turned on, HFI_UNIT must be set to indicate which HFI to use. When PSM2_MULTIRAIL is turned on, PSM2 can reorder and automatically match the HFIs by using the subnet ID. For this to occur, the system administrator needs to configure each subnet to a different ID and then connect the HFIs to those subnets.

PSM2_MULTIRAIL_MAP	This variable overrides any auto-selection and affinity logic in PSM2, regardless of whether PSM2_MULTIRAIL is set to 1 or 2. The syntax is:
	`PSM2_MULTIRAIL_MAP = unit:port,unit:port,...`
	where `unit` (HFI) starts from 0 and `port` is always 1. If only one rail is specified, it is equivalent to a single-rail case.
	Example for dual-rail configuration:
	`PSM2_MULTIRAIL_MAP=1:1,0:1`
	where the first rail is assigned HFI1, port 1 and the second rail is assigned HFI0, port 1.

Variable Name	Description
PSM2_MQ_RNDV_HFI_WINDOW	This variable sets the windowing size in bytes for how PSM2 messages are split for transmission. When PSM2_MULTIRAIL is active, this value controls the granularity at which messages are striped between HFIs. Any value between 1 MB and 4 MB is valid; page aligned values work best. Note that messages will not be striped if the window size is lower than PSM2_MQ_RNDV_HFI_THRESH.
PSM2_MQ_RNDV_HFI_THRESH	This variable sets the eager-SDMA-to-rendezvous switchover threshold in bytes. Further details can be found in the Cornelis Networks Performance Scaled Messaging 2 (PSM2) Programmer's Guide. Messages larger than this threshold will use the PSM2 rendezvous protocol. Messages up to this threshold will be sent using the eager protocol. PSM2 only performs message striping for messages sent via rendezvous protocol. Therefore, you must configure PSM2_MQ_RNDV_HFI_WINDOW to be greater than or equal to PSM2_MQ_RNDV_HFI_THRESH for messages to be striped.

References

- *Cornelis Omni-Path Fabric Performance Tuning User Guide*, "MPI Performance"
- *Cornelis Performance Scaled Messaging 2 (PSM2) Programmer's Guide*

4.4.2 Configuring for CUDA and GPUDirect* Applications

The following lists the general best practices for running MPI-CUDA applications on OPA:

- You must use `libpsm2-cuda` and `hfi1-gpudirect` to get CUDA and GPUDirect support in the OPA stack. Use the `-G` flag with the IFS `INSTALL` script.
- You must use OpenMPI built with CUDA support. The OpenMPI installed when the IFS `INSTALL` script is run with `-G` has CUDA support.
- PSM2 can only handle one GPU per rank. If your node has multiple GPUs, you must use multiple ranks per node.
- `PSM2_CUDA=1` must be set in the ranks' environments.
- `PSM2_GPUDIRECT=1` must be set in the ranks' environments for the ranks to use GPUDirect in sends and receives through OPA.
- To enable GPUDirect if your node has PCI switches, your GPUs and HFIs should be on and under the same PCI switches.
- IOMMU (VT-d on some systems) should be disabled. Consult your BIOS documentation.

In addition to the above, consider the following recommendations when running MPI-CUDA jobs on multi-rail nodes:

- As with non-CUDA messages, only CUDA messages received through rendezvous are eligible to be striped across multiple HFIs on the same fabric.

- PSM2 has different thresholds for CUDA messages:
  - The PSM2 CUDA window threshold is 2 MiB. It is not adjustable.
  - The CUDA rendezvous threshold is 32768 bytes. It is not adjustable.
    - This is larger than the default maximum size at which PSM2 uses GPUDirect when sending CUDA messages. In this case, message striping and GPUDirect are mutually exclusive.
- For the MPI ranks running GPU workloads, the following two situations are possible:
  - If both HFIs are connected to the same socket as the GPU, set:
    - `PSM2_MULTIRAIL=2`, to stripe large messages across both HFIs
    - `PSM2_CUDA=1`, to enables CUDA support in PSM2
    - `PSM2_GPUDIRECT=1`, to enable NVIDIA GPUDirect RDMA in PSM2
  - If the two rails are connected to different sockets, set:
    - `PSM2_MULTIRAIL=0`
    - `HFI_UNIT = <HFI on the socket with GPU>`

Reference

- *Cornelis Omni-Path Fabric Performance Tuning User Guide*

4.5 IPoFabric Configuration

An OPA cluster using IPoFabric, can provide IP communication between nodes on the Cornelis Omni-Path network.

Note: The following information applies to dual-rail, single-fabric systems. For dual-rail, dual-fabric systems, each fabric is independent and will be configured in the same manner. For set up details, refer to the *Cornelis Omni-Path Fabric Software Installation Guide*.

It is possible to increase reliability of a single interface configuration by adding a second HFI card and using IPoIB bonding in active/standby mode. In the case of an adapter or cable failure, the traffic will use the secondary IPoFabric interface.

Reference

- *Cornelis Omni-Path Fabric Host Software User Guide*, "IPoIB Bonding"

4.6 IBM Spectrum Scale Considerations

In dual-rail, single-fabric systems using Spectrum Scale, use the `mmchconfig` command on nodes with dual interfaces to set the `verbsPorts` parameter for supporting file system traffic over both interfaces:

```
mmchconfig verbsPorts="hfi1_0 hfi1_1"
```

Note: The GPFS daemon needs to be restarted on the nodes for the change to take effect. For complete details, refer to Spectrum Scale command reference: https://www.ibm.com/docs/en/spectrum-scale/5.1.0?topic=reference-mmchconfig-command

In addition to using dual rails to increase throughput and reliability for bulk data transfers, Spectrum Scale often uses the IPoFabric address to perform initial connection, maintenance, and management tasks even though it uses RDMA/Verbs for the bulk data transfers. To increase the reliability of a dual-rail storage server, consider enabling IPoFabric bonding on the two interfaces and use the bonded address when adding the server to the Spectrum Scale configuration.

In dual-rail, dual-fabric systems, the `verbsPort` parameter needs to include the fabric number (`2` and `5` set by the Fabric Manager in the example below):

```
mmchconfig verbsPorts="hfi1_0/2  hfi1_1/5"
```

Reference

- *Cornelis Omni-Path Fabric Host Software User Guide*, "IPoIB Bonding" section

4.7 Lustre Considerations

As of Lustre version 2.10, the LNet Multi-Rail feature of Lustre, by default, load balances traffic across the active network interfaces. Lustre version 2.11 and newer has Dynamic Discovery that automatically detects and sets up multi-rail peers.

Note: The following information applies to dual-rail, single-fabric systems. For dual-rail, dual-fabric systems, each fabric is independent and will be configured in the same manner.

If you are using an earlier version of Lustre, additional manual setup is required.

- Hosts with dual HFIs will need to be configured as two active IPoFabric interfaces, like `ib0` and `ib1`.
- Then using either the static configuration method or the dynamic discovery method, add the hosts to the LNet configuration.

Note: The reliability of a dual-rail storage server can be increased by enabling IPoIB bonding on the two interfaces and using the bonded address when adding the server to the Lustre configuration.

References

- *Cornelis Omni-Path Fabric Performance Tuning User Guide*
- *Cornelis Omni-Path IP and LNet Router Design Guide*, "Network Interface Naming Consistency"
- *Cornelis Omni-Path IP and LNet Router Design Guide*, "Adding Static Routes"
- *Cornelis Omni-Path Host Software User Guide*, "IPoIB Bonding"

5 Conclusion

Using dual rail on nodes in a Cornelis Omni-Path system allows the performance of the interconnect to reach up to 200 Gbps capability. While it requires careful consideration with respect to selecting the system configuration and parameter settings, it is straightforward to use Multi-Rails to scale the available bandwidth of Cornelis Omni-Path based clusters.

Appendix A Frequently Asked Questions

Q. What is multi-HFI support in PSM2 and how does it differ from multi-rail?

A. Multi-HFI support is intended to describe the use of multiple HFIs in a system among MPI ranks local to a node to load-balance the hardware resources. It is different from the multi-rail feature that is intended to allow a single process to use all HFIs in the system. For an MPI job with multiple ranks in a node, the default PSM2 behavior depends on the affinity settings of the MPI process. PSM2 defaults to using the HFI that is in the same NUMA node as that of the MPI process. Users can restrict access to a single HFI using the environment variable:

HFI_UNIT=*N* (where valid values of *N* are 0, 1, 2, and 3)

Q. What are some guidelines for using CUDA and Open MPI with Cornelis Omni-Path?

A. When developing CUDA-aware Open MPI applications for OPA-based fabrics, the PSM2 transport is preferred and a CUDA-aware version of PSM2 is provided with all versions of the Cornelis Omni-Path IFS software.

The PSM2 library provides a number of settings that will govern how it will interact with CUDA, including PSM2_CUDA and PSM2_GPUDIRECT, which should be set in the environment before `MPI_Init()` is called. For example:

```
$ mpirun -x PSM2_CUDA=1 -x PSM2_GPUDIRECT=1 --mca mtl psm2 mpi_hello
```

In addition, before calling `MPI_Init()`, each process of the application should select a specific GPU card to use by setting `cudaChooseDevice()`, `cudaSetDevice()`, and similar. The chosen GPU should be within the same NUMA node as the CPU the MPI process is running on. You will also want to use the `mpirun --bind-to-core` or `--bind-to-socket` option to ensure that MPI processes do not move between NUMA nodes.

Q. We have a cluster with two HFIs installed on each server. What should we do to enable both cards and how can we check that both are being used?

A. The HFIs are automatically enabled by the driver; there is nothing further you need to do.

To verify that both HFIs in a server are enabled and working, use the `opainfo` command. A server with two HFIs will have two entries in `opainfo`, each showing details about the HFI's device name, port status, and data counters.

To verify that both cards are being used by an application, you may check the transmit and receive data counters in the output of either `opainfo` or `opapmaquery` before and after running the application.

Tips:

- To view counters: `opapmaquery -h <HFI number>`
- To clear counters: `opapmaquery -o clearportstatus -h <HFI number>`

For example, to clear performance counters on the second HFI in the server, use `opapmaquery -o clearportstats -h 2`.

Another suggestion to help verify that both cards are being used is to run your application on each card individually before running on both together. For example, for an Open MPI command:

```
mpirun -mca btl_openib_if_include=hfi1_0 a.out
mpirun -mca btl_openib_if_include=hfi1_1 a.out
```

If both commands work individually and the expected data counters increment for each card, then that is a good indication that they both work together in a dual rail configuration.

When an application is successfully using dual HFIs, you may also see better performance. In the case of MPI, you will see performance improvements for larger messages (> 64kB).

Q. I am getting lower performance than I expected. Why?

A. Tuning application performance accurately is an extremely challenging task, especially with fast machines and networks. Many, many factors need to be considered; it is not as simple as just compiling and running a stock benchmark application. For numerous suggestions on benchmarking performance, be sure to read https://www.open-mpi.org/faq/?category=tuning#running-perf-numbers.

Also refer to the *Cornelis Omni-Path Fabric Performance Tuning User Guide*.

Pay particular attention to the discussion of processor affinity and NUMA systems. Running benchmarks without processor affinity and/or on CPU sockets that are not directly connected to the bus where the HCA is located can lead to confusing or misleading performance results.

Q. Should both HFIs be on the same NUMA node?

A. This will vary across different node configurations. It is preferable to put the HFI cards on different sockets/NUMA nodes See the "MPI Communications Using PSM2" use case in Table 4 for additional information.

Q. What is the maximum number of HFIs that can be used in a server?

A. The maximum number of HFIs depends on the number of PCI slots available.

Q. Can any application take advantage of Multi-Rail?

A. Yes, provided the application can take advantage of the additional bandwidth the Multi-Rail configuration provides.

Q. How do I make my application use Multi-Rail?

A. See Sections 3 and 4 of this document. If additional support is needed, please contact Cornelis Networks Customer Support.

Q. Can I use the Multi-Rail concept with Virtual Machines? If so, how is this configured?

A. Yes. No special configuration is needed for virtual machines.

Q. Given a particular workload and application, would I benefit by implementing Multi-Rail?

A. Possibly. Please contact Cornelis Networks Customer Support to obtain guidance in making this decision.

Q. How do I know my Multi-rail configuration is working correctly?

A. A simple method is to run the following command on the node you want to check, before and after you run a program. The output will appear as shown in the format below.

```
> opainfo -o stats | egrep "Pkts|hfi" | grep -v MC
hfi1_0:1
           Xmit Pkts                          <value1>
           Rcv Pkts                           <value2>
hfi1_1:1
           Xmit Pkts                          <value3>
           Rcv Pkts                           <value4>
```

When both HFI cards are used, the after values will be higher than the before values.

Q. What kind of performance benefit can I expect to see with the implementation of Multi-Rail?

A. Multiple host interfaces can potentially provide higher throughput and lower message latency. The degree to which this can be achieved in a specific installation depends on many factors like configuration of the nodes, the cluster topology, and the communication patterns of applications running on the cluster. For example, when the network is designed to deliver 2*100 Gbps with dual rails, the applications need to leverage the higher bandwidth available. Conversely, for applications that are limited by a single 100 Gbps interface, a second interface can help.

Q. Are there any system tunings to make Multi-rail work better?

A. Refer to previous sections of this document and this Question for details on tuning in support of the Multi-Rail feature.