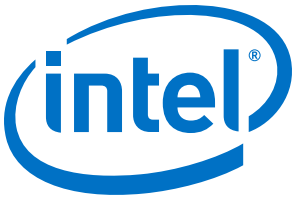


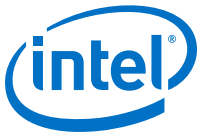
# Intel® Omni-Path Fabric Suite Fabric Manager

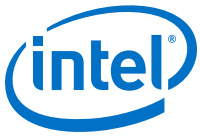
## User Guide

***Rev. 8.0***

***October 2017***

Order No.: H76468-8.0



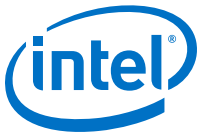


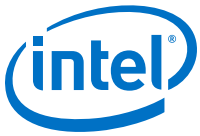
# Revision History

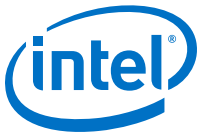
For the latest documentation, go to http://www.intel.com/omnipath/FabricSoftwarePublications.

| Date | Revision | Description |
|---|---|---|
| October 2017 | 8.0 | Updates to this document include:<br>• Added a new parameter, CumulativeTimeoutLimit, to Table 19. SM Fabric Sweep Parameters.<br>• The *Intel® Omni-Path Fabric Suite FastFabric Command Line Interface Reference Guide* has been merged into the *Intel® Omni-Path Fabric Suite FastFabric User Guide*. In this document, all references have been updated appropriately. See the Intel® Omni-Path Documentation Library for details. |
| August 2017 | 7.0 | Updates to this document include:<br>• Updates Pre-Defined Topology Enforcement Level details in Section 2.3.2.4 and Table 7. SM Unicast Routing Parameters.<br>• Updated references to host tools in Section 6.5.1 Device Membership and Security.<br>• Updated MGID details in Section 6.5.8 Pre-Created Multicast Groups.<br>• Clarified limits for Fabric Manager supported configuration of ports and switch ASICs in Section 1.5.2, Section1.5.3, and Table 1. |
| April 2017 | 6.0 | Updates to this document include:<br>• Updated Preface to include a new section "Intel® Omni-Path Documentation Library".<br>• Changed file paths throughout the document to indicate updates to file structures.<br>• Added "Host-Based Fabric Manager or Embedded Fabric Manager Recommendations" in Section "Choosing Between Host and Embedded FM Deployments".<br>• Updates to Table 7. SM Unicast Routing Parameters and Table 16. SM Adaptive Routing Parameters.<br>• Updates to Appendix M Best Practices for ESM Logging. |
| December 2016 | 5.0 | Updates to this document include:<br>• Changed file paths from `.../opt/...` to `.../usr/lib/...`<br>• Added clarifications to Fabric Unicast Routing for Fat Tree usage with respect to core switches<br>• Added Appendix "PM Counters Calculations"<br>• Added new section "Software Architecture Overview" |
| August 2016 | 4.0 | Updates to this document include:<br>• Changed name of ports counter from `SwPortCongestion` to `CongDiscards`<br>• Changed name of ports counter from `SwPortVLCongestion` to `VLCongDiscards`<br>• Removed references to PGI MPIs from Appendix "INSTALL Command" |
| May 2016 | 3.0 | Updates to this document include:<br>• Minor updates to Table 29. PM IntegrityWeights Parameters |
| February 2016 | 2.0 | Updates to this document include:<br>• Minor updates to `StorageLocation` PM Parameter |
| November 2015 | 1.0 | Initial release of this document. |

# Contents







# Figures

# Tables

# Preface

This manual is part of the documentation set for the Intel® Omni-Path Fabric (Intel® OP Fabric), which is an end-to-end solution consisting of Intel® Omni-Path Host Fabric Interfaces (HFIs), Intel® Omni-Path switches, and fabric management and development tools.

The Intel® OP Fabric delivers a platform for the next generation of High-Performance Computing (HPC) systems that is designed to cost-effectively meet the scale, density, and reliability requirements of large-scale HPC clusters.

Both the Intel® OP Fabric and standard InfiniBand* are able to send Internet Protocol (IP) traffic over the fabric, or *IPoFabric*. In this document, however, it is referred to as *IP over IB* or *IPoIB*. From a software point of view, IPoFabric and IPoIB behave the same way and, in fact, use the same `ib_ipoib` driver to send IP traffic over the ib0 and/or ib1 ports.

## Intended Audience

The intended audience for the Intel® Omni-Path (Intel® OP) document set is network administrators and other qualified personnel.

## Intel® Omni-Path Documentation Library

Intel® Omni-Path publications are available at the following URLs:

- Intel® Omni-Path Switches Installation, User, and Reference Guides

  http://www.intel.com/omnipath/SwitchPublications
- Intel® Omni-Path Software Installation, User, and Reference Guides (includes HFI documents)

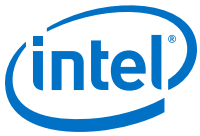
  http://www.intel.com/omnipath/FabricSoftwarePublications
- Drivers and Software (including Release Notes)

  http://www.intel.com/omnipath/Downloads

Use the tasks listed in this table to find the corresponding Intel® Omni-Path document.

| Task | Document Title | Description |
|---|---|---|
| | | |
| **Key:** Shading indicates the URL to use for accessing the particular document. | | |
| • Intel® Omni-Path Switches Installation, User, and Reference Guides: http://www.intel.com/omnipath/SwitchPublications | | |
| • Intel® Omni-Path Software Installation, User, and Reference Guides (includes HFI documents): http://www.intel.com/omnipath/FabricSoftwarePublications (no shading) | | |
| • Drivers and Software (including Release Notes): http://www.intel.com/omnipath/Downloads | | |

*continued...*



| Task | Document Title | Description |
|---|---|---|
| | | |
| Using the Intel® OPA documentation set | *Intel® Omni-Path Fabric Quick Start Guide* | A roadmap to Intel's comprehensive library of publications describing all aspects of the product family. It outlines the most basic steps for getting your Intel® Omni-Path Architecture (Intel® OPA) cluster installed and operational. |
| Setting up an Intel® OPA cluster | *Intel® Omni-Path Fabric Setup Guide* (Old title: *Intel® Omni-Path Fabric Staging Guide*) | Provides a high level overview of the steps required to stage a customer-based installation of the Intel® Omni-Path Fabric. Procedures and key reference documents, such as Intel® Omni-Path user guides and installation guides are provided to clarify the process. Additional commands and BKMs are defined to facilitate the installation process and troubleshooting. |
| Installing hardware | *Intel® Omni-Path Fabric Switches Hardware Installation Guide* | Describes the hardware installation and initial configuration tasks for the Intel® Omni-Path Switches 100 Series. This includes: Intel® Omni-Path Edge Switches 100 Series, 24 and 48-port configurable Edge switches, and Intel® Omni-Path Director Class Switches 100 Series. |
| | *Intel® Omni-Path Host Fabric Interface Installation Guide* | Contains instructions for installing the HFI in an Intel® OPA cluster. A cluster is defined as a collection of nodes, each attached to a fabric through the Intel interconnect. The Intel® HFI utilizes Intel® Omni-Path switches and cabling. |
| Installing host software<br>Installing HFI firmware<br>Installing switch firmware (externally-managed switches) | *Intel® Omni-Path Fabric Software Installation Guide* | Describes using a Text-based User Interface (TUI) to guide you through the installation process. You have the option of using command line interface (CLI) commands to perform the installation or install using the Linux* distribution software. |
| Managing a switch using Chassis Viewer GUI<br>Installing switch firmware (managed switches) | *Intel® Omni-Path Fabric Switches GUI User Guide* | Describes the Intel® Omni-Path Fabric Chassis Viewer graphical user interface (GUI). It provides task-oriented procedures for configuring and managing the Intel® Omni-Path Switch family.<br>Help: GUI online help. |
| Managing a switch using the CLI<br>Installing switch firmware (managed switches) | *Intel® Omni-Path Fabric Switches Command Line Interface Reference Guide* | Describes the command line interface (CLI) task information for the Intel® Omni-Path Switch family.<br>Help: -help for each CLI. |
| Managing a fabric using FastFabric | *Intel® Omni-Path Fabric Suite FastFabric User Guide* (Merged with: *Intel® Omni-Path Fabric Suite FastFabric Command Line Interface Reference Guide*) | Provides instructions for using the set of fabric management tools designed to simplify and optimize common fabric management tasks. The management tools consist of TUI menus and command line interface (CLI) commands.<br>Help: -help and man pages for each CLI. Also, all host CLI commands can be accessed as console help in the Fabric Manager GUI. |
| Managing a fabric using Fabric Manager | *Intel® Omni-Path Fabric Suite Fabric Manager User Guide* | The Fabric Manager uses a well defined management protocol to communicate with management agents in every Intel® Omni-Path Host Fabric Interface (HFI) and switch. Through these interfaces the Fabric Manager is able to discover, configure, and monitor the fabric. |
| | *Intel® Omni-Path Fabric Suite Fabric Manager GUI User Guide* | Provides an intuitive, scalable dashboard and set of analysis tools for graphically monitoring fabric status and configuration. It is a user-friendly alternative to traditional command-line tools for day-to-day monitoring of fabric health.<br>Help: Fabric Manager GUI Online Help. |

***continued...***

| Task | Document Title | Description |
|---|---|---|
| Configuring and administering Intel® HFI and IPoIB driver<br>Running MPI applications on Intel® OPA | *Intel® Omni-Path Fabric Host Software User Guide* | Describes how to set up and administer the Host Fabric Interface (HFI) after the software has been installed. The audience for this document includes both cluster administrators and Message-Passing Interface (MPI) application programmers, who have different but overlapping interests in the details of the technology. |
| Writing and running middleware that uses Intel® OPA | *Intel® Performance Scaled Messaging 2 (PSM2) Programmer's Guide* | Provides a reference for programmers working with the Intel® PSM2 Application Programming Interface (API). The Performance Scaled Messaging 2 API (PSM2 API) is a low-level user-level communications interface. |
| Optimizing system performance | *Intel® Omni-Path Fabric Performance Tuning User Guide* | Describes BIOS settings and parameters that have been shown to ensure best performance, or make performance more consistent, on Intel® Omni-Path Architecture. If you are interested in benchmarking the performance of your system, these tips may help you obtain better performance. |
| Designing an IP or storage router on Intel® OPA | *Intel® Omni-Path IP and Storage Router Design Guide* | Describes how to install, configure, and administer an IPoIB router solution (Linux* IP or LNet) for inter-operating between Intel® Omni-Path and a legacy InfiniBand* fabric. |
| Building a Lustre* Server using Intel® OPA | *Building Lustre* Servers with Intel® Omni-Path Architecture Application Note* | Describes the steps to build and test a Lustre* system (MGS, MDT, MDS, OSS, OST, client) from the HPDD master branch on a x86_64, RHEL*/CentOS* 7.1 machine. |
| Building Containers for Intel® OPA fabrics | *Building Containers for Intel® Omni-Path Fabrics using Docker* and Singularity* Application Note* | Provides basic information for building and running Docker* and Singularity* containers on Linux*-based computer platforms that incorporate Intel® Omni-Path networking technology. |
| Writing management applications that interface with Intel® OPA | *Intel® Omni-Path Management API Programmer's Guide* | Contains a reference for programmers working with the Intel® Omni-Path Architecture Management (Intel OPAMGT) Application Programming Interface (API). The Intel OPAMGT API is a C-API permitting in-band and out-of-band queries of the FM's Subnet Administrator and Performance Administrator. |
| Learning about new release features, open issues, and resolved issues for a particular release | *Intel® Omni-Path Fabric Software Release Notes* | |
| | *Intel® Omni-Path Fabric Manager GUI Release Notes* | |
| | *Intel® Omni-Path Fabric Switches Release Notes* (includes managed and externally-managed switches) | |

## Cluster Configurator for Intel® Omni-Path Fabric

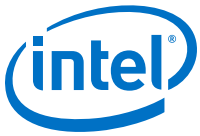
The Cluster Configurator for Intel® Omni-Path Fabric is available at: http://www.intel.com/content/www/us/en/high-performance-computing-fabrics/omni-path-configurator.html.

This tool generates sample cluster configurations based on key cluster attributes, including a side-by-side comparison of up to four cluster configurations. The tool also generates parts lists and cluster diagrams.

## Documentation Conventions

The following conventions are standard for Intel® Omni-Path documentation:

- *Note:* provides additional information.
- ***Caution:*** indicates the presence of a hazard that has the potential of causing damage to data or equipment.



- ***Warning:*** indicates the presence of a hazard that has the potential of causing personal injury.
- Text in blue font indicates a hyperlink (jump) to a figure, table, or section in this guide. Links to websites are also shown in blue. For example:

  See License Agreements on page 15 for more information.

  For more information, visit www.intel.com.
- Text in **bold** font indicates user interface elements such as menu items, buttons, check boxes, key names, key strokes, or column headings. For example:

  Click the **Start** button, point to **Programs**, point to **Accessories**, and then click **Command Prompt**.

  Press **CTRL+P** and then press the **UP ARROW** key.
- Text in `Courier` font indicates a file name, directory path, or command line text. For example:

  Enter the following command: `sh ./install.bin`
- Text in *italics* indicates terms, emphasis, variables, or document titles. For example:

  Refer to *Intel® Omni-Path Fabric Software Installation Guide* for details.

  In this document, the term *chassis* refers to a managed switch.

Procedures and information may be marked with one of the following qualifications:

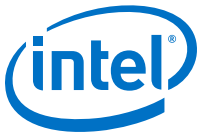
- **(Linux)** – Tasks are only applicable when Linux* is being used.
- **(Host)** – Tasks are only applicable when Intel® Omni-Path Fabric Host Software or Intel® Omni-Path Fabric Suite is being used on the hosts.
- **(Switch)** – Tasks are applicable only when Intel® Omni-Path Switches or Chassis are being used.
- Tasks that are generally applicable to all environments are not marked.

## License Agreements

This software is provided under one or more license agreements. Please refer to the license agreement(s) provided with the software for specific detail. Do not install or use the software until you have carefully read and agree to the terms and conditions of the license agreement(s). By loading or using the software, you agree to the terms of the license agreement(s). If you do not wish to so agree, do not install or use the software.

## Technical Support

Technical support for Intel® Omni-Path products is available 24 hours a day, 365 days a year. Please contact Intel Customer Support or visit http://www.intel.com/omnipath/support for additional detail.





# 1.0 Overview

This section gives an overview of the Intel® Omni-Path Fabric Suite Fabric Manager (FM). For details about the other documents for the Intel® Omni-Path product line, refer to Intel® Omni-Path Documentation Library on page 12 of this document.

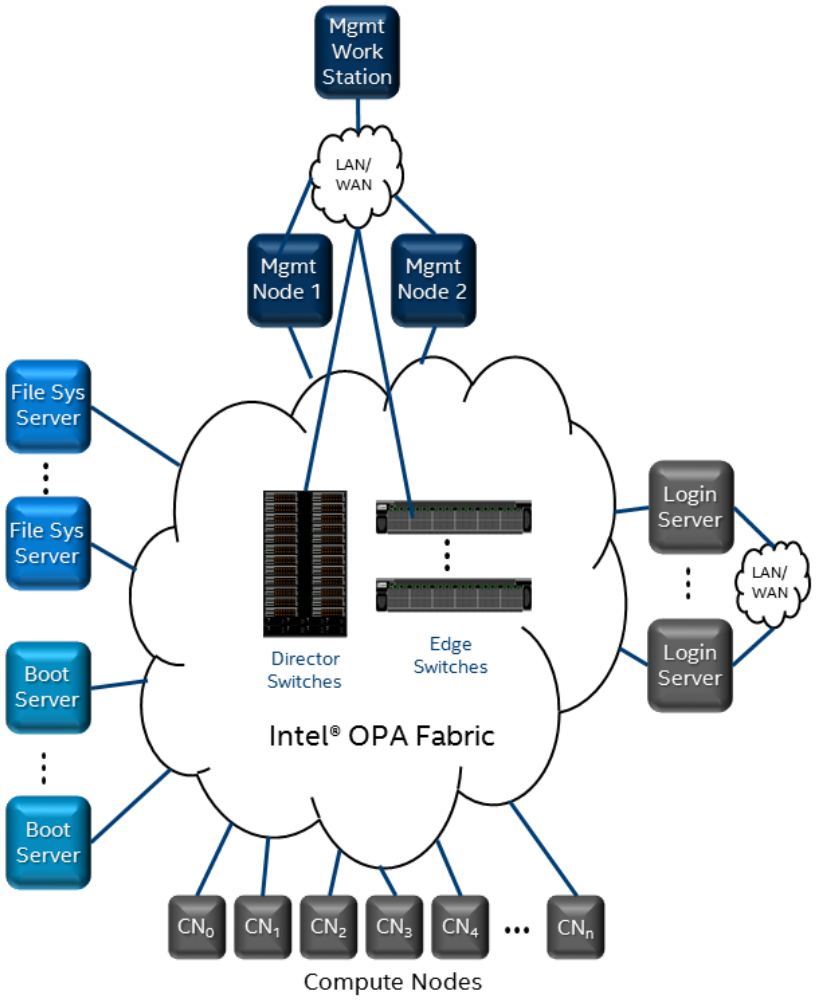
## 1.1 Intel® Omni-Path Architecture Overview

The Intel® Omni-Path Architecture (Intel® OPA) interconnect fabric design enables a broad class of multiple node computational applications requiring scalable, tightly-coupled processing, memory, and storage resources. Options for close "on-package" integration between Intel® OPA family devices, Intel® Xeon® Processors, and Intel® Xeon Phi™ Processors, enable significant system level packaging and network efficiency improvements. When coupled with open standard APIs developed by the OpenFabrics Alliance* (OFA) Open Fabrics Interface (OFI) workgroup, host fabric interfaces (HFIs) and switches in the Intel® OPA family systems are optimized to provide the low latency, high bandwidth, and high message rate needed by large scale High Performance Computing (HPC) applications.

Intel® OPA provides innovations for a multi-generation, scalable fabric, including link layer reliability, extended fabric addressing, and optimizations for many-core processors. High performance datacenter needs are also a core Intel® OPA focus, including link level traffic flow optimization to minimize datacenter-wide jitter for high priority packets, robust partitioning support, quality of service support, and a centralized fabric management system.

The following figure shows a sample Intel® OPA-based fabric, consisting of different types of nodes and servers.

**Figure 1. Intel® OPA Fabric**




To enable the largest scale systems in both HPC and the datacenter, fabric reliability is enhanced by combining the link level retry typically found in HPC fabrics with the conventional end-to-end retry used in traditional networks. Layer 2 network addressing is extended for systems with over ten million endpoints, thereby enabling use on the largest scale datacenters for years to come.

To enable support for a breadth of topologies, Intel® OPA provides mechanisms for packets to change virtual lanes as they progress through the fabric. In addition, higher priority packets are able to preempt lower priority packets to provide more predictable system performance, especially when multiple applications are running simultaneously. Finally, fabric partitioning is provided to isolate traffic between jobs or between users.

The software ecosystem is built around OFA software and includes four key APIs.

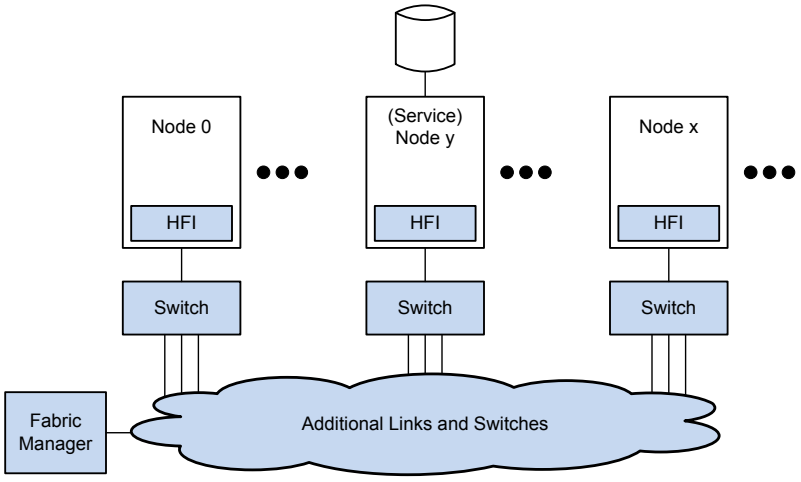
1. The OFA OFI represents a long term direction for high performance user level and kernel level network APIs.

2. The Performance Scaled Messaging 2 (PSM2) API provides HPC-focused transports and an evolutionary software path from the Intel® True Scale Fabric.
3. OFA Verbs provides support for existing remote direct memory access (RDMA) applications and includes extensions to support Intel® OPA fabric management.
4. Sockets is supported via OFA IPoFabric (also called IPoIB) and rSockets interfaces. This permits many existing applications to immediately run on Intel® Omni-Path as well as provide TCP/IP features such as IP routing and network bonding.

Higher level communication libraries, such as the Message Passing Interface (MPI), and Partitioned Global Address Space (PGAS) libraries, are layered on top of these low level OFA APIs. This permits existing HPC applications to immediately take advantage of advanced Intel® Omni-Path features.

Intel® Omni-Path Architecture is an end-to-end solution consisting of Intel® Omni-Path Host Fabric Interfaces (HFIs), Intel® Omni-Path switches, and fabric management and development tools. These building blocks are shown in the following figure.

**Figure 2. Intel® OPA Building Blocks**

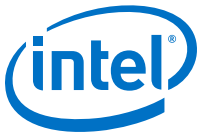



## 1.1.1 Host Fabric Interface

Each host is connected to the fabric through a Host Fabric Interface (HFI) adapter. The HFI translates instructions between the host processor and the fabric. The HFI includes the logic necessary to implement the physical and link layers of the fabric architecture, so that a node can attach to a fabric and send and receive packets to other servers or devices. HFIs also include specialized logic for executing and accelerating upper layer protocols.

## 1.1.2 Intel® OPA Switches

Intel® OPA switches are OSI Layer 2 (link layer) devices, and act as packet forwarding mechanisms within a single Intel® OPA fabric. Intel® OPA switches are responsible for implementing Quality of Service (QoS) features, such as virtual lanes, congestion management, and adaptive routing. Switches are centrally managed by the Intel® Omni-Path Fabric Suite Fabric Manager software, and each switch includes a



management agent to handle management transactions. Central management means that switch configurations are programmed by the FM software, including managing the forwarding tables to implement specific fabric topologies, configuring the QoS and security parameters, and providing alternate routes for adaptive routing. As such, all OPA switches must include management agents to communicate with the Intel® OPA Fabric Manager.

### 1.1.3 Intel® OPA Management

The Intel® OPA fabric is centrally managed and supports redundant Fabric Managers that manage every device (server and switch) in the fabric through management agents associated with those devices. The Primary Fabric Manager is an Intel® OPA fabric software component selected during the fabric initialization process.

The Primary Fabric Manager is responsible for:

1. Discovering the fabric's topology.
2. Setting up Fabric addressing and other necessary values needed for operating the fabric.
3. Creating and populating the Switch forwarding tables.
4. Maintaining the Fabric Management Database.
5. Monitoring fabric utilization, performance, and statistics rates.

The fabric is managed by sending management packets over the fabric. These packets are sent *in-band* (that is, over the same wires as regular network packets) using dedicated buffers on a specific virtual lane (VL15). End-to-end reliability protocols are used to detect lost packets.

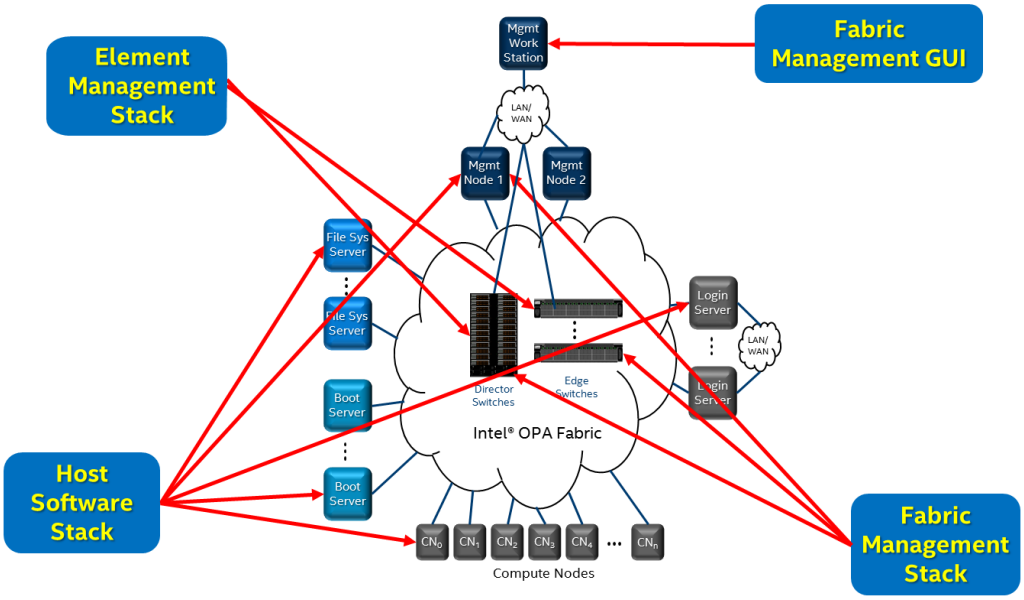
## 1.2 Intel® Omni-Path Software Overview

For software applications, Intel® OPA maintains consistency and compatibility with existing Intel® True Scale Fabric and InfiniBand* APIs utilizing the open source OpenFabrics Alliance* (OFA) software stack on Linux* distribution releases.

#### Software Components

The key software components and their usage models are shown in the following figure and described in the following paragraphs.

**Figure 3. Intel® OPA Fabric and Software Components**




| Software Component Descriptions |
|---|
| **Element Management Stack**<br>• Runs on an embedded Intel processor included in managed Intel® OP Edge Switch 100 Series and Intel® Omni-Path Director Class Switch 100 Series switches.<br>• Provides system management capabilities, including signal integrity, thermal monitoring, and voltage monitoring, among others.<br>• Accessed via Ethernet* port using command line interface (CLI) or graphical user interface (GUI).<br>User documents:<br>• *Intel® Omni-Path Fabric Switches GUI User Guide*<br>• *Intel® Omni-Path Fabric Switches Command Line Interface Reference Guide* |
| **Host Software Stack**<br>• Runs on all Intel® OPA-connected host nodes and supports compute, management, and I/O nodes.<br>• Provides a rich set of APIs including OFI, PSM2, sockets, and OFA verbs.<br>• Provides high performance, highly scalable MPI implementation via OFA, PSM2, and an extensive set of upper layer protocols.<br>• Includes Boot over Fabric mechanism for configuring a server to boot over Intel® Omni-Path using the Intel® OP HFI Unified Extensible Firmware Interface (UEFI) firmware.<br>User documents:<br>• *Intel® Omni-Path Fabric Host Software User Guide* |

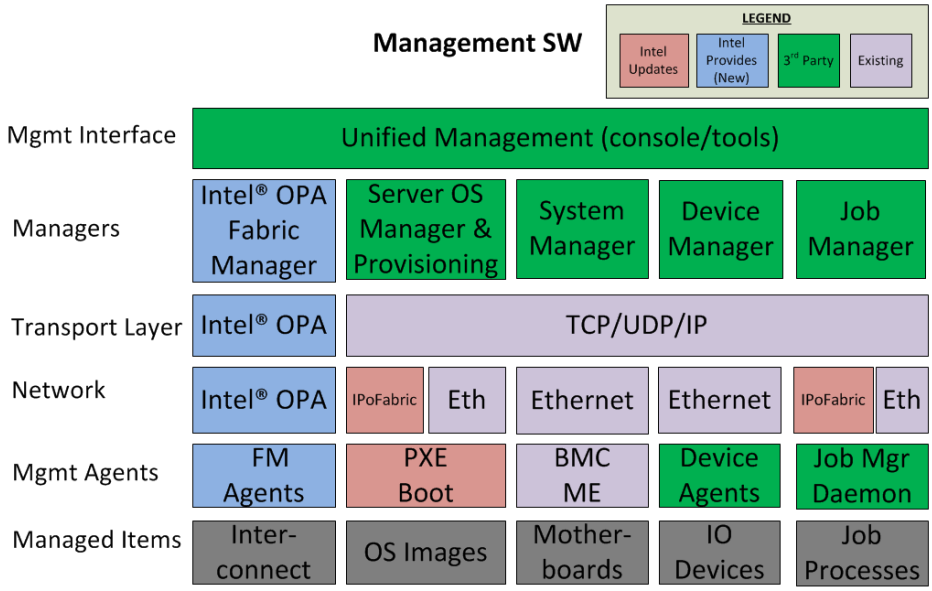
***continued...***

| Software Component Descriptions |
|---|
| • *Intel® Performance Scaled Messaging 2 (PSM2) Programmer's Guide* |
| **Fabric Management Stack**<br>• Runs on Intel® OPA-connected management nodes or embedded Intel processor on the switch.<br>• Initializes, configures, and monitors the fabric routing, QoS, security, and performance.<br>• Includes a toolkit for configuration, monitoring, diagnostics, and repair.<br>User documents:<br>• *Intel® Omni-Path Fabric Suite Fabric Manager User Guide*<br>• *Intel® Omni-Path Fabric Suite FastFabric User Guide* |
| **Fabric Management GUI**<br>• Runs on laptop or workstation with a local screen and keyboard.<br>• Provides interactive GUI access to Fabric Management features such as configuration, monitoring, diagnostics, and element management drill down.<br>User documents:<br>• *Intel® Omni-Path Fabric Suite Fabric Manager GUI Online Help*<br>• *Intel® Omni-Path Fabric Suite Fabric Manager GUI User Guide* |

## 1.3 Management Overview

Intel® Omni-Path Fabric Suite Fabric Manager is designed to provide interconnect management within the context of the overall cluster management solution.

**Figure 4. Management Overview**




As shown in the previous figure, a typical fabric has multiple areas of management to cover various aspects of the cluster. The main areas of management include:

- Interconnect Management
- Server OS Management & Provisioning

- System Management
- Device Management
- Job Management
- Overall Unified Management

### 1.3.1 Interconnect Management

Intel® Omni-Path Fabric Suite Fabric Manager along with the Intel® Omni-Path host stack and Intel® Omni-Path Fabric Suite FastFabric tools provide a comprehensive solution for managing the Intel® Omni-Path Fabric. The Fabric Manager and FastFabric tools are focused on managing the Layer 1-2 Switching Fabric. This includes link and cable management, switch management, fabric QoS, fabric security, and fabric routing. Intel® Omni-Path Fabric Management occurs in-band using Intel® Omni-Path-specific management packets. The Fabric Manager uses a well defined management protocol to communicate with management agents in every Intel® Omni-Path Host Fabric Interface (HFI) and switch. Through these interfaces the Fabric Manager is able to discover, configure, and monitor the fabric.

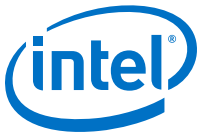
### 1.3.2 Server OS Management and Provisioning

In any cluster, the operating system (OS) versions, features, and configuration installed on each server must be managed. For small clusters, consisting of just a few servers, this may be performed in a more ad-hoc manner. However for larger clusters, there is typically a more unified set of tools to manage OS levels and OS configuration on all the servers. In larger clusters, diskless compute nodes are often used, in which case the Server OS Management subsystem also must be able to help boot the servers and provide the proper OS image to all the servers. This process is often referred to as provisioning and in some cases the OS image loaded on each server may vary depending on the job being run.

There is an existing rich ecosystem of third-party products, open source tools, and established best practices for managing the installation, configuration and boot of server OS images for clusters. As such, Intel® Omni-Path is designed to work seamlessly with this existing ecosystem.

To enable the use of disk-less Intel® Omni-Path servers, an Intel® Omni-Path BIOS option ROM (or integrated directly into the BIOS of some servers) is available. This permits the servers to be booted directly from the Intel® Omni-Path network using standard DHCP and PXE protocols running on top of the standard IPoFabric (a.k.a. IPoIB) protocol. As such, for those choosing to take advantage of the high performance of Intel® Omni-Path for server boot, existing provisioning and server OS management mechanisms based on these standards may be used. For more information, see "Appendix B Client/Server Configuration to Boot Over Fabric" in the *Intel® Omni-Path Fabric Software Installation Guide*.

Intel® Omni-Path also continues to permit out-of-band boot mechanisms, such as through a separate Ethernet* management network to continue to be used when desired.



### 1.3.3 System Management

For larger clusters, server and platform system management is also common. This is focused on managing motherboards and servers and may provide interfaces for controlling BIOS options, power and thermal monitoring and management, FRU (Field Replaceable Unit) inventory and management, and other platform-specific details.

Here too, there is an existing rich ecosystem. Many server vendors provide powerful features and innovative packaging solutions such as blades and other dense packaging solutions designed for large cluster deployments.

Intel® Omni-Path is designed to seamlessly support these existing solutions and can be easily plugged into existing standard servers. Typically these solutions use an out-of-band Ethernet* path to the servers, but it may also use various server vendor-specific solutions within the server platform or within the server rack. Many of these solutions take advantage of the numerous features of Intel server architecture such as the BMC and ME architectures.

### 1.3.4 Device Management

Most clusters have a number of special purpose devices and appliances. A typical example is high performance storage devices designed especially for larger clusters. Each of these devices has a management subsystem provided by the vendor that may integrate into existing device management solutions.

### 1.3.5 Job Management

The primary purpose of a cluster is to run jobs. On smaller clusters or clusters dedicated to running a single job, ad-hoc or manual mechanisms may be used to launch and monitor jobs, such as directly invoking `mpirun`. However on larger clusters or clusters that support multiple jobs and/or users, a job management solution is typically used. These solutions manage a queue of requested jobs, schedule the servers in the cluster to run jobs from the queue, monitor the jobs, and finally clean up after the jobs when they complete. Typical solutions use TCP/IP-based protocols to communicate between the job manager/scheduler and the actual job processes or servers through a job manager daemon. Sometimes the job manager daemon is part of the "runtime" used within the job middleware.

Intel® Omni-Path enables use of existing job management solutions for both in-band (using IPoFabric) and out-of-band (typically Ethernet*) management of jobs.

### 1.3.6 Unified Management

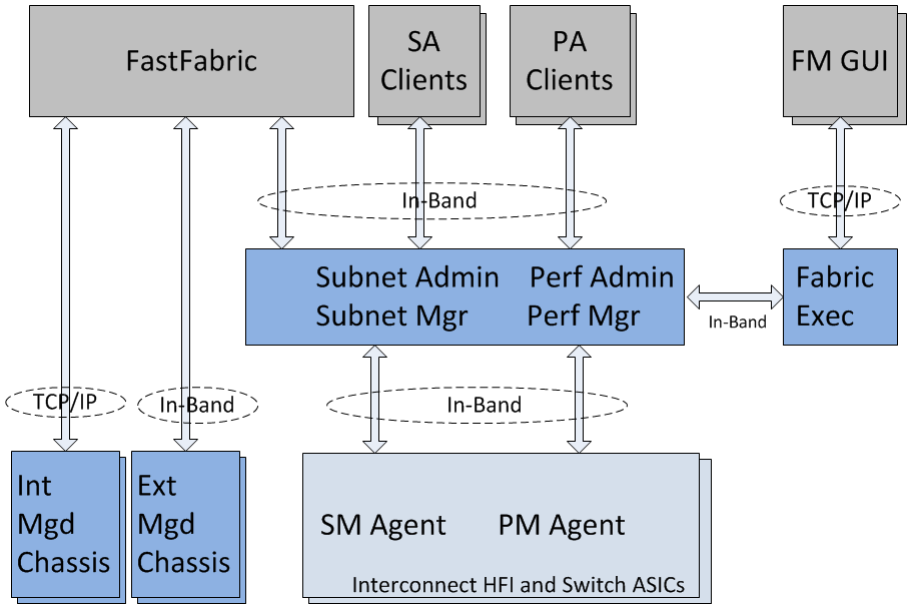
In larger clusters, there are advantages to having a single point of entry and control for all of these management areas. A variety of third-party solutions are available and in many cases these solutions include integration of one or more of the areas. For example, it is common to have job management, Server OS and System Management integrated through a unified manager so that server and OS status can be considered when launching and monitoring jobs.

The Intel® Omni-Path Fabric Suite Fabric Manager provides a number of interfaces and mechanisms that can permit easy integration of Intel® Omni-Path fabrics into this existing ecosystem.

## 1.4 Fabric Management Component Overview

The Fabric Manager architecture is shown in the following figure.

**Figure 5. Component Overview**




The FM is a set of components that perform various functions for the management of the fabric. These components consist of:

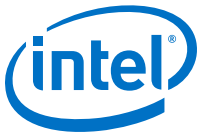
- Subnet Manager (SM)
- Subnet Administration (SA)
- Performance Manager (PM)
- Performance Administration (PA)
- Fabric Executive (FE)

These components interact with Intel® Omni-Path HFIs and switches in the fabric using in-band communications to Subnet Management (SM) agents and Performance Management (PM) agents in each Intel® Omni-PathIntel® Omni-Path device.

Various client applications such FastFabric, Open Fabrics address resolution and other SA and PA clients can interact in-band over the Intel® Omni-Path fabric with the SA and PA.

Out-of-band clients, such as the Fabric Manager GUI can interact over TCP/IP via the FE to access the SA and PA.

In addition, some chassis and firmware management functions are performed by FastFabric directly to internally and externally managed Intel® Omni-Path switching chassis.



### 1.4.1 Subnet Manager

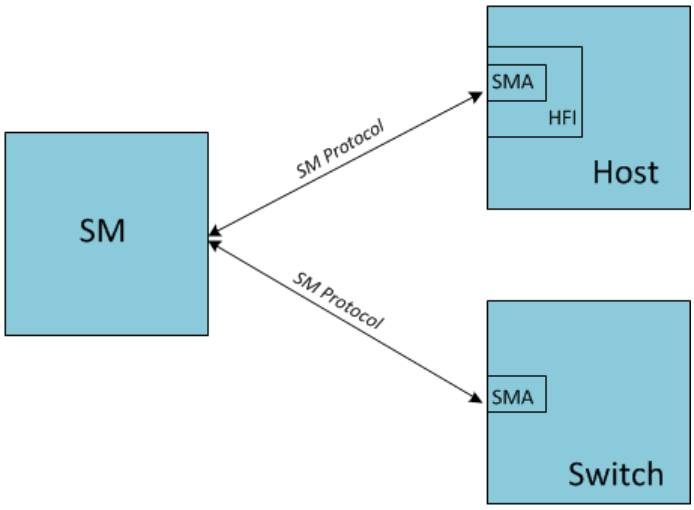
The FM implements a complete Intel® Omni-Path Architecture-compliant Subnet Manager (SM). The SM is responsible for monitoring, initializing and configuring the fabric.

The Subnet Manager component performs all of the necessary subnet management functions. Primarily, the SM is responsible for initializing the fabric and managing its topology. Some of its tasks include:

- Link and port initialization
- Link width downgrade policies
- Route and path assignments
- Local Identifier (LID) address assignments
- Switch forwarding table programming
- Programming Virtual Fabrics ( vFabrics)
  - Quality of Service (QoS)
  - Security
- Sweeping the fabric to discover topology changes, managing those changes when nodes are added and/or deleted
- Enforcing topology and security
- Configuring congestion handling
  - adaptive routing
  - congestion control
  - alternate routes for dispersive routing and failover
- Negotiating and synchronizing redundant SMs
- Arbitration for master and secondary roles between multiple SMs in the fabric

The SM performs all its subnet management and monitoring functions via in-band packets sent over the fabric being managed. The SM communicates with the Subnet Management Agent (SMA) on each node of the fabric, using the SMI packets and SMI protocol. Refer to the following illustration.

**Figure 6. FM Subnet Manager**




One of the critical roles of the SM is the initialization and configuration of routing tables in all the switches. The SM supports a variety of routing algorithms which are discussed in detail later in this guide. Among the capabilities are:

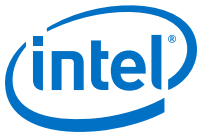
- Support for a wide range of fabric topologies, including Fat Tree, Clos network, and various irregular topologies
- Support for assigning multiple LIDs to end nodes and the carefully balanced programming of alternate routes through the fabric for use by dispersive routing, load balancing and failover techniques by various upper level protocols (ULP)
- Support for Intel® Omni-Path Architecture-compliant multicast, including MLID sharing, pre-creating groups, and other fine tuning of multicast performance
- Advanced monitoring and logging capabilities to quickly react to fabric changes and produce historical logs of changes and problems in the fabric
- Support for Virtual Fabrics with both QoS and partitioning capabilities
- Support for configuring and enabling adaptive routing in Intel® Omni-Path Switch 100 Series

### 1.4.2 Subnet Administration

The Subnet Administration function acts in tight coordination with the SM to perform data storage and retrieval of fabric information. The SM/SA is a single unified entity.

Through the use of SA messages, nodes on the fabric can gain access to fabric information such as:

- Node-to-node path information
- Fabric Topology and configuration
- Event notification
- Application Service information
- Join/Leave Multicast groups
- vFabric Information



- Fabric security configuration and lists of quarantined nodes.

Fundamental information required for address resolution and name services is available to all nodes. However more advanced information about fabric topology, status and configuration is available only to management applications on management nodes.

### 1.4.3 Performance Manager Overview

The Performance Manager component communicates with nodes to collect performance and error statistics. The PM communicates with the Performance Management Agent (PMA) on each node in the fabric, using the PM (GSI) packets.

Examples of the type of statistics collected by the PM include:

- Link Utilization Bandwidth
- Link Packet Rates
- Link Congestion
- Error statistics, such as packet discards, attempted security violations, packet routing errors, etc.

### 1.4.4 Performance Administration

The Performance Administration function acts in tight coordination with the PM to perform data storage and retrieval of fabric performance information. The PM/PA is a single unified entity.

Through the use of PA messages, management nodes on the fabric can gain access to fabric information such as:

- Fabric overall health
- Overall Fabric utilization, congestion, packet rates
- Fabric Error rates
- Traffic and congestion per vFabric
- Traffic and congestion over a PortGroup defined by the sysadmin
- Recent historical fabric performance data
- Sorted lists of ports with the highest utilization, packet rates, congestion, error rates, etc.
- Counters and status for a specific port

### 1.4.5 Fabric Executive

The Fabric Executive component provides for out-of-band access to the FM. It permits the Fabric Manager GUI utility and other tools running on devices not directly connected to the fabric, to communicate over TCP/IP and access the rest of the FM.

The FE exposes a private TCP/IP socket, on which it listens for connect requests from the Fabric Manager GUI applications. The Fabric Manager GUI communicates with the FE using the TCP/IP protocol to retrieve fabric information for presentation to the Fabric Manager GUI user.

The FE then uses in-band Intel® Omni-Path Technology packets to communicate with the other managers (SM, PM). The FE does not have to run on the same host as the other managers.

## 1.5 Embedded and Host FM Solutions

The FM can be deployed as either a host-based or an embedded solution. The host solution uses a Intel® Omni-Path Fabric stack and a Host Fabric Interface (HFI) to access, and manage the fabric. The embedded solution resides on an internally managed switch and uses a switch port to access and manage the fabric. Both solutions provide the set of FM components (SM, SA, PM, PA, FE) aforementioned, although in some cases the feature set and scalability of the FM components may be reduced when run in the embedded solution.

### 1.5.1 Host FM

The host FM deploys all FM components on a Linux server. The FM components are applications that run in the user space, and access the Intel® Omni-Path Architecture HFI using the management datagram (MAD) interface provided by the Open Fabrics Alliance stack.

The host FM can manage both small and large fabrics. It is mandatory for managing large fabrics, as the FM software is able to make use of the large memory resources and high speed processor technology of standard servers.

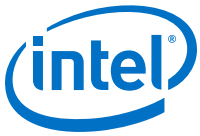
The host FM is installed onto a Linux system as part of the software installation. The utilities are installed into the `/usr/sbin` directory. The configuration file is installed as `/etc/opa-fm/opafm.xml`. It is started, restarted, and stopped using the opafm service (which standard Linux commands such as `systemctl` can use) with the `start`, `restart`, `stop`, or `reload` parameter, respectively.

The Intel® Omni-Path Fabric Suite FastFabric application can configure and control the host FM, query the SA, PA, and FE, and analyze fabric configuration and status. In addition, there are a few host FM control applications that are installed in `/usr/lib/opa-fm/bin` that are discussed later in this guide.

### 1.5.2 Embedded FM

The embedded FM deploys all of the FM components as an embedded solution in an internally managed switch. The maximum fabric configuration supported by the Embedded FM is 100 HFI ports involving less than 20 Switch ASICs. To determine the number of switch ASICs, count as follows: 1 ASIC per Edge switch, 2 ASICs per Director Class Chassis Leaf Module, and 2 ASICs per Director Class Chassis Spine Module. For the small-to-moderate fabrics, the embedded FM provides cost savings in that the fabric is able to utilize an existing resource for the embedded FM purposes.

The embedded FM is accessed through the switch CLI using commands in the Fm command group, as well as a few screens in the Intel® Omni-Path Fabric Chassis Viewer GUI (see *Intel® Omni-Path Fabric Switches GUI User Guide*). The majority of the configuration for the FM is accomplished using an `opafm.xml` file, which can be loaded onto the switch using the Intel® Omni-Path Fabric Chassis Viewer GUI or FastFabric.



The CLI contains many commands that provide access to configuration items and operational status. Both the web interface and CLI provide controls for the embedded FM to be started automatically at chassis boot time. Additionally, they provide the ability to start and stop the embedded FM.

### 1.5.3 Choosing Between Host and Embedded FM Deployments

A host base FM is required for fabrics greater than 100 HFI ports involving more than 20 switch ASICs.

**Table 1. HSM versus ESM Major Capability Differences**

| Capability | HSM | ESM |
|---|---|---|
| Maximum Fabric Node Size | Greater than 100 HFI ports | 100HFI ports or less |
| PA Short Term History | Supported | Not Supported - disabled |
| Dynamic Configuration | Supported | Not Supported - disabled |
| FM Control User Interface | opafmcmd CLI | Chassis CLI |
| FE OOB Security | | |
| File directory location: | Configurable | /mmc0:4 |
| Ciphers Support: | • ECDHE-ECDSA-AES128-GCM-SHA256<br>• ECDHE-ECDSA-AES128-SHA256<br>• DHE-DSS-AES256-SHA | DHE-DSS-AES256-SHA |
| Pre-defined Topology Verification | Supported | Not Supported - disabled |

#### Host-Based Fabric Manager or Embedded Fabric Manager Recommendations

Both fabric managers provide full functionality and the host-based fabric manager can be used in any scenario. The embedded fabric manager is implemented on lightweight internal hardware for smaller fabrics. As a result, the following recommendations exist.

- Director Class Switch configurations - A host-based subnet manager is required. Host-based fabric managers run on a minimally configured server. For redundancy, two or more servers can run the host fabric manager simultaneously.
- Managed Edge switches - For fabrics of more than 100 nodes, which typically occur with three or more 48-port switches in a fabric, a host-based fabric manager is suggested.

| | Host-Based Fabric Manager Supports | Embedded Fabric Manager Supports |
|---|---|---|
| HFIs back-to-back | Yes | Not Applicable |
| 48-port Edge switches - less than 100 nodes | Yes | Yes |
| 48-port Edge switches - greater than 100 nodes | Yes | No |
| 192 or 768 Director Class switch | Yes | No |
| Director Class switch + Edge switches | Yes | No |

## 1.6 Intel® Omni-Path Fabric Suite Fabric Manager GUI and Fabric Manager

The Fabric Manager GUI, a Java application that runs under the Windows* or Linux* operating system, communicates with the FM suite using the Fabric Executive (FE). The Fabric Manager GUI prompts the user for the IP address of the subnet, then opens a TCP/IP connection with the FE. The FE has access to the other FM components, namely the SM/SA and PM/PA. In this way it is able to service requests from the Fabric Manager GUI and furnish relevant information to the Fabric Manager GUI about the fabric.

- The Fabric Manager GUI issues requests to the SM/SA to gather information about the subnet, such as topology and node information.
- The Fabric Manager GUI issues requests to the PM/PA to gather performance and error statistics to permit fabric monitoring.

Refer to the *Intel® Omni-Path Fabric Suite Fabric Manager GUI Online Help* for more information.

## 1.7 Fabric Sweeping

The SM periodically sweeps the fabric. During a sweep the fabric is analyzed, routes are computed, and switches and Host Fabric Interfaces are configured. The SM sweep algorithms attempt to:

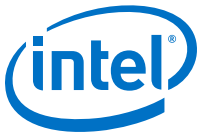
- Balance responsiveness to fabric changes while limiting fabric overhead of the SM
- Efficiently analyze and configure the fabric
- Detect and handle potential hardware limitations
- Handle the possibility that fabric is changing while being analyzed/configured

The SM performs sweeps at fixed intervals. The SM also immediately performs a sweep when a switch reports a port state change trap. Such traps indicate a link has come up or down. Generally traps trigger rapid sweeps to respond to fabric changes. The fixed sweeps are a safety net in case traps are lost or there are switches whose attempts to send traps are failing. To limit overhead, the SM fixed sweep is very slow (default of 5 minutes).

During a sweep, the SM must query and configure many devices in the fabric. These operations occur using the SM protocol over VL15. VL15 may optionally be configured to use flow control, but defaults to non-flow controlled. This combined with the fact that the fabric could be changing while the SM is sweeping, means that packets may be lost during the sweep.

To optimize the performance of the sweep, the SM can issue multiple concurrent packets to a given device or the fabric. The number to issue at once must be carefully balanced between the capabilities of the hardware and the goal of making the sweep faster.

The characteristics of the SM sweep are configurable.



## 1.8 Redundant FMs in a Fabric

It is possible to deploy more than one FM in a fabric. Normally deploying more than one FM is done to provide redundancy. Multiple FMs are a way to ensure that management coverage of the fabric continues unabated in the case of the failure of one of the FMs or the platform on which it is being run.

*Note:* It is important that the fundamental parameters, such as the SubnetPrefix, match in all SMs managing a fabric. Intel also recommends that all redundant SMs are at the same revision level. Failure to follow these recommendations can result in fabric disruptions when SM failover occurs, or if an SM fails to come online. Using the Configuration Consistency Checking feature, the FM can ensure that all redundant FMs have a comparable configuration. See Fabric Manager Configuration Consistency on page 76 for more information.

### 1.8.1 FM State – Master and Standby

When multiple FMs are present in a fabric, one takes on the role of master FM through arbitration. All other FMs take on the role of standby FM and use SM-to-SM communication to monitor the presence of the master FM.

If your intended master FM starts and reports that it is _not_ master check for other FMs running with the `opareport` command.

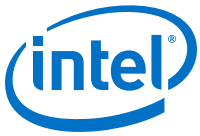
If this command fails to work it may mean that an embedded FM has taken over as master but can not properly run the fabric. In this case check all managed switches to ensure that embedded FMs are not running on them. See Choosing Between Host and Embedded FM Deployments on page 29 for more information on when to chose an embedded FM vs host FM.

## 1.9 Terminology Clarification – "FM" vs. "SM"

In written documentation and verbal communication, there often arises confusion between the terms "FM" and "SM". In some cases they may seem synonymous, while in others they denote distinct differences. Here are some guidelines for the usage of the terms:

- In general, SM refers specifically to the Subnet Manager, while FM refers to the Fabric Manager, consisting of SM/SA, PM/PA, and usually FE.
- When speaking of the product or set of running software processes, SM and FM may be interchangeable, due to the fact that the software is bundled and installed/operated together. Therefore, a phrase such as "start the SM" means to start all of the FM processes.

The scope of the terms is generally interpreted by the context in which it is being used. Therefore, if the specific managers are being discussed, then "FM" means the suite. Likewise, if general fabric management is being considered, then "SM" is being used to specify all of the managers.





# 2.0 SM Features

## 2.1 Virtual Fabrics Overview

Virtual Fabrics (vFabrics) bring to the Intel® Omni-Path Fabric many of the capabilities of Ethernet VLANs and Fibre Channel Zoning.

Using vFabrics, the administrator may slice up the physical fabric into many overlapping virtual fabrics. The administrator's selections determine how the Intel® Omni-Path Technology-specific configuration of the fabric is performed.

The goal of vFabrics is to permit multiple applications to be run on the same fabric at the same time with limited interference. The administrator can control the degree of isolation.

As in Ethernet VLANs, a given node may be in one or more vFabrics. vFabrics may have overlapping or completely independent membership. When IPoIB is in a fabric, typically each vFabric represents a unique IP subnet, in the same way a unique subnet is assigned to different virtual LANs (VLANs).

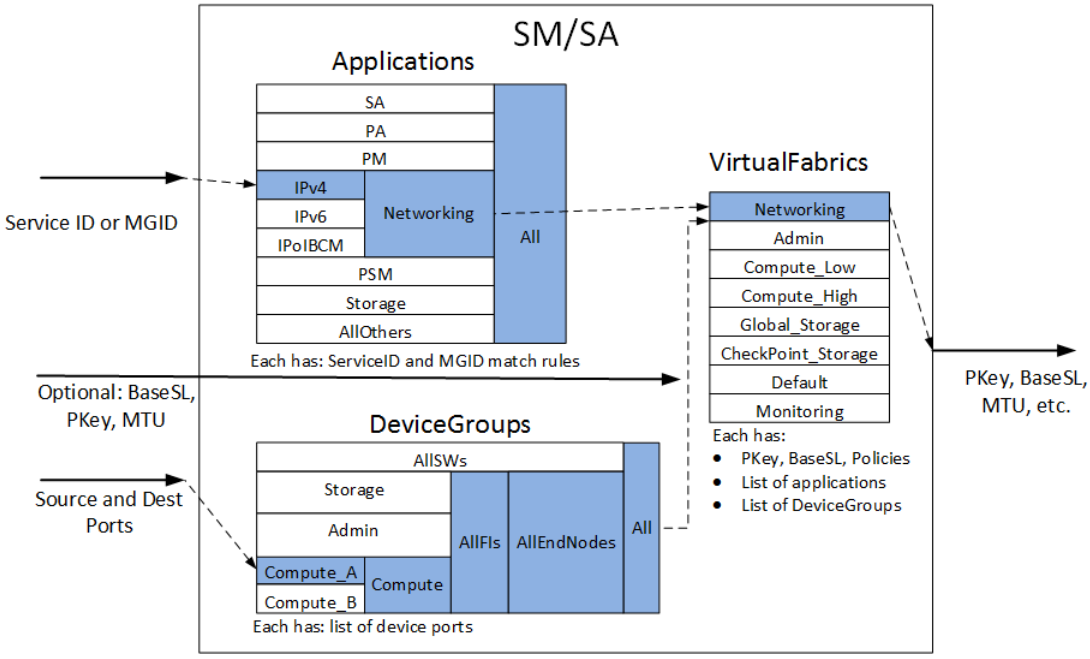
Each vFabric can be assigned Quality of Service (QoS) and security policies to control how common resources in the fabric are shared among vFabrics.

Additionally, as in Fibre Channel Zoning, vFabrics can be used in an Intel® Omni-Path Fabric with shared I/O and storage. vFabrics help control and reduce the number of IO devices visible to each host, and prevent hosts from communicating with each other using storage virtual fabrics. This can further secure the fabric and in some cases, may make making Plug and Play host software and storage-management tools easier to use. Some typical usage models for vFabrics include the following:

- Separating a cluster into multiple vFabrics so that independent applications can run with minimal or no effect on each other.
- Separating classes of traffic. For example, putting a storage controller in one vFabric and a network controller in another, using one wire for all networking becomes secure and predictable.

A vFabric consists of a group of applications that run on a group of devices. For each vFabric the operational parameters of the vFabric can be selected.

**Figure 7. Virtual Fabrics Overview**




As shown in the previous figure, the virtual fabrics definition involves defining a set of applications, device groups, and virtual fabrics. Each application is defined as a set of ServiceIDs and/or Multicast Group IDs (MGIDs). These are defined in terms of pattern match rules (or explicit lists), so that the wide range of 64- and 128-bit inputs can be easily specified.

Applications may also contain lists of other applications. This can permit the definition of higher level applications (such as Networking in the previous example) to reuse existing definitions for lower level applications (such as IPv4, IPv6, and IPoIBCM in the previous example).

Each DeviceGroup consists of a list of device ports. Those device ports may be explicitly listed or matched via Node Description pattern matching. DeviceGroups may also contain lists of other DeviceGroups. This can permit the definition of higher level groups (such as Compute in the previous example) to reuse existing definitions for lower level groups (such as Compute_A and Compute_B in the previous example).

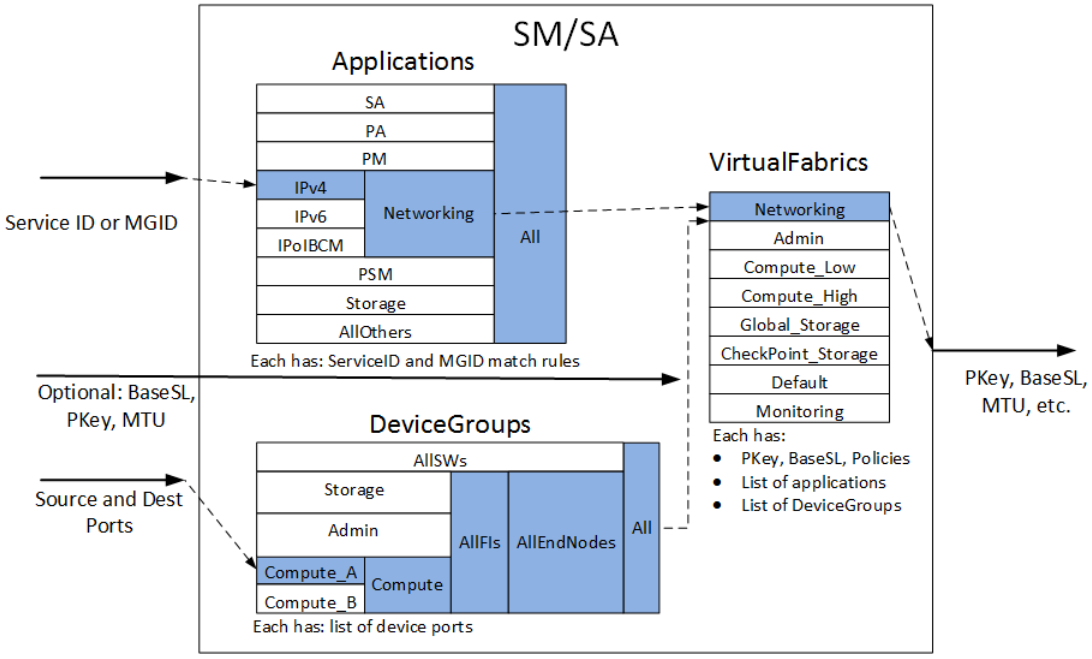
Finally, a Virtual Fabric is defined as a list of applications, a list of device groups, with a set of QoS and security policies, along with the `PKey` and `BaseSL` identifiers for the VirtualFabric.

Within the Open Fabrics Alliance APIs, packets for a given virtual fabric are identified by a `PKey` (indicating security) and a `BaseSL` (indicating QoS). During fabric initialization, the FM will have configured the devices to be aware of these two identifiers and assign the appropriate security (for example, limited/full /none membership) and QoS (for example, MTU, bandwidth, priority, preemption/traffic flow optimization rank, static rate, flow control, HoqLife) policies to these identifiers at each port in the fabric. The settings for these policies may also influence other parameters for the port such as how buffers, VLs, and other resources are allocated to each virtual fabric.

The virtual fabrics mechanisms are closely tied to address resolution and multicast membership in the Open Fabrics Alliance APIs and applications.

When an application makes an address resolution or multicast membership query to the FM, the FM checks the application identifier (ServiceId or Multicast Group ID (MGID)) against the FMs list of applications to identify one or more potential matches. It also checks the source and destination nodes against the FMs list of DeviceGroups to identify one or more potential matching device groups. The FM then looks for virtual fabrics, which include both a matching application and a matching device group and that also match any supplied `BaseSL`, `PKey`, and `MTU`. Then one or more resulting virtual fabrics are used to compose address resolution responses which include the `PKey`, `BaseSL`, and other communications parameters (`MTU`, `StaticRate`, `PktLifeTime`) which the application should use to communicate through the fabric to the identified destinations.

**Figure 8. Example of Virtual Fabrics Resolution**




The previous figure shows an example of the resolution mechanism during address or multicast resolution by SM/SA. In this example a ServiceID (address resolution) or an MGID (multicast join/create) is supplied that matches the IPv4 Application. The IPv4 Application is part of the Networking and All applications, so they also are matched. Also a source and destination are supplied explicitly or implicitly and a DeviceGroup is selected that matches both. In this example, the Compute_A group includes both the source and destination. The Compute_A group is also included in the Compute, AllFIs, AllEndNodes, and All groups, so they also are matched.

*Note:* Depending on the definition of Applications and DeviceGroups, it is possible for a given input to match two or more disjointed entries.

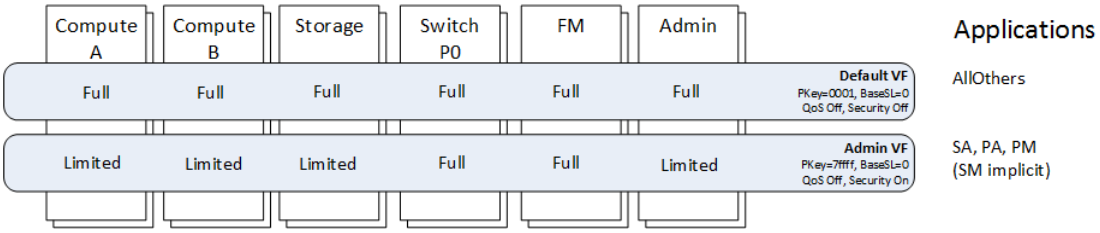
Given the list of matched Applications and matched DeviceGroups, the SM/SA searches the Virtual Fabrics list for an entry that includes both a matched Applications and a matched DeviceGroups. In this example, the Networking VirtualFabric entry

includes the Networking Application and the All DeviceGroup. If supplied, additional input parameters (`BaseSL`, `MulticastSL`, `PKey`, `MTU`) are also compared to the VirtualFabric to confirm a match. Finally, the response includes the `PKey`, `BaseSL`, `MTU`, and other relevant polices from the matched VirtualFabric.

For more information on how address resolution or multicast membership mechanisms in various applications can be integrated with virtual fabrics see Integrating Job Schedulers with Virtual Fabrics on page 179 and Integrating Other Service Applications with Virtual Fabrics on page 181.

Some of the concepts of virtual fabrics are best explained using some simple sample configuration examples.

**Figure 9. Default Virtual Fabrics Configuration**



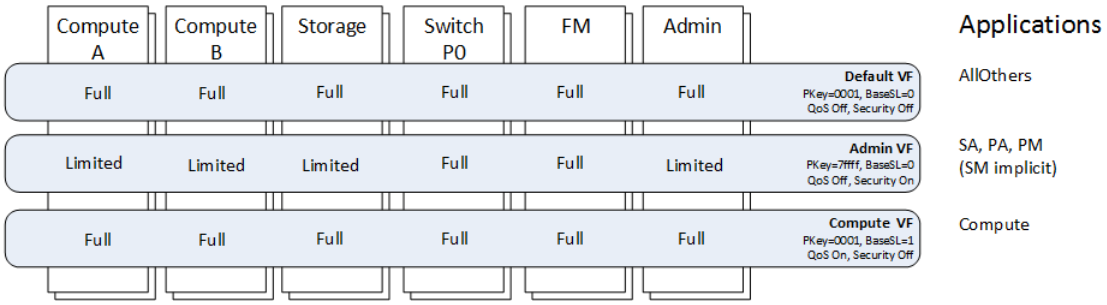

The previous figure shows the default FM configuration of virtual fabrics, including a series of devices, compute nodes A, compute nodes B, storage, switch Port 0s, FMs and other Admin nodes (such as job schedulers). In a given cluster there may be many of each type of device. Horizontally, two virtual fabrics are shown, Default VF and Admin VF. For each of these virtual fabrics a few of the key parameters are indicated, `PKey`, `BaseSL`, `QoS` on/off, and `Security` on/off. Parameters such as the `PKey` and `BaseSL` can be explicitly specified or dynamically assigned by the FM. Lastly on the right are shown the Applications that are associated with each virtual fabric.

In this example, the Default virtual fabric consists of all devices and "All Other" applications (in the default `opafm.xml` file, `AllOthers` is defined as containing `UnmatchedServiceID` and `UnmatchedGID`; see Table 38 on page 125 for more information on `UnmatchedServiceID` and `UnmatchedGID`). This virtual fabric will be used by the majority of applications and is assigned a PKey 0x0001 and BaseSL 0.

This example also shows the Admin virtual fabric. This fabric is used for secure fabric management of the fabric (SA, PA, PM, and implicitly the SM). It includes all the FM nodes and all the switch port 0s (which may also manage devices inside their given switch chassis). All other nodes are limited members. This virtual fabric is assigned PKey 0x7fff and BaseSL 0.

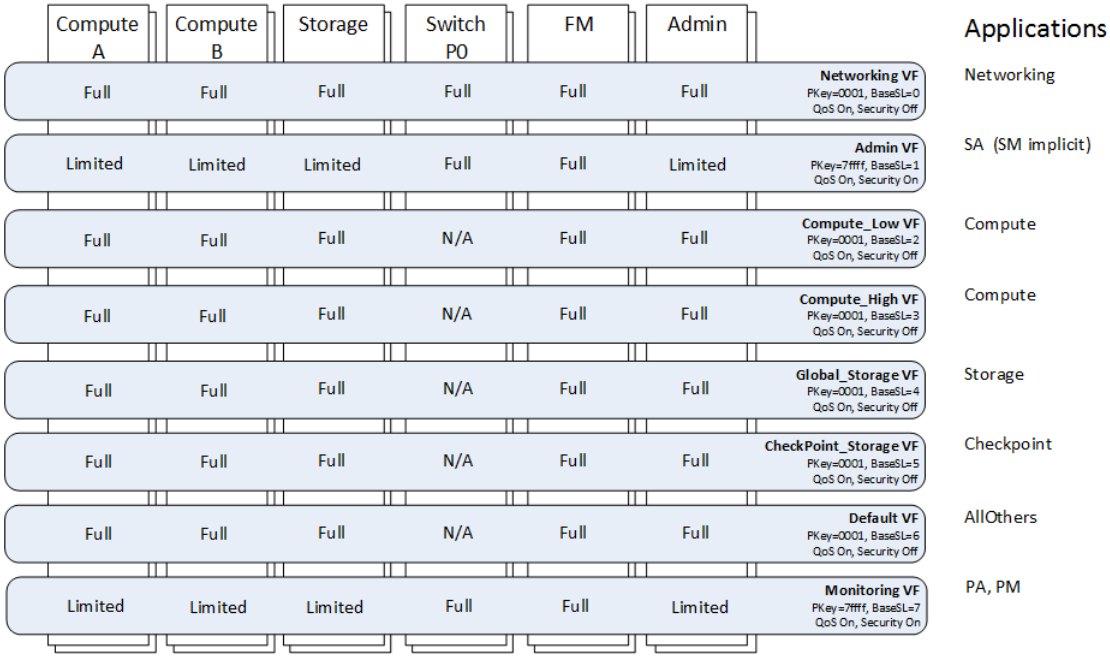
In this configuration, all traffic shares a single BaseSL, so there is no fabric QoS. The Admin traffic is secured so that SA, PA, PM and SM traffic is permitted only between the FM (and switch port 0s) and other devices. However, other devices, such as Compute A devices, cannot attempt to manage other devices in the fabric.

**Figure 10. Simple QoS Virtual Fabrics Configuration**




The previous figure builds on the Default Virtual Fabrics Configuration by providing an additional Compute virtual fabric that includes all devices and the Compute applications. This virtual fabric uses PKey 0x0001 and BaseSL 1. In this configuration, no new security is enabled, however by assigning Compute to a unique BaseSL (and configuring the Compute virtual fabric's QoS policies such as bandwidth and preemption/traffic flow optimization), the Compute fabric's traffic can be separated from other fabric traffic by using a unique BaseSL and unique SCs and VLs throughout the fabric. This example also points out the flexibility and cross matching. The Default and Compute virtual fabrics share a common PKey and therefore have the same security rules. However, the Default and Admin virtual fabrics share a common BaseSL and therefore have the same QoS rules.

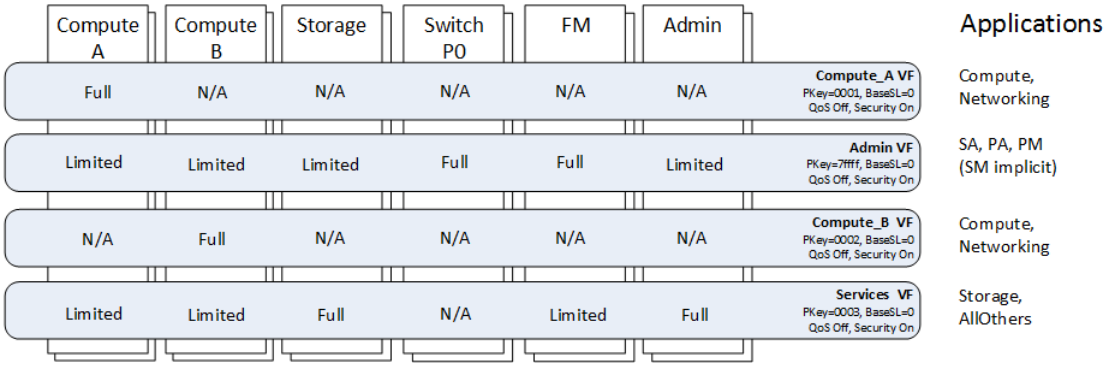
**Figure 11. Advanced QoS Virtual Fabrics Configuration**




The previous figure builds further on the Default Virtual Fabrics Configuration. In this figure eight unique virtual fabrics are defined and each is assigned a unique BaseSL. This configuration has limited security as only management traffic is secured on PKey 0x7fff, while all other traffic is unsecured and uses PKey 0x0001. However, by assigning each virtual fabric a unique BaseSL, they each have unique QoS

characteristics and the performance impacts between virtual fabrics will be controlled by the QoS policies configured for each virtual fabric. In this configuration, due to separate BaseSLs, SCs, and VLs per virtual fabric, the fabric performance statistics can also be monitored per virtual fabric. This occurs in the Performance Manager and is based on analysis of the per VL statistics on each port. This example also shows that, if desired, the management traffic can also be split out. In this example, the Admin virtual fabric is used for SA traffic while the Monitoring virtual fabric is used for PM and PA traffic. This can permit the PM traffic to proceed at regular intervals with less impact from bursty name resolution traffic that may be occurring to the SA.

**Figure 12. Simple Security Virtual Fabrics Configuration**




The previous figure is a security focused example where each virtual fabric is assigned a unique PKey and the Compute A virtual fabric is limited to the Compute A devices. Similarly, the Compute B virtual fabric is limited to the Compute B devices. This means a compute application can only be run within the Compute A or the Compute B devices, but may not use devices from both groups. Both networking and Compute applications are permitted within those two virtual fabrics, so Compute A can be its own separate IP subnet from Compute B. This style of configuration may be useful when multiple departments or tenants of the cluster each need separate subsets of the cluster to run jobs and the sysadmin wants to secure the traffic so applications cannot mistakenly communicate with each other. This example also has an Admin virtual fabric, which is similar to the configuration in Figure 9 on page 35 and which secures the fabric management traffic from the other traffic. Finally, there is a Services virtual fabric that includes the storage and admin nodes as full members, with Storage, and other applications. This permits the storage and admin node to communicate with any node in the fabric, however, it prevents the nodes in Compute A from attempting to communicate with Compute B through this PKey.

The concepts presented in these examples can easily be combined to create more advanced configurations with a mixture and security and QoS polices as required by the given fabric and its operations.

The Fabric Manager also has some capabilities to dynamically adjust virtual fabrics by making changes to the configuration in a live fabric. This can be used to facilitate occasional changes in the needs, such as adding or removing a department or tenant on the fabric.

## 2.2 Quality of Service Policies

Quality of Service (QoS) allows the priority of each vFabric to be configured. QoS policies allow multiple applications to run on the same fabric without interference, through controlling the use of network resources. These policies allow the following:

- Setting minimum bandwidth percentages for high-volume traffic.
- Addressing needs of high-priority, low-volume traffic.
- Place multicast traffic on a separate VL from normal traffic.

QoS policies allow the user to assign a bandwidth minimum to classes of traffic to define how network resources are allocated. For instance, networking traffic can be assigned to one vFabric, storage to a second vFabric, and compute to a third. These classes of traffic can then be assigned a bandwidth percentage to which they will be limited when a link is saturated. Low-volume traffic requires low latency, like administrative or control traffic; the user can specify this as high priority. In addition, when using a mesh or torus fabric, QoS policies can place multicast traffic on a different SL from normal traffic.

### 2.2.1 QoS Operation

Virtual Lanes (VLs) permit multiple logical flows over a single physical link. Each physical link has a number of VLs. Each class of service has a Service Level (SL). The local route header of the packets carries the SL, which identifies the different flows within the subnets. The amount of resources allocated to the VL is based on the bandwidth configured for the vFabric containing the SL. The Subnet Manager programs the SL-to-VL mappings and VL Arbitration tables to achieve the QoS policies.

Applications can query for path records by Service ID to obtain path records associated with the given application. These path records contain the SL associated with the traffic class/QoS vFabric. In turn, this SL is used for packets within this class of traffic.

### 2.2.2 Traffic Flow Optimization

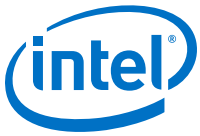
Traffic Flow Optimization (TFO), or preemption is a data flow feature intended to reduce the latency of small packets by allowing such packets to interleave (preempt) large packets at the head of the queue. Without preemption, these smaller packets would be blocked by waiting for a large packet's transfer to complete.

Preemption capability can be enabled per Virtual Fabric (VFs), and the preempting considerations can be ranked among VFs. That is, a hierarchy of preemption among VFs can be established.

- Sizes for small packets (preempting packets) is configurable.
- Sizes for the larger packets (preempted packets) is configurable.

The total amount that a single packet can be preempted by others also can be configured.

See Table 12 on page 96 for more information.



#### 2.2.2.1 Enabling TFO

1. Define high and low priority QoS vFabrics (as outlined in Table 40 on page 133.) This is done by modifying the contents of `/etc/opa-fm/opafm.xml` on the node running the Intel® OPA Fabric Manager.
   a. Ensure `<smallpacket>`, `<largepacket>`, and `<preemptlimit>` allow the workload you want to have preemption capability, based on the message sizes contained within that workload.
   b. Other details and parameters can be found in Table 12 on page 96. Changes to these parameters are updated on HFI or switch ports only when the port is bounced.
2. Restart the Intel® OPA Fabric Manager.

The Intel® OPA Fabric Manager will not modify the preemption settings on an active port. The new parameters will be activated the next time each port is bounced.

When running the workload, specifically ask for the high priority vFabric, as outlined in The Default Partition, IPoIB and vFabrics, and MPI and vFabrics. In the example below we use the BaseSL tag, but this is not mandatory, as other methods are available.

**Example 1. Preemption Configuration Example**

The following example shows defaults for small and large packets and preemption limits. The first vFabric is the default with a QOS=0, meaning this vFabric cannot preempt nor be preempted. The second is a storage-type vFabric. Preemptrank of 1 indicates this vFabric can be preempted by all other ranks. The third vFabric is a latency-sensitive vFabric to be used with a high priority job. A preemptrank of 127 implies this VFabric can preempt all other vFabrics. Another typical strategy is to change QOS=1 for the default vFabric. However, note than this implies that any job not specifically requesting another vFabric can be preempted and not considered high priority.

```
<Preemption>
     <SmallPacket>256</SmallPacket>
    <LargePacket>4096</LargePacket>
    <PreemptLimit>4096</PreemptLimit>
</Preemption>

<VirtualFabric>
      <Name>Default</Name>
      <Enable>1</Enable>
    <QOS>0</QOS>
</VirtualFabric>

<VirtualFabric>
    <Name>Storage</Name>
    <Enable>1</Enable>
    <QOS>1</QOS>
    <Bandwidth>50%</Bandwidth>
    <PreemptRank>1</PreemptRank>
    <BaseSL>6</BaseSL>
    …
</VirtualFabric>

<VirtualFabric>
    <Name>ComputeLat</Name>
    <Enable>1</Enable>
    <QOS>1</QOS>
    <Bandwidth>30%</Bandwidth>
    <PreemptRank>127</PreemptRank>
```

```
    <BaseSL>5</BaseSL>
    …
</VirtualFabric>
```

## 2.3 FM Security

### 2.3.1 Security Overview

One of the most important set of features configurable in the Intel® Omni-Path Fabric Suite Fabric Manager are those related to security, both for in-band and out-of-band communications.

The goal of this added security is to prevent would-be attackers from destabilizing or compromising the integrity of the fabric through an FM external socket connection or by utilizing a compromised node within the fabric. An administrator can control the additional level of security they want the Fabric Manager to enforce on top of the security provided by the Intel® Omni-Path Architecture hardware.

This section is divided into two sub-sections. In-band security details the security options within the FM when managing the fabric through Intel® Omni-Path Architecture hardware. Out-of-band security provides details on the configurable options for socket based connections to the FM's Fabric Executive (FE) for a specific set of services, including the Intel® Omni-Path Fabric Suite Fabric Manager GUI.

### 2.3.2 In-Band Security

#### 2.3.2.1 Fabric Security Quarantine

To enforce security in-band, the Fabric Manager can quarantine nodes that are detected attempting to circumvent fabric security. The method by which the Fabric Manager enforces this quarantine is through link state. As the Fabric Manager is responsible for configuring and activating all ports in the fabric, by leaving a port in an un-activated state, the port is rendered unable to communicate with the fabric through said link.

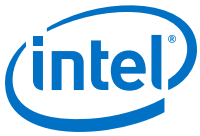
##### 2.3.2.1.1 Quarantined Node List

In addition to every quarantine event being logged, the Fabric Manager also maintains a list of quarantined nodes that is available through an SA query. Please refer to the *Intel® Omni-Path Fabric Suite FastFabric User Guide* for details on opasaquery.

#### 2.3.2.2 Anti-Spoofing Protection

Using information exchanged at Link Negotiation and Initialization (LNI) the Fabric Manager is able to verify the identity of a node against what is reported by the node in attribute responses. If a node is detected attempting to spoof one of the key fields the FM checks, it will be quarantined using the method described above in Fabric Security Quarantine.

The Fabric Manager also configures a field on the switch neighbor of an end node that prevents said node from sending fabric packets with source LIDs outside the range assigned to the node by the FM. If a node attempts to send a packet outside this configured LID range, the neighbor switch immediately drops said packet.



In addition to the LID checks done by the switch port the FM is connected to, for packets sent to the FM, FM also performs per packet validation. When the anti-spoofing protection is enabled, the Fabric Manager verifies proper packet origins and correct P-Key usage for the response compared to the request that was issued.

The anti-spoofing detection of the Fabric Manager is enabled in the following section of the configuration file:

```
<!-- SmaSpoofingCheck enables support for port level SMA security-->
 <!-- checking related features. -->
 <SmaSpoofingCheck>0</SmaSpoofingCheck>
```

#### 2.3.2.3 Management Traffic Denial of Service (DoS) Protection

Intel® Omni-Path Fabric provides a mechanism for preventing Denial of Service (DoS) attacks against management traffic that could cause slow fabric configuration or even prevent full configuration of a large size fabric. The Fabric Manager is responsible for configuring this mechanism on all external switch ports that limits the rate at which management traffic can be sent by an end node. This limit is configurable in the Fabric Manager configuration file and is defined in the following section:

```
<!-- *************** Security Features ******************************-->
<!-- VL15CreditRate: Rate at which to return credits to an HFI that has been -->
<!-- recognized  as attempting to spoof the fabric. Prevents DoS attacks against
-->
<!-- the fabric from a spoofed host. -->
<!-- The field is defined as 1/(2^X) of the full return rate. For instance, if -->
<!-- the field is set to 2, the credit return rate will be 1/(2^2) = 1/4 -->
<!-- (one fourth) the full credit rate when a denial of service attack is
detected. -->
<!-- Valid values are between 0 and 22, with 0 disabling the credit rate limiting
-->
<VL15CreditRate>18</VL15CreditRate>
```

#### 2.3.2.4 Pre-Defined Topology Verification

The ability to validate a fabric layout and configuration against a pre-defined or expected configuration is an important part of the fabric bring-up process on larger scale fabrics. The aim of Pre-Defined Topology Verification is to provide a mechanism within the FM to allow the verification of a pre-defined topology input during a sweep against the real fabric topology.

Pre-Defined Topology Verification relies on a topology input file that describes nodes and links using a combination of Node GUIDs, Node Descriptions and Port numbers. See *Intel® Omni-Path Fabric Suite FastFabric User Guide* (opaxlattopology, opagentopology, opareport) for details creating a topology input file. If a node's actual position and values do not match the pre-defined topology, the node will be quarantined in the same manner as described in Fabric Security Quarantine on page 40. In addition, if a link is discovered that is not defined in the input topology file it will also be quarantined.

This pre-defined topology verification is done on every sweep, so that any dynamic changes that occur within the fabric while the FM is running will be noted in the logs.

Which checks are performed and what the SM does when a check fails can be configured per-field. There are three levels of field enforcement: Disabled, Warn, and Enabled. "Disabled" disables checking for that field; mismatches will not be reported.

"Warn" causes the FM to warn if part of the real topology fails a check. This warning is sent to the FM log output. "Enabled" causes the FM to both report mismatches between the expected and actual topology (same as Warn) as well as quarantine the offending node. NodeGUID, PortGUID, and NodeDesc enforcement settings apply only when UndefinedLink enforcement is set to Warn or Enabled.

See also PreDefinedTopology on page 89 for parameters, default values, and descriptions.

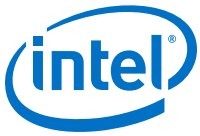
A log message limiter is also provided to limit the number of log messages output by this feature per sweep when a mismatch is detected.

All options are configurable through the Fabric Manager configuration file in the following section:

```
<!-- ***************** Pre-defined Topology Verification ***************** -->
    <!-- The PreDefinedTopology section is used for verifying the layout of  -->
    <!-- the fabric against a topology input file of the expected layout.    -->
    <!-- There are three modes of handling mismatches: Disabled, Warn, and   -->
    <!-- Enabled. Disabled ignores any mismatches on that field, Warn prints -->
    <!-- a warning to the log file, and Enabled prints a warning to the log  -->
    <!-- file and quarantines the node from the fabric.                      -->
    <!-- Field Definitions:                                                  -->
    <!--     Enabled:                                                        -->
    <!--         Whether or not this feature is enabled.                     -->
    <!--     TopologyFilename:                                               -->
    <!--         Fully qualified filename of pre-defined input topology.     -->
    <!--     LogMessageThreshold: Number of warnings to output to log        -->
    <!--         Number of warnings to output to log file before suppressing -->
    <!--         further warnings and only printing a summary at the end of  -->
    <!--         a sweep. Entering 0 disables this threshold.                -->
    <!--     FieldEnforcement:                                               -->
    <!--         Per-field enforcement levels for mismatch handling.         -->
    <!-- Field comparison is done on a validation link. Validation links are -->
    <!-- found using (NodeGUID,PortNum) unless NodeGUID enforcement is       -->
    <!-- DISABLED, in which case (NodeDesc,PortNum) is used to find the      -->
    <!-- link. If there is more than one link matches by (NodeDesc,PortNum)  -->
    <!-- a log warning will be printed and false matches may occur.          -->
    <PreDefinedTopology>
        <LogMessageThreshold>100</LogMessageThreshold>
        <FieldEnforcement>
            <NodeDesc>Warn</NodeDesc>
            <NodeGUID>Warn</NodeGUID>
            <PortGUID>Warn</PortGUID>
            <UndefinedLink>Warn</UndefinedLink>
        </FieldEnforcement>
    </PreDefinedTopology>
```

```
<PreDefinedTopology>
        <Enabled>0</Enabled>
        <TopologyFilename>/etc/opa/topology.0:0.xml</TopologyFilename>
    </PreDefinedTopology>
```

### 2.3.3 Out-of-Band Security



#### 2.3.3.1 Fabric Executive

The FE is the part of the Fabric Manager package that provides OOB LAN network-based applications with access to FM related facilities (i.e., SM/SA, PM/PA, notifications, etc.). The FE provides this support through a protocol running on top of TCP/IP, which enables these FE client applications (FEC) to interface with the FM facilities in real-time.

The FE provides the following means for ensuring that only authorized STL administrators have access to this functionality:

1. FE authenticates the user connecting to the interface. This is to ensure that only STL administrations have access to the information provided by the FE interface.
2. FE provides privacy protection for the channel used to communicate with FEC applications. This is to prevent other non-authorized administration personnel from obtaining STL fabric information directly from the administration network by eavesdropping on the communication between FE and FEC.

The following are all the configuration parameters used to configure security for the FE. These parameters are configurable through the Fabric Manager configuration file:

```
<SslSecurityEnabled>0</SslSecurityEnabled>
    <SslSecurityDir>/usr/local/ssl/opafm</SslSecurityDir>
    <SslSecurityFmCertificate>fm_cert.pem</SslSecurityFmCertificate>
    <SslSecurityFmPrivateKey>fm_key.pem</SslSecurityFmPrivateKey>
    <SslSecurityFmCaCertificate>fm_ca_cert.pem</SslSecurityFmCaCertificate>
    <SslSecurityFmCertChainDepth>1</SslSecurityFmCertChainDepth>
    <SslSecurityFmDHParameters>fm_dh_parms.pem</SslSecurityFmDHParameters>
    <SslSecurityFmCaCRLEnabled>0</SslSecurityFmCaCRLEnabled>
    <SslSecurityFmCaCRL>fm_ca_crl.pem</SslSecurityFmCaCRL>
```

##### <SslSecurityEnabled>

This parameter is used to enable/disable this feature (default value is 1).

The following parameters are required by the OpenSSL interface, in order to establish a secure socket connection over the OOB network.

##### <SslSecurityDir>

This parameter specifies the directory location of OpenSSL-related files (default value `/usr/local/ssl/opafm`).

##### <SslSecurityFmCertificate>

This parameter specifies the certificate PEM file to be used by the STL FM (default value `fm_crt.pem`).

##### <SslSecurityFmPrivateKey>

This parameter specifies the private key PEM file to be used by the STL FM (default value `fm_key.pem`).

##### <SslSecurityFmCaCertificate>

This parameter specifies the Certificate Authority (CA) certificate PEM file to be used by the STL FM (default value `fm_ca_cert.pem`).

#### < SslSecurityCertChainDepth>

This parameter specifies the limit up to which depth certificates in a chain are used during the verification procedure. If the certificate chain is longer than allowed, the certificates above the limit are ignored.

#### <SslSecurityFmDHParameters>

This parameter specifies the Diffie-Hellman parameters PEM file to be used by the STL FM (default value `fm_dh_parms.pem`).

#### <SslSecurityFmCaCRL>

This parameter specifies the CA CRL PEM file to be used by the STL FM (default value `fm_dh_parms.pem`).

#### <SslSecurityFmCaCRLEnabled>

This parameter is used to enable/disable the usage of the CRL PEM file (default value is 1).

## 2.4 SM Algorithms and Features

### 2.4.1 Routing Algorithms in the SM

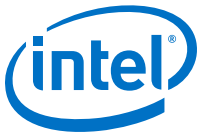
The routing algorithm selects, for every pair of nodes in the fabric, the path to be used for their communications. The actual paths chosen may be based on the path hop count, and link width/speed for each hop. In the context of all the various node-to-node communication paths possible, the SM tries to balance the number of communication paths being routed over each link in the fabric. A variety of strategies for routing are available in the form of different selectable routing algorithms. Refer to Routing Algorithm on page 46 for detailed information.

Spine-first routing is an optional feature that, when enabled, will favor paths that route through the spine switches of a director class switch, rather than a path that is the same length but does not. This helps eliminate cycles (for example, credit loops) on fabrics where such routes are possible. Refer to Shortest Path on page 46 for more information.

To use spine-first routing in a tree where director class switches are not at the core, the fat tree routing algorithm can be used. Refer to Fat Tree on page 46 for more information.

### 2.4.2 LID Mask Control (LMC)

The LMC feature provides a method for assigning multiple local identifiers (LIDs) to a single physical port. This allows for multiple paths through the fabric to be configured between a single pair of nodes. When enabled, the routing algorithm attempts to ensure that these paths are placed on unique hardware when possible in order to reduce disruption in the case of switch failure. Each end-node port will have $2^{LMC}$ LIDs. Refer to LMC, Dispersive Routing, and Fabric Resiliency on page 48 for detailed information.



### 2.4.3 Multicast Group Support

Multicast groups are used to direct one-to-many and many-to-many traffic. Nodes subscribe to multicast groups by issuing requests to the SM. Refer to Fabric Multicast Routing on page 50 for detailed information.

## 2.5 Fabric Unicast Routing

One of the most complex aspects of configuring a large fabric is routing. The SM must configure a routing for the fabric that provides a careful balance between:

- Performance
- Resiliency to fabric disruptions
- Avoidance of deadlocks and credit loops
- Conservation of resources, such as LIDs and routing table entries

As a result of these sometimes conflicting goals, the SM allows for user configuration of many aspects of routing so the administrator can select the criteria for routing the fabric.

### 2.5.1 Credit Loops

Because the Intel® Omni-Path Architecture has credit-based link layer flow control, credit loops are possible. Under high stress, a credit loop can become a fabric deadlock, which will force switch timers to discard packets. These deadlocks and discards can cause significant performance impacts.

The deadlocks are very rare and in practice, they only occur under high bandwidth applications, however it is better to route the fabric to avoid credit loops altogether.

There are many research papers on the topics of routing. Credit loop avoidance is a focus of many of the algorithms. Credit loops are avoidable for all the popular fabric topologies and the SM utilizes algorithms that are designed to avoid credit loops.

Enforcing up/down routing is a method used to avoid credit loops in tree topologies. With up/down routing, when equal cost paths exist which go up or down the tree, preference is given to the up links. This method of routing is used with the shortest path algorithm where the backbone of the fabric is made of director-class switches, by enabling SpineFirstRouting in the SM configuration. When SpineFirstRouting is enabled for equal length paths, the SM always gives preference to routing traffic from a director switch leaf, via the director switch spine, as opposed to using an edge external to the director. This treats the spine as "up" relative to the leaf and results in clean "up"/"down" routing for the fabric. By ensuring that all traffic is using consistent routes, credit loops are prevented.

The fat tree routing algorithm has been extended to also avoid credit loops for fabrics where the backbone of the fabric need not be comprised of director-class switches. It uses spine-first routing for any type of switch at the core of the fabric.

If a credit loop is suspected, the CLI command `opareport -o validatecreditloops` can be used to check the fabric.

### 2.5.2 Routing Algorithm

The routing algorithm selects, for every pair of nodes in the fabric, the path to be used for their communications. The actual paths chosen may be based on the path hop count, and the link width/speed for each hop. In the context of all the various node-to-node communication paths possible, the SM statically load balances the number of communication paths being routed over each link in the fabric. A variety of strategies for routing are available in the form of different selectable routing algorithms.

The SM supports the following routing algorithms:

- Shortest Path on page 46.
- Fat Tree on page 46 - optimized balanced routing for fat tree topologies with credit loop avoidance.
- Device Group Shortest Path on page 47 - a variation of shortest path that can result in better balanced fabrics in some conditions.

The routing algorithm is selectable using the `RoutingAlgorithm` parameter.

#### 2.5.2.1 Shortest Path

This algorithm is the default and works very well for most fabrics. This algorithm always routes using a least cost path. In most fabrics there are many equal cost paths, in which case the SM statically balances the number of paths using each Inter-Switch Link (ISL).

SpineFirstRouting is an optional feature of Shortest Path routing. When SpineFirstRouting is enabled for equal length paths, the SM always gives preference to routing traffic from a director switch leaf, via the director switch spine, as opposed to using an edge switch external to the director. This treats the spine as "up" relative to the leaf and results in clean "up"/"down" routing for the fabric. By ensuring that all traffic is using consistent routes, credit loops are prevented. To use spine first routing in fabrics where director class switches are not at the core of the fabric, the fat-tree routing algorithm can be used.

SpineFirstRouting is enabled by default and has no ill side effects. Unlike simpler algorithms in other SMs, the Intel® SM's shortest path algorithm has sophisticated traffic balancing and routing algorithms which allow it to provide high performance for a wide variety of topologies.

#### 2.5.2.2 Fat Tree

The Fat Tree algorithm generally provides better balancing of ISL traffic with fat tree topologies than the shortest path algorithm and it provides up/down routing for deadlock avoidance. The FM accomplishes the balancing of ISL traffic through identification of the fat tree topology layout and determines up/down routing by calculating which tier in the fabric each switch resides at.

To determine the switch tier, the FM needs to understand how many tiers of switch chips there are in the fabric. If all HFIs and Target Fabric Interfaces (TFIs) are at the same tier, the FM automatically determines the fat tree topology layout. If they are not on the same tier, you can specify which switches are at the core/root of the tree by configuring a `CoreSwitches` device group as an alternate method of topology identification. Only devices that do not communicate with each other should be connected to the core switches. If devices connected to the core switches communicate with each other they could potentially form a credit loop.

Director class switches typically have multiple internal tiers of switch chips. For example, all the Intel® Omni-Path Director Class Switch 100 Series models have two internal tiers of switch chips . A 3-tier fat tree topology can be constructed using edge switches at the lowest tier of the fabric and director class switches at the core.

The Fat Tree algorithm also provides the capability of balancing traffic across device groups. You can configure a `RouteLast` device group to ensure that devices in this group have balanced routing. For instance, if all compute nodes are in this group, better traffic dispersion can be obtained for these devices. This mechanism can be used to load balance over compute and IO nodes. For instance, if all compute nodes are in the device group, routing would be calculated for compute nodes first followed by IO nodes, load balancing across each set of nodes.

#### 2.5.2.3 Device Group Shortest Path

Overall, `dgshortestpath` routing is a form of `Min-Hop` or `Shortestpath` routing, except you can control the order in which routes to end nodes are assigned. This can be used to ensure diversity of routing within groups of devices as well as the entire fabric overall. End nodes that are not members of any listed groups will be routed last.

When the routing algorithm is set to `dgshortestpath` the following section in the `opafm.xml` file is used to configure the algorithm.

```
<DGShortestPathTopology>
<!-- RoutingOrder lists the device groups in the order they should -->
<!-- be handled. Each device group must have been declared in the -->
<!-- DeviceGroups section. -->
<!-- <RoutingOrder> -->
<!-- <DeviceGroup>Compute</DeviceGroup> -->
<!-- <DeviceGroup>All</DeviceGroup> -->
<!-- <DeviceGroup>Storage</DeviceGroup> -->
<!-- </RoutingOrder> --></DGShortestPathTopology>
```

### 2.5.3 Adaptive Routing

A limitation of static routing is that it must be done before traffic begins to flow, hence static routes are balanced using “best guesses” by the FM and sysadmin of potential application traffic patterns. However, once applications start to run, those routes may be non-ideal. Adaptive routing is a powerful capability of all Intel® Omni-Path Switch 100 Series switches, which allows the switches to scalably adjust their routes while the applications are running to balance the routes based on actual traffic patterns.

The Intel adaptive routing solution is highly scalable because it allows the FM to provide the topology awareness and program the switches with the rules for adaptive routing. Then the FM gets out of the way and permits the switches to dynamically and rapidly adjust the routes based on actual traffic patterns.

This approach ensures a scalable solution, because as switches are added, each new switch will work in parallel with others to dynamically adapt to traffic. This approach avoids the FM becoming a bottleneck and scales very well.

Adaptive routing provides a few important capabilities and options:

1. Adaptive routing can rapidly route around fabric disruptions and lost ISLs. When adaptive routing is enabled, this capability automatically occurs and limits the amount of lag time between an ISL going down and the traffic being redirected to alternate routes.

2. Adaptive routing can automatically balance and rebalance the fabric routes based on traffic patterns. It has the unique ability to handle changing traffic patterns that may occur due to different computational phases or the impacts of starting or completing multiple applications which are running on the same fabric.

When Adaptive Routing is enabled, the SM programs each switch with a list of alternate, equal cost routes for each destination in the fabric. When a switch detects that a route has failed (due to congestion or complete failure of a link or switch) it selects from the list of alternates and updates its linear forwarding table (LFT) with the alternate route.

The algorithm the switch uses for choosing the alternate route can be "Random" (chose the alternate randomly), "Greedy" (chose the alternate that is least busy) or "GreedyRandom" (if there are multiple alternates that aren't busy, randomly choose from them).

The configuration setting `Threshold` determines how busy a route must be before it will be rerouted by Adaptive Routing, and the setting `ARFrequency` determines how frequently the switch checks for congestion. If the `LostRouteOnly` setting is enabled, traffic is only rerouted if a route completely fails.

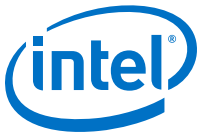
The following is the adaptive routing section of the `opafm.xml-sample` file:

```
<!-- Configures support for AdaptiveRouting in Intel Switches -->
    <AdaptiveRouting>
      <!-- 1 = Enable, 0 = Disable -->
      <Enable>0</Enable>
      <!-- When set, only adjust routes when they are lost. -->
      <!-- If not set, adjust routes when they are lost and -->
      <!-- when congestion is indicated. -->
      <LostRouteOnly>0</LostRouteOnly>
      <!-- Algorithm the switch should use when selecting an egress port -->
      <!-- Algorithms are currently 0 = Random, 1 = Greedy and -->
      <!-- 2 = GreedyRandom. -->
      <Algorithm>0</Algorithm>
      <!-- Update Frequency: Specifies the minimum time between -->
      <!-- AR adjustments. Values range from 0 to 7 and are read as 2^n -->
      <!-- times 64 ms. Default is 4, or about 1 second  . -->
      <ARFrequency>4</ARFrequency>
      <!-- Congestion threshold above which switch uses adaptive routing. -->
      <!-- Congestion threshold is per-VL and measured by tag consumption
percentage. -->
      <!-- Values range from 0 to 7. -->
      <!-- 7, 6, 5, 4 correspond to 55%, 60%, 65%, and 70%, respectively. -->
      <!-- 3, 2, 1 correspond to 80%, 90%, and 100%, respectively. -->
      <!-- 0 means "Use firmware default" -->
      <Threshold>0</Threshold>
    </AdaptiveRouting>
```

### 2.5.4 LMC, Dispersive Routing, and Fabric Resiliency

The SM also supports LID Mask Control (LMC). LMC allows for more than 1 LID to be assigned to each end node port in the fabric, specifically $2^{LMC}$ LIDs will be assigned. This allows the SM to configure the fabric with multiple routes between each end node port, allowing applications to load balance traffic (for example, using algorithms such as PSM's dispersive routing) across multiple routes or provide rapid failover using techniques like Alternate Path Migration (APM).

The Intel® Performance Scaled Messaging (PSM) layer can take advantage of LMC to provide dispersive routing and load balance MPI across multiple routes.



When LMC is configured with a non-zero value, the SM assigns routes with the following goals in priority order:

1. Each LID for a given destination port is given as unique a route as possible through the fabric, using in order of preference:
   a. completely different switch chassis where possible;
   b. ASICs when possible and different switch chassis are not possible;
   c. at least different ports in the same switch when different ASICs or chassis are not possible.

   This approach provides optimal resiliency so that fabric disruptions can be recovered from using APM and other rapid failover techniques.
2. The overall assignment of Base LIDs to ISLs is statically balanced, such that applications that only use the Base LID will see balanced use of the fabric.
3. The assignment of alternate LIDs to ISLs is statically balanced, such that applications that use multiple LIDs for load balancing may see additional available bandwidth through the fabric core.

#### 2.5.4.1 PathRecord Path Selection

When a non-zero LMC value is used, the SM will have multiple paths available between pairs of nodes. The FM permits configuration of the SM/SA to specify which combinations of paths should be returned and in what order. Most multi-path applications will use the paths in the order given, so the first few returned is typically used for various failover and dispersive routing techniques.

Most applications use the first path or only the first few paths. When LMC!=0, there can be N=($2^{LMC}$) addresses per port. This means there are $N^2$ possible combinations of SLID and DLID which the SA could return in the Path Records. However there are really only N combinations that represent distinct outbound and return paths. All other combinations are different mixtures of those N outbound and N return paths.

Also important to note is that LMC for all HFIs are typically the same, while LMC for switches will usually be less. Generally redundant paths and/or having a variety of paths is not critical for paths to switches (which are mainly used for management traffic), but can be important for applications communicating HFIs to HFI.

The FM `Path Selection` parameter controls what combinations are returned and in what order. For examples below let's assume SGID LMC=1 (2 LIDs) and DGID LMC=2 (4 LIDs)

- `Minimal` – return no more than 1 path per lid: SLID1/DLID1, SLID2/DLID2 (since SGID has 2 lids stop)
- `Pairwise` – cover every lid on both sides at least once: SLID1/DLID1, SLID2/DLID2, SLID1/DLID3, SLID2/DLID4
- `OrderAll` – cover every combination, but start with pairwise set: SLID1/DLID1, SLID2/DLID2, SLID1/DLID3, SLID2/DLID4 SLID1/DLID2, SLID1/DLID4, SLID2/DLID1, SLID2/DLID3
- `SrcDstAll` – cover every combination with simple all src, all dst: SLID1/DLID1, SLID1/DLID2, SLID1/DLID3, SLID1/DLID4 SLID2/DLID1, SLID2/DLID2, SLID2/DLID3, SLID2/DLID4

### 2.5.5 Handling Fabric Changes

Programming using Directed Route (DR) SMPs is less performant than LID routing (LR) due to a significant amount of latency introduced at each intermediate hop on the path to the destination. LR SMPs are used whenever possible to improve performance at scale. When programming linear forwarding tables (LFTs) on switches, a mixed LR-DR approach is used. This method uses LR routing up to the last hop in the path and then uses DR for the last hop. This approach is used to program the minimal amount of LFT route data to enable the LR path from the SM to the destination switch, a maximum of two LFT blocks. The SM then switches to pure LR SMPs to program the remaining LFT data.

A wave model is used to program the routing tables. The first wave of switches includes the set of switches connected to the SM. The next wave starts with the SM doing LID based routing to these connected set of switches and initializing the next set of switches that are connected to these switches with one hop DR from those switches and so on. This wave order also ensures that each new wave is fully LID routable from the SM, that is, the routes to the new wave are such that they will always pass through switches that already have been fully initialized.

To minimize disruption during the programming of the LFT data, the programming of full LFT blocks is done in parallel for the set of switches in each wave by striping LFT blocks across the switches. For example block N for all switches are written and then block N+1 and so on.

When handling fabric changes, the fabric is analyzed to reduce the amount of changes to the routing tables in an effort to minimize fabric disruptions. If HFIs are added or deleted, the LFT changes will be limited to the affected routes. Instead of recalculating routing tables, the loss of an HFI results in the removal of the route for the specific HFI. The addition of an HFI results in the insertion of the route into the existing LFT blocks. Only those LFT blocks with added or deleted HFI routes will be programmed on the switches. This reduces fabric programming time and minimizes fabric disruption due to unnecessary routing recalculations.

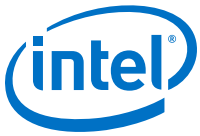
## 2.6 Fabric Multicast Routing

In addition to unicast, Intel® Omni-Path Architecture also supports multicast. Multicast allows a single packet sent by an application to be delivered to many recipients. Intel® Omni-Path Architecture Multicast is used for IPoIB broadcast for TCP/IP address resolution protocol (ARP), IPoIB multicast for UDP/IP multicast applications, and can also be directly used by other applications.

Intel® Omni-Path Architecture supports separate routes per Multicast Group. Each multicast group is identified by a unique 128-bit Multicast GID. Within the fabric, the SM assigns each active multicast group a 24-bit Multicast LID.

To implement multicast, the SM must construct spanning tree routes throughout the fabric that will deliver exactly one copy of each sent packet to every interested node. The SM must configure such routes within the limitations of the hardware. Namely:

- There may be varied MTU and speed capabilities for different switches and Inter Switch Links.
- The fabric topology and hardware may change after applications have joined a multicast group



To support efficient yet dependable routing of multicast, the SM allows the user to configure and control Multicast Routing. The root selection algorithm gives the user the ability to influence the choice of the root of the spanning tree. The default is to choose a switch that is at the core of the fabric, that is, one with the least total cost to all other switches.

### 2.6.1 Handling Fabric Changes

The SM must make the realizable decision for a multicast group at the time an application creates/joins a multicast group. This means the SM must determine if there is a path with the appropriate speed and MTU to meet the requested capabilities of the multicast group.

However, later fabric changes could make the multicast group unrealizable. For example, removal or downgrade of high speed links, loss of switches, changes to switch MTU or speed configuration, to name a few. Unfortunately, in this case there is no standard way in the Open Fabrics Alliance APIs to notify end nodes that the multicast group is no longer viable.

To address this situation, the SM performs stricter multicast checking at Join/Create time. This means a multicast join/create is rejected if there are any switch-to-switch links that do not have at least the MTU or rate requested for the multicast group, reducing the chance that a simple fabric failure or change (for example, loss of one link) could make the group unrealizable.

The `DisableStrictCheck` parameter controls this capability. When 1, this parameter disables the strict checking and accepts Join/Create request for which at least one viable fabric path exists. By default the parameter is 0, which allows for more strict checking.

In addition, the `MLIDTableCap` parameter is used to configure the maximum number of Multicast LIDs available in the fabric. This must be set to a value less than or equal to the smallest Multicast forwarding table size of all the switches that may be the in the fabric. It defaults to 1024, which is below the capability of all current Intel® Omni-Path switches. Using a value below the capability of the switches can prevent errant applications from creating an excessive number of multicast groups.

### 2.6.2 Conserving Multicast LIDs

IPv6 (and possibly other applications) can create numerous multicast groups. In the case of IPv6, there is one Solicited-Node multicast group per HFI/TFI port. This can result in an excessively large number of multicast groups. Also in large fabrics, this quickly exceeds `MLIDTableCap`. For example, a 10,000 node fabric with IPv6 would need over 10,000 multicast groups.

To address this situation, the SM can share a single MLID among multiple Multicast groups. Such sharing means both the routes, and destinations are shared. This may deliver some unrequested multicast packets to end nodes, however unneeded packets are silently discarded by the transport layer in the HFI/TFI and have no impact on applications.

The SM allows the administrator to configure sets of multicast groups which will share a given pool of Multicast LIDs. This is accomplished using the MLIDShare sections in the configuration file.

MLID sharing can conserve the hardware MLID tables so other uses of multicast can be optimized/efficient.

By default the SM shares a pool of 500 LIDs among all IPv6 solicited-node multicast groups. Thus in fabrics of 500 nodes or less, a unique LID is used for every multicast group. However in larger fabrics, LIDs are shared so that there are still over 500 unique LIDs available for other multicast groups, such as the IPoIB broadcast group and other multicast groups that may be used by applications.

### 2.6.3 Precreated Multicast Groups

The first end node that joins a multicast group also creates the multicast group. When a multicast group is created, critical parameters such as the MTU and speed of the multicast group are also established. The selection of these values must carefully balance the performance of the multicast group against the capabilities of the hardware which may need to participate in the group in the future. For example if an application on an HFI with a 4K MTU creates a 4K multicast group, it prevents subsequent joins of the group by 2K MTU HFIs.

Some ULPs and applications, such as IPoIB, require key multicast groups, such as the IPv4 broadcast group, to be pre-created by the SM.

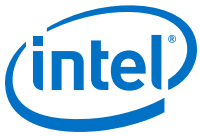
Pre-created multicast group configurations are specified in the `MulticastGroup` sections of the SM configuration files. When the multicast groups are pre-created, their MTU and speed are defined by the SM configuration file, allowing the administrator to be able to account for anticipated hardware capabilities and required performance.

To simplify typical configurations, a MulticastGroup section, which does not explicitly specify any MGIDs, will implicitly include all the multicast groups specified by the IPv4 and IPv6 standards for IPoIB. As such typical clusters using IPoIB need not explicitly list the MGIDs for IPoIB.

### 2.6.4 Multicast Spanning Tree Root

Multicast routing is performed by computing a spanning tree for the fabric. When constructing a spanning tree for a multicast group, MTU and rate must be considered. If the SM first constructs a spanning tree for the largest MTU and rate found in the fabric and if that spanning tree is complete (i.e., includes all switches) then that spanning tree is sufficient for all groups as it will support all smaller MTUs and rates. If it's not complete, the SM will use MTUs and rates smaller than the maximum until a complete spanning tree is computed. Using a common spanning tree reduces computational time required in fabric programming. The spanning tree has a root switch and spans throughout the fabric to reach all of the switches and HFIs that are members of the multicast group. Since this tree is common across multicast groups, it is optimized to compute the MFT data associated with each switch only once. This common data is augmented with ports participating in a given multicast group at the switch when programming the multicast forwarding tables (MFTs).

The FM allows the root of the spanning tree to be configured using the `Sm.Multicast.RootSelectionAlgorithm` parameter. A goal of the spanning tree calculation is to select a switch at the center of the fabric for the root of the spanning tree. A good heuristic for a switch at the center of the fabric is a switch that has the least total cost to all other switches. Another option is a switch that has the least worst case cost to other switches.



On fabric change, the root switch using cost heuristics can change, resulting in a reconstruction of the spanning tree and potential disruption due to reprogramming of the MFTs. The SM's `MinCostImprovement` parameter can determine how much improvement is needed before a new spanning tree root is selected. Disruption to in-flight multicast traffic can be avoided or limited to cases where the fabric has changed significantly enough to provide sufficient benefit to justify a change by using these parameters.

The SM's DB Sync capability synchronizes the multicast root between the master and standby SMs. During SM failover the multicast root can be retained and limit disruption to multicast traffic in flight.

### 2.6.5 Multicast Spanning Tree Pruning

A complete tree unconditionally includes all switches. When HFIs request to join or leave the multicast group the SM only needs to program the switch immediately next to the HFI. This is optimized to program only those MFT blocks which have changed since the last sweep.

A pruned tree omits switches that do not have HFIs as members of the group, as well as intermediate switches that do not need to be in the group. A pruned tree reduces multicast traffic internal to the fabric when only a small subset of nodes are part of a given multicast group. However, the time to add or remove HFIs from the group can be significantly higher as many intermediate switches may need to also be programmed for the group.

The default is a complete tree. This has been found to work very well in High Performance Computing environments. Such environments typically have very little multicast traffic with the vast majority of traffic being IPoIB ARP packets which need to be broadcast to all nodes running IPoIB. The default allows IPoIB hosts to come up and down quicker as may be common in environments that restart compute nodes between jobs.

## 2.7 Congestion Control Architecture

The objective of Intel® Omni-Path Congestion Control Architecture (CCA) is to reduce the propagation of fabric congestion during oversubscription scenarios such as many-to-one traffic. By configuring CCA-specific switch and fabric interface parameters, the fabric can throttle the packet transmit rate on the sending hosts, thereby preventing the spread of congestion through the fabric.

Below is a listing of some of the core ideas in Intel® Omni-Path Architecture's CCA implementation:

1. Intel® Omni-Path Architecture's approach to fabric congestion is holistic in nature and addresses congestion through a combination of features: Adaptive routing, dispersive routing and CCA.
2. Intel® Omni-Path Architecture's Adaptive Routing and Dispersive routing features are designed to optimize and balance traffic in the core ISLs of the fabric.
3. The CCA implementation in the FM is focused on preventing the spread of congestion into the core fabric from over subscribed HFIs (receivers of many to one traffic).

CCA is configured by the SM and automatically used by Intel® Omni-Path HFIs for both Verbs and PSM traffic. In addition, CCA can be configured per MPI application through a simple set of PSM environment variables. Additional PSM debug logging can be enabled to facilitate analysis of PSM CCA events when needed.

The FM configuration file allows CCA settings to be configured for switches and fabric interfaces.

## 2.8 Packet and Switch Timers

The Intel® Omni-Path Architecture allows for assorted timers and lifetimes to be set to avoid unforeseen situations that can cause progress to be stalled causing widespread impacts from localized situations (for example, a hung server, broken ISL, and so on). The Intel® OP Architecture SM allows these timers to be configured by the administrator.

### 2.8.1 Switch Timers

Every switch supports the following timers:

- HeadOfQueueLife (HoqLife)
- SwitchLifeTime
- VLStallCount

These three parameters are applicable to the fabric globally. However, HoqLife may be specified per Virtual Fabric.

These can be used to relieve fabric congestion and avoid fabric deadlocks by discarding packets. Discards help prevent back pressure from propagating deep into the core of the fabric, however such discards cause end nodes to time-out and retransmit the lost packets.

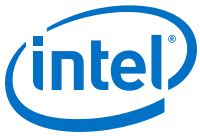
If a packet stays at the Head of a Switch Egress Port for more than HoqLife, it is discarded. Similarly a packet queued in a switch for more than SwitchLifetime is discarded. SwitchLifetime and HoqLife can also be set to infinite in which case no discards will occur.

VLStallCount controls a second tier, more aggressive discard. If VLStallCount packets in a row are discarded due to HoqLife by a given VL on an egress port, that egress port's VL enters the VL Stalled State and discards all that VL's egress packets for 8*HoqLife.

Packets discarded for any of these reasons are included in the TxDiscards and Congestion Discards counters for the Port, which can be queried using FastFabric. Such discards are also included in the Congestion information monitored by the PM and available using FastFabric tools such as `opatop` and `opapaquery` and the Fabric Manager GUI. A congestion that is severe enough to cause packet discards is given a heavy weight in the PM Congestion Group so that it does not go unnoticed.

### 2.8.2 Packet LifeTime

Within an HFI every Reliable Queue Pair (QP) has a time-out configured. If there is no acknowledgment (ACK) for a transmitted QP packet within the time-out, the QP retries the send. There is a limit on retries (up to 7) after which the QP fails with a Retry Timeout Exceeded error.



Intel® Omni-Path Architecture defines that the timeout for a QP should be computed based on the Packet LifeTime reported by the SA in a PathRecord. The LifeTime represents the one way transit time through the fabric. Therefore, the actual QP timeout is at least 2x the Packet LifeTime (plus some overhead to allow for processing delays in the HFI at each end of the fabric).

Careful selection of Packet LifeTime (and QP time-outs) is important. If time-outs are set too large, then the impact of a lost packet could be significant. Conversely if the time-outs are set too low, then minor fabric delays could cause unnecessary retries and possibly even Retry Timeout Exceeded errors and the resulting disruption of applications.

The SM allows for two approaches to configure Packet LifeTime:

- Constant
- Dynamic

*Note:* Some applications, especially MPI middleware, have independent configuration of QP time-outs and ignore the values provided by the SM. For such applications configuration of SM Packet LifeTime has no effect.

The constant approach causes the SM to return the same value for Packet LifeTime for all queries.

The dynamic approach causes the SM to return a different value for the Packet LifeTime depending on the number of hops through the fabric in the given path. This allows the SM to account for the fact that longer routes have more opportunities for congestion, queuing delays, etc, and should have larger time-outs. When using the dynamic approach, a unique Packet LifeTime can be configured for one hop to nine hop paths. Paths greater than nine hops will all use the nine hop value.

## 2.9 SM Sweep Optimizations

### 2.9.1 Optimized Fabric Programming

The Intel® Omni-Path Architecture allows the SM to route SMA packets using "Directed Routed" or "LID Routed" mechanisms. Packets can also be routed with a mixture of these two mechanisms. LID routed packets follow the normal routing used by other traffic and are fully routed by hardware with low latency. Directed Routed packets have the route explicitly specified in the packet, hop by hop, and allows the SM to access components in the fabric prior to having the Switch Routing tables fully programmed. Directed route packets are typically routed in switch firmware and experience higher latency at each switch hop. Typically LID-routed SM packets incur much lower latency while traversing the fabric.

To optimize fabric programming in large fabrics, the FM supports "Early LID Routing." The SM programs routing tables in a "cresting wave" approach such that the majority of fabric programming can use LID-routed packets or packets that are LID-routed all the way to their final hop. The net result of this advanced mechanism is fast programming of the fabric and therefore more rapid fabric initialization and change handling.

### 2.9.2 Scalable SMA Retries

Normally, SMA packets are not flow controlled. However, to optimize fabric programming time, the FM can be configured to issue multiple SMA packets in parallel. This approach can result in occasional packet loss, which can have a negative effect on the SMs performance. To allow for rapid recovery from such packet loss, while not causing excessive retries to sluggish nodes, the FM allows configuration of packet retry intervals and retry limits. By default a randomized backoff algorithm is used to balance the overhead of retries against the need for a rapid recovery from the occasional lost packet. In addition the FM allows flow control to be enabled or disabled for the management VL (VL15).

### 2.9.3 Cascade Activate

When new links come on-line, they start in Linkup/Init state. After the SM has completed programming a given port, it moves the port to Armed. After the SM completes programming all ports in the fabric, it needs to bring them to Active, so they can be used for normal non-management traffic.

The Cascade Activate feature of the SM permits the final move to Active to be automated using the switches in the fabric. When Cascade Activate is enabled, the SM can Active all the ports in the fabric with as few as one packet. This mechanism permits the Armed ports to be treated like dominos, where the arrival of a single packet into the switch on an armed port results in all the armed ports of the switch being activated and all the newly activated ports can then trigger their neighbor ports to continue the chain reaction to activate other ports in the fabric. The result is a very rapid armed-to-active transition for a newly initialized fabric.

As a safety net, after performing the cascade activation, the SM still makes a sweep of all the switches and double checks that all previously armed ports have moved to active. If any are found that did not activate, the SM activates them directly.

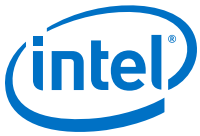
### 2.9.4 Cable Info Caching

To facilitate fabric deployment verification and fabric analysis, the SM can retrieve cable info data from each cable in the fabric. Because the cable EEPROM access times can be slow, the SM provides optimizations in this area.

To optimize SM sweep time, the cable info is cached and only retrieved from links when they transition to Linkup/Init. In addition, the SM can further optimize the gathering of cable info by limiting the accesses to a single end of each cable. This optimization assumes the transceivers on each end of an optical cable are the same and will report only the manufacturing information for one end of the cable. If desired, the gathering of cable info can also be disabled.

## 2.10 Link Bandwidth Negotiation

The Fabric Manager has the ability to set bandwidth limits for links it accepts into the fabric. This capability is enabled by configuring Link Speed and Link Width policies. The width policy allows the administrator to set a minimum width for a link that is acceptable. For example, the administrator can specify a link width policy that allows only links containing more than a minimum number of data lanes to join the fabric.



After a link is accepted into the fabric it may, at a later time, dynamically downgrade its lane count due to detected errors. The Fabric Manager has the ability to limit how many lanes are dropped in this manner. If the link attempts to drop more lanes than the SM allows, the link will bounce and reenter the initial stages of negotiation.

For both of these mechanisms, when the FM finds a port that is out of policy, it leaves the port in Init (or bounces it), lights its beaconing LED, and sets the `LinkInitReason` for the port to indicate a reason of "Out of Policy". These mechanisms make it easier for a technician or sysadmin to identify such ports and understand why they are not being included in the fabric.

## 2.11 Fabric Diagnostics

To aid in fabric diagnostics, several port level status fields are tracked by the SM.

`LinkInitReason` stores the reason why the SM left the port in the Init state. The SM clears this field when transitioning a link out of the Init state.

`LinkDownReason` tracks the last reason a link went down. The SM reads this field and stores it to make available through SA queries.

`PortErrorAction` is a field programmed by the FM that allows the administrator to select error conditions that should result in a port being taken down.

## 2.12 Port Beaconing

When the FM encounters a link or port with problems, it will turn on the beaconing LED on one or both sides of the link. This occurs in the following situations:

- A port fails to arm or activate when instructed by the SM
- A link bandwidth violation occurs
- A port is on a quarantined node

The FM will turn off a port's beaconing LED next time it transitions the port from the Init state to Armed. In a typical use case, the LED would signal a port problem that is then addressed by the system administrator. Once resolved, the port would be bounced, which results in a new SM sweep and the SM arming the port and turning off the LED.

## 2.13 Fabric Change Detection

An important aspect of the Subnet Manager (SM) is its ability to rapidly detect and respond to fabric changes. This is accomplished using the Fabric Sweep process. The SM can start a fabric sweep due to any of the following:

- Fabric Change Trap from an HFI or a Switch. Typically it would be a port going up or down due to reboot, cable insertion/removal, and so on.
- Scheduled periodic sweep (`SweepInterval`).
- Initial FM startup or when FM takes over as Master
- Manually requested by user (`/usr/lib/opa-fm/bin/opafmcmd smForceSweep`)
- When a new redundant FM is started or stopped

The primary mechanism for detecting changes is traps. Traps are asynchronous notifications sent to the SM by devices (typically switches) when ports go up/down, when capabilities change, or links are unstable. The periodic sweep is used as a safety net in case a trap is lost or a device does not issue traps. All Intel® Omni-Path Fabric switches sold by Intel support traps.

In extremely rare situations, you may choose to force a sweep. This is generally not necessary when using fabrics constructed using Intel supplied switches.

### 2.13.1 Handling Unstable Fabrics

During each sweep the SM analyzes the fabric and identifies what, if anything, has changed since the last sweep. If changes are detected, the SM recomputes the routes for the fabric and reprograms the devices in the fabric as needed. The route recalculation is done in a manner that limits the amount of reprogramming needed in the fabric to minimize the disruption.

During the analysis and reprogramming process the fabric may still be changing. In this case errors may occur. When more than the set `SweepErrorsThreshold` parameter of non-recoverable errors (such as a device going offline mid sweep) occur during a single sweep, the SM abandons the sweep and starts it over, to obtain a complete and accurate view of the fabric. The SM does not abandon the sweep more than the set `SweepAbandonThreshold` parameter of consecutive times, after which it will do the best it can. This helps to handle a fabric that is constantly changing, such as a fabric with an unstable link.

Similarly, if a port issues more than the `TrapThreshold` number of changes per minute, the SM considers the link unstable and disables the port, removing it from the fabric, preventing any traffic to be routed over it. In this case, the SM turns on the LED of the disabled port to physically flag the port as disabled.

### 2.13.2 Tolerance of Slow Nodes

In rare cases, nodes under a heavy load, such as when running high stress MPI applications, will be slow to respond to SM sweeps. To avoid disrupting application runs, the SM can be configured using `NonRespTimeout` and `NonRespMaxCount` to be more tolerant of such devices and assume their capabilities have not changed since the last successful sweep.

The trade-off in increasing the tolerance is that loss of nodes due to node hangs or crashes will be detected much slower. Typically this capability is only relevant to HFIs. The risk in this feature mainly applies to nodes that hang but keep their link up so that the neighbor switch does not report a port state change.

*Note:* Intel recommends setting `RENICE_IB_MAD=yes` in `/etc/rdma/rdma.conf` to ensure rapid responses by the SMA and actually reduce overhead by avoiding the cost of retries. This option is enabled by default when using Intel® OFA Delta. Intel® Omni-Path Software is installed using the INSTALL script supplied in the IntelOPA-* software packages.

### 2.13.3 Multicast Join/Leave Denial of Service

In rare cases, nodes send excessive multicast creates/deletes several times a second for the same group causing continuous SM sweeps. To stop the continuous SM sweeps, Multicast (MC) Denial of Service (DOS) can be set up in the configuration file to monitor the MC DOS Threshold, set up the interval of monitoring, and either bounce the port or disable the port.

Example:

```
<!-- When McDosThreshold is zero, monitoring of MC DOS is disabled. -->
    <McDosThreshold>0</McDosThreshold>
    <!-- Default interval is 60 seconds.  -->
    <McDosInterval>60</McDosInterval>
    <!-- McDosAction, the action to take if MC DOS is suspected. -->
    <!--    0 = Port will be disabled.  -->
    <!--    1 = Port will be bounced.  -->
    <McDosAction>0</McDosAction>
```

## 2.14 Cable Information

The FM can obtain detailed information about every cable in a fabric. The information resides in EEPROMs found in every cable connector.

Here are some examples of cable information display formats with increasing levels of detail in each example below:

```
QSFP: PassiveCu,   2m VENDOR_NAME   P/N PART NUM         Rev B


QSFP: PassiveCu,   2m VENDOR NAME   P/N PART NUM         Rev B
      Power Class 1, 1.5W max S/N SERIAL NO         Mfg YYYY/MM/DD-LOT
      OPA Cert? N  OPA Rates: 4x25G                     OUI VENDOR_OUI


QSFP: PassiveCu,   2m VENDOR NAME   P/N PART NUM         Rev B
      Power Class 1, 1.5W max S/N SERIAL NO         Mfg YYYY/MM/DD-LOT
      OPA Cert? N  OPA Rates: 4x25G                     OUI VENDOR_OUI
    Cable Type: Passive copper cable
    Max Temp: 0 C


QSFP: PassiveCu,   2m VENDOR NAME   P/N PART NUM         Rev B
      Power Class 1, 1.5W max S/N SERIAL NO         Mfg YYYY/MM/DD-LOT
      OPA Cert? N  OPA Rates: 4x25G                     OUI VENDOR_OUI
    Cable Type: Passive copper cable
    Max Temp: 0 C
    TX SI: CDR: N/A EQ: Fixed Cap: N Auto Cap: N Squelch En: N
    RX SI: CDR: N/A Emph Cap: N Ampl Cap: N
```

## 2.15 SM Loop Test

### 2.15.1 Loop Test Introduction

The SM Looptest is a diagnostic test facility in the SM. As part of this test, the SM stress tests inter-switch links (ISLs) by continuously passing traffic through them. Other tools such as FastFabric or the Fabric Manager GUI can be used to monitor the links for signal integrity issues or other errors. The advantage of the Looptest is that it provides a guaranteed way to test all of the ISLs in the fabric without the need for a large number of hosts or applications.

When the Looptest is started, the SM deliberately sets up certain additional routes with loops in addition to the normal routes that are set up to enable communication with end ports. These loop routes only include ISLs and are not used for communication with end ports, but are used only to stress test the ISLs. Each loop route starts from a given switch, A, uses one of its ISLs to another switch and passes through a set of other ISLs and switches (say B, C, D) and ends up back at switch A but via a different ISL.

*Warning:* Loop test should not be run when applications are running on the fabric. The loop test introduces high volume traffic on the switch ISLs, which would slow down normal application traffic.

*Warning:* Because there are additional LIDs for each of the loop route LIDs, you may observe a large number of additional LIDs in the switch LID Forwarding Tables (LFTs) when the loop test is running.

As part of loop test the SM sets up a large number of these loop routes so as to cover all ISLs in the fabric. The SM associates a LID with each of the loop routes in such a way that a packet sent to that LID enters the loop and spins around the loop utilizing the ISLs on that loop. The SM injects a packet to into each loop by sending a packet to each of the LIDs associated with the loop routes. Those packets loop around the loops and continuously pass traffic on the ISLs.

Once loop test is stopped, the SM invalidates the loop LIDs, which in turn would cause the loop packets to be dropped by the switches and stop the utilization of the switch ISLs.

### 2.15.2 Important Loop Test Parameters

- **Loop path length** – An argument to `opafmcmd` for `smLooptestPathLength` that represents the number of hops the SM checks to find a loop that leads back to the same switch is termed as the loop path length.
- **Number of packets injected** – This argument can be set in `opafm.xml` using `LoopTestPackets` or `opafmcmd smLooptestInjectPackets`. It represents the number of 256-byte packets the SM injects into each of the loops. These packets loop around on the ISLs.

These parameters can be configured as part of the loop test.

### 2.15.3 Loop Test Fast Mode

Intel recommends the customers use the fast mode for ISL validation and link integrity testing. In the fast mode, the loop test does not attempt to include each ISL in all possible loops, but includes it in at least the specified number of loops (this value is controlled using the MinISLRedundancy parameter). When using fast mode the computations are less expensive and finish faster, which allows the loop test to be

started quickly. The loop test fast mode uses a MinISLRedundancy value of four by default (for example, each ISL is included in at least four loops). There is an important reason for including an ISL in multiple loops – when an ISL goes down in the fabric, the loop that this ISL is part of, is broken and therefore other ISLs in the loop will no longer be tested. But if each ISL is included in multiple loops, the other ISLs in the broken loop are also part of other loops and will continue to see traffic and therefore continue to get tested (albeit at a slightly lower utilization).

In typical fast mode operations (with the default MinISLRedundancy of 4), injecting five packets into each loop is sufficient to get a high utilization on the ISLs.

### 2.15.4 Loop Test Default Mode

By default the loop test runs in default mode. In the default mode, the SM uses an exhaustive approach to set up loop routes and includes each ISL in as many loops as possible. This ensures that each ISL is exactly in the same number of loops and hence will see the same amount of utilization. But finding all possible loops is computationally intensive and can take a long time.

In the default mode, to keep the computations to find loops to a manageable level, the SM by default uses only a path length of three i.e., three hop loops. Three hop loops are usually sufficient for checking ISLs when there are external cables between leafs of the same switch chassis or in other small fabrics but is not sufficient for checking all ISLs in large fabrics that involve multiple switch chassis. For such fabrics the Loop Test Fast Mode should be used.

In typical default mode operations, injecting one packet into each loop is sufficient to get a high utilization on the ISLs included in the loops.

### 2.15.5 SM Loop Test Setup and Control Options

There are two ways to run the SM loop test. The test can be run using CLI commands or can be configured to run using the FM Configuration file. Both of these methods are discussed in the following sections.

- Run the SM Loop Test using opafmcmd CLI Commands on page 61
- Set up the Configuration File to Run the SM Loop Test on page 63

### 2.15.6 Run the SM Loop Test using opafmcmd CLI Commands

#### Requirements to run the SM Loop Test

- FM must be running for host or embedded nodes in order for the SM loop test to run.
- The FM must be master.
- To run in fast mode:
  - Fast mode is enabled when `smLooptestFastModeStart` is used to start the test.
  - Enabling fast mode automatically sets the path length to 4 and the inject on each sweep to disabled.
  - The fast mode can only be disabled by running the `smLooptestStop` command that stops the loop test completely.

### Loop Setup Options

- Set up the path length to set the number of hops that the FM checks to find a loop that leads back to the same switch:

```
/usr/lib/opa-fm/bin/opafmcmd smLooptestPathLength
```

  Values are 2-4 with a default of 3

  Loop path length is set to 4 for Fast Mode

  *Note:* Intel recommends setting this parameter before starting loop test. If the parameter is changed after starting loop test, packets must be injected with the `opafmcmd smLooptestInjectPackets` option.

- Set the minimum number of loops in which to include each ISL:

```
/usr/lib/opa-fm/bin/opafmcmd smLooptestMinISLRedundancy
```

  Applicable only when running in Fast Mode

  Default is 4

### Packet Injection Options

- Packets are injected if the loop test is started normally and a number of packets has been specified as an argument to the command:

```
/usr/lib/opa-fm/bin/opafmcmd smLooptestStart 4
```

- Five packets are injected by default if loop test is started in Fast Mode:

```
/usr/lib/opa-fm/bin/opafmcmd smLooptestFastModeStart
```

  In fast mode, once loop test is started packets are injected only once. They are not injected at every sweep.

- Packets can be injected once the loop test has been started:

```
/usr/lib/opa-fm/bin/opafmcmd smLooptestInjectPackets numPkts
```

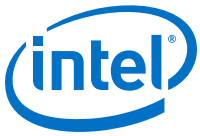
- Packets can be injected on each sweep:

```
/usr/lib/opa-fm/bin/opafmcmd smLooptestInjectEachSweep
```

  The `opafmcmd smLooptestInjectEachSweep` parameter can be used in both fast and default mode loop test case to control whether each sweep will inject a packet or not. In fast mode, the default is not to inject on each sweep, but it can be changed to inject on each sweep after the loop test is started. In default mode, this parameter can be changed either before or after loop test is started to enable packet injection on each sweep.

### Other SM Loop Test Commands

For a full list of commands that can be used for the SM Loop Test, refer to FM Loop Test on page 168.



To stop the SM Loop Test:

```
/usr/lib/opa-fm/bin/opafmcmd smLooptestStop
```

## 2.15.7 Set up the Configuration File to Run the SM Loop Test

The FM configuration file can be set up to automatically run the SM loop test when the master FM is started. The parameters are added to the `Miscellaneous` section of the configuration file and can include the following three items:

- Run at start up:

  ```
  <LoopTestOn>1</LoopTestOn>
  ```

  1 = enabled

  0 = disabled

- Run in fast mode:

  ```
  <LoopTestFastMode>1</LoopTestFastMode>
  ```

  1 = enabled

  0 = disabled

  Once fast mode is enabled, whenever loop test is started it will use the fast mode.

- Set the number of packets injected for each test run:

  ```
  <LoopTestPackets>4</LoopTestPackets>
  ```

  With fast mode, you need around four packets to get a high utilization on the links.

## 2.15.8 Reports from SM LoopTest

The following reports can be requested for the SM loop test.

### SM Loop Test Show Loop Paths

The following is an example of the SM Loop Test Show Loop Paths Report.

```
# /usr/lib/opa-fm/bin/opafmcmd smLooptestShowLoopPaths
   Connecting to LOCAL FM instance 0
   Successfully sent Loop Test Path show for node index (all) to local SM instance

   Node Idx: 1, Guid: 0x00066a2400000000 Desc 9024 #0
   Node Idx: 2, Guid: 0x00066a2400000003 Desc 9024 #3

-------------------------------------------------------------------------------
   Node     Node                              Node     Path
   Idx      Lid            NODE GUID        #Ports    LID      PATH[n:p->n:p]
   ----   ---------  --------------------   -------- ------   ------------------
    1       0x0001    0x00066a2400000000      36      0x0080  1:1->2:1 2:2->1:2
    1       0x0001    0x00066a2400000000      36      0x0081  1:1->2:1 2:3->1:3
    1       0x0001    0x00066a2400000000      36      0x0082  1:1->2:1 2:4->1:4
    1       0x0001    0x00066a2400000000      36      0x0083  1:1->2:1 2:5->1:5
```

```
                                        *
                                        *
                                        *
```

### SM Loop Test Show Switch LFT

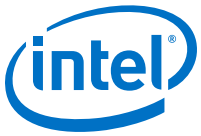
The following is an example of the SM Loop Test Show Switch LFT Report.

```
# /usr/lib/opa-fm/bin/opafmcmd smLooptestShowSwitchLft

Connecting to LOCAL FM instance 0
Successfully sent Loop Test LFT show for node index (all) to local SM instance

Node[0001]  LID=0x0001  GUID=0x00066a2400000000 [9024 #0] Linear Forwarding Table
   LID      PORT
   ------   ----
   0x0001   0000
   0x0002   0001
   0x0003   0010
   0x0004   0020
   0x0005   0021
   0x0006   0022
   0x0007   0023
      *
      *
      *
```

### SM Loop Test Show Topology

The following is an example of the SM Loop Test Show Topology Report.

```
# /usr/lib/opa-fm/bin/opafmcmd smLooptestShowTopology
Connecting to LOCAL FM instance 0
Successfully sent Loop Test topology show to local SM instance

sm_state= MASTER count= 29796 LMC= 0, Topology Pass count= 4, Priority= 0, Mkey=
0x
0000000000000000

--------------------------------------------------------------------------------
george HFI-1
--------------------------------------------------------------------------------
Node[  0] => 0079159a00117500  (1)  ports=1, path=
Port ----  GUID  ----  (S)    LID      LMC      _VL_ __MTU__ __WIDTH__
___SPEED____
 CAP_MASK N#  P#
 1   0079159a00117500 4  LID=000d  LMC=0000   2  1  4k  2k  1X/4X  4X 2.5-10
2.5
0761086a  1  19

--------------------------------------------------------------------------------
9024 #0
--------------------------------------------------------------------------------
Node[  1] => 0000000000066a24  (2)  ports=36, path= 1
Port ---- GUID ----   (S)   LID       LMC      _VL_ __MTU__ __WIDTH__ __SPEED___
 CAP_MASK N# P#
 0   0000000000066a24 4  LID=0001  LMC=0000   8  8  2k  2k 1X/4X 4X  2.5/5 5.0
  00100848 1   0 1
 1   0000000000000000 4                       8  8  2k  2k 1X/4X 4X  2.5/5 5.0
  00000048 2   1
 2   0000000000000000 4                       8  8  2k  2k 1X/4X 4X  2.5/5 5.0
  00000048 2   2
                                        *
                                        *
                                        *
```



### SM Loop Test Show Configuration

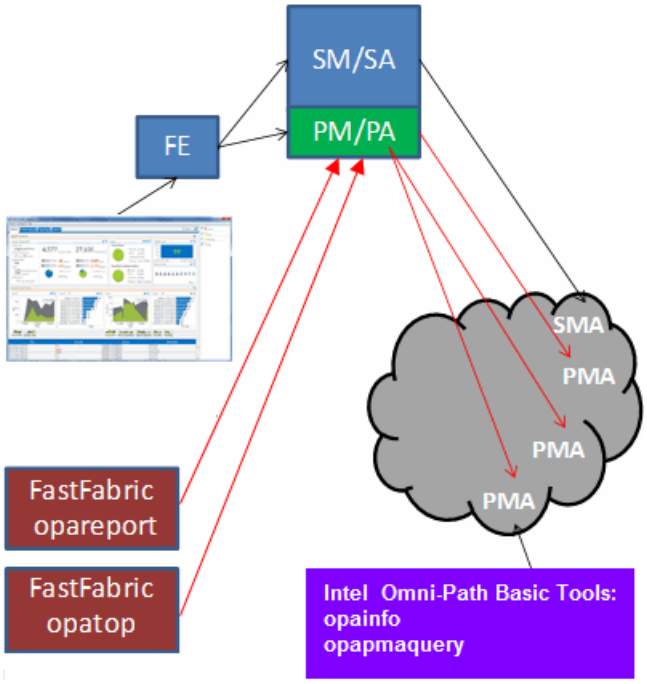
The following is an example of the SM Loop Test Show Configuration Report.

```
# /usr/lib/opa-fm/bin/opafmcmd smLooptestShowConfig
 Connecting to LOCAL FM instance 0
 Successfully sent Loop Test configuration show to local SM instance
 Loop Test is running with following parameters:
  Max Path Length    #Packets    Inject Point
----------------    -------    ----------------
       4             00005       All Nodes

FastMode=0, FastMode MinISLRedundancy=4, InjectEachSweep=1, TotalPktsInjected
since start=0
```

# 3.0 PM Features

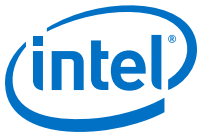
The Performance Manager (PM) is the Fabric Management entity responsible for monitoring fabric information related to the port and virtual lane level counters and the picture that they convey. Each port in the fabric monitors provides counters that tally information such as the amount of data transmitted and received, as well as occurrences of errors that indicate problems at the port and link level. Refer to the following figure for an example of the management paths.

**Figure 13. Management Paths**




The PM performs regular sweeps of the fabric to gather and summarize both error and performance statistics. This capability allows for centralized control of PMAs and can avoid the potential confusion when tools (`opareport` and so on) directly clear PMA counters. To avoid the need to clear counters, all Intel® Omni-Path devices support 64-bit counters, which will not overflow within the life of the cluster. As such direct clearing of counters is not recommended, however some tools, such as `opareport` still provide this capability.

The Performance Administration (PA) entity allows a centralized control point for querying the performance data in the fabric. The relationship of the PM and PA mirrors that of the Subnet Manager and Subnet Administration (SM/SA) entities; the PM is the "workhorse," communicating with the Performance Management Agents (PMAs) in the



fabric, and, with the use of a calculation "engine," stores the information in the PA database. The PA is then available as a service, similar to the SA, to provide performance information to client applications in the fabric.

*Note:* The PM when enabled takes control of all PMAs in the system. As such:

- Tools that directly access the PMAs and clear counters should not be used. Such as `opareport -C -M`, and chassis port thresholding (`ismAutoClearConf` for Intel® Chassis should be set to disabled)
- Tools that query the PMA counters may yield unexpected results.

## 3.1 Port Groups

The PA separates the ports in the fabric into groups, which it then monitors according to the performance of the ports in the group. The following pre-defined groups are built into the PM/PA:

- All – all ports in the fabric
- HFIs – all ports on HFIs in the fabric
- SWs – all ports on switches in the fabric

Non-Enhanced and Enhanced Switch Port 0 ports both support PMA statistics.

Within groups, the ports are defined as internal and external.

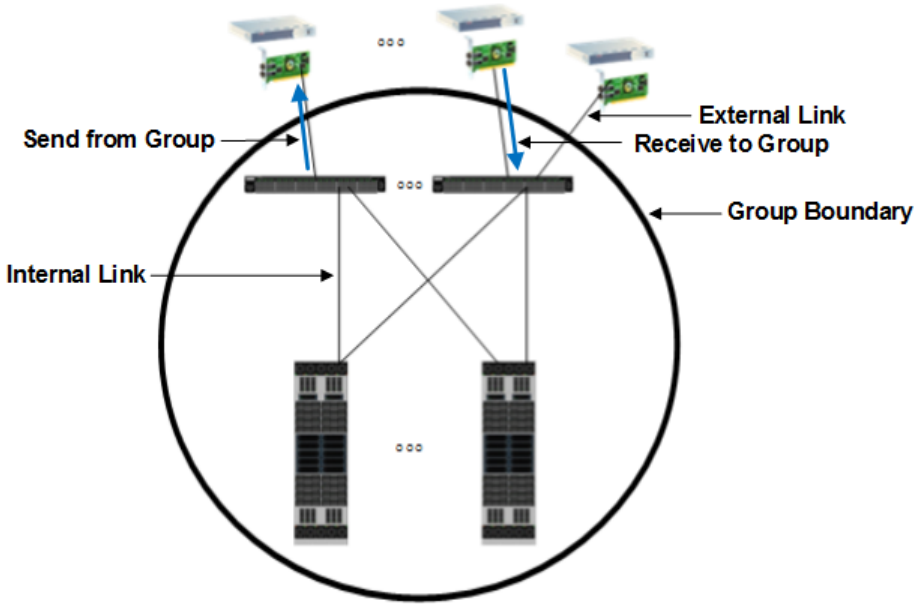
You can also create user defined port groups using device groups (discussed in DeviceGroup Parameters on page 127). Each `PmPortGroup` must have a unique name. The monitor tag is used to link to a device group that is defined elsewhere in the XML file. There is also an enable tag that can be used to disable the user-defined PortGroup.

The layout is as follows:

```
<PmPortGroups>
  <PmPortGroup>
     <Name>Rack1PG</Name>
     <Monitor>Rack1DG</Monitor>
     <Enable>1</Enable>
  </PmPortGroup>
</PmPortGroups>
```

A port is considered an Internal Port when it and its neighbor are both in the group. For example ISLs are Internal to the SWs group. Refer to the following figure for an example of internal and external links. Switch Port Zero is considered internal to any group the Switch is a part of.

**Figure 14. Internal and External Links**




A port for which one side of the link is in the group but its neighbor is not in the group, is considered an External port. For example a link between an HFI and a Switch tabulates both the Switch Port and the HFI port as External to the SWs group.

For External ports, Send (data leaving the group) and Recv (data entering the group) statistics are accumulated. Error counters are also accumulated to reflect the overall status of the link.

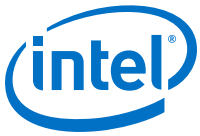
*TFI* refers to any host port that is running a Device Management Agent. Such agents are typically only run on target device nodes such as Block Storage targets running SRP.

*HFI* refers to any host port that is not running a Device Management Agent. This will typically include most or all of the hosts in a cluster (compute, file system, management, login, etc). If the OFA SRP target implementation is mistakenly run, it starts a device management agent and the FM categorizes the node as a TFI, in which case PM monitoring will not include the node in the HFIs PM group. As desired if needed, the sample `opafm.xml` configuration file includes a sample PmDeviceGroup that monitors TFIs; this group is disabled by default.

## 3.2 PM Sweeps

The PM performs regular sweeps of the fabric to get and analyze PMA counters for all the devices in the fabric. The PM Sweep Interval is configurable using the `Pm.SweepInterval` parameter to specify the number of seconds. If set to 0, PM automatic sweeps are disabled. If `Pm.SweepInterval` is not specified, the default is 10, therefore it will be on by default.

Most, but not all, of the counters are 64-bit. As such, the need for the PM to clear a counter should be extremely rare. Sixty-four-bit counters take over a hundred years to overflow.



All Intel® Omni-Path devices support the 64-bit counters, and therefore, the PM sweep interval can be set much larger. Larger values (e.g., 60 seconds) reduce the frequency at which data is gathered by the PM and so decrease fabric jitter and overhead on the FM node. Small values (such as the default of 10 seconds or lower) increase the frequency of the PM's data gathering, providing much more granularity to the data when using analysis tools such as `opareport`, `opatop`, and `opapaquery`. Each sweep typically involves only a few packets per node, so even at faster sweep rates, overhead per node is quite modest.

The majority of information the PM monitors is available on switches, so if desired, HFI jitter can be slightly reduced by setting `ProcessHFICounters` to 0. This option also reduces FM overhead and increases PM performance and scalability.

See the *Intel® Omni-Path Fabric Suite FastFabric User Guide* for more information about tools that can query and/or analyze the data available from the PM (`opareport`, `opatop`, `opapaquery`, and so on).

### 3.2.1 Scalable PMA Retries

To optimize fabric programming time, the FM can be configured to issue multiple PMA packets in parallel. This approach can result in occasional packet loss, which can have a negative effect on PM performance.

To allow for rapid recovery from such packet loss, while not causing excessive retries to sluggish nodes, the FM allows for scalable retries which increase their timeout with each attempt.

## 3.3 PA Images

The PM is constantly monitoring the port counter information in the fabric using periodic sweeps. Each sweep gathers the port counters in the fabric and performs reduction and summary calculations of the data to provide group level information in the PA.

Each sweep is stored as a PA image. An image is a historical representation of the fabric data, a time slice. The number of images stored in the PA is finite, with the oldest images being replaced with new ones as resources dictate. The images are identifiable and may be retrieved for examination as long as they still exist in the PA (if they have not been replaced with a newer image due to resource constraints).

The recent history is retained in the memory of the PM. The amount of recent history is configurable using `Pm.TotalImages`.

The PM also supports a `FreezeFrame` capability for the history. This can allow a client to freeze an existing image to allow extended analysis without subsequent sweeps replacing the image. This is used both for the viewing of any non-live data in `opatop` and the bookmarking of historical data for later analysis. The number of freeze frames is configurable using `Pm.FreezeFrameImages`. This permits tools such as `opatop` to view recent historical data. The history includes performance, error and topology information. When viewing recent history changes, any of these areas can be viewed and analyzed.

## 3.4 PA Client Queries

PA services available to clients include information about the PM/PA configuration, groups, virtual fabrics, images, and ports.

Using a set of Performance Administration MAD packets, client applications in FastFabric communicate with the PA to retrieve data that is then available to show to the end user.

Examples of queries supported by the PA include:

- PM/PA configuration – information such as sweep rate, maximum number of images, and so on.
- List of groups or virtual fabrics
- Group or virtual fabric configuration – list of ports in the group or virtual fabric, including node information
- Group or virtual fabric information – a summary of the statistical categories being monitored for the group or virtual fabric
- Group or virtual fabric focus – a "top talkers" or "top offenders" type summary giving the highest or lowest in contributors to a particular focus area such as bandwidth or error situation
- Port counter management, including retrieval
- Image management, including configuration retrieval and freeze frame management

## 3.5 Error Thresholding

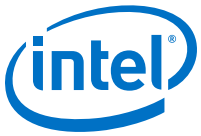
Thresholds can be configured for each category of PMA Counter. These thresholds are used for summary information reported by `opatop` and other FastFabric tools. In addition the `PM.ThresholdsExceededMsgLimit` parameters in the FM configuration can specify how many threshold exceeded events for each class of counters should be logged. This logging can provide a long term history of fabric errors.

## 3.6 Counter Classification

The PM classifies performance and error counters into a number of higher level categories. These categories indicate various issues in the fabric, such as "congestion," "integrity," and "security." The counters of both ports on a link are analyzed, and a value is computed using both sides. So, for example, both the transmit-side counter and the receive-side counter of a link are considered in the computation. The intent of the computation is to narrow the root cause of the issue to the appropriate side (transmit/receive) of the link. Tools such as `opatop` report the category values for the associated port. Detailed drill-down displays can then show the condition and the devices for both sides of the link.

Bandwidth (MBps) and Packet Rate (KPps) are tallied on the send side, using the maximum of the associated port's send counters and the neighbor port's receive counters.

The following table shows the relationship between what counters belong to each category. `Integrity`, `Congestion`, and `SmaCongestion` use a weighted sum specified in the configuration file. `SmaCongestion` uses the same weighting used for



Congestion category, but applies the weights to the associated virtual lane 15 (management VL) counter. `Bubble` uses the maximum value between the sum of the associated port's `XmitWastedBW` and `XmitWaitData` and the neighbor port's `PortRcvBubble`. Security category is calculated similarly to the way `Bandwidth` and `Packet Rate` are calculated.

To minimize space needed in error logs, shortened strings are used to denote reported counters, as shown in the table below. See PM Counters Calculations on page 276 for more details on counters.

**Table 2. PM Conditions and Associated Ports Counter**

| Category | Log String | Associated Port | Log String | Neighbor Port |
|---|---|---|---|---|
| Bandwidth (MBps) (MAX used) | TxD | PortXmitData | RxD | PortRcvData |
| Packet Rate (KPps) (MAX used) | TxP | PortXmitPkts | RxP | PortRcvPkts |
| Integrity (Weight sum) | LWD.Tx | LinkWidthDowngrade Transmit Active Width | EBO | ExcessiveBufferOverrunErrors |
| | LWD.Rx | LinkWidthDowngrade Receive Active Width | | |
| | LQI | LinkQualityIndicator | | |
| | Unc | UncorrectableErrors | | |
| | LD | LinkDowned | | |
| | RxE | PortRcvErrors | | |
| | FMC | FMConfigErrors | | |
| | LER | LinkErrorRecovery | | |
| | LLI | LocalLinkIntegrityErrors | | |
| Congestion (Weight sum) | CD | CongDiscards | RxF<br>RxFPct | PortRcvFECN<br>PortRcvFECN Percentage |
| | RxB | PortRcvBECN (only HFIs) | | |
| | RxBPct | PortRcvBECN Percentage | | |
| | MkF | PortMarkFECN | | |
| | MkFPct | PortMarkFECN Percentage | | |
| | TxTC | PortXmitTimeCong | | |
| | TxTCPct | PortXmitTimeCong Percentage | | |
| | TxW | PortXmitWait | | |
| | TxWPct | PortXmitWait Percentage | | |
| SMA Congestion (Weight sum) | VLCD[15] | [15].VLCongDiscards | VLRxF[15] | [15].PortVLRcvFECN |
| | VLRxB[15] | [15].PortVLRcvBECN (only HFIs) | | |
| | LMkF[15] | [15].PortVLMarkFECN | | |
| | VLTxTC[15] | [15].PortVLXmitTimeCong | | |
| | VLTxW[15] | [15].PortVLXmitWait | | |
| Security (sum) | TxCE | PortXmitConstraintErrors | RxCE | PortRcvConstraintErrors |

***continued...***

| Category | Log String | Associated Port | Log String | Neighbor Port |
|---|---|---|---|---|
| Routing | RxSR | PortRcvSwitchRelayErrors | | |
| Bubble (MAX used) | WBW | XmitWastedBW | RxBb<br>RxBbPct | PortRcvBubble<br>PortRcvBubble Percentage |
| | TxWD | XmitWaitData | | |
| | TxBbPct | PortXmitBubble Percentage | | |

The results of the associations of conditions to ports is as follows:

- **Integrity** – Associated with a port that is having errors in the Physical (Phy) and Link Layers, as well as errors in hardware. The Integrity value is computed as a weighted sum, the weight for each counter is configurable.
- **Congestion** – Associated with the port that is generating more traffic than the neighbor or ports downstream of the neighbor can handle and is causing backpressure in the fabric. The value is computed as a weighted sum and the weight of each counter is configurable.
- **SmaCongestion** – Associated with the management VL of a port that is generating more management traffic than the neighbor or ports downstream of the neighbor can handle and is causing backpressure in the fabric. The value is computed as a weighted sum. SmaCongestion uses the VL 15 counters associated with the port counters used by the congestion category.
- **Bubble** – Associated with the port that is sendling idle bubbles in the middle of packets. The value is computed as a maximum between the sum of XmitWastedBW and XmitWaitData and the neighbor's PortRcvBubble.
- **Security** – Associated with the port that is sending packets that violate security.
- **Routing** – Associated with the port that is receiving packets it cannot route. This is only applicable to switch ports.

*Note:* The PortRcvRemotePhysicalErrors counter is not included in any of these conditions. This counter indicates packets that were marked bad due to discovery of a packet error during switch cut through. When such errors occur the counters included in Integrity allow the issue to be pinpointed. In contrast this counter occurs for every hop along the route and is not very useful for the root causing the Integrity issue.

### 3.6.1 Congestion Statistics

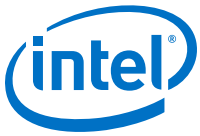
These computed counters parameterize certain counters. The computed counters include:

- `DiscardsPct10` – Percentage of packets discarded relative to total packets sent.
- `UtilizationPct10` – Percentage of data sent through the port relative to the maximum throughput.

See Integrity Weights on page 115 and Congestion Weights on page 115 for more information.

### 3.6.2 Histogram Data

For each condition a histogram is tracked. Each histogram has a set of buckets that count the number of ports in a given group with a certain range of performance/ errors.



For Bandwidth, ten histogram buckets are tracked, each 10% wide. The buckets each count how many ports in a given group had this level of utilization relative to wire speed.

For the remaining conditions (Integrity, Congestion, SmaCongestion, Bubble, Security, Routing), five histogram buckets are tracked. The first four are 25% wide and indicate the ports whose value is the given percentage of the configured threshold. The fifth bucket counts the number of ports whose event rate matched or exceeded the threshold. The overall summary status for a given group is based on the highest event rate for any port tabulated by the group.

## 3.7 PM Running Counters to Support opareport

The PM maintains an internal, single set of running counters. The PM updates these running counters on each sweep and they reflect running totals of all the counters on all the ports. The running totals are accessed via the PA. These counters can be used by tools such as `opareport`, `opafabricanalysis`, and `opaallanalysis` to help analyze long term counter trends.

You can clear the running totals for a given port (or all ports) using the PA. Accessing or clearing the PM running totals does not affect the actual PMA hardware counters. As such, the values shown in the running totals tend not to match the values shown in the PMA or results from low level tools such as `opapmaquery`, `opainfo`, or the switch chassis CLI command `ismPortStats` that directly access the PMA.

The PM internally retains one copy of the running counters for each port.

When the PM is running, `opareport` accesses data from the PM, therefore options such as `-o` errors and `-C` access PM Running Counters and do not affect the actual hardware PMA counters unless the PmaDirect command line option is added to the opareport query.

## 3.8 Short-Term PA History

The PA stores a small number of images in memory, usually 10. This number is limited by the amount of RAM on the machine, as the images are stored in memory and can be quite large. The objective of Short-Term PA history is to save more than just those images, so that fabric performance can be better tracked and analyzed over the last 24 hours or more. This is done by compressing the images and saving them to the disk. Client programs can then access the historical data using PA MADs, in the exact same manner that the in-RAM images are accessed (`opatop`, `opapaquery`).

To save disk space, the data in the images can be combined into "composite images" before they are stored. A composite image is essentially the same as a regular image, but it covers a longer length of time. For example, if the regular images cover 10 second intervals, and there are three images per composite image, then the data in the composite image would represent a 30 second interval.

Short-Term PA History images can also be frozen, similar to how in-RAM images are frozen. Freezing a Short-Term PA History image causes it to be cached in memory. It remains cached until it is released, or a different Short-Term PA History image is frozen. Only one Short-Term PA History image can be frozen at a time. Frozen Short-Term PA History images do not count towards the number of freeze-frames of in-RAM images.

Short-Term PA History images persist across PM restarts. When the PM starts up, it checks to see if there are already any Short-Term PA History files stored on the disk. If this is the case, it will be able to access those older files, as well as any new files that it creates.

The PM has been updated, allowing you to change the weight and threshold of PA categories. This enables you to recalculate values using already stored port data. Also, PA query time, memory usage, and disk space usage will decrease with the new PM History version. To change thresholds and weights, edit the `opafm.xml` file, and restart the FM.

*Note:* The FM no longer supports the previous short term history (STH) file after this change. The old files do not need to be removed as they will age out normally.

## 3.9 PA Failover

The PA supports seamless failover without loss of counter historical data.

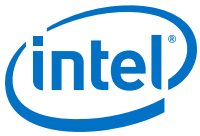
A standby FM does not sweep the fabric. Instead, the standby FM monitors and synchronizes with the master FM. This synchronization permits the standby FM to take over the fabric seamlessly. In the case of the PM, the standby PM synchronizes in memory and on disk PM sweep data from the master.

The synchronization (dbsync) is performed in-band via an FM private communication protocol. The synchronization of PM data is managed and controlled by the master FM. The synchronization performance depends on PM parameters, fabric size, and number of virtual Fabrics enabled. Composite images are synchronized as a single image, so for larger fabrics, use of composite images can reduce synchronization overhead.

The master FM configuration includes controls over the frequency of synchronization and a limit on the data rate consumed by the synchronization of PA data.

The PA (and SA) of a standby FM are not active. As such a standby FM do not respond to PA queries. All PA queries should be directed to the master FM.

The purpose of the PA Failover feature is to allow PA responsibilities to transition (failover) from the master FM to a standby FM in situations where the MASTER has been replaced (failure or preemption/traffic flow optimization by a higher priority FM). Within a FM, the SM and PM have a "unified" relationship in that they access common data structures in memory. As such they failover together as a unit. The PM and PA also access common data structures in memory and failover together.



# 4.0 FM Features

## 4.1 Redundant FMs in a Fabric

It is possible to deploy more than one FM in a fabric. Normally deploying more than one FM is done to provide redundancy. Multiple FMs are a way to ensure that management coverage of the fabric continues unabated in the case of the failure of one of the FMs or the platform on which it is being run.

*Note:* It is important that the fundamental parameters, such as the SubnetPrefix, match in all SMs managing a fabric. Intel also recommends that all redundant SMs are at the same revision level. Failure to follow these recommendations can result in fabric disruptions when SM failover occurs, or if an SM fails to come online. Using the Configuration Consistency Checking feature, the FM can ensure that all redundant FMs have a comparable configuration. See Fabric Manager Configuration Consistency on page 76 for more information.

### 4.1.1 FM Role Arbitration

FMs arbitrate for the master role using in-band Intel® Omni-Path messages. Arbitration is based on the SM Priority parameter and the GUID of the port on which the FM is running.

FMs arbitrate for the master roles based first on the SM Priority parameter, which is a configurable value in the range 0-15, followed by the GUID of the port on which the FM is running (includes switches and HFIs).

The arbitration rules are:

- The FM with the highest priority value is master FM.
- In the case of FMs with equally high priority values, the FM with the lowest port GUID is the master FM.

### 4.1.2 Master FM Failover

Failover from master to standby FM is essentially seamless. Because the master and standby FMs participate in a database synchronization mechanism, the new master FM is equipped with enough data to continue managing the fabric upon takeover.

The change of master FMs occurs without loss of multicast membership and minimal disruption of routing. In most instances, data traffic will continue without interruption.

### 4.1.3 Master Preservation – Sticky Failover

Multiple FMs in a fabric can be configured to support "sticky" failover. Normally when the user configures the FMs using the SM Priority value to designate the hierarchy of master and standby FMs, that order always holds true during arbitration. Thus, upon failover of the master, a standby becomes the new master until the original master is restarted, whereby the original master reclaims the master role.

With the sticky failover feature, the FMs can be configured so that once a FM takes over the master role, it retains that role even if the original master returns to the fabric and arbitration rules dictate that it should reclaim the role. Using the sticky failover feature minimizes disruption to the fabric when connectivity to the host running the master FM is unstable.

#### 4.1.3.1 PM – Master, Standby, Failover, Sticky Failover

The PM is built into the SM, therefore the master PM will always be the exact same FM as the master SM.

#### 4.1.3.2 Sticky Failover and Elevated Priority

Elevated priority is supported for the SM/PM. It is the priority the manager will run at, once it becomes master. This feature defaults to off (0).

By configuring an elevated priority, the user can configure a fabric to only failover managers when the master fails, rather than re-negotiating whenever a new manager comes online. This can be achieved by configuring the elevated priority to be higher than any manager's normal priority.

When elevated priority is configured, a renegotiation at normal priorities can be forced using the restore priority operation. This will reduce the master's priority back to normal and renegotiate for master. When the present master is not the preferred master, this will allow a failover back to the normally preferred master to occur. For a host FM this is accomplished using the `opafmcmd` utility. For an embedded FM, this is accomplished using the `smRestorePriority` chassis CLI command.

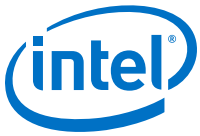
The intention of this feature is to allow failover to occur only once when a manager is experiencing intermittent failures, rather than repeatedly passing control back and forth, disrupting fabric operations. Therefore, it is a best practice to configure all FM managers with the same elevated priority. However, this is not required.

A typical configuration would be an elevated priority of 15 for all FMs, a normal priority of 8 for the preferred master and a normal priority of 1 for the preferred standby.

*Note:* Using elevated priorities permits a secondary FM to boot first and then if the preferred primary FM boots afterward, the preferred primary takes over and elevate its priority. If the preferred primary boots first, it runs without an elevated priority until a secondary comes online, at which time the preferred primary remains master and elevate its priority.

### 4.1.4 Fabric Manager Configuration Consistency

When there are redundant FMs in a fabric, it is important they all have the same configuration for key fabric operational parameters. (However, other parameters, such as FM priority, can purposely be different between FMs.) This is required such that



upon FM failover there will be no disruption nor changes in fabric operation. To ensure FM configuration consistency, there is an optional configuration check feature in the FM that is enabled by default. The Configuration Consistency Checking is applied to the Subnet Manager and Performance Manager.

When enabled, the configuration between redundant managers is checked using checksums. If an operational inconsistency is detected a message can be logged or the standby manager can be changed to inactive. See Command Line Interface (CLI) Taxonomy on page 176 for more information about CLI commands such as `config_diff` which can help compare FM configuration files.

When an inactivation is performed, for the SM, the standby SM state is set to `InActive`. For the PM the standby shuts down with a log message. Multiple FEs operate independently, and configurations are not checked for consistency. An `InActive` SM can only be reactivated by manually resolving the configuration inconsistencies and restarting the FM using the command `systemctl restart opafm`.

#### 4.1.4.1 Parameters Excluded from Configuration Consistency Checking

The following set of parameters are excluded from configuration consistency checking. See Fabric Manager Configuration on page 80 for descriptions of the parameters.

##### Common and Shared Configuration

The following parameters are excluded for all managers:

HFI, Port, Debug, RmppDebug, Priority, ElevatedPriority, LogLevel, LogFile, LogMode, *_LogMask, SyslogMode, Name, PortGUID, CoreDumpLimit, and CoreDumpDir

Disabled Virtual Fabrics are excluded from the consistency checksum. For enabled Virtual Fabrics, no parameters are excluded from the consistency checksum.

##### SM Configuration

The following SM-specific parameters are excluded: TrapLogSuppressTriggerInterval, SmPerfDebug, SaPerfDebug, DebugVf, DebugJm, DebugLidAssign, LoopTestOn, LoopTestPackets, LoopTestFastMode, LID, DynamicPortAlloc, and SaRmppChecksum

Also all MLIDShared, CongestionControl, and AdaptiveRouting parameters are ignored when the given section is not enabled.

##### PM Configuration

The following PM-specific parameters are excluded:

ThresholdsExceededMsgLimit (entire section) and StorageLocation (in the ShortTermHistory section).

##### FE Configuration

Multiple FEs are independent, and the configurations are not checked for consistency.

#### 4.1.4.2 Changes Needed to Run ESM/HSM as Redundant Pairs

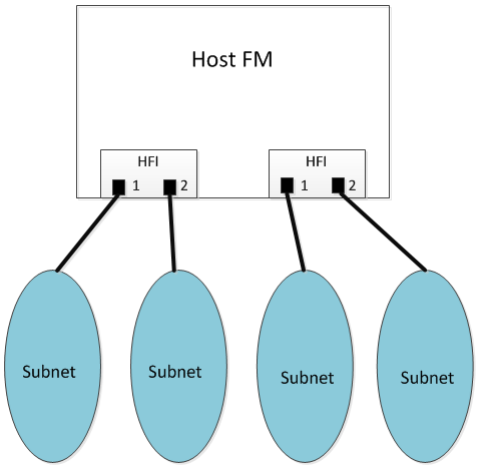
To run the Host Subnet Manager (HSM)/Embedded Subnet Manager (ESM) as redundant pairs, you must modify the default HSM configuration file. See Host FM on page 28 and Embedded FM on page 28 for more details. Change the following parameters:

- disable `ShortTermHistory`
- decrease the `SubnetSize` to match the value of ESM `SubnetSize`

## 4.2 Multiple Subnet Support in Host FM

The host FM suite may be used to manage more than one subnet, sometimes described as *multiple planes*. Multiple planes are distinct isolated fabrics for which there is no interaction between ports in different fabrics. Refer to the following figure.

**Figure 15. Multiple Subnet Support in Host FM**




As long as a host server has multiple HFI ports, the host FM can be configured to run separate *instances* over each HFI port available. Each HFI port is cabled to a separate plane, and the configuration denotes which port the instance should use to access the fabric.

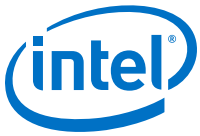
The host FM can support four planes with four separate HFI ports on the server.

Each FM instance is a separate process, and there is no interdependency among FM instances. An individual instance can be started or stopped using the command:

```
/usr/lib/opa-fm/bin/opafmctrl.sh [start|stop] -i #
```

## 4.3 Subnet Configuration Example

Fabric managers can run on various hosts. In smaller clusters the fabric manager may be run on a server that also performs other management functions (job scheduling, provisioning, etc). In larger clusters management functions should be separated onto different servers per management function.



*Note:* Intel does not recommend running compute application software on hosts that are dedicated to Subnet management.

The following table shows an example subnet in which hosts A and B share subnet management responsibilities; host A runs the primary Subnet Manager while host B is the standby. A Fabric Executive is active on both hosts.

**Table 3. Subnet Configuration Example**

| Host | FE | PM | SM |
|---|---|---|---|
| A | yes | yes | yes |
| B | yes | yes | yes |
| C,D,... | no | no | no |

## 4.4 FM Logging

The FM performs logging to aid fabric and FM debug. In the embedded environment, the user uses the logging controls to dictate where the log messages are sent (for example, internal buffer, external syslog server). In the host environment, the logs are sent to the system log on that system and may also be sent to a centralized syslog server.

Log messages follow the Intel convention of severities, for instance, FATAL, ERROR, WARNING, INFO, and so on.

Many important fabric events are logged by the SM, such as ports going up or down, nodes not answering, nodes appearing or disappearing, SMs joining or leaving the fabric, SM state transitions, synchronization errors, and so on.

# 5.0 Fabric Manager Configuration

**Table 4. Intel® Omni-Path Architecture Standard Terms**

| Term | Definition |
|---|---|
| Endnode | An endnode is any node that contains a fabric interface. Also referred to as HFI or TFI, two specific types of endnodes. |
| Host | One or more HFIs governed by a single memory/CPU complex. |
| Subnet | A set of ports using Intel® Omni-Path Architecture, and associated links, that have a common subnet ID and are managed by a common Subnet Manager. |
| Port | Location on a fabric interface or switch to which a link connects. There may be multiple ports on a single fabric interface. Switches contain more than one port by definition. |

This section describes how to configure a host or embedded FM.

The configuration file, `opafm.xml`, defines the following:

- FM management entities to be run on this FM host or chassis for each subnet to which the host is attached (by definition a chassis is only connected to one subnet).
- Operational parameters for the fabric, such as routing and time-outs.
- vFabrics to be configured.
- Configuration for each FM component, such as sweep rates, logging options, and retries.

The configuration file formats for the host and embedded FMs are the same. There are some additional parameters that are available on the host FM. If those parameters are present in the configuration file, they are ignored on the embedded FM, therefore you can use the same configuration file for both host and embedded fabric managers.

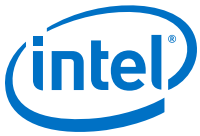
To make a change to a FM's configuration, edit the `opafm.xml` configuration file and restart the FM.

On a host this is done by editing `/etc/opa-fm/opafm.xml` then restarting the FM service, `systemctl opafm restart`.

On a chassis this is done by downloading the `opafm.xml` file to a host, editing it, uploading the new file to the chassis, then restarting the FM using the `smControl` command. The FastFabric Tools also provide CLI and interactive tools to assist in managing the configuration and operation of embedded FMs. Refer to the *Intel® Omni-Path Fabric Suite FastFabric User Guide* for information on using FastFabric to transfer the xml file to the switch.

The use of XML for the FM configuration file offers a number of significant and powerful advantages:

- XML is a standard; there are many third-party tools that can edit and read it



- Third-party editors understand XML and can do text highlighting and syntax checking
- Existing FastFabric tools such as `opaxmlextract`, `opaxmlindent`, `opaxmlfilter`, and `opaxmlgenerate` can edit the file
- Existing Linux tools such as `xml_grep` can search in the `.xml` file

## 5.1 Manager Instances

A given FM host can run multiple FM instances. Each instance is assigned to a single local HFI port. An Embedded FM in a chassis supports only one instance.

*Note:* A host can be connected to as many subnets as the number of ports its HFIs contain. If a host contains one HFI with two ports, and both ports are connected to different subnets, the host can run a maximum of two PM, FE, and SM instances, if adequate memory is available.

### 5.1.1 Configuration File Syntax

The `opafm.xml` file uses standard XML syntax to define configuration settings.

While XML is quite powerful, only a handful of basic syntaxes are needed to create FM configuration files.

Comments are ignored during parsing and may be used to annotate the file as needed. The following is a comment:

```
<!-- this is a comment -->
```

In the sample configuration file, comments are kept to a single line for readability, however a comment can span multiple lines if needed.

A parameter is specified using an XML tag such as:

```
<parameter>value</parameter>
```

All XML tag names are case sensitive. Each parameter starts with the start tag `<parameter>` and ends with a corresponding end tag: `</parameter>`. The names of the parameters are described in the remainder of this section and are shown in the sample `opafm.xml` configuration file. The parameter value is specified between the start and end tags.

The majority of parameters have numeric values. Numbers can be specified in decimal or hex. Hex numbers should start with 0x. For example: 0x1c is the hex representation of 28 decimal.

A section in the file can contain multiple related parameters. Sections also begin with a start tag and end with an end tag. For example:

```
<section>
<parameter1>value</parameter1>
<parameter2>value</parameter2>
</section>
```

A section may contain zero or more parameters. Some sections contain other sections. The indentation in the sample `opafm.xml` configuration file is used for readability but indentation is not required. If a section contains no parameters, it can be omitted.

Throughout this guide, the abbreviation `section.parameter` or `section.subsection` is used to indicate a section containing a given parameter or subsection.

The xml parser quietly ignores any unrecognized parameter or section. This capability allows for forward and backward compatibility of the configuration files. It will be possible to use newer configuration files (which may have additional parameters) on an older version of the FM. This has the negative risk that a mistyped parameter may be silently ignored and defaulted. To help avoid such mistakes it is recommended to start with the sample configuration file and edit or cut and paste parameters as needed. Also the `config_check` tool can be run in strict mode (`-s` option) to report any unrecognized tags.

The default FM configuration file provided shows all of the tags and provides detailed comments as to their meaning.

The first line in the file must be the following:

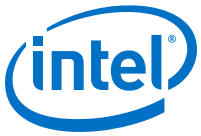
```
<?xml version="1.0" encoding="utf-8"?>
```

The `opafm.xml` configuration file is organized into a few top level sections as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<Config>
<!-- Common FM configuration, applies to all FM instances/subnets  -->
<Common>
</Common>
<!-- A single FM Instance/subnet -->
<Fm>
</Fm>
<!-- A single FM Instance/subnet -->
<Fm>
</Fm>
<!-- A single FM Instance/subnet -->
<Fm>
</Fm>
<!-- A single FM Instance/subnet -->
<Fm>
</Fm>
</Config>
```

The `Common` section defines parameters that apply to all FM instances. In a typical configuration file, most parameters are defined in the `Common` section and the `Fm` sections only define a few instance specific parameters. If needed, any parameter set in the `Common` section may be overridden for a given FM Instance by also defining it in the `Fm` section.

The embedded FM only supports a single FM instance. Only the first `Fm` section in the `opafm.xml` file is used by the embedded FM, any additional `Fm` sections are ignored.



The `Common` and each `Fm` sections have a similar layout. They each contain the following subsections:

```
<Applications>
</Applications>
<DeviceGroups>
</DeviceGroups>
<VirtualFabrics>
</VirtualFabrics>
<Shared>
</Shared>
<Sm>
</Sm>
<Fe>
</Fe>
<Pm>
</Pm>
```

The `Applications`, `DeviceGroups` and `VirtualFabrics` sections are used to configure vFabrics. This will be covered in greater detail in Virtual Fabrics on page 124.

The `Shared` section defines parameters that apply to all managers (`Sm`, `Fe`, and `Pm`). This section typically defines logging and redundancy options. If needed, any parameter set in the `Shared` section may be overridden for a given manager by also defining it in the `Sm`, `Fe`, and/or `Pm` sections.

## 5.2 Fabric Manager Shared Parameters

The following table describes parameters that may be used in the `Shared` subsection of either the `Common` or `Fm` sections.

**Table 5. Shared Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| SubnetSize | 9216 | Maximum number of endports (connected HFI and TFI ports) that the fabric is expected to support. Because several internal parameters are derived from this, it's important that this number not be set to less than the true number of connected end ports in the fabric.<br>For the embedded SM, if this value exceeds the capabilities of the chassis, the value given will be appropriately reduced. Refer to the *Intel® Omni-Path Fabric Switches Command Line Interface Reference Guide* for information on ESM subnet sizes supported. |
| CoreDumpLimit | unlimited | Max core file size in bytes. Can be unlimited or a numeric value. Suffix of K, M or G can be used to aid specification of larger values, such as 400M for 400 Megabytes.<br>A value of 0 disables generation of core dumps.<br>For the Embedded FM, this parameter is ignored.<br>This capability uses the core dump ulimit (setrlimit) capabilities of Linux and sets the soft limit. If the hard limit is set to 0 or the hard limit is smaller than CoreDumpLimit the FM will not be able to create dumps and a message will be logged on FM startup. |

***continued...***

| Parameter | Default Value | Description |
|---|---|---|
| CoreDumpDir | `/var/crash/opafm` | Directory to dump to. A unique file will be created for each core file per the standard Linux kernel configuration This directory will be created, but its parent directory must pre-exist.<br>For the Embedded FM, this parameter is ignored.<br>Default parent directory of `/var/crash/` may not exist on some systems. Make sure to create this directory or use an appropriate existing directory name before enabling FM core dumps. |
| Priority<br>ElevatedPriority | 0<br>0 | The startup priority of each manager in this `Fm Instance`. 0-15 is allowed. Priority and Elevated Priority control failover for SM and PM. Priority is used during initial negotiation, high Priority wins. ElevatedPriority is assumed by the winning master. This can provide sticky failover and prevent fallback when previous master comes back online. |
| LogLevel | 2 | Sets log level option for SM, PM, and FE:<br>0 = disable vast majority of logging output<br>1 = fatal, error, warn (syslog CRIT, ERR, WARN)<br>2 = +notice, (syslog NOTICE, INFO)<br>3 = +INFO (syslog DEBUG)<br>4 = +VERBOSE and some packet data (syslog DEBUG)<br>5 = +debug trace info (syslog DEBUG)<br>This parameter is ignored for the Embedded FM. Refer to the *Intel® Omni-Path Fabric Switches GUI User Guide* for information on configuring chassis logging options. |
| LogFile | | Sets log output location for SM, PM, and FE. By default (or if this parameter is empty) log output is accomplished using syslog. However, if a LogFile is specified, logging will be done to the given file. LogMode further controls logging.<br>This parameter is ignored for the Embedded FM. Refer to the *Intel® Omni-Path Fabric Switches GUI User Guide* for information on configuring chassis logging options. |
| LogMode | 0 | Controls mode for logging by SM, PM, and FE.<br>low two bits control logging as follows:<br>Low Bit (0/1):<br>0 - use normal logging levels<br>1 - logging is quieted by downgrading the majority of fatal, error, warn and infiniinfo log messages to level 3 (INFO) and only outputting user actionable events when LogLevel is 1 or 2.<br>Next Bit (0/2) (only affects logging when LogFile specified):<br>0 - user actionable events go to syslog and LogFile<br>2 - when LogFile specified, nothing goes to syslog<br>This parameter is ignored for the Embedded FM. Refer to the *Intel® Omni-Path Fabric Switches GUI User Guide* for information on configuring chassis logging options. |
| SyslogFacility | local6 | For the Host FM, controls what syslog facility code is used for log messages.<br>Allowed values are: auth, authpriv, cron, daemon, ftp, kern, local0-local7, lpr, mail, news, syslog, user, or uucp.<br>For the Embedded FM, this parameter is ignored. |

***continued...***

| Parameter | Default Value | Description |
|---|---|---|
| ConfigConsistencyCheckLevel | 2 | Controls the Configuration Consistency Check for SM, and PM. Checking can be completely disabled, or can be set to take action by deactivating Standby SM or PM if configuration does not - pass the consistency check criteria.<br>0 = disable Configuration Consistency Checking<br>1 = enable Configuration Consistency Checking without taking action (only log a message)<br>2= enable Configuration Consistency Checking and take action (log message and move standby to inactive state) |
| Debug<br>RmppDebug | 0 | Additional parameters for debug/development use. These enable debugging modes for SM, PM, and FE. RmppDebug can be overridden by other RmppDebug in individual components. |
| CS_LogMask<br>MAI_LogMask<br>CAL_LogMask<br>DVR_LogMask<br>IFS_LogMask<br>SM_LogMask<br>SA_LogMask<br>PM_LogMask<br>PA_LogMask<br>FE_LogMask<br>APP_LogMask | 0x1fff | For advanced users, these parameters can provide more precise control over per subsystem logging. For typical configurations these should be omitted and the LogLevel parameter should be used instead.<br>For each subsystem there can be a LogMask. The mask selects severities of log messages to enable and is a sum of the following values:<br>0x1=fatal<br>0x2=actionable error<br>0x4=actionable warning<br>0x8=actionable notice<br>0x10=actionable info<br>0x20=error<br>0x40=warn<br>0x80=notice<br>0x100=progress<br>0x200=info<br>0x400=verbose<br>0x800=data<br>0x1000=debug1<br>0x2000=debug2<br>0x4000=debug3<br>0x8000=debug4<br>0x10000=func call<br>0x20000=func args<br>0x40000=func exit<br>For Embedded SM corresponding Chassis Logging must also be enabled and SM configuration applies to all managers.<br>For Host SM, the Linux syslog service will need to have an appropriate level of logging enabled. |

For example:

```
<Shared>
<SubnetSize>2560</SubnetSize>
<CoreDumpLimit>0</CoreDumpLimit>
<CoreDumpDir>/var/crash/opafm</CoreDumpDir>
<Priority>0</Priority>
<ElevatedPriority>0</ElevatedPriority>
<LogLevel>2</LogLevel>
<LogMode>0</LogMode>
<SyslogFacility>local6</SyslogFacility>
```

```
<ConfigConsistencyCheckLevel>2</ConfigConsistencyCheckLevel>
<RmppDebug>0</RmppDebug>
</Shared>
```

## 5.3 Controlling FM Startup

The embedded FM's startup is controlled using the Chassis CLI commands `smConfig`, `smControl` and `smPmStart`. For the embedded FM the various `<Start>` parameters are ignored.

*Note:* The remainder of this section applies only to the host FM.

Unlike other `Shared` section parameters, the `Start` parameter is specially handled. The `Start` parameter may be used in the `Fm.Shared` section or any `Sm`, `Fe`, or `Pm` section (in `Common` or `Fm`). However, it cannot be used in the `Common.Shared` section and will be ignored if it is used there. In order for a given FM Instance to be started, the `Fm.Shared` section must have a Start section parameter with a value of 1. Within a given FM Instance, only the managers whose `Start` section parameters are also 1 (they default to 1) will be started.

The following example starts the SM and FE for the first FM Instance, but it will not start the PM. The second FM Instance will not be started at all.

```
<Common>
<Sm>
<Start>1</Start>
</Sm>
<Fe>
<Start>1</Start>
</Fe>
<Pm>
<Start>01</Start>
</Pm>

</Common>
<Fm>
<Shared>
<Start>1</Start>
</Shared>
</Fm>
<Fm>
<Shared>
<Start>0</Start>
</Shared>
</Fm>
```

The following example starts the SM and PM for the first FM Instance, but it does not start the FE. The second FM Instance will be started with an SM, FE, and PM.

```
<Common>
<Sm>
<Start>1</Start>
</Sm>
<Fe>
<Start>10</Start>
</Fe>
<Pm>
<Start>1</Start>
</Pm>
</Common>
<Fm>
```

```
<Shared>
<Start>1</Start>
</Shared>
<FePm>
<Start>1</Start> <!-- override Common.FePm -->
</FePm>
</Fm>
<Fm>
<Shared>
<Start>1</Start>
</Shared>
</Fm>
```

Typically, individual managers are controlled in the `Common` section and FM Instance startup is controlled in the `Fm.Shared` section. In most configurations, only a single FM Instance runs with all managers started, in which case the default `opafm.xml` file can be used as provided.

All instances are defined in a single `opafm.xml` configuration file. The sample configuration xml below defines the four FM instances in Figure 15 on page 78.

```
<Fm>
  <Shared>
    <Name>fm0</Name>
    <Hfi>1</Hfi>
    <Port>1</Port>
    <SubnetPrefix>0xfe80000000001001</SubnetPrefix>
  </Shared>
  <Sm>
    <Start>1</Start>
  </Sm>
  <Pm>
    <Start>1</Start>
  </Pm>
  <Fe>
    <TcpPort>3245</TcpPort>
    <Start>1</Start>
  </Fe>
</Fm>

<Fm>
  <Shared>
    <Name>fm1</Name>
    <Hfi>1</Hfi>
    <Port>2</Port>
    <SubnetPrefix>0xfe80000000001002</SubnetPrefix>
  </Shared>
  <Sm>
    <Start>1</Start>
  </Sm>
  <Pm>
    <Start>1</Start>
  </Pm>
  <Fe>
    <TcpPort>3246</TcpPort>
    <Start>1</Start>
  </Fe>
</Fm>

<Fm>
  <Shared>
    <Name>fm2</Name>
    <Hfi>2</Hfi>
    <Port>1</Port>
    <SubnetPrefix>0xfe80000000002001</SubnetPrefix>
  </Shared>
  <Sm>
```

```
    <Start>1</Start>
  </Sm>
  <Pm>
    <Start>1</Start>
  </Pm>
  <Fe>
    <TcpPort>3247</TcpPort>
    <Start>1</Start>
  </Fe>
</Fm>

<Fm>
  <Shared>
    <Name>fm3</Name>
    <Hfi>2</Hfi>
    <Port>2</Port>
    <SubnetPrefix>0xfe80000000002002</SubnetPrefix>
  </Shared>
  <Sm>
    <Start>1</Start>
  </Sm>
  <Pm>
    <Start>1</Start>
  </Pm>
  <Fe>
    <TcpPort>3248</TcpPort>
    <Start>1</Start>
  </Fe>
</Fm>
```

*Note:* The SM and the PM are unified. As such in order to enable the PM to Start, its corresponding SM must be enabled. However it is valid to enable the SM to start without enabling the PM.

## 5.4 SM Parameters

The following tables describe parameters that can be used in the `Sm` subsection of either the `Common` or `Fm` sections.

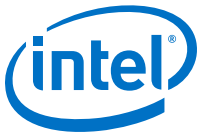
Any parameter that can be used in the `Common.Shared` section may also be used in the `Common.Sm` or `Fm.Sm` sections.

### 5.4.1 SM Redundancy

The parameters in the following table control SM Failover. They complement the Priority and ElevatedPriority parameters that are used in the `Shared` section and can also be used in the `Sm` section if needed.

**Table 6. SM Redundancy Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| MasterPingInterval | 5 | MasterPingInterval is the interval in seconds at which the secondary SM pings the master (the ping occurs using an inband SM packet). |
| MasterPingMaxFail | 3 | Number of times a secondary's ping of the master must fail before renegotiation of a new master occurs. |
| DbSyncInterval | 15 | Secondary SMs synchronize their fabric database with the master at DbSyncInterval minutes. If set to 0, Db synchronization is disabled. |



## 5.4.2 Fabric Routing

The following Routing Algorithms are supported:

- shortestpath - pick shortest path and balance lids on ISLs
- fattree — shortest path with better balancing for fat tree topology. The fattree algorithm uses up/down routing to avoid credit loops caused by routing cycles. Given equal cost up/down paths, only the up routes will be used.
- dgshortestpath - a variation of shortest path. When the routing algorithm is set to `dgshortestpath` the following section is used to configure the algorithm. Overall, `dgshortestpath` routing is a form of `Min Hop` or `Shortestpath` routing, except you can control the order in which routes to end nodes are assigned. This can be used to ensure diversity of routing within groups of devices as well as the entire fabric overall. End nodes that are not members of any listed groups will be routed last.

The parameters in the following table control Fabric Unicast Routing.

**Table 7. SM Unicast Routing Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| RoutingAlgorithm | shortestpath | Selects the routing algorithm. This can be `shortestpath`, `fattree`, or `dgshortestpath`. See Fabric Unicast Routing on page 45 for more information. |
| SpineFirstRouting | 1 | A routing option applicable to all routing algorithms to avoid credit loops in complex fabrics with Intel switches. Given equal length routes, routes through chassis spine first. Hence avoids loops caused by routing via edge/leaf switches instead of spines. See Fabric Unicast Routing on page 45 for more information. |
| Lmc | 0 | Lmc for LID assignment to HFIs and TFIs. $2^{Lmc}$ LIDs are assigned per port. See Fabric Unicast Routing on page 45 for more information. |
| LmcE0 | 0 | Lmc for LID assignment to Switches with an Enhanced Port 0 capability. $2^{Lmc}$ LIDs are assigned per Switch Port 0 with Enhanced Port 0 capability. See Fabric Unicast Routing for more information. |
| ForceRebalance | 0 | When enabled, routing is rebalanced with every change to the fabric (i.e., HFIs added or removed). Normally to reduce fabric disruption and improve FM sweep times, when the fabric changes the FM attempts to make minimal changes to the routing tables. This may lead to some imperfect balancing of routes. However, in most fabrics, changes to design are not normally happening, but reboots of servers and perhaps switches are. Disabling `ForceRebalance` optimizes reaction to such reboots with the least impact to the rest of the fabric and when the given nodes come back online, the fabric should once again be balanced. |

## 5.4.3 PreDefinedTopology

The `PreDefinedTopology` section is used for verifying the layout of the fabric against a topology input file of the expected layout.

There are three modes of handling mismatches: Disabled, Warn, and Enabled. Disabled ignores any mismatches on that field, Warn prints a warning to the log file, and Enabled prints a warning to the log file and quarantines the node from the fabric.

**Table 8. PreDefinedTopology Field Definitions**

| Field | Default Value | Description |
|---|---|---|
| Enabled | 0 | Whether or not this feature is enabled. |
| TopologyFilename | None | Fully qualified filename of pre-defined input topology. |
| LogMessageThreshold | 0 | Number of warnings to output to log. Number of warnings to output to log file before suppressing further warnings and only printing a summary at the end of a sweep. Entering 0 disables this threshold. |
| FieldEnforcement<br>• UndefinedLink<br>• NodeGUID<br>• NodeDesc<br>• PortGUID | Disabled | Per-field enforcement levels for mismatch handling. Parent Element for NodeGUID, NodeDesc, and PortGUID field enforcement elements, which are described below.<br>Enforcement levels are the same for all enforcement elements:<br>• Disabled<br>• Warning<br>• Enabled<br><PreDefinedTopology><br><Enabled>0</Enabled><br><TopologyFilename></TopologyFilename><br><LogMessageThreshold>100</<br>LogMessageThreshold><br><FieldEnforcement><br><NodeDesc>Warn</NodeDesc><br><NodeGUID>Warn</NodeGUID><br><PortGUID>Warn</PortGUID><br><UndefinedLink>Warn</UndefinedLink><br></FieldEnforcement><br></PreDefinedTopology> |

See also Pre-Defined Topology Verification on page 41 for related information.

### 5.4.4 Fat Tree Topology

When the `RoutingAlgorithm` is set to fattree, the `FatTreeTopology` section is used to configure fat tree parameters. To properly handle routing decisions, the SM needs to understand some fabric topology information. Specifically, how many tiers are in the fat tree and which tier a switch belongs to. If all HFIs are on the same tier, the SM can easily determine the fat tree topology and tier a given switch resides in. If they are not, you can create a "CoreSwitches" device group indicating the core switches in the fat tree.

```
<!-- *************** Fat Tree Topology *************************** -->
    <!-- When the RoutingAlgorithm is set to fattree, the following -->
    <!-- section is used to configure fat tree parameters.  -->
    <!-- FatTreeTopology:  -->
    <!--    To properly handle routing decisions, the SM need to understand -->
    <!--    some fabric topology information.  Specifically, how many -->
    <!--    tiers are in the fat tree and which tier a switch belongs to. -->
    <!--    If all FIs are on the same tier, the SM can easily determine -->
    <!--    the fat tree topology and tier a given switch resides in. -->
    <!--    If they are not, a list of CoreSwitches will be used -->
    <!--    to identify the topology. -->
    <FatTreeTopology>
```

```
    <!-- The number of tiers in the fat tree -->
    <TierCount>3</TierCount>
    <!-- Indicates whether or not the HFIs are on the same tier in the -->
    <!-- fat tree. -->
    <FIsOnSameTier>1</FIsOnSameTier>
    <!-- The following must be setup if the HFIs are not on the same tier. -->
    <!-- It identifies the root/core switches in the fat tree. -->
    <!-- <CoreSwitches>core_device_group</CoreSwitches> -->
    <!-- Nodes may be specified for exclusion from initial round-robin -->
    <!-- to give better route balancing of remaining nodes. -->
    <!-- This may be useful in assymetric fat trees or to -->
    <!-- initially balance across compute nodes in the tree. -->
    <!-- <RouteLast>hfi_device_group</RouteLast>  -->
  </FatTreeTopology>
```

The parameters in the following table control Fat Tree Topology.

**Table 9. SM Fat Tree Topology Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| TierCount | 3 | The number of tiers in the fat tree. |
| FIsOnSameTier | 1 | Discovery algorithm a bit more streamlined if FIs on same tier, but this is not a requirement. |
| RouteLast | | When the fattree routing algorithm is in use, nodes may be specified for exclusion from initial round-robin to give better route balancing of remaining nodes. This may be useful in asymmetric fat trees or to initially balance across compute nodes in the tree. This can be done by specifying a group of HFIs with the `RouteLast` device group. |
| CoreSwitches | | A core switch device group can be configured to specify the switches at the core of the fabric. This must be used if the HFIs are not at the same tier in the fat tree. |

### 5.4.5 Intel® Omni-Path Technology-compliant Multicast

Intel® Omni-Path Architecture has a limitation in that the SM must make the realizable decision for a Multicast group at Multicast Join/Create time. However later fabric changes (removal of links, loss of switches) could make the multicast group unrealizable, but there is no notice in Intel® OP that the SM could send to the end node.

To address this situation, the SM performs stricter Multicast checking at Join/Create time. This means a Multicast join/create is rejected if there are any switch-to-switch links that do not have at least the MTU or Rate requested for the Multicast group. The rejection reduces the chance that a simple fabric failure could make the group unrealizable.

The parameters in the following table control Fabric Multicast Routing. These are all part of the `Multicast` subsection within the `Sm` section.

## Table 10. SM Multicast Routing Parameters

| Parameter | Default Value | Description |
|---|---|---|
| DisableStrictCheck | 0 | When 1, disables the strict checking and accepts Join/Create for which at least 1 viable fabric path exists. |
| MLIDTableCap | 1024 | Number of Multicast LIDs available in fabric. Must be set to less than or equal to the smallest Multicast forwarding table size of all switches in fabric. |
| MulticastGroup | | Describes a set of multicast groups that should be pre-created by the SM. This subsection can be used multiple times with each occurrence specifying a different set of multicast groups to be created. |
| EnablePruning | 0 | Should multicast spanning tree be pruned to its minimal size.<br>When disabled (0), multicast join and leave is optimized by building the spanning tree to include all switches, such that a HFI join/leave only requires a single switch update<br>When enabled (1), multicast traffic will only propagate through the minimal set of switches, reducing overhead in the fabric<br>Recommend to be disabled (0) for typical fabrics where multicast is mainly used to support IPoIB ARP. |
| RootSelectionAlgorithm | LeastTotalCost | Controls how the root switch for the multicast spanning tree is selected. One of the following algorithm names can be specified to select the root switch for the spanning tree (Names of algorithm are not case sensitive).<br>`LeastTotalCost` – A switch with the smallest sum of costs to other switches will be selected.<br>`LeastWorstCaseCost` – A switch that has the least worst case cost to other switches will be selected.<br>`SMNeighbor` – The Switch next to the SM will be selected.<br>`LeastTotalCost` and `LeastWorstCaseCost` tend to select a switch that is more in the center of the fabric as the root of the multicast spanning tree. With these options, the information about the selected switch is also communicated to the standby SMs so that the multicast spanning tree computed by the standby SMs will match that of the master SM. That way if the master SM fails over to a standby SM, the multicast spanning tree for the fabric will still remain the same and the multicast forwarding tables in the switches do not have to be reprogrammed. The multicast traffic will not be disrupted.<br>Because the `SMNeighbor` algorithm will select the switch next to the SM, if a master SM fails over to the standby, the spanning tree computed by the standby SM will differ from that of the master and the multicast forwarding tables in the switch will have to be reprogrammed which can result in some disruptions to the multicast traffic. |
| MinCostImprovement | 50% | When using `LeastTotalCost` or `LeastWorstCaseCost`, the `MinCostImprovement` parameter controls when the multicast spanning tree root should be changed |

***continued...***

| Parameter | Default Value | Description |
|---|---|---|
| | | if there are changes in the number of switches in the fabric. The root will be changed only if in the new topology a switch's cost is `MinCostImprovement`(expressed as a percentage) better than that of the current root switch or when the current root switch goes offline.<br>A higher value of `MinCostImprovement` would result in the multicast spanning tree root not being changed too frequently and therefore would cause less disruptions in multicast traffic. A smaller value would select the best switch and provide better multicast latency but can cause disruptions to multicast traffic upon fabric changes. |

### 5.4.6 Pre-Created Multicast Groups

Multicast Groups which are pre-created by the SM are all part of the `MulticastGroup` subsection of the `Multicast` subsection within the `Sm` section. Each `MulticastGroup` section defines one or more Multicast Groups which will be pre-created by the SM.

OFA Delta requires a pre-created `MulticastGroup` for IPoIB. The configuration can specify other groups that are also needed. Every pre-created `MulticastGroup` can have one or more MGIDs. The MGID must be unique among all `MulticastGroups` within an FM instance and must be able to match a single VF.

When defined at `Common` level, the MGID must be unique within all instances. MGIDs are specified as two 64-bit values separated by a colon (:). A single MGID can be specified as `<MGID>0xabc:0x123567</MGID>` in the `MulticastGroup` section. If no MGIDs are specified, the `IPv4 broadcast`, `IPv4 all nodes` and the `IPv6 all nodes` MGIDs will be created.

The following is a sample of IPoIB IPv4 and IPv6 multicast for all VFs which have IPoIB as an application.

```
<MulticastGroup>
        <Create>1</Create>
        <MTU>2048</MTU>
        <Rate>25g</Rate>
        <!-- <SL>0</SL> -->
        <QKey>0x0</QKey>
        <FlowLabel>0x0</FlowLabel>
        <TClass>0x0</TClass>
</MulticastGroup>
```

The following is a sample of IPoIB, IPv4, and IPv6 multicast for `0x9001/0x1001` PKey. This can be useful if there are multiple IPoIB vFabrics and different multicast parameters (Rate, MTU, etc.) are desired for each IPoIB vFabric. Because IPoIB MGID includes PKey, we specify PKey not VirtualFabric. MGIDs specified must use the Full PKey (0x8000-bit set).

```
<MulticastGroup>
        <Create>0</Create>
        <PKey>0x1001</PKey>
        <!-- PKey 0x9001/0x1001 is part of IPv4 MGID below -->
        <!-- MGID = 0xffFS401bPPPP0000:00000000GGGGGGGG -->
        <!-- where F=flags, S=scope, P=PKey and G=IP Multicast Group -->
```

```
        <MGID>0xff12401b90010000:0x00000000ffffffff</MGID> <!-- bcast -->
        <MGID>0xff12401b90010000:0x0000000000000001</MGID> <!-- all nodes -->
        <MGID>0xff12401b90010000:0x0000000000000002</MGID> <!-- all routers -->
        <MGID>0xff12401b90010000:0x0000000000000016</MGID>
        <!-- PKey 0x9001/0x1001 is part of IPv6 MGIDs below -->
        <!-- MGID = 0xffFS601bPPPPGGGG:GGGGGGGGGGGGGGGG -->
        <!-- where F=flags, S=scope, P=PKey and G=IP Multicast Group -->
        <MGID>0xff12601b90010000:0x0000000000000001</MGID> <!-- all nodes -->
        <MGID>0xff12601b90010000:0x0000000000000002</MGID> <!-- all routers -->
        <MGID>0xff12601b90010000:0x0000000000000016</MGID>
        <MTU>2048</MTU>
        <Rate>10g</Rate>
        <QKey>0x0</QKey>
        <FlowLabel>0x0</FlowLabel>
        <TClass>0x0</TClass>
</MulticastGroup>
```

### Understanding Multicast matching VirtualFabrics

Rule: All MGIDs of all pre-created MulticastGroups must be able to match to a VirtualFabric (VF).

In order for a MC group to find a matching VF, all these rules should apply:

- Matching VF name. If no name, then do not use for matching.
- Matching PKeys. If no PKey, then do not use for matching.
- MC group Rate should be smaller than or equal to the VF Rate. If MC Rate is missing Fabric Manager uses the default value of 25g.
- MC group MTU should be smaller than or equal to VF MTU. If MC MTU is missing Fabric Manager uses the default value of 2048.
- The application property of the candidate VF should have the same MGID associated to the MC group MGID.
- If for any reason, a single MGID matches more than one VF or no VF at all, that will be considered a configuration error and the FM will not start.

Example

Using the provided `/etc/opa-fm/opafm.xml` we will describe two situations:

- The MulticastGroup section specifies a PKey of 0x0002. Therefore, it can only match a VirtualFabric with PKey 0x0002. By default, no VirtualFabric has such a PKey. So the MGIDs of this group can never be assigned to a VF. The system will fail to start because a config-defined MC group could not find a matching VF.
- MulticastGroups section without an explicit <MGID> tag will attempt to create the IPoIB MGIDs (such as 0xff12401bPPPP0000:0x0000000000000001) for ALL VirtualFabrics with the Networking application (or AllOthers if none contain Networking). Where 'PPPP' in the MGID is replaced by said VF's PKey. So the first MulticastGroup will create an McMemberRecord (as seen in `opasaquery -o mcmember`) for the Networking VF and use PKey 0x0002. Then the second MulticastGroup will try to create an McMemberRecord for the same MGID. Because an MGID must be unique, the system will signal that it detected Duplicates MGIDs and the FM will fail to start.

The parameters in the following table define the policies and controls for the Multicast Groups.

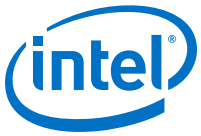


**Table 11. SM Multicast Group Pre-Creation Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Create | 1 | Enables or disables the creation of the given set of Multicast Groups. This provides a convenient way to disable a `MulticastGroup` section without needing to delete it from the configuration file. |
| VirtualFabric<br>or<br>PKey | | Controls the virtual fabric for which the `MulticastGroup` is created. Alternatively a `PKey` may be specified. If neither is specified, IPv4/6-type MGIDs will be created for all Virtual Fabrics which contain those MGIDs in the <Application> section. |
| Rate | 25g | The Static Rate for the multicast group. Only nodes and paths which have a rate greater than or equal to this value will be able to join the group. This also sets the upper bound for the performance of the multicast group. Rate is specified in the natural format and may be one of 25g, 50g, 75g or 100g. The `Rate_Int` tag and IBTA Int values are no longer supported.<br>Rate must be specified in natural format using `Rate`. |
| MTU | 2048 | The MTU for the multicast group. Only nodes and paths that have an MTU greater than or equal to this value will be able to join the group. This also sets the upper bound for the message sizes that may be sent to the multicast group.<br>MTU must be specified in the natural format, one of 2048, 4096, 8192, or 10240. The `MTU_Int` tag and IBTA Int values are no longer supported. |
| SL | | The Service Level for the Multicast Group. If specified, it must match the MulticastSL of the corresponding Virtual Fabric. If unspecified, this will default to the MulticastSL of said Virtual Fabric. |
| QKey | 0 | The QKey to be used for the group. |
| FlowLabel | 0 | The FlowLabel to be used for the group.<br>This value is not presently used. |
| TClass | 0 | The Traffic Class to be used for the group.<br>This value is not presently used. |

## 5.4.7 Fabric Programming

SwitchLifetime, HoqLife, and VLStallCount can be used to relieve fabric congestion and avoid fabric deadlocks by discarding packets. Discards help prevent back pressure from propagating deep into the core of the fabric, however such discards cause end nodes to need to timeout and retransmit.

If a packet stays at the Head of a Switch Egress Port for more than HoqLife, it is discarded. Similarly a packet queued in a switch for more than SwitchLifetime is discarded.

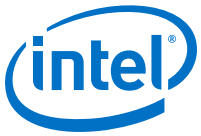
Specified as integer time using ns, us, ms, s, m, or h as units. SwitchLifetime and HoqLife can also be set to infinite. VLStallCount controls a second tier more aggressive discard. If VLStallCount packets in a row are discarded due to HoqLife by a given VL on an egress port. That egress port's VL enters the VL Stalled State and discards all that VL's egress packets for 8*HoqLife. Packets discarded for any of these reasons will be included in the TxDiscards counter which is queryable via FastFabric.

The parameters in the following table control Fabric Configuration and Programming.

## Table 12. SM Fabric Configuration Parameters

| Parameter | Default Value | Description |
|---|---|---|
| TimerScalingEnable | 0 | HoqLife can be automatically adjusted based on Virtual Fabric configurations and preemption/traffic flow optimization configurations. This feature is Timeout Scaling, and can be enabled below. Furthermore, HoqLife can be specified per Virtual Fabric, overriding the global definition below. When HoqLife is specified at the Virtual Fabric level, the timeout scaling is automatically disabled for that VF. |
| SwitchLifetime | 33ms | A packet queued in a switch for more than `SwitchLifetime` is discarded. Specified as integer time using ns, us, ms, s, m, or h as units. Can also be set to infinite. |
| HoqLife | 8ms | If a packet stays at the Head of a Switch Egress Port for more than HoqLife, it is discarded. Specified as integer time using ns, us, ms, s, m, or h as units. Can also be set to infinite.<br>HoqLife can be automatically adjusted based on Virtual Fabric configurations and preemption/traffic flow optimization configurations. This feature is Timeout Scaling, and can be enabled below. Furthermore, HoqLife can be specified per Virtual Fabric, overriding the global definition below.<br>When HoqLife is specified at the Virtual Fabric level, the timeout scaling is automatically disabled for that VF. |
| VLStallCount | 7 | Controls a second tier more aggressive discard. If `VlStallCount` packets in a row are discarded due to HoqLife by a given VL on an egress port. That egress port's VL enters the VL Stalled State and discards all that VL's egress packets for 8*`HoqLife`. |
| CongestionControl | | Configures support for OPA Congestion Control. For Congestion Control configuration information refer to SM Congestion Control on page 102 |
| AdaptiveRouting | | Configures support for AdaptiveRouting in Intel® OPA Switches. For Adaptive Routing configuration information refer to SM Adaptive Routing on page 103. |
| SaRespTime | 1s | Maximum anticipated SA response time. |
| PacketLifetime | 1s | When `DynamicPacketLifetime` is disabled, the `PacketLifetime` is reported as the `PKtLifetime` for all paths. |
| DynamicPacketLifetime | | Configures support for `DynamicPacketLifetime`. For `Dynamic Packet Lifetime` configuration information refer to Table 18 on page 104. |
| SmaBatchSize | 4 | Max parallel requests to a given SMA. |
| PathSelection | Minimal | Controls PathRecord path selection and ordering. Most applications will use the first path or only the first few paths. When LMC=0 this setting makes no difference because there is only a src/dst address per pair of ports. However when LMC!=0, there can be N=(1<<LMC) addresses per port. This means there are $N^2$ possible combinations of SLID and DLID which the SA could return in the Path Records. However there are really only N combinations which represent distinct outbound and return paths. All other combinations are different mixtures of those N |

***continued...***

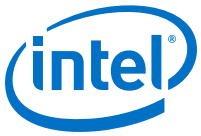


| Parameter | Default Value | Description |
|---|---|---|
| | | outbound and N return paths. Also important to note, is that LMC for all FIs are typically the same, while LMC for switches will be less. Generally redundant paths and/or having a variety of paths is not critical for paths to switches, but can be important for applications talking HFI to HFI.<br>Controls what combinations are returned and in what order. For the examples below, let's assume SGID LMC=1 (2 LIDs) and DGID LMC=2 (4 LIDs).<br>`Minimal` - return no more than 1 path per lid SLID1/DLID1, SLID2/DLID2 (since SGID has 2 lids stop).<br>`Pairwise` - cover every lid on both sides at least once SLID1/DLID1, SLID2/DLID2, SLID1/DLID3, SLID2/DLID4.<br>`OrderAll` - cover every combination, but start with pairwise set SLID1/DLID1, SLID2/DLID2, SLID1/DLID3, SLID2/DLID4 SLID1/DLID2, SLID1/DLID4, SLID2/DLID1, SLID2/DLID3.<br>`SrcDstAll` - cover every combination with simple all src, all dst SLID1/DLID1, SLID1/DLID2, SLID1/DLID3, SLID1/DLID4 SLID2/DLID1, SLID2/DLID2, SLID2/DLID3, SLID2/DLID4. |
| QueryValidation | 0 | When 1, enables IBTA compliant SA query operation for `GetTable` (PathRecord). As such a SGID and NumbPath is required.<br>When 0, allows interop with some non-compliant OFA Delta queries and permits `GetTable` (PathRecord) to specify either a SGID or SLID and defaults `numbPath` to 127 if not specified. |
| SmaBatchSize | 2 | This and the following two parameters control how many concurrent SMA requests the SM can have in flight while programming the SMAs in the fabric. |
| MaxParallelReqs | 4 | Max devices to req in parallel. |
| VL15FlowControlDisable | 1 | VL15FlowControlDisable can be used to turn off the link-level reliable transport flow control behavior on VL15.<br>When reliable transport flow control has been disabled for a VL, packets will be discarded by that VL when the packet length is larger than current available VL credit. |
| VL15CreditRate | 18 | Rate at which to return credits to an HFI that has been recognized as attempting to spoof the fabric. Prevents DoS attacks against the fabric from a spoofed host.<br>The field is defined as $1/(2^X)$ of the full return rate. For instance, if the field is set to 2, the credit return rate will be $1/(2^2) = 1/4$ (one fourth) the full credit rate when a denial of service attack is detected.<br>Valid values are between 0 and 20, with 0 disabling the credit rate limiting.<br>Valid values are between 0 and 20, with 0 disabling the credit rate limiting |
| MinSharedVLMemory | 0 | MinSharedVLMemory is an integer representing a percentage of overall Buffer memory that will be guaranteed to be shared among the ports (provided that the port supports shared memory). If not specified, defaults to 0.<br>DedicatedVLMemMulti is an integer multiplier that can increase the amount of dedicated memory per VL. It is specified in multiples of MTUs for that VL.<br>If not specified, defaults to 1. |

***continued...***

| Parameter | Default Value | Description |
|---|---|---|
| DedicatedVLMemMulti | 1 | |
| WireDepthOverride | -1 | WireDepthOverride is an integer value (in bytes) that overrides the wire depth returned in the port info. If not specified, defaults to -1.<br>A value of -1 indicates to use the value from port info.<br>A value of 0 means to ignore Wire Depth, and use Replay Depth.<br>Any other value indicates a byte-value to use instead of wire depth from port info. |
| ReplayDepthOverride. | -1 | ReplayDepthOverride is an integer value (in bytes) that overrides the buffer depth returned <!-- in the port info. If not specified, defaults to -1. Similar to WireDepthOveried as above.<br><!-- A value of -1 indicates to use the value from port info.<br>A value of 0 means to ignore Replay Depth, and use Wire Depth.<br>Any other value indicates a byte-value to use instead of replay depth from port info.<br>If both ReplayDepthOverride and WireDepthOverride are 0, then no bytes are considered for data outstanding on the wire. |
| LinkPolicy | | Link Downgrades:<br>When two ports are linked together, the SM selects the allowable settings based on the settings supported by the hardware of the port pair. The link can chose to downgrade width by dropping lanes that are not performing adequately. The lane will drop up to MaxDroppedLanes from the initial negotiated width.<br>LinkPolicy:<br>A policy can be enabled for Link Width or Link Speed. When enabled, the SM will enforce the specified policy on each link, and disable those in violation. The following values for policy are valid:<br>SpeedPolicy:<br>• "Supported" - The SM will disable links whose active speed is not highest speed supported in firmware<br>• "12.5G" - The SM will disable any links operating at lower than 12.5Gbps<br>• "25G" - The SM will disable any links operating at lower than 25 Gbps<br>WidthPolicy<br>• "Supported" - The SM will disable links whose active width is not widest width supported in firmware<br>• "2x" - The SM will disable any links operating with less than 2 lanes<br>• "3x" - The SM will disable any links operating with less than 3 lanes<br>• "4x" - The SM will disable any links operating with less than 4 lanes-<br>*Note:* The "Supported" quality policy is implicitly enabled when --><br><!-- using any of the available policies. |
| Preemption | | |
| SmallPacket | 256 | Maximum size for a small packet to be a preempting packet. |

***continued...***



| Parameter | Default Value | Description |
|---|---|---|
| | | SmallPktLimit minimum is 32 (but hardware may round this up) |
| LargePacket | 4096 | Minimum size for a large packet to be a preemptible packet.<br>LargePacket maximum is 8192.<br>LargePacket can be greater than MTU, a packet can be up to 128 bytes larger than MTU due to headers and trailers.<br>If LargePacket limit is much larger than MTU (for example, 8K on a link with 4K MTU), then the limit would simply never apply. LargePacketLimit having a maximum of 8192 also simply means a 10K packet is always eligible for preemption when preemption is enabled.<br>A sensible configuration of LargePacketLimit will set it to a value less than the largest anticipated packet on the preempted VLs, otherwise preemption would never occur. |
| PreemptLimit | 4096 | Limit on total number of bytes to be preempted. |
| NoReplyIfBusy | 0 | SA to return busy status (when set to 0) or else no response if busy (when set to 1) |
| SmaSpoofingCheck | 1 | Support port-level SMA security checking features |

#### 5.4.7.1 Switch and HFI Congestion Control

The parameters in the following table control Congestion Control configuration for the Switch and HFI parameters in the fabric. The following is a sample of the Congestion Control section of the FM configuration file.

```
<CongestionControl>
      <!-- 1 = Enable, 0 = Disable -->
      <Enable>0</Enable>
      <!-- Turn on additional debug logging for CCA -->
      <!-- 1 = Enable, 0 = Disable -->
      <Debug>0</Debug>
      <!-- CC settings applicable to all switches. -->
      <!-- Default settings for switch are selected to provide the best -->
      <!-- congestion control for various traffic mixes but may require -->
      <!-- tuning for best performance -->
      <Switch>
        <!-- OPA CC SwitchCongestionSetting:Threshold -->
        <!-- A value in the range 0 to 15.  Higher values indicate more -->
        <!-- aggressive congestion thresholds. -->
        <Threshold>1</Threshold>
        <!-- OPA CC SwitchCongestionSetting:Packet_Size -->
        <!-- Minimum packet size to mark in units of credits. -->
        <PacketSize>0</PacketSize>
        <!-- IBTA CC SwitchCongestionSetting:CS_ReturnDelay -->
        <CsReturnDelay>0</CsReturnDelay>
        <!-- OPA CC SwitchCongestionSetting:Marking_Rate -->
        <MarkingRate>0</MarkingRate>
        <!-- OPA CC SwitchCongestionSetting:Control_Map bit0 indicates that -->
        <!-- the victim mask is valid-->
        <!-- A value of 0 will cause marking of only sources of congestion -->
        <!-- A value of 1 will cause marking of both sources and victims of
congestion -->
        <VictimMarkingEnable>1</VictimMarkingEnable>
      </Switch>
      <!-- CC settings applicable to all HFIs. -->
      <!-- When using Intel HFIs, the settings for Congestion Control can -->
      <!-- be configured with the following environment variables such as -->
```

```
      <!-- the ones shown here and may require tuning for best performance -->
      <!-- PSM_DISABLE_CCA 0 -->
      <!-- PSM_CCTI_INCREMENT 1 -->
      <!-- PSM_CCTI_TIMER 1 -->
      <!-- PSM_CCTI_TABLE_SIZE 128 -->
      <Fi>
        <!-- OPA CC FICongestionEntry:CCTI_Increase -->
        <Increase>5</Increase>
        <!-- OPA CC FICongestionEntry:CCTI_Timer -->
        <Timer>10</Timer>
        <!-- OPA CC FICongestionEntry:Trigger_Threshold -->
        <Threshold>8</Threshold>
        <!-- OPA CC FICongestionEntry:CCTI_Min -->
        <Min>0</Min>
        <!-- OPA CC CongestionControlTable:CCTI_Limit -->
        <Limit>127</Limit>
        <!-- Maximum injection rate delay in us, used to determine how the -->
        <!-- congestion control table entries are generated. -->
        <DesiredMaxDelay>21</DesiredMaxDelay>
      </Fi>
    </CongestionControl>
```

**Table 13. Congestion Control Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Enable | 0 | Enables (1) or disables (0) Congestion Control. |
| Debug | 0 | Enables (1) or disables (0) additional debug logging for CCA |
| Switch | N/A | CC settings applicable to all switches. Default settings for switch are selected to provide the best congestion control for various traffic mixes but may require tuning for best performance. Refer to Congestion Control Switch Settings on page 100 for the switch parameters. |
| Fi | N/A | CC settings applicable to third party HFIs. When using Intel® HFIs, the settings for Congestion Control can be configured with the following environment variables such as the ones shown here and may require tuning for best performance<br>PSM_DISABLE_CCA 0<br>PSM_CCTI_INCREMENT 1<br>PSM_CCTI_TIMER 1<br>PSM_CCTI_TABLE_SIZE 128 |

### 5.4.7.2 Congestion Control Switch Settings

The parameters in the following table control Congestion Control settings applicable to all switches. Default settings for a switch are selected to provide the best congestion control for various traffic mixes but may require tuning for best performance.

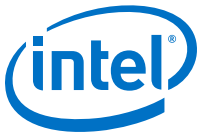


**Table 14. Congestion Control Switch Settings Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Threshold | 1 | A value in the range 0 to 15. Higher values indicate more aggressive congestion thresholds. |
| PacketSize | 0 | Minimum packet size to mark in units of credits. |
| MarkingRate | 0 | The marking rate for switch congestion |
| VictimMarkingEnable | 1 | Control_Map bit0 indicates that the victim mask is valid.<br>A value of 0 will cause marking of only sources of congestion<br>A value of 1 will cause marking of both sources and victims of congestion |

### 5.4.7.3 Congestion Control HFI Settings

When using PSM, the settings for Congestion Control can be configured with the following environment variables and may require tuning for best performance:

```
PSM_DISABLE_CCA 0
PSM_CCTI_INCREMENT 1
PSM_CCTI_TIMER 1
PSM_CCTI_TABLE_SIZE 128
```

The parameters in the following table control Congestion Control settings applicable to HFIs. Default settings for an HFI are selected to provide the best congestion control for various traffic mixes but may require tuning for best performance.

**Table 15. Congestion Control HFI Settings Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Increase | 1 | HFI Congestion Control Table Index (CCTI) Increase. The amount by which a CCTI will be increased, when a packet which was marked as having gone through a point of congestion, is received. |
| Timer | 1 | HFI CCTI Timer that is a cyclic timer set up by the congestion manager, associated with an SL, which on expiration is reset and decreases the CCTI . |
| Threshold | 8 | HFI Trigger Threshold When a CCTI is equal to the Trigger Threshold this event is triggered and information pertaining to the event is logged by the Channel Adapter. |
| Min | 0 | HFI CCTI minimum. This is the lowest value that CCTI can be reduced to; the default value is zero. |
| Limit | 127 | CCTI_Limit is the bounding value for a CCTI. CCTI cannot be greater than CCTI_Limit. |
| DesiredMaxDelay | 8300 | Maximum injection rate delay in ns, used to determine how the congestion control table entries are generated. |

### 5.4.7.4 SM Congestion Control

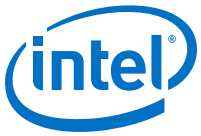
The parameters in the following table control OPA Congestion Control configuration for the switches and HFIs in the fabric. The following is a sample of the Congestion Control section of the FM configuration file.

```
<CongestionControl>
<Enable>0</Enable>
<Switch>
<Threshold>1</Threshold>
<PacketSize>0</PacketSize>
<MarkingRate>0</MarkingRate>
<VictimMarkingEnable>1</VictimMarkingEnable>
</Switch>
<Fi>
<Increase>1</Increase>
<Timer>1</Timer>
<Threshold>8</Threshold>
<Min>0</Min>
<Limit>127</Limit>
<DesiredMaxDelay>8300</DesiredMaxDelay>
</Fi>
</CongestionControl>
```

**Table 16. SM Congestion Control Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Enable | 0 | 1 = Enable `Congestion Control`<br>0 = Disable `Congestion Control` |
| Switch | | Section for `CongestionControl` settings applicable to all switches. The parameters for the switches are shown in the following six rows. |
| Switch.Threshold | 1 | A value in the range 0 to 15. Higher values indicate more aggressive congestion thresholds. |
| Switch.PacketSize | 0 | Minimum packet size to mark in units of credits. |
| Switch.MarkingRate | 0 | The rate for marking both sources and victims. |
| Switch.VictimMarkingEnable | 1 | `Control_Map` bit0 indicates that the victim mask is valid. A value of 0 causes marking of only sources of congestion. A value of 1 causes marking of both sources and victims of congestion. |
| FI | | Section for `CongestionControl` settings applicable to all HFIs. The parameters for the HFIs are shown in the following seven rows. |
| Fi.Increase | 1 | The amount by which a CCTI will be increased when a packet which was marked as having gone through a point of congestion is received. |
| Fi.Timer | 1 | When the timer expires it is reset to its specified value, and the CCTI is decremented by one. |
| Fi.Threshold | 8 | When the CCTI is equal to this value, an event is logged in the FI's cyclic event log. |

*continued...*



| Parameter | Default Value | Description |
|---|---|---|
| Fi.Min | 0 | This is the lowest value that CCTI can be reduced to; the default value is zero. |
| Fi.Limit | 127 | CCTI_Limit is the bounding value for a CCTI; CCTI cannot be greater than CCTI_Limit. |
| Fi.DesiredMaxDelay | 8300 | Maximum injection rate delay in ns, used to determine how the congestion control table entries are generated. |

### 5.4.7.5 SM Adaptive Routing

The parameters in the following table control Adaptive Routing configuration for the Intel switches in the fabric. The following is a sample of the Adaptive Routing section of the FM configuration file.

```
<!-- Configures support for AdaptiveRouting in Intel Switches -->
    <AdaptiveRouting>
      <!-- 1 = Enable, 0 = Disable -->
      <Enable>0</Enable>
      <!-- When set, only adjust routes when they are lost. -->
      <!-- If not set, adjust routes when they are lost and -->
      <!-- when congestion is indicated. -->
      <LostRouteOnly>0</LostRouteOnly>
      <!-- Algorithm the switch should use when selecting an egress port -->
      <!-- Algorithms are currently 0 = Random, 1 = Greedy and -->
      <!-- 2 = GreedyRandom. -->
      <Algorithm>0</Algorithm>
      <!-- Update Frequency: Specifies the minimum time between -->
      <!-- AR adjustments. Values range from 0 to 7 and are read as 2^n -->
      <!-- times 64 ms. Default is 4, or about 1 second  . -->
      <ARFrequency>4</ARFrequency>
      <!-- Congestion threshold above which switch uses adaptive routing. -->
      <!-- Congestion threshold is per-VL and measured by tag consumption
percentage. -->
      <!-- Values range from 0 to 7. -->
      <!-- 7, 6, 5, 4 correspond to 55%, 60%, 65%, and 70%, respectively. -->
      <!-- 3, 2, 1 correspond to 80%, 90%, and 100%, respectively. -->
      <!-- 0 means "Use firmware default" -->
      <Threshold>0</Threshold>
    </AdaptiveRouting>
```

**Table 17. SM Adaptive Routing Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Enable | 0 | Enables (1) or disables (0) Adaptive Routing. |
| LostRouteOnly | 0 | When set (1), only adjust routes when they are lost.<br>If not set (0), adjust routes when they are lost or when congestion is detected. |
| Algorithm | 0 | Algorithm the switch should use when selecting an egress port. Algorithms are currently 0 = Random, 1 = Greedy and 2 = GreedyRandom. Default is 0. |
| ARFrequency | 4 | Update Frequency: Specifies the minimum time between AR adjustments. Values range from 0 to 7 and are read as 2n times 64 ms. Default is 4, or about 1 second. |

***continued...***

| Parameter | Default Value | Description |
|---|---|---|
| | | The higher the number, the less frequently the AR adjusts, as the number represents a longer time span between adjustments. |
| Threshold | 0 | Congestion threshold above which switch uses adaptive routing. Values range from 0 to 7. 7,6,5,4 correspond to 55%, 60%, 65% and 70%. 30%, 20% and 10% correspond to 80%, 90% and 100%. 0 means "Use firmware default". Default value is 0.<br>Higher Percentage means that higher congestion is required before Adaptive routing takes control. Higher Percentage is less sensitive, less adaptive. |

### 5.4.7.6 Dynamic Packet Lifetime

`DynamicPacketLifetime` and `PacketLifetime` control the `PktLifetime` reported in PathRecord queries. `PktLifetime` is used by IBTA compliant applications to set the Queue Pair timeout and retry intervals. Per the IBTA algorithm the Queue Pair timeout will typically be two-times or four-times these values. When `DynamicPacketLifetime` is enabled, the `PktLifetime` reported will depend on number of switch hops. `Hops01` is `PktLifetime` for one-hop paths, `Hops02` is two-hop paths, etc. When `DynamicPacketLifetime` is disabled, the `PacketLifetime` is reported as the `PKtLifetime` for all paths. The following is a sample of the Dynamic Packet Lifetime section of the FM configuration file.

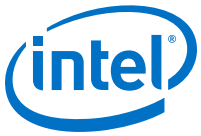
```
<PacketLifetime>1s</PacketLifetime>
<DynamicPacketLifetime>
      <Enable>1</Enable>
      <Hops01>67ms</Hops01>
      <Hops02>134ms</Hops02>
      <Hops03>134ms</Hops03>
      <Hops04>268ms</Hops04>
      <Hops05>268ms</Hops05>
      <Hops06>268ms</Hops06>
      <Hops07>268ms</Hops07>
      <Hops08>536ms</Hops08>
      <Hops09>536ms</Hops09>
</DynamicPacketLifetime>
```

The parameters in the following table define Packet Lifetime values based on HopCount of the path.

**Table 18. SM DynamicPacketLifetime Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Enable | 1 | Enables or disables the use of dynamic packet lifetime. This provides a convenient way to disable this section without needing to delete it from the configuration file |
| Hops01 | 67ms | Packet Lifetime for one-hop paths |
| Hops02 | 134ms | Packet Lifetime for two- hop paths |
| Hops03 | 134ms | Packet Lifetime for three-hop paths |
| Hops04 | 268ms | Packet Lifetime for four-hop paths |
| Hops05 | 268ms | Packet Lifetime for five-hop paths |

*continued...*



| Parameter | Default Value | Description |
|---|---|---|
| Hops06 | 268ms | Packet Lifetime for six-hop paths |
| Hops07 | 268ms | Packet Lifetime for seven-hop paths |
| Hops08 | 536ms | Packet Lifetime for eight-hop paths |
| Hops09 | 536ms | Packet Lifetime for nine or more hop paths |

### 5.4.8 Fabric Sweep

The Fabric sweep is the process by which the SM discovers fabric changes and then reprograms the parts of the fabric affected by the changes. The SM sweeps immediately upon fabric changes based on traps from the switches. Because traps can be lost, the SM also has a slow periodic sweep at `SweepInterval` to verify fabric configuration. The following is a sample of the Fabric Sweep section of the FM configuration file.

```
<!-- ***************** Fabric Sweep ************************** -->
<!-- The SM sweeps immediately upon fabric changes based on traps from -->
<!-- the switches.  Since traps can be lost, the SM also has a slow -->
<!-- periodic sweep at SweepInterval to verify fabric config. -->
<SweepInterval>300</SweepInterval> <!-- max seconds between sweeps -->
<IgnoreTraps>0</IgnoreTraps> <!-- don't sweep nor log when traps occur -->
<!-- The SM waits up to RespTimeout milliseconds for responses. -->
<!-- Upon a timeout, up to MaxAttempts are attempted for a given request -->
<MaxAttempts>3</MaxAttempts>
<RespTimeout>250</RespTimeout> <!-- in milliseconds -->
<!-- SM will start with MinRespTimeout as the timeout value for requests -->
<!-- and use multiples of this value for subsequent attempts if there is -->
<!-- a timeout in the previous attempt. SM will keep retrying till -->
<!-- cumulative sum of timeouts for retries is less than -->
<!-- RespTimeout multiplied by MaxAttempts. -->
<!-- If MinRespTimeout is set to 0, upon timeout, up to MaxAttempts -->
<!-- are attempted with each attempt having a timeout of RespTimeout -->
<MinRespTimeout>35</MinRespTimeout> <!-- in milliseconds -->
<!-- When there are a large number of fabric changes at once, the SM -->
<!-- could have lots of errors while attempting to access/program -->
<!-- devices which disappeared mid-sweep.  If the SM has more than -->
<!-- SweepErrorsThreshold in a given sweep, it will give up and start -->
<!-- the sweep over.  Should SweepAbandonThreshold sweeps fail in a row -->
<!-- the SM will will do its best to complete the sweep as is. -->
<SweepErrorsThreshold>0</SweepErrorsThreshold>
<SweepAbandonThreshold>3</SweepAbandonThreshold>
<!-- If a given port issues more than TrapThreshold traps/minute -->
<!-- it will be disabled as an unstable port.  0 disables this feature. -->
<!-- The traps managed by this threshold are Traps 129-131. -->
<!-- These include traps for port Local Link Integrity threshold, -->
<!-- Excessive Buffer Overrun, and Flow Control Update watchdog timer. -->
<!-- Valid values to enable this feature are 10-100. -->
<TrapThreshold>0</TrapThreshold>
<!-- TrapThresholdMinCount is minimum number of traps required -->
<!-- to consider that TrapThreshold rate has been been reached. -->
<!-- For example if TrapThreshold is set to 10 traps/minute and -->
<!-- TrapThresholdMinCount is set to 5, the port will be disabled  -->
<!-- after 5 traps are received at the rate of 10 traps per minute -->
<!-- i.e. 5 traps in 30 seconds. -->
<!-- This value must be greater than 2. Default is 10. Ignored if -->
<!-- TrapThreshold is set to 0.-->
<!-- Larger values will increase accuracy of detecting trap rate -->
<!-- but also increase the time between a trap surge and the SM -->
<!-- disabling a port. Very small values can lead to a port being -->
<!-- disabled just after a few traps. -->
<TrapThresholdMinCount>10</TrapThresholdMinCount>
<!-- When Suppress1x is enabled, the SM will not Activate 1x links and -->
<!-- will take Down any Active 1x links found -->
```

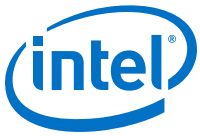
```
<Suppress1x>0</Suppress1x>
<!-- Multicast join/create/deletes result in a SM sweeps. -->
<!-- A pathological node can cause a denial of service attack by  -->
<!-- execessive creates and deletes of multicast groups.  The -->
<!-- following limits the number of multicast Set/Delete sequences -->
<!-- that a node can issue before the SM takes action to quiet  -->
<!-- the node. -->
<!-- The number of deletes allowed within a given interval is limited -->
<!-- by the following parameters.  -->
<!-- When McDosThreshold is zero, monitoring of MC DOS is disabled. -->
<McDosThreshold>0</McDosThreshold>
<!-- Default interval is 60 seconds.  -->
<McDosInterval>60</McDosInterval>
<!-- McDosAction, the action to take if MC DOS is suspected. -->
<!--    0 = Port will be disabled.  -->
<!--    1 = Port will be bounced.  -->
<McDosAction>0</McDosAction>
<!-- Sometimes when a node is under heavy load, it may fail to respond -->
<!-- to SMA queries for a while.  In order to prevent the SM from -->
<!-- dropping such nodes from the fabric, the SM will allow a node to be -->
<!-- non responsive for up to NonRespMaxCount sweeps and NonRespTimeout -->
<!-- seconds before dropping it from the fabric. -->
<NonRespTimeout>600</NonRespTimeout> <!-- in seconds -->
<NonRespMaxCount>3</NonRespMaxCount>
```

The parameters in the following table control Fabric Sweep.

**Table 19. SM Fabric Sweep Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| SweepInterval | 300 | The SM sweeps immediately upon fabric changes based on traps from the switches. Since traps can be lost, the SM also has a slow periodic sweep at a maximum of SweepInterval seconds to verify fabric configuration. |
| IgnoreTraps | 0 | Do not sweep or log when traps occur when enabled (1). |
| MaxAttempts<br>RespTimeout<br>MinRespTimeout | 3<br>250<br>35 | The SM spends up to `RespTimeout` multiplied by `MaxAttempts` per packet. These allow two modes of operation.<br>When `MinRespTimeout` is non-zero: the SM starts with `MinRespTimeout` as the time-out value for requests and use multiples of this value for subsequent attempts if there is a time-out in the previous attempt. SM keeps retrying until the cumulative sum of time-outs for retries is less than `RespTimeout` multiplied by `MaxAttempts`. This approach is recommended and reacts quickly to lost packets while still allowing adequate time for slower SMAs to respond.<br>When `MinRespTimeout` is zero: upon a time-out, up to `MaxAttempts` are attempted with each attempt having a time-out of `RespTimeout`. This approach is provided for backward compatibility with previous SM versions. |
| CumulativeTimeoutLimit. | 300 | The maximum time, in seconds, that the FM will spend waiting on all timeouts during a single sweep. Beyond this limit, the FM will abandon the current sweep, temporarily quarantine any ports that timed out, and resweep. This is intended to limit the amount of time the FM spends in any one sweep and increase responsiveness in the presence of disruptions.<br>Setting this value to zero disables the timeout limit. |
| SweepErrorsThreshold<br>SweepAbandonThreshold | 0<br>3 | When there are a large number of fabric changes at once, the SM could have lots of errors while attempting to access/program devices which disappeared mid-sweep. If the SM |

*continued...*



| Parameter | Default Value | Description |
|---|---|---|
| | | receives more than the set SweepErrorsThreshold parameter of errors in a given sweep, it gives up and starts the sweep over. When the SM abandons more than the set `SweepAbandonThreshold` parameter of sweeps in a row the SM does its best to complete the sweep. |
| TrapThreshold | 0 | If a given port issues more than the `TrapThreshold` traps/minute value it is disabled as an unstable port. 0 disables this feature. |
| TrapThresholdMinCount | 10 | Minimum number of traps required to reach the `TrapThreshold` rate. For example, if `TrapThreshold` is set to 10 traps per minute and `TrapThresholdMinCount` is set to 5, the port is disabled after 5 traps are received at the rate of 10 traps per minute (for example: 5 traps in 30 seconds). This parameter value must be greater than 2, and the default parameter value is 10. This parameter is ignored if `TrapThreshold` is set to 0. Larger values increase the accuracy of detecting the trap rate, but also increase the time between a trap surge and the SM disabling a port. Very small values can result in a port being disabled after a few traps. |
| Suppress1x | 0 | When `Suppress1x` is enabled, the SM does not activate 1x links and takes down any active 1x links found |
| McDosThreshold | 0 | Multicast Denial of Service Threshold (McDos)<br>A pathological node can cause a denial of service attack by excessive creates and deletes of multicast groups. The following limits the number of multicast Set/Delete sequences that a node can issue before the SM takes action to quiet the node.<br>The number of deletes allowed within a given interval is limited by the following parameters.<br>When McDosThreshold is zero, monitoring of MC DOS is disabled. |
| McDosInterval | 60 | McDos Interval<br>The interval in seconds for the number of deletes allowed.<br>Default is 60. |
| McDosAction | 0 | McDos Action<br>The action to take if McDos is suspected.<br>0 = Port will be disabled.<br>1 = Port will be bounced. |
| NonRespTimeout<br>NonRespMaxCount | 600<br>3 | When a node is under heavy load, it may fail to respond to SMA queries for a while. In order to prevent the SM from dropping such nodes from the fabric, the SM will allow a node to be non responsive for up to `NonRespMaxCount` sweeps and `NonRespTimeout` seconds before dropping it from the fabric. |

## 5.4.9 SM Logging and Debug

When nodes appear or disappear from the fabric, a message is logged. The SM Logging/Debug section defines how many messages are logged, and if additional SM sweep and SA query debug information is logged. The following is a sample of the SM Logging/Debug section of the FM configuration file.

```
<!-- ***************** SM Logging/Debug *************************** -->
<!-- When nodes appear or disappear from the fabric, a message is logged -->
<!-- This can set a threshold on how many such messages to output per -->
```

```
<!-- sweep.  Once NodeAppearanceMsgThreshold messages are logged in a -->
<!-- given sweep, the remainder are output at a lower log level (INFO) -->
<!-- Hence avoiding excessive log messages when significant -->
<!-- fabric changes occur.  0 means no limit. -->
<NodeAppearanceMsgThreshold>100</NodeAppearanceMsgThreshold>
<SmPerfDebug>0</SmPerfDebug> <!-- log additional SM sweep info -->
<SaPerfDebug>0</SaPerfDebug> <!-- log additional SA query info -->
```

The parameters in the following table control SM Logging.

**Table 20. SM Logging and Debug Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| NodeAppearanceMsgThreshold | 100 | This can set a threshold on how many log messages to output per sweep. Once `NodeAppearanceMsgThreshold` messages are logged in a given sweep, the remainder are output at a lower log level (INFO) therefore avoiding excessive log messages when significant fabric changes occur.<br>`0` means no limit. |
| SmPerfDebug | 0 | If `1`, then log additional SM sweep info. |
| SaPerfDebug | 0 | If `1`, then log additional SA query info.<br>Log additional SA query info `SM_0_sa_debug_perf:dec`<br>• `<Debug>0</Debug>` #SM_0_debug:dec<br>• `<RmppDebug>0</RmppDebug>` #SM_0_sa_debug_rmpp:dec |

## 5.4.10 Miscellaneous

In addition the SM supports the parameters described in the following sections.

### 5.4.10.1 LID

LID can be set for this SM as shown in the following table.

**Table 21. Additional SM Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| LID | 0 | LID for this SM, 0=pick any available.<br>0 is recommended. |

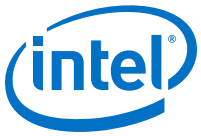
### 5.4.10.2 SM Overrides of the Common.Shared Parameters

The `Common.Shared` parameters can be overridden in the SM using the parameters described in the following table.

**Table 22. Additional Sm Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Priority | 0 | 0 to 15, higher wins. |
| ElevatedPriority | | 0 to 15, higher wins. |

*continued...*



| Parameter | Default Value | Description |
|---|---|---|
| LogLevel | 2 | *Note:* Overrides the `Common.Shared` LogLevel Settings<br>Sets log level option for SM:<br>• 0 = disable vast majority of logging output.<br>• 1 = fatal, error, warn (syslog CRIT, ERR, WARN).<br>• 2 = +notice, INFIIINFO (progress messages) (syslog NOTICE, INFO).<br>• 3 = +INFO (syslog DEBUG).<br>• 4 = +VERBOSE and some packet data (syslog DEBUG).<br>• 5 = +debug trace info (syslog DEBUG) This parameter is ignored for the Embedded FM. Refer to *Intel® Omni-Path Fabric Switches Command Line Interface Reference Guide* for information on configuring chassis logging options. |
| LogFile | | *Note:* Overrides `Common.Shared` setting.<br>Sets log output location for SM. By default (or if this parameter is empty) log output is accomplished using syslog. However, if a LogFile is specified, logging will be done to the given file. LogMode further controls logging. This parameter is ignored for the Intel® Embedded FM. Refer to *Intel® Omni-Path Fabric Switches Command Line Interface Reference Guide* for information on configuring chassis logging options. |
| SyslogFacility | Local6 | *Note:* Overrides `Common.Shared` setting.<br>For the Host FM, controls what syslog facility code is used for log messages. Allowed values are: auth, authpriv, cron, daemon, ftp, kern, local0-local7, lpr, mail, news, syslog, user, or uucp. For the Embedded FM, this parameter is ignored. |
| ConfigConsistencyCheckLevel | 2 | Controls the Configuration Consistency Check for SM. If specified for an individual instance of SM, overrides Shared settings. Checking can be completely disabled, or can be set to take action by deactivating Standby SM if configuration does not pass the consistency check criteria.<br>• 0 = disable Configuration Consistency Checking<br>• 1 = enable Configuration Consistency Checking without taking action (only log a message)<br>• 2= enable Configuration Consistency Checking and take action (log message and move standby to inactive state) |

### 5.4.10.3 Additional SM Parameters for Debug and Development

The SM supports the parameters in the following table to aid diagnosis and debug. Only use these parameters under the direction of your support representative.

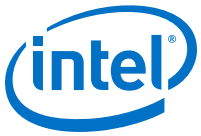
**Table 23. SM Debug Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| LoopTestOn | 0 | Cable loop test enable. |

***continued...***

| Parameter | Default Value | Description |
|---|---|---|
| | | When SM starts, if `LoopTestOn` is set to 1 a loop test is started in normal mode. The `LoopTestPackets` setting specifies how many packets will be injected into the loop test when started.<br>`LoopTest` is a good way to validate ISLs between switches in a cluster.<br>By default Loop test runs in default mode. In the default mode, the SM uses an exhaustive approach to setup loop routes and will include each ISL in as many loops as possible. This ensures that each ISL is exactly in the same number of loops and hence will see the same amount of utilization. But finding all possible loops is computationally intensive and can take a long amount of time. |
| LoopTestFastMode | 0 | Puts `LoopTest` in fast mode when set to 1, when started based on `LoopTestOn` setting.<br>Under this mode, loop test doesn't attempt to include each ISL in all possible loops, but includes it in at least the specified number of loops (this value is controlled via the `MinISLRedundancy` parameter).<br>In typical fast mode operations (with the default `MinISLRedundancy` of 4), injecting four packets into each loop is sufficient to get a high utilization on the ISLs. |
| LoopTestPackets | 0 | Number of packets to inject. If the XML tag is not set or commented out in the XML configuration file but loop test has been enabled in the configuration file, then a loop test starts but since the packet count is zero, no packets will be injected into the loops for testing. However, you can manually inject packets using one of the inject CLI commands. |
| LIDSpacing | 0 | Spacing of LIDs to test LFT |
| SaRmppChecksum | 0 | RMPP internal checksum |
| DynamicPortAlloc | 1 | This parameter is for development use only. |
| TrapLogSuppressTriggerInterval | 30 | If traps are received from the same port within the interval (in seconds), logging of traps from that port are suppressed.<br>Log suppression is disabled if set to 0. |
| CS_LogMask<br>MAI_LogMask<br>CAL_LogMask<br>DVR_LogMask<br>IF3_LogMask<br>SM_LogMask<br>SA_LogMask<br>PM_LogMask<br>PA_LogMask<br>FE_LogMask<br>APP_LogMask | 0x00000000<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff | Alternative to use of LogLevel. For advanced users, these parameters can provide more precise control over per subsystem logging. For typical configurations these should be omitted and the `LogLevel` parameter should be used instead.<br>For each subsystem there can be a `LogMask`. The mask selects severities of log messages to enable and is a sum of the following values:<br>0x1=fatal<br>0x2=actionable error<br>0x4=actionable warning<br>0x8=actionable notice<br>0x10=actionable info<br>0x20=error<br>0x40=warn<br>0x80=notice<br>0x100=progress<br>0x200=info<br>0x400=verbose<br>0x800=data<br>0x1000=debug1<br>0x2000=debug2 |

***continued...***



| Parameter | Default Value | Description |
|---|---|---|
| | | 0x4000=debug3<br>0x8000=debug4<br>0x10000=func call<br>0x20000=func args<br>0x40000=func exit<br>For embedded FM corresponding Chassis Logging must also be enabled and Sm configuration applies to all managers.<br>For Host FM, the Linux syslog service needs to have an appropriate level of logging enabled. |

## 5.5 FE Parameters

The following tables describe parameters that can be used in the `Sm` subsection of either the `Common` or `Fm` sections.

Any parameter that can be used in the `Common.Shared` section can also be used in the `Common.Fe` or `Fm.Fe` sections.

### 5.5.1 FE Overrides of the Common.Shared Parameters

The Common.Shared parameters can be overridden in the FE using the parameters described in the following table.

**Table 24. Additional FE Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| LogLevel | 2 | *Note:* Overrides the Common LogLevel Settings<br>Sets log level option for FE:<br>• 0 = disable vast majority of logging output<br>• 1 = fatal, error, warn (syslog CRIT, ERR, WARN)<br>• 2 = +notice, INFIIINFO (progress messages) (syslog NOTICE, INFO)<br>• 3 = +INFO (syslog DEBUG)<br>• 4 = +VERBOSE and some packet data (syslog DEBUG)<br>• 5 = +debug trace info (syslog DEBUG) This parameter is ignored for the Embedded FM. Refer to the *Intel® Omni-Path Fabric Switches Command Line Interface Reference Guide* for information on configuring chassis logging options. |
| LogFile | | *Note:* Overrides `Common.Shared` setting.<br>Sets log output location for FE. By default (or if this parameter is empty) log output is accomplished using syslog. However, if a LogFile is specified, logging will be done to the given file. LogMode further controls logging. This parameter is ignored for the Embedded FM. Refer to the *Intel® Omni-Path Fabric Switches Command Line Interface Reference Guide* for information on configuring chassis logging options. |
| SyslogFacility | Local6 | *Note:* Overrides `Common.Shared` setting. |

***continued...***

| Parameter | Default Value | Description |
|---|---|---|
| | | For the Host FM, controls what syslog facility code is used for log messages. Allowed values are: auth, authpriv, cron, daemon, ftp, kern, local0-local7, lpr, mail, news, syslog, user, or uucp.<br>For the Embedded FM, this parameter is ignored |

### 5.5.2 Additional FE Parameters for Debug and Development

The FE supports the parameters in the following table to aid diagnosis and debug. Only use these parameters under the direction of your support representative.

**Table 25. FE Debug Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Debug | 0 | *Note:* Overrides Debug setting from `Common.Shared`<br>Additional parameters for debug/development use - This enables debugging modes for FE. |
| RmppDebug | 0 | *Note:* Overrides RmppDebug setting from `Common.Shared`<br>If 1, then log additional FE info with regards to RMPP or the Reliable Message Passing Protocol. |
| CS_LogMask<br>MAI_LogMask<br>CAL_LogMask<br>DVR_LogMask<br>IF3_LogMask<br>SM_LogMask<br>SA_LogMask<br>PM_LogMask<br>PA_LogMask<br>FE_LogMask<br>APP_LogMask | 0x00000000<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff | Alternative to use of LogLevel. For advanced users, these parameters can provide more precise control over per subsystem logging. For typical configurations these should be omitted and the `LogLevel` parameter should be used instead.<br>For each subsystem there can be a `LogMask`. The mask selects severities of log messages to enable and is a sum of the following values:<br>0x1=fatal<br>0x2=actionable error<br>0x4=actionable warning<br>0x8=actionable notice<br>0x10=actionable info<br>0x20=error<br>0x40=warn<br>0x80=notice<br>0x100=progress<br>0x200=info<br>0x400=verbose<br>0x800=data<br>0x1000=debug1<br>0x2000=debug2<br>0x4000=debug3<br>0x8000=debug4<br>0x10000=func call<br>0x20000=func args<br>0x40000=func exit<br>For embedded FM corresponding Chassis Logging must also be enabled and SM configuration applies to all managers.<br>For Host FM, the Linux syslog service needs to have an appropriate level of logging enabled. |

### 5.5.3 FE Instance Specific Parameter

The parameter shown in the following table can be used only in the `Fm.Fe` section.

**Table 26. FE Instance Specific Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| TcpPort | 3245 | TCP socket to listen for Fabric Manager GUI on |

## 5.6 PM Parameters

The following tables describe parameters that can be used in the `Pm` subsection of either the `Common` or `Fm` sections.

Any parameter that can be used in the `Common.Shared` section can also be used in the `Common.Pm` or `Fm.Pm` sections.

### 5.6.1 PM Controls

The parameters shown in the following table set up the options for when and how the PM monitors the fabric.

**Table 27. PM Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| ServiceLease | 60 | ServiceRecord lease with SA in seconds. |
| SweepInterval | 10 | The PM constantly sweeps and computes fabric statistics If the `SweepInterval` is set to `0`, the PM will not perform sweeps. But instead if queried it will do an immediate PMA operation.<br>Tools such as `opatop` require PM `SweepInterval` be non-zero.<br>The default in the sample FM configuration file is ten seconds. However when upgrading from previous FM releases, if a FM configuration file is used without this value specified, a default of `0` is used. This permits upgrades to operate in a mode comparable to the existing configuration and requires specific user action to enable the new 6.0 and above PM features. |
| MaxClients | 3 | The maximum number of concurrent PA client applications (For example, `opareport`, `opatop`, `oparfm`) running against the same PM/PA. |
| TotalImages<br>FreezeFrameImages | 10<br>5 | The PM can retain recent fabric topology and performance data. Each such dataset is referred to as an Image. Images allow for access to recent history and/or Freeze Frame by clients. Each image consumes memory, so care must be taken not to take an excessive amount of memory, especially for larger fabrics.<br>`TotalImages` - total images for history and freeze<br>`FreezeFrameImages` - max unique frozen images |
| FreezeFrameLease | 60 | IF a PA client application hangs or dies, after this set time, all its frozen images will be released.<br>Specified in seconds |

## 5.6.2 PA Category Parameters

### 5.6.2.1 Thresholds

The Parameters shown in the following table set the thresholds for each category and exceeding these values for each category will print a log warning. 0 causes the given class of errors to be ignored.

**Table 28. Threshold Parameters**

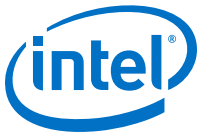
| Parameter | Default Value | Description |
|---|---|---|
| Integrity | 100 | Threshold for logging a warning indicating a possible error condition. |
| Congestion | 100 | Threshold for logging a warning indicating a possible error condition. |
| SmaCongestion | 100 | Threshold for logging a warning indicating a possible error condition. |
| Bubble | 100 | Threshold for logging a warning indicating a possible error condition. |
| Security | 10 | Threshold for logging a warning indicating a possible error condition. |
| Routing | 100 | Threshold for logging a warning indicating a possible error condition. |

### 5.6.2.2 Threshold Exceeded Message Limit

The parameters shown in the following table limit how many ports which exceed their PM Thresholds are logged per sweep. These can avoid excessive log messages when extreme fabric problems occur. These parameters can be used in the `ThresholdsExceededMsgLimit` section.

**Table 29. PM ThresholdsExceededMsgLimit Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Integrity | 10 | Maximum ports per PM Sweep to log if exceeds configured threshold. A value of 0 suppresses logging of this class of threshold exceeded errors. |
| Congestion | 0 | Maximum ports per PM Sweep to log if exceeds configured threshold. A value of 0 suppresses logging of this class of threshold exceeded errors. |
| SmaCongestion | 0 | Maximum ports per PM Sweep to log if exceeds configured threshold. A value of 0 suppresses logging of this class of threshold exceeded errors. |
| Bubble | 0 | Maximum ports per PM Sweep to log if exceeds configured threshold. A value of 0 suppresses logging of this class of threshold exceeded errors. |
| Security | 10 | Maximum ports per PM Sweep to log if exceeds configured threshold. A value of 0 suppresses logging of this class of threshold exceeded errors. |
| Routing | 10 | Maximum ports per PM Sweep to log if exceeds configured threshold. A value of 0 suppresses logging of this class of threshold exceeded errors. |



### 5.6.2.3 Integrity Weights

The parameters shown in the following table control the weights for the individual counters which are combined to form the `Integrity` count. These parameters can be used in the `IntegrityWeights` section.

**Table 30. PM IntegrityWeights Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| LocalLinkIntegrityErrors | 0 | Weight for LocalLinkIntegrityErrors counter.<br>0 causes counter to be ignored |
| RcvErrors | 100 | Weight for RcvErrors counter.<br>0 causes counter to be ignored |
| ExcessiveBufferOverruns | 100 | Weight for ExcessiveBufferOverruns counter.<br>0 causes counter to be ignored |
| LinkErrorRecovery | 0 | Weight for LinkErrorRecovery counter.<br>0 causes counter to be ignored |
| LinkDowned | 25 | Weight for LinkDowned counter.<br>0 causes counter to be ignored |
| UncorrectableErrors | 100 | Weight for UncorrectableErrors counter.<br>0 causes counter to be ignored |
| FMConfigErrors | 100 | Weight for FMConfigErrors counter.<br>0 causes counter to be ignored |
| LinkQualityIndicator | 40 | Weight applied to the LQI normalization equation. Calculation is 2^(5-LQI)-1.<br>0 causes counter to be ignored |
| LinkWidthDowngrade | 100 | Weight applied to the LWD normalization equation. Calculation is equal to the number of active lanes down:<br>LinkWidth.Active - LinkWidthDowngrade.RxActive<br>0 causes counter to be ignored |

### 5.6.2.4 Congestion Weights

The parameters shown in the following table control the weight to use for each individual counter when computing congestion, which are combined to form the `Congestion` count. Integrity errors can also cause congestion.

Pct (Percentage) means the specified counter is divided by the appropriate data transfer counter, then normalized to a predetermined range before being weighted and summed into the category.

**Table 31. PM CongestionWeights Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| XmitWaitPct | 10 | Weights for XmitWeight counter divided by an associated data transfer counter and normalized to a predetermined range of 0.01% to 1%. 0 causes given counter to be ignored. |
| CongDiscards | 100 | Weights for SwPortCongestion counter.<br>0 causes given counter to be ignored. |

***continued...***

| Parameter | Default Value | Description |
|---|---|---|
| RcvFECNPct | 5 | Weights for RcvFECN counter divided by an associated data transfer counter and normalized to a predetermined range of 0.1% to 10%.<br>0 causes given counter to be ignored. |
| RcvBECNPct | 1 | Weights for RcvBECN counter divided by an associated data transfer counter and normalized to a predetermined range of 0.1% to 10%.<br>0 causes given counter to be ignored. |
| XmitTimeCongPct | 25 | Weights for XmitTimeCongestion counter divided by an associated data transfer counter and normalized to a predetermined range of 0.1% to 10%.<br>0 causes given counter to be ignored. |
| MarkFECNPct | 25 | Weights for MArkFECN counter divided by an associated data transfer counter and normalized to a predetermined range of 0.1% to 10%.<br>0 causes given counter to be ignored. |

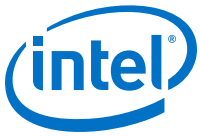
### 5.6.3 PM Sweep Operation Control

The parameters shown in the following table control the operation of the PM during each sweep.

**Table 32. PM Sweep Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Resolution | | Resolution determines the number of LocalLinkIntegrity or LinkErrorRecovery errors must occur before the PMA will include them in the ErrorCounterSummary. Most counters are 64 bits wide and are expected to not ever saturate in the up-time of a port. |
| .LocalLinkIntegrity | 8000000 | |
| .LinkErrorRecovery | 100000 | |
| ErrorClear | 7 | This controls when the PM clears PMA Error counters<br>0 = clear when non-zero<br>1 = clear when 1/8 of individual counters max<br>2 = clear when 2/8 of individual counters max<br>...<br>7 = clear when 7/8 of individual counters max |
| ClearDataXfer | 0 | Enable clearing of Data Transfer Counters. |
| Clear64bit | 0 | Enable clearing of 64-bit Error Counters. |
| Clear32bit | 1 | Enable clearing of 32-bit Error Counters. |
| Clear8bit | 1 | Enable clearing of 8-bit Error Counters. |
| ProcessHFICounters | 1 | Enable processing (sweeping) of HFI Counters. |
| ProcessVLCounters | 1 | Enable processing (sweeping) of VL Counters. |
| PmaBatchSize | 2 | Maximum concurrent PMA requests the PM can have in flight while querying the PMAs in the fabric. |

*continued...*



| Parameter | Default Value | Description |
|---|---|---|
| MaxParallelNodes | 10 | Maximum nodes to concurrently issue parallel requests to a given PMA. |
| MaxAttempts<br>RespTimeout<br>MinRespTimeout | 3<br>250<br>35 | The PM will spend up to `RespTimeout * MaxAttempts` per packet. These allow two modes of operation.<br>When `MinRespTimeout` is non-zero, the PM will start with `MinRespTimeout` as the time-out value for requests and use multiples of this value for subsequent attempts if there is a time-out in the previous attempt. PM will keep retrying until the cumulative sum of time-outs for retries is less than `RespTimeout` multiplied by `MaxAttempts`. This approach is recommended and will react quickly to lost packets while still allowing adequate time for slower PMAs to respond.<br>When `MinRespTimeout` is zero, upon a time-out, up to `MaxAttempts` are attempted with each attempt having a time-out of `RespTimeout`. This approach is provided for backward compatibility with previous PM versions. |
| SweepErrorsLogThreshold | 10 | Maximum number of PMA node or Port warning messages to output per sweep with regard to nodes that cannot be properly queried. |

## 5.6.4 PM Overrides of the Common.Shared Parameters

The `Common.Shared` parameters can be overridden in the PM using the parameters described in the following table.

### Table 33. Additional PM Parameters

| Parameter | Default Value | Description |
|---|---|---|
| LogLevel | 2 | *Note:* Overrides the `Common.Shared` LogLevel Settings<br>Sets log level option for PM:<br>• 0 = disable vast majority of logging output<br>• 1 = fatal, error, warn (syslog CRIT, ERR, WARN)<br>• 2 = +notice, INFIINFO (progress messages) (syslog NOTICE, INFO)<br>• 3 = +INFO (syslog DEBUG)<br>• 4 = +VERBOSE and some packet data (syslog DEBUG)<br>• 5 = +debug trace info (syslog DEBUG) This parameter is ignored for the Embedded FM. Refer to the *Intel® Omni-Path Fabric Switches Command Line Interface Reference Guide* for information on configuring chassis logging options. |
| LogFile | | *Note:* Overrides `Common.Shared` setting.<br>Sets log output location for PM. By default (or if this parameter is empty) log output is accomplished using syslog. However, if a LogFile is specified, logging is done to the given file. LogMode further controls logging. This parameter is ignored for the Embedded FM. Refer to the *Intel® Omni-Path Fabric* |

***continued...***

| Parameter | Default Value | Description |
|---|---|---|
| | | *Switches Command Line Interface Reference Guide* for information on configuring chassis logging options. |
| SyslogFacility | Local6 | *Note:* Overrides `Common.Shared` setting.<br>For the Host FM, controls what syslog facility code is used for log messages. Allowed values are: auth, authpriv, cron, daemon, ftp, kern, local0-local7, lpr, mail, news, syslog, user, or uucp. For the Embedded FM, this parameter is ignored. |
| ConfigConsistencyCheckLevel | 2 | Controls the Configuration Consistency.<br>Check for PM. If specified for an individual instance of PM, will override Shared settings. Checking can be completely disabled, or can be set to take action by deactivating Secondary PM if configuration does not pass the consistency check criteria.<br>• 0 = disable Configuration Consistency Checking<br>• 1 = enable Configuration Consistency Checking without taking action (only log a message)<br>• 2 = enable Configuration Consistency Checking and take action (log message and shutdown Secondary PM) |
| Priority | 0 | 0 to 15, higher wins. |
| ElevatedPriority | | 0 to 15, higher wins. |

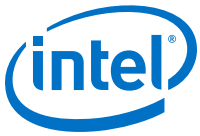
## 5.6.5 PM Short-Term History PM Parameters

The following parameters are used to enable and customize the off-loading of RAM-resident images to disk in order to preserve counter history for a long period of time.

### Table 34. Short-Term History PM Parameters

| Parameter | Default Value | Description |
|---|---|---|
| Enable | 1 | Enable Short Term History. Will automatically disable when running on ESM. |
| StorageLocation | `/var/usr/lib/opa-fm/pm0_pahistory` | The absolute path where the history files will be stored.<br>StorageLocation is a PM instance-specific parameter (similar to TcpPort for the FE).<br>The default value is `/var/usr/lib/opa-fm/pm0_pahistory` (where '0' is the number of the instance). If there are multiple instances of the PM, then `StorageLocation` must be unique for each of them. |
| TotalHistory | 24 | The total number of hours of history that will be stored. This is a limit on the total amount of data stored, and not a limit on a file's age. PM downtime does not count towards the total time. |

*continued...*



| Parameter | Default Value | Description |
|---|---|---|
| ImagesPerComposite | 3 | Determines how many images will be compounded into a single image as part of writing to the file. A higher number will save disk space but will result in a lower data granularity. Must not be 0. |
| MaxDiskSpace | 1024 | A cap on how much disk space (in MiB) the short term history is allowed to use. If this size is exceeded, the oldest files will be deleted to save space. |
| CompressionDivisons | 8 | Determines how many divisions will be used to concurrently compress or decompress data. Recommend less than or equal to number of processing cores of the management node, must not exceed 32. |

### 5.6.6 PM/PA Fail-over Parameters

The following parameters are used to enable and control the PM Fail-over extension.

**Table 35. PM/PA Fail-over Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| ImageUpdateInterval | 5 | `ImageUpdateInterval` is defined as the interval at which the MASTER PM updates STANDBY PMs with an image (when an image is available); otherwise image updates occur as they are available. Note that if multiple STANDBY PMs exist, the MASTER PM updates each STANDBY concurrently during the `ImageUpdateInterval`. In the Event PA Short Term History is enabled on all PMs, the MASTER PM will update all STANDYBY PMs with disk-resident images once all RAM-resident images have been updated.<br>`ImageUpdateInterval` must be less than the PM `SweepInterval` in order for the MASTER PM to be able to send images fast enough to keep up with new images as well as catch-up on older images. If `ImageUpdateInterval` is greater than `SweepInterval`, then `ImageUpdateInterval` is set equal to `SweepInterval` and a warning message is logged.<br>Setting `ImageUpdateInterval` to 0 turns off the transfer of images to STANDBY PMs.<br>The default value for `ImageUpdateInterval` is based upon the (`SweepInterval` / 2) rounded down to the nearest integer; must be at least 1. |

### 5.6.7 Additional PM Parameters for Debug and Development

The PM supports the parameters in the following table to aid diagnosis and debug. Only use these parameters under the direction of your support representative.

**Table 36. PM Debug Parameters**

| Parameter | Default Value | Description |
|---|---|---|
| Debug | 0 | *Note:* Overrides Debug setting from `Common.Shared`. |

***continued...***

| Parameter | Default Value | Description |
|---|---|---|
| | | Additional parameters for debug/development use. This enables debugging modes for PM. |
| RmppDebug | 0 | *Note:* Overrides RmppDebug setting from `Common.Shared`.<br>If 1, then log additional PM info with regards to RMPP or the Reliable Message Passing Protocol. |
| CS_LogMask<br>MAI_LogMask<br>CAL_LogMask<br>DVR_LogMask<br>IF3_LogMask<br>SM_LogMask<br>SA_LogMask<br>PM_LogMask<br>PA_LogMask<br>FE_LogMask<br>APP_LogMask | 0x00000000<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff<br>0x000001ff | Alternative to use of `LogLevel`. For advanced users, these parameters can provide more precise control over per subsystem logging. For typical configurations these should be omitted and the `LogLevel` parameter should be used instead.<br>For each subsystem there can be a `LogMask`. The mask selects severities of log messages to enable and is a sum of the following values:<br>0x1=fatal<br>0x2=actionable error<br>0x4=actionable warning<br>0x8=actionable notice<br>0x10=actionable info<br>0x20=error<br>0x40=warn<br>0x80=notice<br>0x100=progress<br>0x200=info<br>0x400=verbose<br>0x800=data<br>0x1000=debug1<br>0x2000=debug2<br>0x4000=debug3<br>0x8000=debug4<br>0x10000=func call<br>0x20000=func args<br>0x40000=func exit<br>For embedded FM corresponding Chassis Logging must also be enabled and SM configuration applies to all managers.<br>For Host FM, the Linux syslog service will need to have an appropriate level of logging enabled. |

## 5.7 FM Instance Shared Parameters

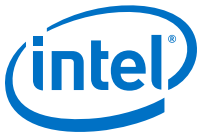
Any parameter that can be used in the `Common.Shared` section may also be used in the `Fm.Shared` section.

The following table describes parameters that are used in the `Shared` subsection of `Fm` section.

### Table 37. FM Instance Shared Parameters

| Parameter | Default Value | Description |
|---|---|---|
| Name | fm0 | Name for the given FM instance. Also used to mark log messages with `_sm`, `_fe`, `_pm`, appended |

*continued...*



| Parameter | Default Value | Description |
|---|---|---|
| | | This parameters is ignored for the Embedded FM. |
| Hfi<br>Port<br>PortGUID | 1<br>1<br>0 | Each FM runs on a single local HFI port. The port may be specified by `Hfi` and `Port` or by `PortGuid`<br>`Hfi` = Local HFI to use for Fm Instance, 1=First HFI<br>`Port` = Local HFI port to use for Fm instance, 1=First Port<br>`PortGUID` = Local HFI port to use for Fm instance<br>If `PortGUID` is `0`, it is ignored and `Hfi/Port` is used. When `PortGUID` is non-zero, it will be used and `Hfi/Port` will be ignored.<br>These parameters are ignored for the Embedded FM. |
| SubnetPrefix | 0xfe80000000000000 | Unique subnet prefix to assign to this Fabric. The same subnet prefix must be used by all redundant SMs on a given fabric. |

## 5.8 Changing Parameters and Impacts to System Operation and Performance

Parameters that are very likely to disrupt a live system should generally not be changed while applications are running. To minimize disruptions, Intel recommends the following steps.

1. Stop all Fabric Managers (see Stopping the Fabric Manager on page 138).
2. Move all host-to-switch links to `Init`, which stops all fabric applications, using either of these methods:
   - Reset all hosts or bounce their fabric links
   - Reset all switches
3. Complete the change to the FM configuration.
4. Restart all the Fabric Managers.

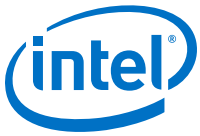
The following actions may be disruptive and should be avoided while applications are running;

- Increasing `LMC`, `LMCE0`

Parameters that may disrupt some applications or impact application performance and whose changes must be carefully considered on a live fabric are the following:

- Decreasing `LMC`, `LMCE0` - some applications could continue to use LIDs whose assignments have changed and may fail or enter recovery and reconnection modes.
- Changes to PathRecord settings (`PathSelection`) - Applications that use PathRecords to select end-to-end addresses may ignore any resultant changes and continue to use the previous settings until the application is restarted or it refetches its PathRecords. This can change what paths applications use (only applicable when `LMC != 0`) and cause application performance changes or impacts.

- Activating or deactivating pre-defined pre-Enabled vFabrics - this is intended to be permitted on a live fabric. However applications attempting to use a deactivated vFabric will fail after the vFabric is deactivated. In general such applications should be stopped prior to deactivating the given vFabric. Activation of a pre-defined pre-enabled vFabric is typically safe.
- Changing device groups or applications associated with an active vFabric - applications or nodes whose assignments have changed may fail or continue executing with the old PathRecords and therefore not fully obey the change.
- Changes to an active vFabric's bandwidth, timeouts, priority, or preemption/traffic flow optimization rank - this will change arbitration and scheduling within the fabric. Application operation should not be impacted; however, performance and scheduling of fabric bandwidth will change to the new settings in a timely manner. Applications that use PathRecords to compute end-to-end timeouts may ignore any resultant changes in their timeouts and continue to use the previous timeouts until the application is restarted or refetches its PathRecords. Typically, changes to timeouts are small and have limited impact, but under extreme congestion situations, this could cause unexpected application performance impacts.
- Changes to FM VL buffer allocation parameters (`MinSharedVLMem`, `DedicatedVLMemMulti`, `WireDepthOverride`, `ReplayDepthOverride`). These parameters tune VL buffering and therefore affect application performance.
- Changes to these parameters are activated only when a port is bounced.
- Changes to Preemption (`LargePacket`, `SmallPacket`, `PreemptLimit`) - These parameters tune preemption/traffic flow optimization and therefore affect application performance.
- Changes to CongestionControl (`CongestionControl`).
- Changes to routing algorithm (`RoutingAlgorithm`, `SpineFirstRouting`, `ForceRebalance`, `FatTreeTopology`, `DGShortestPathTopology`, `AdaptiveRouting`) - these can result in rerouting of the fabric and may briefly disrupt traffic. If incorrect (or much better than previous) choices are made, fabric and application performance can be greatly affected (for better or worse).
- `ForceAttributeRewrite` - this can undo previous adaptive routing decisions in switches and result in a short term change to application performance.
- Changing FM security parameters (`PreDefinedTopology`, `VL15CreditRate`, `SmaSpoofingCheck`, `McDosThreshold`, `McDosInterval`, `McDosAction`, `SmAppliance`) - in general accurate changes should not impact existing applications. However, mistakes or tightening of security may disable nodes or applications in the cluster that were running prior to the change. Such changes may be intentional and desired if the goal is to tighten security and impact such applications or nodes.
- Multicast - changes to FM Multicast section of parameters may impact assignments of MLIDs to multicast groups or QoS parameters (`Mtu`, `Rate`) or Security parameters (`Pkey`) associated with a given multicast group. Typically on FM restart, end nodes must rejoin multicast groups and if these parameters change there can be minor disruptions of multicast traffic via lost packets. Most multicast applications will ride through these short term disruptions without error.
- Changes to fabric timeouts (`TimerScalingEnable`, `SwitchLifetime`, `HoqLife`, `VLStallCount`). On uncongested fabrics minor changes often do not affect applications. Significant decreases in values may cause congestion



mitigation mechanisms in switches to fire sooner, resulting in packet discards and changes to application performance. Significant increases may slow switch mitigation and allow congestion to propagate further and impact application performance.

- Changes to Link Policies (`HFILinkPolicy`, `ISLLinkPolicy`) - if all active links are currently "in policy" this will have no impact to existing applications. However, mistakes or tightening of link policies may disable nodes or links in the cluster that were running prior to the change. Such changes may be intentional and desired if the goal is to tighten link policies and impact such nodes or links.
- Changes to SA timeouts (`SaRespTime`, `NoReplyIfBusy`) - will impact how long applications will wait for the SA and how the SA handles excessive requests when busy. For most applications this has limited impact but under extreme situations application behavior and performance with regard to SA queries can change.
- Changes to FM SubnetSize - this alters the memory used by the FM for selected buffering and may impact FM performance. In general there should be no impact to existing applications beyond changes to the FM's responsiveness to PA and SA requests.
- QueryValidation - Intel recommends not changing this as non-compliant OFA applications may cease to work if it changes from 0 to 1.
- LID - may change the SM's LID and briefly disrupt application queries to the SA.

Parameters that can safely be changed on a live system without impacting applications are the following:

- PM/PA parameters (may impact active management and monitoring applications, but will not impact non-management applications running on the cluster). This includes adding or removing PM port groups, altering DeviceGroups that are only used as PmPortGroups.
- FM controls on FM core dumps (`CoreDumpLimit`, `CoreDumpDir`)
- Changes to FM failover and DB Sync parameters (`Priority`, `ElevatedPriority`, `ConfigConsistencyCheckLevel`, `MasterPingInterval`, `MasterPingMaxFail`, `DbSyncInterval`). However, such changes may result in a different FM becoming master or the movement of inconsistent FMs to standby.
- Changes to FM sweep time, SMA timeouts/retries, SM strategies (`VL15FlowControlDisable`, `SmaBatchSize`, `MaxParallelReqs`, `SweepInterval`, `IgnoreTraps`, `MaxAttempts`, `RespTimeout`, `MinRespTimeout`, `SweepErrorsThreshold`, `SweepAbandonThreshold`, `TrapThreshold`, `TrapThresholdMinCount`, `NonRespTimeout`, `NonRespMaxCount`, `SwitchCascadeActivateEnable`, `NeighborNormalRetries`) - this mainly impacts SM responsiveness to fabric changes and SM handling of non-responsive nodes. Typically has no application impact. However if adjustments are such that nodes are now not responsive within expected time frames, SM could drop some nodes from the fabric.
- Changes to FM logging (`LogLevel`, `LogMode`, `SyslogFacility`, `Debug`, `RmppDebug`, `*LogMask*`, `NodeAppearanceMsg`, `SmPerfDebug`, `SaPerfDebug`, `PortBounceLogLimit`).
- Changes to FE configuration `(SslSecurity*`.

# 6.0 Virtual Fabrics

The configuration of vFabrics consists of the following sections:

- Applications - describes applications that can run on one or more end nodes
- DeviceGroups - describes a set of end nodes in the fabric
- VirtualFabrics - defines a vFabric consisting of a group of applications, a set of devices, and the operating parameters for the vFabric

Each Application, DeviceGroup, and VirtualFabric must be given a unique name that is used to reference it. Applications, DeviceGroups, and VirtualFabrics may be defined in the `Common` or `Fm` sections. Those defined in the `Common` section apply to all `Fm` Instances. Those defined in an `Fm` section describe additional Applications, DeviceGroups, or VirtualFabrics that are specific to the given `Fm` instance.

## 6.1 Virtual Lane Support

Different ports (nodes) connected in the fabric cluster can support varying number of Virtual Lanes (VLs). Generally, Intel® Omni-Path products support eight (8) VLs, but future products may support a different number of VLs. When configuring vFabrics, every SL in use maps to a unique VL. This imposes a limit of 8 SL assignments. To enable more QoS vFabrics, you must configure them to share SLs (see Sharing SLs Between Multiple vFabrics on page 133).

## 6.2 Dynamic Fabric Configuration

Dynamic Fabric Configuration is a feature designed for fabric cluster environments where it is desirable to change Virtual Fabrics on-the-fly, (such as multi-tenant environments). Virtual fabrics can be modified in the configuration file and applied to the fabric *without* stopping and restarting the SM. These changes are also non-disruptive to the fabric cluster.

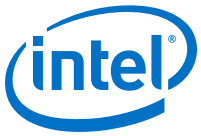
These characteristics are:

- Virtual Fabric Bandwidth settings
- Virtual Fabric active/standby

Other changes to the configuration file require a restart of the SM and may also be disruptive to the cluster.

## 6.3 Application Parameters

Applications are defined within the `Applications` section. This section contains zero or more `Application` sections.



The layout is as follows:

```
<Applications>
    <Application>
      <!-- application parameters -->
    </Application>
</Applications>
```

Each `Application` section has zero or more `ServiceIDs` subsections and/or `MGIDs` subsections. These are matched against PathRecord and Multicast SA queries so that the returned SLID/DLID, PKey, SL, and so on are appropriate for the vFabric that contains the application(s).

`ServiceIDs` subsections are 64-bit values that identify applications within a PathRecord query. In many ways ServiceIDs are the Intel® Omni-Path's equivalent of TCP socket ports. ServiceIDs are typically used within the Intel® Omni-Path Communication Manager protocol to identify the application making a connection request. When an application issues a PathRecord query to the SA, the ServiceID in the query is compared against the `ServiceID` subsections in the various `VirtualFabrics` sections.

ServiceIDs are assigned by application writers and standards bodies such as IEEE. Consult with the application supplier to determine the Service IDs used by the application.

Multicast GIDs (MGIDs) are 128-bit values that identify multicast groups for Unreliable Datagram applications such as IPoIB. MGIDs are represented as two 64-bit values separated by a colon (:). For example: `0xabc:0x123567`

This way of representing 128-bit values is the same as used in other FastFabric Toolset commands such as `opasaquery`, `opashowmc`, and so on. Applications not used in any vFabric have no effect.

*Note:* The default `opafm.xml` configuration file contains many standard preconfigured applications that are referenced in VirtualFabrics, which can be created by the administrator.

*Note:* Intel® Omni-Path mechanisms establish connections for the association of applications to VirtualFabrics, by using the ServiceID in PathRecord SA queries (for unicast applications) and the use of MGIDs in McMemberRecord SA queries (for multicast applications).

*Note:* Some unicast applications, notably `openmpi`, `mvapich` and `mvapich2`, use nonstandard mechanisms to establish connections. Therefore, the unicast applications must have their PKey and BaseSL manually configured consistent with the vFabric used.

**Table 38. Application Parameters**

| Parameter | Description |
|---|---|
| Name | Name for Application.<br>Every Application must have a unique name. The name must be unique among all Application names within an FM instance. When defined at the Common level, the name must be unique within all instances. |

***continued...***

| Parameter | Description |
|---|---|
| | The name is limited to 64 characters and is case sensitive. |
| ServiceID | A single 64-bit service ID to match against. |
| ServiceIDRange | A range of service IDs to match against.<br>Any service ID within the range (inclusive) is considered a match.<br>The range is two 64-bit values separated by a dash such as:<br>0-0xffffffffffffffff |
| ServiceIDMasked | A masked compare of service ID to match against.<br>Matches service IDs that when ANDed with second value (the mask) match the first value.<br>The mask is two 64-bit values separated by a * such as:<br>0x120003567*0xff000ffff |
| MGID | A single 128-bit MGID to match against. |
| MGIDRange | A range of MGIDs to match against.<br>Any MGID within the range (inclusive) is considered a match.<br>The range is two 128-bit values separated by a dash such as:<br>0:0-0xffffffffffffffff:0xffffffffffffffff |
| MGIDMasked | A masked compare of MGID to match against.<br>Matches MGIDs that when ANDed with second value (the mask) match the first value.<br>The mask is two 128-bit values separated by a * such as:<br>0xabc:0x120003567*0xfff:0xff000ffff |
| Select | Special selection cases.<br>The following selection cases can be used as an easy catch-all:<br>• `UnmatchedServiceID` - matches all applications' service IDs that match none of the vFabrics after filtering by `src/dest/requestor`.<br>• `UnmatchedMGID` - matches all applications' MGIDs that match none of the vFabrics after filtering by `src/dest/requestor`.<br>• `SA` - allows an application to be specified that includes SA queries. This allows SA query operations to be assigned to an appropriate vFabric. SA queries must use the default Partition Key (0x7fff or 0xffff). However, other aspects of SA access can be controlled (SL, and so on).<br>• `PA` - allows an application to be specified that includes all PM sweep traffic. This allows PM operations to be assigned to an appropriate vFabric in a manner similar to that of the SA selector. Applications containing PM traffic must use the management Partition Key. Only **one** VF may reference any application group containing PM traffic. Additionally, all ports in the fabric must be either a full or limited member of this VF.<br>The listed specifiers are case insensitive.<br>By having multiple select parameters, multiple special cases can be combined as needed in the same application section. |
| IncludeApplication | This includes all of the service IDs, MGIDs, and special selections in the given application.<br>Loops (including the parent application) are not allowed.<br>There is a nesting limit of 32. |

A working example configuration for Applications is provided in the default FM configuration file.

## 6.4 DeviceGroup Parameters

Device groups are defined within the `DeviceGroups` section. This section contains zero or more `DeviceGroup` sections.

The layout is as follows:

```
<DeviceGroups>
    <DeviceGroup>
      <!-- device group parameters -->
    </DeviceGroup>
</DeviceGroups>
```

Each `DeviceGroup` section can have one or more devices (nodes and ports). Devices are matched against PathRecord and Multicast SA queries so that the returned SLID/DLID, PKey, SL, and so on are appropriate for the vFabric that contains the device(s).

Devices in a `DeviceGroup` but not found in the fabric are ignored.

When security is enabled for a vFabric, further measures will be taken, using PKeys, to secure the vFabric and ensure devices in other vFabrics cannot talk to devices in the given vFabric. Such security includes hardware enforced per-packet PKey checking and enforcement.

*Note:* To effectively use the `DeviceGroups` section, the administrator must carefully list the required devices in each `DeviceGroup` section. FastFabric Toolset commands, such as `opasaquery` and `opareport`, can help generate the lists.

*Note:* Portions of `opareport` XML output, such as `opareport -o brnodes -x`, can be cut and pasted into the appropriate `DeviceGroup` sections.

There is a limit of the number of Secure VirtualFabrics a given device's port can be in. That limit is dependent on the PKey capabilities of the hardware.

The Intel® Omni-Path switch ports support 32 PKeys. Since the FM supports a limit of 32 vFabrics, all Intel® Omni-Path switch ports can support the full set of Secure VirtualFabrics.

The current Intel® Omni-Path HFI port supports 16 PKeys, so each HFI port is limited to participating in 16 Secure VirtualFabrics.

In a typical customer configuration, Switch Port 0 only needs to be part of the admin or default vFabric, for example, PKey 0xffff.

When using non-secure VirtualFabrics, the FM may consolidate multiple VirtualFabrics to a single PKey so that a configuration is valid that contains more vFabrics than the PKey capability.

The following table lists the parameters and their descriptions for the `DeviceGroup` subsection under the `DeviceGroups` section.

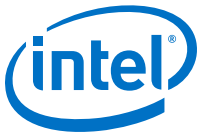
**Table 39. DeviceGroup Parameters**

| Parameter | Description |
|---|---|
| Name | Name for DeviceGroup. |

*continued...*

| Parameter | Description |
|---|---|
| | Every DeviceGroup must have a unique Name. The name must be unique among all DeviceGroup names within an FM instance. When defined at the Common level must be unique within all instances.<br>The name is limited to 64 characters and is case sensitive. |
| SystemImageGUID | Include all of the ports and nodes within the given system as identified by its 64-bit System Image GUID. |
| NodeGUID | Selects all of the ports in the node (a Fabric Interface or switch is a single node) as identified by its 64-bit node GUID. |
| PortGUID | Selects the given port as identified by its 64-bit port GUID. |
| NodeDesc | Selects all nodes matching the given name.<br>Limited wildcard support allows a single NodeDesc to match multiple nodes. * represents zero or more of alphabetic, numeric, and "-" ","="." "_" 'characters. ? represents zero or one alphabetic, numeric, or "-" "," "=" "." "_" character. [##-##] represents numbers within a specified range.<br>While easier to configure, use of this mechanism is less secure than specification using GUIDs. It's very easy for a systems node description to be changed.<br>You can also specify a port range to include on a matching node description. You can use the following syntax to specify a port range : [##-##] (i.e.: switch:[2-4] would match node name "switch", ports 2, 3, and 4) |
| NodeType | Selects all ports on all nodes of the specified node type. The following types may be specified:<br>• FI - All Fabric Interfaces<br>• SW - All Switches<br>The listed types are case insensitive.<br>By having multiple NodeType parameters, multiple node types can be combined as required in the same DeviceGroup section. |
| Select | Special selection cases.<br>The following selection cases can be used as an easy catch-all:<br>• All - All Devices<br>• Self - This FM instance's port<br>• SWE0 - Every Switch Port 0 with Enhanced Port 0 in Capability Mask. By definition also includes all Embedded SMs and managed spines of Intel® Internally Managed Switches.<br>• AllMgmtAllowed - Selects every HFI port connected to a switch port that is configured to support a management node.<br>The listed selection cases are case insensitive.<br>By having multiple Select parameters, multiple selections can be combined as required in the same DeviceGroup section. |
| IncludeGroup | This will include all the devices in the given DeviceGroup.<br>Loops (including the parent DeviceGroup) are not allowed.<br>It is valid to have more than one specification match the same device; in which case, the device is only included in the DeviceGroup once.<br>There is a nesting limit of 32. |

*Note:* The SystemImageGUID, NodeGUID, PortGUID, and NodeDesc for devices presently in the fabric can be identified by using opareport or opareport -o comps. If required opareport -x or opareport -o comps -x provides an XML output from which individual NodeGUID, PortGUID, SystemImageGUID, or NodeDesc lines can be cut/pasted into the required DeviceGroup sections.



A working example `DeviceGroups` configuration is provided in the default FM configuration file.

*Note:*

The `AllSMs DeviceGroup` has been deprecated, and users should use `AllMgmtAllowed` instead.

## 6.5 VirtualFabric Parameters

vFabrics are defined within the `VirtualFabrics` section. This section contains zero or more `VirtualFabric` sections.

The layout is as follows:

```
<VirtualFabrics>
    <VirtualFabric>
      <!-- virtual fabric parameters -->
    </VirtualFabric>
</VirtualFabrics>
```

Each `VirtualFabric` contain the following:

- One or more groups of devices
- One or more sets of Applications
- Administrator policies

The vFabrics control the security configuration for the given set of devices and applications in the overall fabric.

Each `VirtualFabric` section can have one or more `Applications` and `DeviceGroups` sections. Both the devices and the applications are matched against PathRecord and Multicast SA queries. The returned SLID/DLID, PKey, SL, and so on, are appropriate for the vFabric that contain the involved devices and applications.

### 6.5.1 Device Membership and Security

Devices are specified by the devicegroup name in the following ways in the `VirtualFabric` section:

- As Full Members:

  ```
  <Member>group_name</Member>
  ```

  Such devices may talk to any other `Member` or `LimitedMember`.

- As Limited Members:

  ```
  <LimitedMember>group_name</LimitedMember>
  ```

When `Security` is `1` (On), `LimitedMembers` are not permitted to talk to other `LimitedMembers`. However, `LimitedMembers` can always talk to `Members`. `LimitedMembers` cannot join multicast groups in the vFabric.

When security is on for a vFabric, PKeys and switch hardware enforcement is used to secure the vFabric and enforce the Members and LimitedMembers rule. Security also ensures that devices in other vFabrics cannot talk to devices in the given vFabric. This security includes, hardware enforced per-packet PKey checking, and enforcement by switches and end nodes.

If Security is 0 (Off), LimitedMembers are treated the same as Members. This allows the user to easily turn off Security for a vFabric without changing the rest of the definition. If Security is not specified under VirtualFabric, vFabric Security will default to 0.

Member and LimitedMember can each be specified more than once per VirtualFabric if required. If a device is in both the Members and LimitedMembers DeviceGroups subsection, it is treated as a Member. This allows All to be specified as a LimitedMember; then selected Members can be specified, ensuring the VirtualFabric includes all devices while allowing a limited set of Members.

Devices in a DeviceGroup but not found in the fabric are ignored.

By default the FM picks an available PKey for the vFabric. When Security is off, the SM may share the same PKey among multiple vFabrics.

If required a user-selected PKey can be specified.

PKey must be specified for applications that do not use SA PathRecord queries, including MPIs that use non-standard mechanisms for job startup.

*Note:* When secure vFabrics are used, every host port must be a member of at least one vFabric for proper operation of host tools such as opainfo. If a host port is not a member of any vFabric these tools will be unable to access the local port.

### 6.5.2 Application Membership

Applications are specified by the application name in the VirtualFabric section.

### 6.5.3 Policies

The QoS Policy, MaxMTU, and MaxRate can be specified for a vFabric. This is one way to restrict the capabilities of the vFabric and influence the performance available to applications and devices within the vFabric.

### 6.5.4 Quality of Service Parameters

When setting up QoS within vFabrics, the user should identify the maximum bandwidth for each vFabric when the link is saturated. High-priority, low-volume groups can be configured with the HighPriority setting. The user can also configure the BaseSL and MulticastSL used to enforce this bandwidth with the BaseSL setting. If not configured, the BaseSL is assigned, and the MulticastSL is assigned to the BaseSL. The amount of configured bandwidth can be specified using the Bandwidth setting and cannot exceed 100 percent. If the configured bandwidth does exceed 100 percent, a parser error is given. If there is a mixture of QoS and non-QoS vFabrics configured, all non-QoS vFabrics are assigned the same BaseSL. Any unconfigured bandwidth is assigned to that SL.

### 6.5.5 The Default Partition

The Intel® Omni-Path Architecture requires every fabric to have a default partition. At a minimum this vFabric is used by all end nodes to interact with the SM/SA.

To meet this requirement there must be an enabled vFabric with the following items:

- A PKey of 0x7fff (or 0xffff)
- The SA application (for example, an Application with `<Select>SA</Select>`)
- The only enabled vFabric that includes the SA application
- Have `All` or `AllMgmtAllowed` (for example, a DeviceGroup with `<Select>AllMgmtAllowed</Select>`) as a `Member`
- Have `All` (for example, a DeviceGroup with `<Select>All</Select>`) as a `Member` or `LimitedMember`
- Additional Applications as needed
- Additional DeviceGroups as `Members` or `LimitedMembers` as needed
- All other vFabric policies (QoS, Security, MTU...) may be set as needed
- The PM application (for example, an Application with `<Select>PM</Select>`.
- If the PM is enabled - The only enabled vFabric that includes the PM application
- An application referencing the PA's ServiceID (for example, an Application with `<ServiceID>0x1100d03c34845555</ServiceID>`)

*Note:* It is a requirement that all chassis-managed spines (for example, SWE0) be a member of the default partition so that the chassis can access and manage its own leaf switch chips.

### 6.5.6 IPoIB and vFabrics

vFabrics are configured within the hardware in the order in which they appear in the configuration file. When IPoIB runs, it uses the first PKey on the given port for the default ( hfi1_0...) network device. Therefore, it is best to place the Networking/IPoIB vFabric first.

IPoIB starts with the PKey for the IPoIB interface, and uses that to define the MGID of the VLAN's broadcast multicast group. Many aspects of the IPoIB VLAN are defined by the multicast group itself. Among them are the MTU for the VLAN.

A given port or node can participate in more than one IPoIB subnet. Each such subnet must have its own unique PKey. For vFabrics other than the first, the PKey should be manually specified in the `VirtualFabric` section and the PKey must be supplied to IPoIB. On some Linux systems with the Intel® Omni-Path Fabric or OFA Delta stack, additional IPoIB virtual interfaces can be created by a command such as:

```
echo 0x1234 > /sys/class/net/ib0/create_child
```

The PKey given is ORed with 0x8000 to define the PKey for the multicast group. This creates an ib0.9234 interface that can be assigned the appropriate IP address and IP parameters.

The operation of Linux with multiple IPoIB subnets is very similar to the use of IP over Ethernet when VLANs are being used. It is up to the administrator which network interfaces are actually used and assigned IP addresses. This is done using the standard `ifcfg` files.

### 6.5.7 MPI and vFabrics

MPIs implementations such as openmpi, mvapich, and mvapich2, do not make PathRecord requests, and do not use ServiceIDs to determine connection parameters.

To use vFabrics in conjunction with those MPIs, the PKey and BaseSL or MulticastSL must be manually specified in the `VirtualFabric` section. The selected PKey and SL also need to be specified to MPI at job startup. Some examples of this are shown in the `/opt/iba/src/mpi_apps/ofed*.params` files that are provided with Intel® FastFabric Toolset.

When using MPI with the Intel® PSM API, path record queries can be enabled in conjunction with the Distributed SA. In which case there is no need to manually specify the PKey and SL at MPI job startup. Refer to the Intel® Omni-Path Fabric OFA Delta Host Software User Guide for more information about enabling Path Record queries in PSM.

### 6.5.8 Pre-Created Multicast Groups

The `Multicast.MulticastGroup` section of the `opafm.xml` configuration file can specify multicast groups that should be pre-created by the SM. If neither a VirtualFabric nor PKey is specified for a given pre-created MulticastGroup, the group will be created for a single vFabric that contains the given MGID as an application and the remaining group properties (Rate, MTU, SL) match the candidate VF.

If MGIDs are specified for the MulticastGroup section, the group must match exactly one VirtualFabric. If no MGID is specified, the MulticastGroup section must match at least one VF, and no matching VFs may share PKeys.

When no MGIDs are explicitly specified, the necessary IPoIB multicast groups for IPv4 and IPv6 are pre-created against the selected VirtualFabric/PKeys (all applicable vFabrics if no specific VirtualFabric/PKey selected). When such automatic pre-creation occurs, the PKey assigned to the vFabric is inserted into the MGIDs per the IPoIB standard.

An example of this capability is provided in the sample configuration file.

### 6.5.9 Securing the Default Partition

When using a secured default partition with vFabrics and redundant FMs, it is recommended to explicitly specify the nodes/ports running the FMs as Members of the default partition's vFabric.

Similarly, if using Intel®FastFabric Toolset in conjunction with a secure default partition, it will be necessary to specify the nodes/ports running Intel® FastFabric Toolset as Members of the default partitions. Failure to do so limits the operations that Intel® FastFabric Toolset can perform and the nodes that Intel® FastFabric Toolset can manage.

### 6.5.10 Multiple vFabrics with Same PKey

When multiple vFabrics are specified with the same PKey, they share a single PKey. When this occurs, the security for the vFabrics is the logical "OR" of the security for the two. If security is off in both, there are no limited members (only full members). If security is on for either (or both), security is imposed for both.

When two vFabrics share the same PKey, the list of members is the combined list from both vFabrics. `Members` is the sum of members in both, and `LimitedMembers` is the sum of limited members in both.

### 6.5.11 Sharing SLs Between Multiple vFabrics

It is possible for the user to specify multiple QoS vFabrics to share SLs. The vFabrics sharing SLs must be configured with the same QoS settings (HighPriority, PreemptRank, FlowControlDisable, HoqLife, PktLifeTimeMult). Multicast isolation can be achieved within a vFabric by specifying a MulticastSL that differs from its BaseSL. If an SL is used for multicast isolation by one vFabric, it may not be specified as the BaseSL of another vFabric. It may, however, be used as another vFabric's MulticastSL. If the user does not specify SLs for a QoS vFabric, it will be assigned a unique BaseSL.

If a unique MulticastSL is specified for a vFabric, the bandwidth assigned to the vFabric will be split evenly between the BaseSL and the MulticastSL. A vFabric may be configured to have 0% bandwidth if it shares SLs with another vFabric whose bandwidth is non-zero.

Non-QoS vFabrics may not specify SLs or bandwidth. They will all share the same BaseSL and one share of the unallocated bandwidth.

### 6.5.12 Parameters

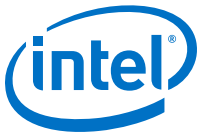
**Table 40. VirtualFabric Parameters**

| Parameter | Description |
|---|---|
| Name | Name for `VirtualFabric`.<br>Every `VirtualFabric` must have a unique name. The name must be unique among all `VirtualFabric` names within an `FM` instance. When defined at the `Common` level must be unique within all instances.<br>The name is limited to 64 characters and is case sensitive. |
| Enable | Enable (1) or Disable (0) a vFabric.<br>When Disabled (0), the VirtualFabric is ignored. This allows the user to easily disable a VirtualFabric without deleting its definition. |
| QoS | When On (1), the Subnet Manager provides QoS for this vFabric and between other vFabrics.<br>When Off (0), the Subnet Manager is free to manage SLs and VLs as it chooses, and there are no guarantees. |
| Standby | Active (0) or Standby (1). When Standby (1), the vFabric is initialized but no traffic can flow. Active (0) and Standby (1) status can be changed without restarting the FM using Dynamic Reconfiguration. |
| HighPriority | If set to 1, this indicates the vFabric is for high-priority traffic that does not require any bandwidth limiting. This would typically include management or control traffic, which is low bandwidth, but critical |

***continued...***

| Parameter | Description |
|---|---|
| | to process in a timely manner. An example is SA traffic where there is no reason to restrict bandwidth since it is low volume, but it needs to be serviced at a high priority.<br>When priority is set to High, any bandwidth allocation is ignored for this vFabric. |
| Bandwidth | This is the minimum percentage (0%-100%) of bandwidth that should be given to this vFabric relative to other low-priority vFabrics. When there is no contention, this vFabric could get more than this amount.<br>If unspecified, the SM evenly distributes the remaining bandwidth among all the vFabrics with unspecified bandwidth.<br>Total Bandwidth cannot exceed 95% for enabled Virtual Fabrics with QoS enabled when there is at least 1 enabled VF with QoS disabled.<br>Total Bandwidth cannot exceed 100% for enabled Virtual Fabrics. VFs with QoS disabled share at least 5% BW and all QoS enabled VFs with unspecified bandwidth require at least 1% BW.<br>If HighPriority is specified, this field is ignored. |
| PktLifeTimeMult | Amount to multiply PktLifeTime by when reported by SM for this vFabric. This can permit extra time in PathRecords (and therefore end-to-end timeouts) to account for delays in low-priority vFabrics that are given low-bandwidth allocations.<br>The value is rounded up to the next power of two.<br>0 is invalid; default is 1. |
| BaseSL | Allows a specific SL (0-15) to be used for the vFabric. SM selects value if unspecified. |
| MulticastSL | Allows a specific SL (0-15) to be used for multicast traffic. SM assigns it to the BaseSL if unspecified. |
| Security | When On (1), the Subnet Manager provides security within this vFabric and between other vFabrics. `LimitedMembers` cannot talk to each other in the vFabric.<br>When Off (0), the Subnet Manager is free to manage routes and PKeys as it chooses, and there are no guarantees. `LimitedMembers` can talk to each other in the vFabric. Default is 0. |
| Member | A `DeviceGroup.Name` that should be in the VirtualFabric.<br>Can be specified more than once per VirtualFabric if required. |
| LimitedMember | A `DeviceGroup.Name` with limited membership in the VirtualFabric. When Security is off, this is functionally the same as a `DeviceGroup` specified using Members.<br>Can be specified more than once per VirtualFabric if required. |
| Application | An `Application.Name` that should be in the VirtualFabric.<br>Can be specified more than once per VirtualFabric if required. |
| PKey | Partition Pkey to use for vFabric.<br>By default the Subnet Manager picks an available PKey. However, if required, a user-selected PKey can be specified.<br>The PKey is a 16-bit value, and the high bit is ignored. The Subnet Manager uses the appropriate high-bit based on the Security and `Member`/`LimitedMember` status per device. |
| MaxMTU | Maximum MTU for SM to return in any PathRecord or Multicast group for the VirtualFabric.<br>Actual values returned may be further reduced by hardware capabilities or if the `PathRecord` or `Multicast` group is requested to have a smaller MTU.<br>However, SM considers it an error to create a `Multicast` group with MTU larger than that of the VirtualFabric. |

*continued...*



| Parameter | Description |
|---|---|
| | The value can also be stated as Unlimited. If not specified, the default MaxMTU is unlimited. |
| MaxRate | Maximum static rate for SM to return in any PathRecord or Multicast group for the VirtualFabric.<br>Actual values returned may be further reduced by hardware capabilities or if the `PathRecord` or `Multicast` group is requested to have a smaller rate.<br>However, SM considers it an error to create a `Multicast` group with rate larger than that of the VirtualFabric.<br>The value can also be stated as Unlimited. If not specified, the default MaxRate is unlimited. |
| PreemptionRank | Preemption capability can be configured per Virtual Fabric in terms of a rank which is a value ranging from 0 to 127. Rank 0 indicates that this VF cannot preempt nor be preempted. Ranks of a higher value can preempt ranks of a lower value (except rank 0). If QOS is disabled then preemption is disabled and the rank is 0. |
| HoqLife | Head of queue time. This is specified at the SM level, but can be overriden at the Virtual Fabric level. When HoqLife is specified at the Virtual Fabric level, the timeout scaling is automatically disabled for that VF. |
| FlowControlDisable | Enable (0) or Disable (1) link level flow control. When link layer flow control has been disabled, packets will be discarded by that VL when there are insufficient credits. Disabling flow control for a vFabric can affect traffic on VLs shared with other Virtual Fabrics |

A working example VF configuration is provided in the default FM configuration file.

# 7.0 Installation and Setup

## 7.1 Installing the Host FM on Linux

*Note:* The Host FM software requires that the same version of the Intel® HFI host stack be installed with at least the HFI drivers. These drivers must be set to **startup**.

The installation provides an interactive INSTALL and is packaged as a tgz. Intel recommends using the `./INSTALL`. The rpms are still within the tgz file and can be installed using standard rpm commands if required. The INSTALL command installs the FM. When using the INSTALL command located in the `IntelOPA-FM.*` or `IntelOPA-IFS.*` directories, the installation process interactively prompts the user to keep or upgrade the FM configuration file.

*Note:* For Intel OPA-IFS installations, refer to the *Intel® Omni-Path Fabric Software Installation Guide*.

*Note:* After installing the FM, the user must reboot the server or use the following startup procedures for the new installation to take effect.

## 7.2 Controlling the FM

The following CLI command controls the FM, with commands to start, stop, restart, and other functions.

opafm /syntax

`opafm` allows the user to control the FM service. By default, the service will spawn all enabled instances of the FM and its components. Instances can be enabled/disabled in `opafm.xml`.

*systemctl [start|stop|restart|reload|status] opafm*

opafm Options

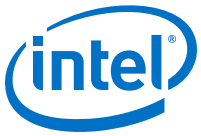
`start` – Starts all configured managers.

`stop` – Stops all configured managers.

`restart` – Restarts all configured managers.

`reload` – Reloads the configuration (this option only handles changes to the <Start> parameters).

`status` – Shows status (running, not running, or disable) for all managers.



`start`, `stop`, `restart`, `reload`, and `status simultaneously` controls all instances of all FM components and managers.

opafmctrl /syntax

`opafmctrl` allows the user to manage the instances of the FM that are running after the `opafm` service has been started.

```
/usr/lib/opa-fm/bin/opafmctrl [start|stop|restart] [-i instance]
[component|compname|insname]...]
```

`-i` *instance* – This option can be specified multiple times and indicates that only specific instances (as configured in the `opafm.xml` configuration file) are to be started or stopped. This value should be an integer value greater than or equal to 0. Without this option, all instances are acted on.

*component*|*compname*|*insname* – This option can be specified more than once and can be any of the following:

- *Component*:
  - `sm` - Subnet manager
  - `fe` - Fabric executive
- *Compname*:

  A specific component/manager name such as `fm0_sm` or `fm1_fe` (manager names are formed by combining an instance name and one of the manager names listed previously, separated by an underscore).
- *Insname*:

  A specific instance name such as `fm0` or `fm1` (instance names are defined in the `Fm.Name` parameter in the configuration file).

When `-i` is used, only the specified components in the instance are started. The `-i` option can be specified more than once to select multiple instances.

When a *compname* is specified, that component is started (regardless of `-i`).

When an *insname* is specified, all components of that instance are started.

If components are specified, then all components in selected instances are started.

If no arguments are specified, all instances are acted on.

opafm Examples

To start the SM for instance 0:

```
/usr/lib/opa-fm/bin/opafmctrl start -i 0 sm
```

To start the SM for instances 0 and 1:

```
/usr/lib/opa-fm/bin/opafmctrl start -i fm0 -i fm1 sm
```

To start the SM for all instances:

```
/usr/lib/opa-fm/bin/opafmctrl start sm
```

To start all managers for instances 0 and 1:

```
/usr/lib/opa-fm/bin/opafmctrl start -i 0 -i 1
```

To start the SM only for instance 1:

```
/usr/lib/opa-fm/bin/opafmctrl start fm1_sm
```

*Note:* `Start`, `restart`, and `sweep` only act on instances and managers enabled in the configuration file using their corresponding `Start` parameters.

## 7.3 Starting the Fabric Manager

1. Log into the Fabric Manager system as `root` or as a user with `root` privileges.
2. Start the Fabric Manager:

   ```
   systemctl start opafm
   ```

3. Verify that all the tasks are up and running:

   ```
   systemctl status opafm
   ```

   *Note:* The default configuration runs Fabric Manager on port 1 of the first HFI in the system.

   To run the Fabric Manager on either port 2, or ports 1 and 2, you must edit the `/etc/opa-fm/opafm.xml` configuration file.

## 7.4 Stopping the Fabric Manager

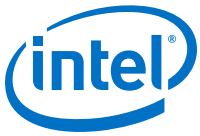
To stop the Fabric Manager, follow these steps:

1. Log into the Fabric Manager system as `root` or as a user with `root` privileges.
2. Stop the Fabric Manager:

   ```
   systemctl stop opafm
   ```

## 7.5 Removing the Fabric Manager

Refer to the *Intel® Omni-Path Fabric Host Software User Guide* for removal procedures for the Fabric Manager.



## 7.6 Automatic Startup

`systemctl [enable|disable] opafm` toggles automatic startup of the `opafm` service and spawns all the enabled instances of the FM on boot. For more information, refer to the *Intel® Omni-Path Fabric Suite FastFabric User Guide* for information on controlling the automatic startup of FM.

# 8.0 Host Fabric Manager Commands

This section describes the Intel® Omni-Path Fabric Suite Fabric Manager Host commands and provides syntax, options, and a sample output for each one.

## 8.1 opafmcmd

Executes a command to a specific instance of the Fabric Manager (FM). This command can be used, for example, to query the configuration attributes of a particular instance of an FM as well as issue subnet management commands.

### Syntax

```
opafmcmd [-i fm_instance] cmd [args]
```

### Options

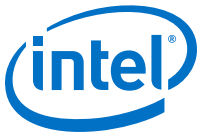
`--help` — Produces full help text.

`-i fm_instance` — Specifies the number of the FM instance to act on.

Range = 0 to 7. Default = 0.

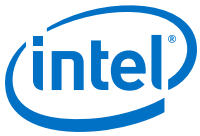
`cmd [args]` — Specifies the commands and arguments (if applicable) to be run. Values include:

| | |
|---|---|
| `smForceSweep` | Makes the Subnet Manager (SM) sweep now. |
| `smRestorePriority` | Restores the normal priority of the SM, if it is currently elevated. |
| `smShowCounters` | Gets statistics and performance counters from the SM. |
| `smResetCounters` | Resets SM statistics and performance counters. |
| `smStateDump` | Dumps internal SM state into specified directory. |
| `smLogLevel` | Sets the SM logging level. Values include:<br>• 0=NONE+ |



| | |
|---|---|
| | • 1=WARN+<br>• 2=INFINI_INFO+<br>• 3=INFO+<br>• 4=VERBOSE+<br>• 5=DEBUG2+<br>• 6=DEBUG4+<br>• 7=TRACE+ |
| `smLogMode` | Sets the SM log mode flags. Values include:<br>• 0/1 1=downgrade non-actionable<br>• 0/2 2=logfile only |
| `smLogMask` | Sets the SM log mask for a specific subsystem to the value given. For a list of subsystems and mask bit meanings, see the files `/etc/opa-fm/opafm.xml` or `/usr/share/opa-fm/opafm.xml`. |
| `smPerfDebug` | Toggles performance debug output for SM. |
| `saPerfDebug` | Toggles performance debug output for Subnet Administration (SA). |
| `saRmppDebug` | Toggles Reliable Message Passing Protocol (RMPP) debug output for SA. |
| `pmShowCounters` | Gets statistics and performance counters for the Performance Manager (PM). |
| `pmResetCounters` | Resets statistics and performance counters for the PM. |
| `pmDebug` | Toggles debug output for PM. |
| `pmRmppDebug` | Toggles RMPP debug output for PM. |
| `feLogLevel` | Sets the Fabric Executive (FE) logging level. Values include: |

| | |
|---|---|
| | • 0=NONE+<br>• 1=WARN+<br>• 2=NOTICE+<br>• 3=INFO+<br>• 4=VERBOSE+<br>• 5=DEBUG2+<br>• 6=DEBUG4+<br>• 7=TRACE+ |
| `feLogMode` | Sets the FE log mode flags.<br>• 0/1 1=downgrade non-actionable<br>• 0/2 2=logfile only |
| `feLogMask` | Sets the FE log mask for a specific subsystem to the value given. For a list of subsystems and mask bit meanings, see the files `/etc/opa-fm/opafm.xml` or `/usr/share/opa-fm/opafm.xml` |
| `feDebug` | Toggles debug output for FE. |
| `feRmppDebug` | Toggles RMPP debug output for FE. |
| `smLooptestStart` | Starts loop test in normal mode. Specify the number of 256 byte packets. Default = 0. |
| `smLooptestFastModeStart` | Starts loop test in fast mode. Specify the number of 256 byte packets. Default = 5. |
| `smLooptestStop` | Stops loop test. Returns switch LFTs back to normal. |
| `smLooptestInjectPackets` | Enter *`numPkts`* to send to all switch loops. Default = 1. |
| `smLooptestInjectAtNode` | Enter the switch node index to inject loop packets. Default = 0. |
| `smLooptestInjectEachSweep` | Sets whether to inject or stop injecting packets each sweep. |



| | |
|---|---|
| | • Enter `1` to inject packets each sweep.<br>• Enter `0` to stop injecting each sweep. |
| `smLooptestPathLength` | Sets the loop path length. Range = 2 - 4. Default = 3. |
| `smLooptestMinISLRedundancy` | Sets the minimum number of loops in which to include each ISL. Default = 4. |
| `smLooptestShowLoopPaths` | Displays the loop paths given node index or all loop paths. Default = all. |
| `smLooptestShowSwitchLft` | Displays a switch LFT given node index or all switches LFTs. Default = all. |
| `smLooptestShowTopology` | Displays the topology for the SM Loop Test. |
| `smLooptestShowConfig` | Displays the current active loop configuration. |
| `smForceRebalance` | Toggles Force Rebalance setting for SM. |
| `smAdaptiveRouting` | Displays or modifies Adaptive Routing setting for SM. If no arguments are entered, displays current setting. Enter `0` to Disable. Enter `1` to Enable. |
| `smForceAttributeRewrite` | Sets rewriting of all attributes upon resweeping.<br>• Enter `0` to Disable.<br>• Enter `1` to Enable. |
| `smSkipAttrWrite` | Specifies the bitmask of attributes to be skipped (not written) during sweeps. Enter `–help` for list of options. |
| `smPauseSweeps` | Pauses SM sweeps. |
| `smResumeSweeps` | Resumes SM sweeps. |

### Example

```
opafmcmd smForceSweep
opafmcmd –i 2 smLogLevel 3
```

## 8.2 opafmcmdall

Executes a command to all instances of the Fabric Manager (FM) listed in the options. The behavior is similar to `opafmcmd`.

### Syntax

```
opafmcmdall [-i fm_instance] cmd [args]
```

### Options

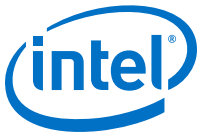
`--help` Produces full help text.

`-i fm_instance` Specifies the number of the FM instance to act on.

Range = 0 to 7. Default = 0.

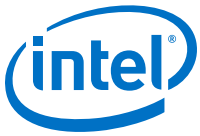
`cmd [args]` Specifies the commands and arguments (if applicable) to be run. Values include:

| | |
|---|---|
| `smForceSweep` | Makes the Subnet Manager (SM) sweep now. |
| `smRestorePriority` | Restores the normal priority of the SM, if it is currently elevated. |
| `smShowCounters` | Gets statistics and performance counters from the SM. |
| `smResetCounters` | Resets SM statistics and performance counters. |
| `smStateDump` | Dumps internal SM state into specified directory. |
| `smLogLevel` | Sets the SM logging level. Values include:<br>• 0=NONE+<br>• 1=WARN+<br>• 2=INFINI_INFO+<br>• 3=INFO+<br>• 4=VERBOSE+ |



| | |
|---|---|
| | • 5=DEBUG2+<br>• 6=DEBUG4+<br>• 7=TRACE+ |
| `smLogMode` | Sets the SM log mode flags. Values include:<br>• 0/1 1=downgrade non-actionable<br>• 0/2 2=logfile only |
| `smLogMask` | Sets the SM log mask for a specific subsystem to the value given. For a list of subsystems and mask bit meanings, see the files `/etc/opa-fm/opafm.xml` or `/usr/share/opa-fm/opafm.xml`. |
| `smPerfDebug` | Toggles performance debug output for SM. |
| `saPerfDebug` | Toggles performance debug output for Subnet Administration (SA). |
| `saRmppDebug` | Toggles Reliable Message Passing Protocol (RMPP) debug output for SA. |
| `pmShowCounters` | Gets statistics and performance counters for the Performance Manager (PM). |
| `pmResetCounters` | Resets statistics and performance counters for the PM. |
| `pmDebug` | Toggles debug output for PM. |
| `pmRmppDebug` | Toggles RMPP debug output for PM. |
| `feLogLevel` | Sets the Fabric Executive (FE) logging level. Values include:<br>• 0=NONE+<br>• 1=WARN+<br>• 2=NOTICE+ |

| | |
|---|---|
| | • 3=INFO+<br>• 4=VERBOSE+<br>• 5=DEBUG2+<br>• 6=DEBUG4+<br>• 7=TRACE+ |
| `feLogMode` | Sets the FE log mode flags.<br>• 0/1 1=downgrade non-actionable<br>• 0/2 2=logfile only |
| `feLogMask` | Sets the FE log mask for a specific subsystem to the value given. For a list of subsystems and mask bit meanings, see the files `/etc/opa-fm/opafm.xml` or `/usr/share/opa-fm/opafm.xml` |
| `feDebug` | Toggles debug output for FE. |
| `feRmppDebug` | Toggles RMPP debug output for FE. |
| `smLooptestStart` | Starts loop test in normal mode. Specify the number of 256 byte packets. Default = 0. |
| `smLooptestFastModeStart` | Starts loop test in fast mode. Specify the number of 256 byte packets. Default = 5. |
| `smLooptestStop` | Stops loop test. Returns switch LFTs back to normal. |
| `smLooptestInjectPackets` | Enter *`numPkts`* to send to all switch loops. Default = 1. |
| `smLooptestInjectAtNode` | Enter the switch node index to inject loop packets. Default = 0. |
| `smLooptestInjectEachSweep` | Sets whether to inject or stop injecting packets each sweep.<br>• Enter `1` to inject packets each sweep.<br>• Enter `0` to stop injecting each sweep. |



| | |
|---|---|
| `smLooptestPathLength` | Sets the loop path length. Range = 2 - 4. Default = 3. |
| `smLooptestMinISLRedundancy` | Sets the minimum number of loops in which to include each ISL. Default = 4. |
| `smLooptestShowLoopPaths` | Displays the loop paths given node index or all loop paths. Default = all. |
| `smLooptestShowSwitchLft` | Displays a switch LFT given node index or all switches LFTs. Default = all. |
| `smLooptestShowTopology` | Displays the topology for the SM Loop Test. |
| `smLooptestShowConfig` | Displays the current active loop configuration. |
| `smForceRebalance` | Toggles Force Rebalance setting for SM. |
| `smAdaptiveRouting` | Displays or modifies Adaptive Routing setting for SM. If no arguments are entered, displays current setting. Enter `0` to Disable. Enter `1` to Enable. |
| `smForceAttributeRewrite` | Sets rewriting of all attributes upon resweeping.<br>• Enter `0` to Disable.<br>• Enter `1` to Enable. |
| `smSkipAttrWrite` | Specifies the bitmask of attributes to be skipped (not written) during sweeps. Enter `–help` for list of options. |
| `smPauseSweeps` | Pauses SM sweeps. |
| `smResumeSweeps` | Resumes SM sweeps. |

### Example

```
opafmcmdall smForceSweep
# sends command to all FMs

opafmcmdall -i 2 -i 4 smLogLevel 3
# sends command to only FM2 and FM4
```

## 8.3 opafmconfigcheck

Parses and verifies the configuration file of a Fabric Manager (FM). Displays debugging and status information.

### Syntax

```
opafmconfigcheck [-s] [-c config_file] [-v] [-d]
```

### Options

| | |
|---|---|
| `--help` | Produces full help text. |
| `-s` | Enables strict check mode; validates multicast and VF settings. This option points out inconsistencies or invalid settings in VF and multicast configurations. |
| `-c config_file` | Specifies configuration file. Default = `/etc/opa-fm/opafm.xml` |
| `-v` | Displays debugging and status information. |
| `-d` | Displays configuration checksum information. |

### Example

```
opafmconfigcheck
opafmconfigcheck -v
opafmconfigcheck -sv
```

## 8.4 opafmconfigdiff

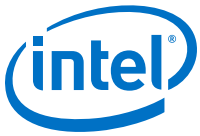
Performs a file difference between two configuration files corresponding to two FM instances described by *file1* and *file2*.

### Syntax

```
opafmconfigdiff [-f] [-l] [-d 'diff_args'] file1 file2
```

### Options

| | |
|---|---|
| `--help` | Produces full help text. |



| | |
|---|---|
| `-f` | Filters out FM parameters that are not part of the consistency check. Removes configuration tags that do not cause consistency checks on the FM to fail from diff. |
| `-l` | Includes comments in XML to indicate original line numbers. |
| `-d 'diff_args'` | Specifies additional arguments to add to `diff` command. For example, enter `uw` for unified format ignoring whitespace. |
| *file1 file2* | Specifies the names of the configuration files to be compared. |

### Example

```
opafmconfigdiff /etc/opa-fm/opafm.xml  /usr/share/opa-fm/opafm.xml
opafmconfigdiff -f /etc/opa-fm/opafm.xml  /usr/share/opa-fm/opafm.xml
opafmconfigdiff -d -uw /etc/opa-fm/opafm.xml  /usr/share/opa-fm/opafm.xml
```

## 8.5 config_convert

Converts an oldconfig file from an older release into a comparable FM XML configuration file using `opafm_src.xml`.

Syntax

```
config_convert [-d] [-D] old_file changed to /usr/share/opa-fm/etc/opafm_src.xml
```

Path

```
/usr/share/opa-fm/opafm_src.xml
```

Options

`-d` – Show tags not found in `old_file`.

`-D` – Extra debug output to `stderr`.

*old_file* – An `fm0_sm.config` file to convert.

`/usr/share/opa-fm/opafm_src.xml` – File that describes mapping of old parameters to XML parameters. This exact filename should be specified.

Notes

The converted file is output to stdout.

When using an embedded FM, this command is also available on hosts with FastFabric installed in the `/usr/lib/opa/fm_tools` directory. In which case the `/usr/lib/opa/fm_tools/opafm_src.xml` file can be used as the mapping file.

Examples

```
config_convert /usr/share/opa-fm/fm0_sm.config.VERSION
/usr/share/opa-fm/etc/opafm_src.xml > my_fm_config.xml
```

## 8.6 config_generate

Interactively generates a FM XML configuration file.

Syntax

```
config_generate [-e] dest_file
```

Path

```
 /usr/lib/opa-fm/bin/
```

Options

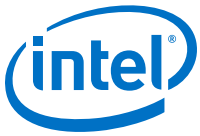
`-e` – Generate file for embedded FM (For example, don't prompt for features not applicable to embedded such as multiple FM instances). Default is to generate a file for host FM.

*dest_file* – Name of file to generate.

Notes

This command presently allows interactive selection of the following FM configuration parameters:

- SubnetSize
- LMC (multi-LID control)
- AdaptiveRouting (Enable, LostRouteOnly)
- LogMode
- NodeAppearanceMsgThreshold
- Which FM instances should be enabled
- Name for each FM Instance
- IPoIB MulticastGroup Rate and MTU for each FM instance
- FM primary or secondary status for failover for each FM instance
- Sticky failover
- SubnetPrefix for each FM instance
- Performance Manager
  - Start
  - Sweep Interval
  - Logging Thresholds
  - Number of PM/PA clients
  - Number of historical images to retain



- Fabric Executive
  - Start
  - SslSecurityEnable

When using an embedded FM, this command is also available on hosts with FastFabric installed in the `/usr/share/opa/fm_tools` directory.

Examples

```
config_generate my_fm_config.xml
```

## 8.7 fm_capture

Provides a capture of FM configuration and present status to aid Intel support in troubleshooting problems.

Syntax

```
fm_capture [fm_instance ...]
```

Path

```
/usr/lib/opa-fm/bin/
```

Options

*fm_instance* – One or more FM instance numbers, by default all running instances will be captured.

Notes

A dated tgz file will be created in the current directory.

This command is automatically included in the information gathered by `opacapture`. This command should only be used if directed by Intel Support.

Examples

```
fm_capture 0
```

## 8.8 smpoolsize

A utility that determines the SM memory requirements of a particular fabric size.

Syntax

```
smpoolsize -n numFIs -s numSwType1 [-p numPortsSwType1] [-S numSwType2] [-P
numPortsSwType2] [-A numActivePortsSwType2] [-l lmc]
```

Path

```
/usr/lib/opa-fm/bin/
```

Options

`-n numFIs` – Number of HFIs.

`-s numSwType1` – Number of Switches of Type 1.

`-p numPortsSwType1` – Number of ports on Switch of Type 1.

`S -S numSwType2` – Number of Switches of Type 2.

`P -P numPortsSwType2` – Number of ports on Switch of Type 2.

`A -A numActivePortsSwType2` – Number of active ports on Switch of Type 2.

`l -l lmc` – LID Mask Control.

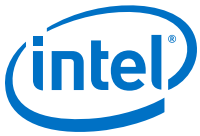
Notes

The size computed is a rough estimate and does not account for the other managers (PM, FE). The size of a fabric with 244 HFIs and thirty 24-port switches is 16,223,042 bytes.

Examples

A 244-node fabric with thirty 24-port switch chips and an LMC of 0 would be input as follows:

```
-> smpoolsize -n 244 -s 30 -p 24 -l 0
```



# 9.0 Embedded Fabric Manager Commands and Configuration

## 9.1 Viewing the Fabric

Information about the fabric can be obtained using the following CLI commands:

- `smPKeys`
- `smShowLids`
- `smShowMcMember`
- `smShowServices`
- `smShowInform`
- `smShowCounters`
- `smShowLidMap`
- `smShowTopology`
- `smShowVFInfo`

Refer to the *Intel® Omni-Path Fabric Suite FastFabric User Guide* for more information on these commands and their options.

### 9.1.1 Determining the Master Fabric Manager

To determine if this FM is the master, use the following CLI command: `smControl`.

Similar information can also be obtained using the following CLI commands:

- `opafabricinfo`
- `opatop`
- `opasaquery`
- `opareport`

## 9.2 Subnet Management Group CLI Commands

The following pages define the group CLI commands and give the syntax, options and a sample output of each. Most of the commands act only against the Management Card you are logged into. To act on the FM running on the slave management card, you must log into that card.

### 9.2.1 Operational Commands

The following commands control the configuration and operation of the FM.

#### 9.2.1.1 smControl

Starts and stops the embedded FM.

**Syntax**

```
smControl [start | stop | restart | status]
```

**Options**

`start` Starts the embedded FM.

`stop` Stops the embedded FM.

`restart` Restarts the embedded FM. (Starts it if it's not already running.)

`status` Prints out the embedded FM status.

**Example**

```
-> smControl start
Starting the SM...
```

#### 9.2.1.2 smAdaptiveRouting

Displays or dynamically sets SM Adaptive Routing when the feature is configured.

**Syntax**

```
smAdaptiveRouting [runningMode]
```

**Options**

*runningMode* 0 = adaptive routing is disabled.

1 = adaptive routing is enabled.

**Example**

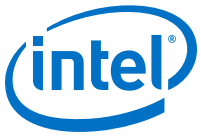
```
-> smAdaptiveRouting
SmAdaptiveRouting is 0 (disabled)
```

**Notes**

The subnet manager must be running to use this command. Changes made with this command affect only the currently running SM in a fabric with multiple SMs running. Changes are lost if the SM is restarted or the chassis is rebooted. To make changes permanent, edit the Fabric Manager XML configuration file.

#### 9.2.1.3 smConfig

Configures startup parameters of the embedded subnet manager.



### Syntax

```
smConfig [query] [startAtBoot yes|no] [startOnSlaveCmu yes|no]
```

### Options

`query` Displays present settings, no change.

`startAtBoot`

`yes` Displays present settings, no change.

`no` Does not start the subnet manager at chassis boot.

`startOnSlaveCmu` Starts the subnet manager at chassis boot.

`yes` Starts the subnet manager on the slave CMU.

`no` Does not start the subnet manager on the slave CMU.

### Examples

Option 1

```
-> smConfig
Start at boot? [Y]
Start on slave CMU? [N]
```

Option 2

```
-> smConfig startAtBoot yes startOnSlaveCmu yes
Saving....
Saving complete...
```

### Notes

Use this command to configure the subnet manager. Changes to these parameters do not take effect until the next reboot of the Chassis Management Cards.

This command is only available on the master chassis management card.

## 9.2.1.4 smPmStart

Controls the start of the performance manager (PM) and Fabric Executive (FE) during subnet manager (SM) start-up.

### Syntax

```
smPmStart [enable | disable | none]
```

#### Options

`enable` Enables the start of the PM and FE at SM start-up.

`disable` Enables the start of the FE and disables the PM at SM start-up.

`none` Disables the start of PM and Fabric Executive (FE) at SM start-up.

#### Example

```
-> smPmStart
SM is enabled
PM is enabled
FE is enabled
-> smPmStart disable
SM is enabled
PM is disabled
FE is enabled
```

#### Notes

The configuration can only be changed from the master Chassis Management Card.

### 9.2.1.5 smShowConfig

Displays the XML configuration file.

#### Syntax

```
smShowConfig [-infoOnly | -contentOnly] [-noprompt]
```

#### Options

`-infoOnly` Displays the timestamp for the XML configuration file.

`-contentOnly` Displays the contents of the XML configuration file.

`-noprompt` Do not prompt to 'Continue' for each page of displayed output.

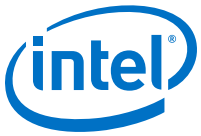
#### Examples

Example 1

```
->smShowConfig -infoOnly
XML config file loaded 09:43:07    04/09/2015
```

Example 2

```
->smShowConfig
XML config file loaded 09:43:07 04/09/2015
<?xml version="1.0" encoding="utf-8"?>
<Config>
<!-- Common FM configuration, applies to all FM instances/subnets -->
<Common>
<!-- Various sets of Applications which may be used in Virtual Fabrics -->
```



```
<!Applications defined here are available for use in all FM instances. -->
<!-- Additional Applications may be defined here or per FM instance. -->
<!-- Applications specified per FM instance will add to -->
<!-- instead of replace those Application definitions. -->
<Applications>
...
...
...
Continue? [Y]
```

#### Notes

With no arguments, the XML configuration file timestamp and contents are displayed, one screen at a time. Enter `Y` or `Enter` at the prompt to continue displaying command output. Enter `N` at the prompt to terminate the output.

The `-infoOnly` and `-contentOnly` flags limit the information that is displayed. Use the `-noprompt` flag to send all output to the screen at once.

This command is only available on the master Chassis Management Card.

### 9.2.1.6 smForceSweep

Forces a fabric sweep by the embedded subnet manager.

#### Syntax

```
smForceSweep
```

#### Options

None.

#### Example

```
-> smForceSweep
```

#### Notes

This command has no output message. To see the resulting sweep information, the "Info" level log messages must be turned on. Refer to smLogLevel on page 166, smLogMode on page 167, and smLogMask on page 167.

### 9.2.1.7 smShowTopology

Displays the current LID assignments for the devices in the fabric.

#### Syntax

```
smShowTopology
```

#### Options

None.

#### 9.2.1.8 smShowVFInfo

Displays Virtual Fabric (VF) information.

*Note:* The subnet manager must be running to use this command.

**Syntax**

```
smShowVFInfo
```

**Options**

None.

#### 9.2.1.9 smRestorePriority

Restores normal priorities from elevated states for the SM and PM.

**Syntax**

```
smRestorePriority [sm|all]
```

**Options**

`sm` Restore normal SM priority.

`all` Restore normal priorities for the SM and PM.

**Example**

```
-> smRestorePriority
```

**Notes**

This command restores the normal priorities of various subnet managers after they have elevated their priority as a result of a failover. Issuing this command allows the "unsticking" of a sticky failover. Issuing this command without arguments restores the normal priorities of the SM. The priority of the PM is based on the priority of the SM.

### 9.2.2 FM Queries

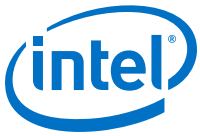
The following commands query the state of the fabric and the fabric manager. The information provided in most of these queries can also be obtained by CLI commands such as `opareport`, `opasaquery`, `opafabricinfo`, or `opashowmc`.

#### 9.2.2.1 smShowLids

Displays all fabric LID information as known by the subnet manager.

**Syntax**

```
smShowLids
```



### Options

None.

### Notes

Use this command to display the current LID assignments for the devices in the fabric. This command requires the given chassis to be the master FM.

Similar information can also be obtained using the CLI commands on the management node:

- `opasaquery`
- `opareport`

## 9.2.2.2 smShowMcMember

Displays multicast member information in the embedded subnet manager.

### Syntax

```
smShowMcMember [-h]
```

### Options

`-h` Display the host name as part of the output.

### Example

```
-> smShowMcMember
Multicast Groups:
  join state key: F=Full N=Non S=SendOnly Member
0xff12601bffff0000:00000001ffffd5bb (c001)
  qKey = 0x00000000  pKey = 0xFFFF  mtu = 4  rate = 3  life = 19  sl = 0
  0x0011750000ffd5bb F
0xff12401bffff0000:00000000ffffffff (c000)
  qKey = 0x00000000  pKey = 0xFFFF  mtu = 4  rate = 3  life = 19  sl = 0
  0x00117501a0007116 F  0x001175020003fffd5 F  0x00117500a00001ac F
  0x00117501a000015d F  0x00117500a00001a3 F  0x00117500a00001dc F
  0x00117500a000035a F  0x0011750000ffd5c2 F  0x0011750000ffd664 F
  0x0011750000ffd9c2 F  0x0011750000ffd9f8 F  0x0011750000ffd5b9 F
  0x0011750000ffda4a F  0x0011750000ffd5bb F  0x0011750000ffd9de F
```

### Notes

Use this command to display multicast member information in the subnet manager. This command is not available unless the subnet manager is in Master mode.

Similar information can also be obtained using the CLI command on the management node:

- `opashowmc`

## 9.2.2.3 smShowServices

Displays subnet administration service records of the subnet manager.

### Syntax

```
smShowServices
```

### Options

None.

### Notes

The components (fields) of each service record are displayed. Each service record is stored in a location identified by a *Slot* number that is displayed before any component of that service record. If a group of slots does not contain service records, the first slot of the empty group is displayed as `empty`.

This command states that the SM is in the STANDBY mode if the SM is not in MASTER mode.

Similar information can also be obtained using the CLI command on the management node:

- `opasaquery -o service`

## 9.2.2.4 smShowInform

Displays event forwarding (inform) table in the embedded subnet manager.

### Syntax

```
smShowInform
```

### Options

None.

### Notes

Use this command to display the event forwarding (inform) table in the subnet manager. This command is not available unless the subnet manager is in the Master mode.

Similar information can also be obtained using the CLI command on the management node:

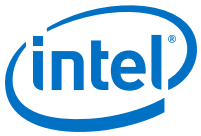
- `opasaquery -o inform`

## 9.2.2.5 smListSecurityFiles

Displays the FM security files stored in the flash.

### Syntax

```
smListSecurityFiles [-showSingleLine]
```



### Options

`showSingleLine` Displays the file names on one line.

### Example

```
-> smListSecurityFiles -showSingleLine
proc list_esm_security_files { } {
##
## list_esm_security_files
## ------------------------
## return the list of FM security files in the given chassis
##
## Usage:
##        list_esm_security_files
## Arguments:
##        none
## Returns:
##        list of .pem files in chassis
##        -code error on failure
## Additional Information:
##        The global timeout is changed by this routine

            global spawn_id expect_out spawn_out timeout
            global expecting

            send_chassis_cmd "smListSecurityFiles -showSingleLine"

            # this could return no *.pem files, in which case "none" is output
            # The \r\n is needed to bound the + so we get all the data on the line
    set out [expect_list 60 "{files: \[0-9A-Za-z._ \]+\[\r\n\]}" { "usage"
"Error" "Failed" "roblem" "not found"} ]
            expect_chassis_prompt 60
            #log_message "out=$out"

            # The \r\n is needed to bound the + so we get all the data on the line
            # there may be trailing spaces, TCL lists will ignore
    regexp {: ([0-9A-Za-z._ ]+)[\r\n]+} $out line ret
            #log_message "ret=$ret"
            if { [ regexp -nocase "none" "$ret" ] } {
                        set ret ""
            }

            return "$ret"
}
```

## 9.2.2.6 smShowLidMap

Displays the LID-to-port GUID map for the subnet manager.

### Syntax

```
smShowLidMap
```

### Options

None.

### Example

```
Edge-> smShowLidMap
-------------------------------------------------------------------------------
SM is currently in the MASTER state, with Topology Pass count = 3
-------------------------------------------------------------------------------
Lid 0x0001: guid = 0x001175010165b157, pass = 3, phkpstl057 hfi1_0
Lid 0x0002: guid = 0x001175010265baf7, pass = 3, OmniPth00117501ff65baf7
Lid 0x0003: guid = 0x001175010165ac3a, pass = 3, phkpstl058 hfi1_0
Lid 0x0004: guid = 0x001175010165ad44, pass = 3, phkpstl059 hfi1_0
Lid 0x0005: guid = 0x001175010165ae43, pass = 3, phkpstl060 hfi1_0
Lid 0x0006: guid = 0x0000000000000000, pass = 0
Lid 0xbfff: guid = 0x0000000000000000, pass = 0
```

### Notes

Use this command to display the LID-to-port GUID map of the subnet manager. The pass count for a LID is incremented each time the SM sweep detects that LID.

If LMC has been used to assign multiple LIDs to a node, those assignments are reflected in the output.

This command is not available unless the subnet manager is in the Master mode.

Similar information can also be obtained using the CLI command on the management node:

- `opasaquery`
- `opareport -o lids`

## 9.2.2.7 smPKeys

Displays partition keys (PKeys) in the PKey table.

*Note:* The subnet manager must be running to display PKeys.

Identical information can also be obtained using the CLI command on the management node:

```
opasaquery -o pkey
```

### Syntax

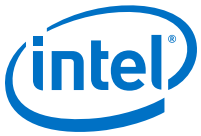
```
smPKeys
```

### Options

None.

### Example

```
-> smPKeys
LID: 0x00000002 PortNum:  0 BlockNum:  0
      0-   7:  0x8001  0x7fff  0xffff  0x0000  0x0000  0x0000  0x0000  0x0000
      8-  15:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     16-  23:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     24-  31:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
```



```
-------------------------------------------------------------------------------
LID: 0x00000002 PortNum:  9 BlockNum:  0
      0-   7:  0x8001  0x0000  0xffff  0x0000  0x0000  0x0000  0x0000  0x0000
      8-  15:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     16-  23:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     24-  31:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
-------------------------------------------------------------------------------
LID: 0x00000002 PortNum: 12 BlockNum:  0
      0-   7:  0x8001  0x0000  0xffff  0x0000  0x0000  0x0000  0x0000  0x0000
      8-  15:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     16-  23:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     24-  31:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
-------------------------------------------------------------------------------
LID: 0x00000002 PortNum: 41 BlockNum:  0
      0-   7:  0x8001  0x0000  0xffff  0x0000  0x0000  0x0000  0x0000  0x0000
      8-  15:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     16-  23:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     24-  31:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
-------------------------------------------------------------------------------
LID: 0x00000002 PortNum: 44 BlockNum:  0
      0-   7:  0x8001  0x0000  0xffff  0x0000  0x0000  0x0000  0x0000  0x0000
      8-  15:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     16-  23:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     24-  31:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
-------------------------------------------------------------------------------
LID: 0x00000003 PortNum:  1 BlockNum:  0
      0-   7:  0x8001  0x7fff  0xffff  0x0000  0x0000  0x0000  0x0000  0x0000
      8-  15:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     16-  23:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     24-  31:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
-------------------------------------------------------------------------------
LID: 0x00000004 PortNum:  1 BlockNum:  0
      0-   7:  0x8001  0x7fff  0xffff  0x0000  0x0000  0x0000  0x0000  0x0000
      8-  15:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     16-  23:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     24-  31:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
-------------------------------------------------------------------------------
LID: 0x00000005 PortNum:  1 BlockNum:  0
      0-   7:  0x8001  0x7fff  0xffff  0x0000  0x0000  0x0000  0x0000  0x0000
      8-  15:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     16-  23:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     24-  31:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
-------------------------------------------------------------------------------
LID: 0x00000001 PortNum:  1 BlockNum:  0
      0-   7:  0x8001  0x7fff  0xffff  0x0000  0x0000  0x0000  0x0000  0x0000
      8-  15:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     16-  23:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
     24-  31:  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000  0x0000
```

### 9.2.2.8 smShowCounters

Displays various statistics and counters maintained by the subnet manager.

#### Syntax

```
smShowCounters
```

#### Options

None.

### Example

```
-> smShowCounters
COUNTER:                                THIS SWEEP LAST SWEEP      TOTAL
------------------------------------- ---------- ---------- ----------
   SM State transition to DISCOVERY:           0          0          2
      SM State transition to MASTER:           0          0          1
     SM State transition to STANDBY:           0          0          1
    SM State transition to INACTIVE:           0          0          0
      Total transmitted SMA Packets:         123        711       2181
          Direct Routed SMA Packets:         123        711       2122
             LID Routed SMA Packets:           0          0         40
              SMA Query Retransmits:           0          0         18
    SMA Query Retransmits Exhausted:           0          0          3
                  SM TX GET(Notice):           0          0          0
                  SM TX SET(Notice):           0          0          0
                 SM RX TRAP(Notice):           0          0          0
          SM TX TRAPREPRESS(Notice):           0          0          0
         SM TX GET(NodeDescription):           0        148        444
                SM TX GET(NodeInfo):           0        148        444
              SM TX GET(SwitchInfo):           0          6         18
...
```

### Notes

This command is not available unless the subnet manager is in Master mode.

## 9.2.2.9 smResetCounters

Resets various statistics and counters maintained by the subnet manager.

### Syntax

```
smResetCounters
```

### Options

None.

### Example

```
-> smResetCounters
```

## 9.2.2.10 pmShowCounters

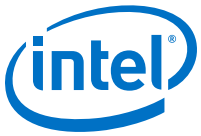
Displays various statistics and counters maintained by the performance manager (PM).

### Syntax

```
pmShowCounters
```

### Options

None.



### Example

```
-> pmShowCounters
                                COUNTER: THIS SWEEP LAST SWEEP      TOTAL
---------------------------------------- ---------- ---------- ----------
                            PM Sweeps:          0          1      32039
          Ports whose PMA failed query:          0          0        206
    Nodes with 1 or more failed Ports:           0          0        197
        Total transmitted PMA Packets:           0        228    7307203
                 PMA Query Retransmits:          0          0       7418
      PMA Query Retransmits Exhausted:           0          0        206
             PM TX GET(ClassPortInfo):           0          0         98
        PM TX GET(PortSamplesControl):           0          0          0
         PM TX GET(PortSamplesResult):           0          0          0
              PM TX GET(PortCounters):           0        172    5507335
              PM TX SET(PortCounters):           0         35    1119563
      PM TX GET(PortCountersExtended):           0         11     352409
        PM TX GET(VendorPortCounters):           0          0          0
        PM TX SET(VendorPortCounters):           0         10     320380
                     PM RX GETRESP(*):           0        228    7299579
                   PM RX STATUS BUSY:            0          0          0
                PM RX STATUS REDIRECT:           0          0          0
                PM RX STATUS BADCLASS:           0          0          0
              PM RX STATUS BADMETHOD:            0          0          0
           PM RX STATUS BADMETHODATTR:           0          0          0
                PM RX STATUS BADFIELD:           0          0          0
                 PM RX STATUS UNKNOWN:           0          0          0
             PA RX GET(ClassPortInfo):           0          0          0
                   PA RX GET(GrpList):           0          0         13
                   PA RX GET(GrpInfo):           0          0         82
 .... ....
 .... ....
 .... ....
```

## 9.2.2.11 pmResetCounters

Resets various statistics and counters maintained by the performance manager (PM).

### Syntax

```
pmResetCounters
```

### Options

None.

### Example

```
-> pmResetCounters
```

## 9.2.2.12 pmShowRunningTotals

Displays the running total counters for all ports in the fabric maintained by the performance manager (PM).

### Syntax

```
pmShowRunningTotals
```

#### Options

None.

### 9.2.3 FM Configuration Queries

The following commands query the individual configuration attributes of the FM. Some of these commands can also make non-persistent changes to the FM. Any such non-persistent changes will be lost the next time the FM or chassis is restarted. To make persistent changes to the FM configuration the desired `opafm.xml` file must be downloaded to the chassis.

#### 9.2.3.1 smLogLevel

Displays or dynamically sets the subnet manager logging level.

#### Syntax

```
smLogLevel [loglevel]
```

#### Options

*loglevel* Logging level. Options include:

0 NONE+

1 WARN+

2 INFINI_INFO+

3 INFO+

4 VERBOSE+

5 DEBUG2+

6 DEBUG4+

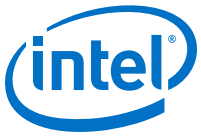
7 TRACE+

#### Example

```
-> smLogLevel
Log Level:2
```

#### Notes

The subnet manager must be running to use this command. Changes made with this command affect only the currently running SM in a fabric with multiple SMs running. Changes are lost if the SM is restarted or the chassis is rebooted. To make changes permanent, edit the Fabric Manager XML configuration file.



### 9.2.3.2 smLogMode

Displays or dynamically sets the subnet manager logging mode.

#### Syntax

```
smLogMode [logmode]
```

#### Options

*logmode* Logging mode. Options include:

0 Use normal logging levels.

1 Logging is quieted by downgrading the majority of fatal, error, warn, and info log messages.

3 (INFO) and only outputting user actionable events when LogLevel is 1 or 2.

#### Example

```
-> smLogMode
Log Mode:0
```

#### Notes

The subnet manager must be running to use this command. Changes made with this command affect only the currently running SM in a fabric with multiple SMs running. Changes are lost if the SM is restarted or the chassis is rebooted. To make changes permanent, edit the Fabric Manager XML configuration file.

### 9.2.3.3 smLogMask

Displays or dynamically sets the subnet manager logging mask for a specific subsystem.

#### Syntax

```
smLogMask subsystem [mask]
```

#### Options

*subsystem* Subsystem. Options include: CS, MAI, CAL, DVR, IF3, SM, SA, PM, PA, FE, APP

*mask* Bit mask for logging to enable.

For the bit mask layout, see Table 5 on page 83.

### Example

```
-> smLogMask SA
SA Log Mask: 0x1ff
```

### Notes

The subnet manager must be running to use this command. Changes made with this command affect only the currently running SM in a fabric with multiple SMs running. Changes are lost if the SM is restarted or the chassis is rebooted. To make changes permanent, edit the Fabric Manager XML configuration file.

## 9.2.4 FM Loop Test

The following commands set up and perform loop test in the fabric. The Loop Test should be performed after an installation or major changes have occurred in the fabric or cluster to help validate that data is being transported as intended.

### 9.2.4.1 smLooptestStart

Starts the SM Loop Test in normal mode with the specified number of 256 byte packets. If the SM has not been previously started, this command starts the SM.

*Note:* The Loop Test only operates if the SM is in the Master state.

In the default mode, the SM uses an exhaustive approach to set up loop routes and includes each ISL in as many loops as possible. This ensures that each ISL is in the same number of loops and therefore sees the same amount of utilization. However, finding all possible loops is computationally intensive and can take a long time. In most cases, Intel recommends you use fast mode.

### Syntax

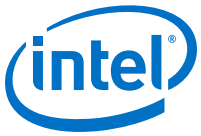
```
smLooptestStart [packets]
```

### Options

*packets* The number of 256 byte packets used when starting the SM Loop Test. Valid values = 0 - 10. Default = 0. If the number of packets is 0, then no packets are injected.

### Example

```
-> smLooptestStart
Waiting for SM to complete startup....N|2015/09/15 14:24:17.180U: Thread
"esm_top"
(0xccace3f0)
        MSG:NOTICE|SM:OmniPath GUID=0x00117501e30027xx:port 0|COND:#5 SM state
to master|NODE:OmniPath GUID=0x00117501e3002711:port
0:0x00117501e3002711|DETAIL:transition from DISCOVERING to MASTER
topology_loopTest: DONE
W|2015/09/15 14:24:17.280U: Thread "esm_top" (0xccace3f0)
        MSG:WARNING|SM:OmniPath GUID=0x00117501e30027xx:port 0|COND:#1
Redundancy lost|NODE:OmniPath GUID=0x00117501e3002711:port
0:0x00117501e3002711|DETAIL:SM redundancy not available
topology_loopTest: DONE
```



```
..................done
The SM Loop Test is being started
Loop Test is setup, but no packets have been injected and no traffic is running
```

### 9.2.4.2 smLooptestFastModeStart

Starts the SM Loop Test in fast mode with the specified number of 256 byte packets. If the SM has not been previously started, this command starts the SM.

*Note:* The Loop Test only operates if the SM is in the Master state.

Intel recommends you use the fast mode for ISL validation and link integrity testing. In fast mode, the loop test does not attempt to include each ISL in all possible loops, but includes it in at least the specified number of loops (using the `MinISLRedundancy` parameter). In typical fast mode operations, using the default `MinISLRedundancy` value = 4, injecting five packets into each loop is sufficient to get a high utilization on the ISLs.

#### Syntax

```
smLooptestFastModeStart [packets]
```

#### Options

*packets* The number of 256 byte packets used when starting the SM Loop Test in Fast Mode. Valid values = 0 - 10. Default = 5. If the number of packets is 0, then no packets are injected.

#### Example

```
-> smLooptestFastModeStart
Waiting for SM to complete startup...Local LID changed to: 0
.N|2015/09/15 14:19:28.280U: Thread "esm_top" (0xcca0d828)
        MSG:NOTICE|SM:OmniPath GUID=0x00117500e30027xx:port 0|COND:#5 SM state
to master|NODE:OmniPath GUID=0x00117500e30027xx:port
0:0x00117500e3002711|DETAIL:transition from DISCOVERING to MASTER
Local LID changed to: 1
Local LID changed to: 1
topology_loopTest: DONE
W|2015/09/15 14:19:28.390U: Thread "esm_top" (0xcca0d828)
        MSG:WARNING|SM:OmniPath GUID=0x00117500e30027xx:port 0|COND:#1
Redundancy lost|NODE:OmniPath GUID=0x00117500e3002711:port
0:0x00117500e3002711|DETAIL:SM redundancy not available
topology_loopTest: DONE
..................done
The SM Loop Test is being started in Fast Mode
```

### 9.2.4.3 smLooptestStop

Stops the SM Loop Test.

#### Syntax

```
smLooptestStop
```

### Options

None.

### Example

```
-> smLooptestStop
Waiting for SM to complete shutdown...
A|2015/09/15 14:21:46.500U: Thread "esm_Start" (0x85738dd8)
        ESM: SM Control: Initiating shutdown of the subnet manager. Some errors
and
warnings are common during this process 0
N|2015/09/15 14:21:46.500U: Thread "esm_Start" (0x85738dd8)
        MSG:NOTICE|SM:OmniPath GUID=0x00117500e3002711:port 0|COND:#7 SM
shutdown|NODE:OmniPath GUID=0x00117500e3002711:port 0:0x00117500e3002711
........N|2015/09/15 14:21:54.720U: Thread "INVALID" (0xcca13ac8)
        MSG:NOTICE|SM:OmniPath GUID=0x00117500e3002711:port 0|COND:#13 SM state
to inactive|NODE:OmniPath GUID=0x00117500e3002711:port
0:0x00117500e3002711|DETAIL:transition from MASTER to NOTACTIVE
...A|2015/09/15 14:21:57.720U: Thread "esm_Start" (0x85738dd8)
        ESM: SM Control: Subnet manager shutdown complete. 0
...................done
The SM Loop Test is being stopped
```

### Notes

Use this command to stop the SM Loop Test. Returns switch LFTs back to normal.

*Note:* This command will stop SM if it was started by either the `smLooptestStart` command or the `smLooptestFastModeStart` command. If SM was started using the `smcontrol start` command, this command will not stop SM.

## 9.2.4.4 smLooptestInjectPackets

Injects packets into the SM Loop Test.

### Syntax

```
smLooptestInjectPackets [packets]
```

### Options

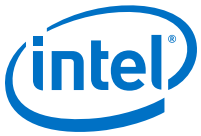
*packets* The number of packets to inject into the SM Loop Test. Valid values are 1 - 10 (default = 1).

### Example

```
-> smLooptestInjectPackets 2
Sending 2 packets to all loops
Packets have been injected into the SM Loop Test
-> topology_loopTest: DONE
```

## 9.2.4.5 smLooptestInjectAtNode

Injects packets to a specific switch node for the SM Loop Test.



### Syntax

```
smLooptestInjectAtNode [node index]
```

### Options

*node index* The node index of the switch to inject packets.

### Example

```
-> smLooptestInjectAtNode 3
Sending 2 packets to node index 3
Packets have been injected into the SM Loop Test for node 3
-> topology_loopTest: DONE
```

## 9.2.4.6 smLooptestInjectEachSweep

Enables/disables packet injected on each sweep for the SM Loop Test.

### Syntax

```
smLooptestInjectEachSweep setting
```

### Options

*setting* Options include:

1 Inject packets on each sweep.

0 Do not inject packets on each sweep for the SM Loop Test.

### Example

```
-> smLooptestInjectEachSweep 1
sm_looptest_inject_packets_each_sweep: loop test will inject packets every sweep,
numPackets=2
The SM Loop Test will inject packets every sweep
```

## 9.2.4.7 smLooptestPathLength

Sets the loop path length for the SM Loop Test.

### Syntax

```
smLooptestPathLength [length]
```

### Options

*length* The loop path length for the SM Loop Test. Valid values are 2, 3 (default), and 4.

### Example

```
-> smLooptestPathLength 3
The SM Loop Test path length has been set to 3
-> topology_loopTest: DONE
```

## 9.2.4.8 smLooptestMinISLRedundancy

Sets the minimum number of loops in which to include each ISL for the SM Loop Test in Fast Mode.

### Syntax

```
smLooptestMinISLRedundancy [loops]
```

### Options

*loops* The minimum number of loops to include in each ISL for the SM Loop Test. If no value is entered, the default (default = 4) is used.

*Note:* This command is only applicable if running the Loop Test in Fast Mode.

### Example

```
->  smLooptestMinISLRedundancy 3
-> topology_loopTest: DONE
```

## 9.2.4.9 smLooptestShowLoopPaths

Displays the loop paths for the SM Loop Test.

### Syntax

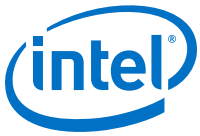
```
smLooptestShowLoopPaths [node index]
```

### Options

*node index* The node index of the node to print the loop paths. If no value is entered, the default (all nodes) is used.

### Example

```
-> smLooptestShowLoopPaths
Node Idx: 0, Guid: 0x00117500e3002711 Desc OmniPath GUID=0x00117500e3002711
------------------------------------------------------------------------------
 Node  Node                        Node    Path
 Idx   Lid        NODE GUID       #Ports    LID    PATH[n:p->n:p]
 ---- ------  ------------------  ------  ------  ---------------
    0 0x0001  0x00117500e3002711    36    0x0040  0:9->0:33
    0 0x0001  0x00117500e3002711    36    0x0042  0:33->0:9
------------------------------------------------------------------------------
```



```
There are 2 total loop paths of <=4 links in length in the fabric!
Two LIDs are used per loop path to inject packets in clockwise and anti-clockwise
directions
```

### 9.2.4.10 smLooptestShowSwitchLft

Displays the switch LID Forwarding Table (LFT) for the SM Loop Test.

#### Syntax

```
smLooptestShowSwitchLft [node index]
```

#### Options

*node index* The node index of the switch for which to print the switch LFT. If no value is entered, the default (all switches) is used.

#### Example

```
-> smLooptestShowSwitchLft
Node[0000]  LID=0x0001  GUID=0x00117500e3002711
[OmniPath GUID=0x00117500e3002711] Linear Forwarding Table
   LID    PORT
 ------   ----
 0x0001   0000
 0x0005   0031
 0x0009   0017
 0x0010   0011
 0x0016   0021
 0x001d   0022
 0x0021   0025
 0x0040   0009
 0x0041   0033
 0x0042   0033
 0x0043   0009
```

### 9.2.4.11 smLooptestShowTopology

Displays the topology for the SM Loop Test.

#### Syntax

```
smLooptestShowTopology
```

#### Options

None.

#### Example

```
-> smLooptestShowTopology
sm_state = MASTER   count = 481   LMC = 0, Topology Pass count = 4, Priority = 0,
Mkey = 0x0000000000000000
--------------------------------------------------------------------------------
OmniPath GUID=0x00117500e3002711
--------------------------------------------------------------------------------
Node[  0] => 00117500e3002711  (2)  ports=36, path=
        Port ----  GUID  ----  (S)   LID        LMC     _VL_   __MTU__  __WIDTH__
___SPEED____  CAP_MASK   N#  P#
           0 00117500e3002711   4  LID=0001  LMC=0000   1  1    4k  4k  1X-8X  4X
2.5-10   5.0  0200004a    0   0
           9 0000000000000000   4                       1  1    2k  2k     4X  4X
```

```
2.5-10  10.0  00000000    0  33
          11 0000000000000000   4                      1  1    2k  2k     4X  4X
2.5-10  10.0  00000000    1   1
          17 0000000000000000   4                      1  1    2k  2k     4X  4X
2.5-10  10.0  00000000    2   1
          21 0000000000000000   4                      1  1    2k  2k     4X  4X
2.5-10  10.0  00000000    3   1
          22 0000000000000000   4                      1  1    2k  2k     4X  4X
2.5-10  10.0  00000000    4   1
          25 0000000000000000   4                      1  1    2k  2k     4X  4X
2.5-10  10.0  00000000    5   1
          31 0000000000000000   4                      1  1    2k  2k     4X  4X
2.5-10  10.0  00000000    6   1
          33 0000000000000000   4                      1  1    2k  2k     4X  4X
2.5-10  10.0  00000000    0   9  9
-------------------------------------------------------------------------------
st164 HFI-1
-------------------------------------------------------------------------------
Node[  1] => 001175000100d050  (1)  ports=2, path=11
        Port ----  GUID  ----  (S)   LID        LMC     _VL_   __MTU__  __WIDTH__
___SPEED____  CAP_MASK   N#  P#
          1 001175000100d051   4  LID=0010  LMC=0000   8  1    2k  2k  1X/4X  4X
2.5-10  10.0  02510868    0  11 11
-------------------------------------------------------------------------------
st10 HFI-1
-------------------------------------------------------------------------------
Node[  2] => 00117500007eaa56  (1)  ports=2, path=17
        Port ----  GUID  ----  (S)   LID        LMC     _VL_   __MTU__  __WIDTH__
___SPEED____  CAP_MASK   N#  P#
          1 00117500007eaa56   4  LID=0009  LMC=0000   2  1    4k  2k  1X/4X  4X
2.5-10  10.0  07610868    0  17 17
-------------------------------------------------------------------------------
st166 HFI-1
-------------------------------------------------------------------------------
Node[  3] => 00117500007ec376  (1)  ports=1, path=21
        Port ----  GUID  ----  (S)   LID        LMC     _VL_   __MTU__  __WIDTH__
___SPEED____  CAP_MASK   N#  P#
          1 00117500007ec376   4  LID=0016  LMC=0000   2  1    4k  2k  1X/4X  4X
2.5-10  10.0  07610868    0  21 21
-------------------------------------------------------------------------------
compute000 HFI-1
-------------------------------------------------------------------------------
Node[  4] => 0011750300032de8  (1)  ports=2, path=22
        Port ----  GUID  ----  (S)   LID        LMC     _VL_   __MTU__  __WIDTH__
___SPEED____  CAP_MASK   N#  P#
          1 0011750300032de9   4  LID=001d  LMC=0000   8  1    2k  2k  1X/4X  4X
2.5-10  10.0  02510868    0  22 22
-------------------------------------------------------------------------------
compute001 HFI-1
-------------------------------------------------------------------------------
Node[  5] => 0011750300033694  (1)  ports=2, path=25
        Port ----  GUID  ----  (S)   LID        LMC     _VL_   __MTU__  __WIDTH__
___SPEED____  CAP_MASK   N#  P#
          1 0011750300033695   4  LID=0021  LMC=0000   8  1    2k  2k  1X/4X  4X
2.5-10  10.0  02510868    0  25 25
-------------------------------------------------------------------------------
st9 HFI-1
-------------------------------------------------------------------------------
Node[  6] => 00117500007eaa1c  (1)  ports=1, path=31
        Port ----  GUID  ----  (S)   LID        LMC     _VL_   __MTU__  __WIDTH__
___SPEED____  CAP_MASK   N#  P#
          1 00117500007eaa1c   4  LID=0005  LMC=0000   2  1    4k  2k  1X/4X  4X
2.5-10  10.0  07610868    0  31 31
```

### 9.2.4.12 smLooptestShowConfig

Displays the configuration for the SM Loop Test.

#### Syntax

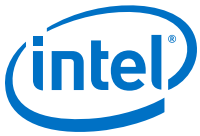
```
smLooptestShowConfig
```

#### Options

None.

#### Example

```
-> smLooptestShowConfig
Loop Test is running with following parameters:
 Max Path Length    #Packets    Inject Point
 ---------------    --------    ------------
         4            00004         All Nodes
FastMode=1, FastMode MinISLRedundancy=4, InjectEachSweep=0, TotalPktsInjected
since start=4
```



# Appendix A Bubble Definition

Bubbles occur when an egress port outputs idle flits in the middle of a packet.

In Intel® OPA bubbles can occur for a couple reasons:

1. The ingress port to the switch that was supplying the packet experienced a link replay (PIP) in the middle of the packet. At that time a bubble may occur in the middle of the packet while replay occurs. Such bubbles are typically on the order of a few hundred nanoseconds.
2. The ingress port to the switch had a packet preemption/traffic flow optimization. In this case the ingress port starts receiving packet A and the switch egress port X starts outputting A. A is then preempted at the ingress link with packet B, which is destined to port Y. For the duration of packet B, port X has nothing to output and hence outputs idles. After packet B completes, packet A resumes and port X completes sending packet A. A telephone analogy would be that at the ingress port we put packet A on hold to service packet B. The egress port X is idled (much like the person you put on hold) until packet A is resumed.

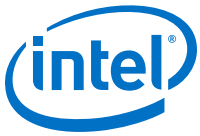
Under high load the fabric tends to self compress bubbles. Under high load the packets end up briefly queued in the switch. When packets are queued in the switch, the ingress port is merely feeding a packet buffer, but the egress packet is not actually sending that packet yet (the egress port is busy sending another packet previously queued). In this scenario, bubbles due to preemption or link replay, which interrupt packets from the ingress port tend to self compress out and not cause bubbles on the egress port.

# Appendix B Command Line Interface (CLI) Taxonomy

The Intel® Omni-Path Software CLI provides a robust multi-tiered command set. The CLIs are designed with common options and style. The tiered approach splits commands into the following categories.

*Note:* The set of `opaextract*` commands are scripts and can be copied to create new customized tools.

- Monitoring and Diagnostics
  - High Level Analysis and Monitoring
  - Mid-Tier CLI Analysis and Diagnostics
  - Lower Level CLI Diagnostics
  - Deep Level CLI Diagnostics and Data
- Testing
  - Verbs Benchmarking and Testing
  - MPI Benchmarking and Testing
  - Fabric Stress Testing
- Management
  - Host Management
  - Switch Management
  - Installation
  - Link and Port
  - Fabric Manager
  - Topology
  - QoS
  - SRP IOUs
  - Routing
  - Distributive SA
  - Data Gathering
  - Fabric Verification
- Others
  - Wireshark
  - Additional Commands
  - Miscellaneous



## Table 41. CLI Taxonomy

| Category | CLIs |
|---|---|
| **Monitoring and Diagnostics** | |
| High Level Analysis/Monitoring | Intel® Omni-Path Fabric Suite FastFabric TUI<br>Intel® Omni-Path Fabric Suite Fabric Manager GUI<br>Intel® Omni-Path Fabric Suite FastFabric CLI: opatop, opalinkanalysis, opallanalysis (opachassisanalysis, opaesmanalysis, opahostsmanalysis, opafabricanalysis) |
| Mid-Tier CLI Analysis/Diagnosis | Intel® Omni-Path Fabric Suite FastFabric CLI: opareports, opareport, opaextractbadlinks, opaextractlink, opaextractsellinks, opaextractstat2]\<br>Intel® Omni-Path Fabric Host Software Basic CLI: opafabricinfo |
| Lower Level CLI Diagnosis | Intel® Omni-Path Fabric Suite FastFabric CLI: opaextractlids, opaextracterror, opaextractperf, opaextractstat, opashowallports |
| Deep Level CLI Diagnosis/Data | Intel® Omni-Path Fabric Suite FastFabric CLI: opafequery, opafirmware, opaportinfo, opashowmc, opapaquery<br>Intel® Omni-Path Fabric Host Software Basic CLI: opahfirev, opapmaquery, opasmaquery, opasaquery |
| **Testing** | |
| Verbs Benchmarks/Tests | OFA Delta utilities bundled in the `IntelOPA-IFS` package for convenience: ibping, ib_send_bw, ib_atomic_bw, ib_atomic_lat, ib_read_bw, ib_read_lat, ib_send_lat, ib_write_bw, ib_write_lat, ibv_rc_pingpong, ibv_srq_pingpong, ibv_uc_pingpong, ibv_ud_pingpong |
| MPI Benchmarks/Tests | /usr/src/opa/mpi_apps |
| Stress Tests | Intel® Omni-Path Fabric Suite FastFabric CLI: opacabletest, opacheckload |
| **Management** | |
| Multi-Switch External Management | Intel® Omni-Path Fabric Suite FastFabric CLI: opaswitchadmin, opagenswitches |
| Multi-Switch Internal Management | Intel® Omni-Path Fabric Suite FastFabric CLI: opachassisadmin, opacmdall, opapingall, opagenchassis, opagenesmchassis, opasetupssh |
| Multi-Host Management | Intel® Omni-Path Fabric Suite FastFabric CLI: opacmdall, opapingall, opadownloadall, opauploadall, opafindgood, opahostadmin, opascpall, opasetupssh, opaverifyhosts |
| Installation | Intel® Omni-Path Fabric Host Software Basic CLI: opaconfig, opa_config_ff, opa_config_fm |
| Link and Port Management | Intel® Omni-Path Fabric Suite FastFabric CLI: opaenableports, opadisableports, (opaextractbadlinks, opaextractsellinks generate input format for opaenable/disableports), opadisablehosts, opaswdisableall<br>Intel® Omni-Path Fabric Host Software Basic CLI: opainfo, opaportinfo, opapmaquery, opasmaquery, opaportconfig |
| Fabric Manager (FM) Management | Intel® Omni-Path Fabric Suite FastFabric CLI: opafmcmd, opafmcmdall, opafmconfigcheck, opafmconfigdiff<br>service opafm |
| Topology Analysis and Management | Intel® Omni-Path Fabric Suite FastFabric CLI: opareport -o verify*, opagentopology, opareport -o links, opaextractlink, opaextractsellinks |
| SRP IOUs | Intel® Omni-Path Fabric Suite FastFabric CLI: opareport -o ious |
| Link Issue Analysis | Intel® Omni-Path Fabric Suite FastFabric TUI<br>Intel® Omni-Path Fabric Suite Fabric Manager GUI |

***continued...***

| Category | CLIs |
|---|---|
| | Intel® Omni-Path Fabric Suite FastFabric CLI: opatop, opaallanalysis, opaextractbadlinks, opareport (-o errors, slow* mis*), opashowallports |
| QoS Analysis | Intel® Omni-Path Fabric Suite FastFabric CLI: opareport (-o vfinfo, vfmember, bfrctrl), opareport -V -o comps -d 10 (dumps all QoS config)<br>Intel® Omni-Path Fabric Host Software Basic CLI: opasmaquery, opasaquery -o vfinfo, opasaquery -o path, other opasaqueries for sc, sl, vl tables |
| Routing Analysis | Intel® Omni-Path Fabric Suite FastFabric CLI: opareport (-o portusage, treepathusage, pathusage, portgroups, validateroutes, validatepgs, validatecreditloops, linear, mcast) |
| Customer Support Data Gathering Host and Switch | Intel® Omni-Path Fabric Suite FastFabric CLI: opacaptureall, opacapture |
| Topology Generation and Conversions | Intel® Omni-Path Fabric Suite FastFabric CLI: opagentopology, opareport -o topology, opaxlattopology, opaxlattopology_cust (deprecated), opaxmlgenerate |
| General XML File Utilities | Intel® Omni-Path Fabric Suite FastFabric CLI: opaxmlextract, opaxmlfilter, opaxmlindent |
| Verify Fabric (VF) query for launch integration | Intel® Omni-Path Fabric Host Software Basic CLI: opagetvf, opagetvf_env |
| Distributive Subnet Administrator (SA) | Intel® Omni-Path Fabric Host Software Basic CLI: opa_osd_dump, opa_osd_exercise, opa_osd_perf, opa_osd_query |
| **Other** | |
| Wireshark | Intel® Omni-Path Fabric Host Software Basic CLI: opapacketcapture |
| Partial Support | OFA Delta utility bundled in the `IntelOPA-IFS` package for convenience: ibv_devinfo (MTU shows <= 4K even for 8K, fw_ver N/A, does not support extended physical states for down ports) |
| Unsupported | ibdiagnet, ibdiagpath, ibdiaggui, ibtopodiff, ibdmchk, ibdmsh, ibdmtr |
| Others Supported | Linux* command: ibsrpdm<br>ibv_devices |
| Miscellaneous | Intel® Omni-Path Fabric Suite FastFabric CLI: opaexpandfile, opafirmware |

# Appendix C Integrating Job Schedulers with Virtual Fabrics

Clusters deployed with multiple virtual fabrics may provide multiple QoS levels for compute jobs, security for separation of traffic, QoS to separate compute from other fabric services such as storage and management, etc. When a cluster is configured with multiple virtual fabrics, it is important to ensure that jobs are launched on the proper virtual fabric.

Many job schedulers provide integrated and automated mechanisms to assign, queue and launch jobs within the proper virtual fabric. However in some cases when launching jobs manually, or when using a job scheduler where the user must provide the final scripts for launching the job (such as scripts that invoke `mpirun`) it may be necessary to directly identify the virtual fabric.

In such cases, it can generally be assumed that the whole job will be run within a single virtual Fabric. This implies:

- A single PKey will be used for all communications within the job
- A single Service Level (SL) or set of Service Levels will be used for all communications within the job
- A single MTU may be used for all communications within the job

As a short cut, in most cases, it can also be assumed that a single StaticRate, PktLifeTime, and other parameters may be acceptable for the whole job. However depending on fabric status, this may not always be true.

Given these assumptions the PKey, SL, and MTU for the desired virtual fabric can be determined and supplied to all ranks in the job. This is often done using parameters to `mpirun`.

The PKey, SL, and MTU for the desired virtual fabric can be determined in a few ways:

1. If the virtual fabric configuration explicitly specified the PKey, SL and MTU to be used, these values can be provided *a priori* to the job launch in a hardcoded manner. If no MTU was specified or an MTU of unlimited was specified, a value of 8192 can typically be used. Note that this approach is the most error prone, but is sometimes the easiest way to get started.
2. The parameters of the virtual Fabric can be determined at job launch time via one of the Intel® Omni-Path tools. Various tools can provide the necessary information such as:
    a. `opagetvf` – This command can provide the essential information about a given virtual fabric. A typical usage would be to specify the virtual fabric by name. The output of this tool is in a simple colon delimited format which can

be easily scripted. There is also a related tool, `opagetvf_env`, that can be included in bash scripts. `opagetvf_env` provides a set of bash scripting functions that can obtain the same information.

```
$ opagetvf -d ComputeLow
ComputeLow:5:0x3:0:8192:unlimited:0x3

. /usr/sbin/opagetvf_env    # defines bash function opagetvf_func
# openMPI example
# These 4 lines select a Virtual Fabric by name and configure PKEY, SL,
MTU
opagetvf2_func "-d 'ComputeLow'" OMPI_MCA_btl_openib_pkey   \
    OMPI_MCA_btl_openib_ib_service_level OMPI_MCA_btl_openib_mtu
export OMPI_MCA_mtl_psm_ib_pkey=$OMPI_MCA_btl_openib_pkey
export OMPI_MCA_mtl_psm_ib_service_level=
$OMPI_MCA_btl_openib_ib_service_level
export MPI_CMD_ARGS="$MPI_CMD_ARGS -x PSM_MTU=$OMPI_MCA_btl_openib_mtu"
# MVAPICH2 example
opagetvf_func "-d 'ComputeLow'" MV2_DEFAULT_PKEY
MV2_DEFAULT_SERVICE_LEVEL MTU
export MPI_CMD_ARGS="-genv MV2_DEFAULT_PKEY $MV2_DEFAULT_PKEY -genv
MV2_DEFAULT_SERVICE_LEVEL $MV2_DEFAULT_SERVICE_LEVEL -genv PSM_PKEY
$MV2_DEFAULT_PKEY -genv HFI_SL $MV2_DEFAULT_SERVICE_LEVEL"
[ -n "$MTU" ] && export MPI_CMD_ARGS="$MPI_CMD_ARGS -genv MV2_DEFAULT_MTU
IBV_MTU_$MTU -genv PSM_MTU $MTU
# then invoke the appropriate mpirun using $MPI_CMD_ARGS as some of the
arguments
```

For more information, see the *Intel® Omni-Path Fabric Host Software User Guide*.

b. `opareport -o vfinfo` – This command provides a list of all active virtual fabrics. Its output can be easily scripted using the `-x` option and `opaxmlextract`. See the *Intel® Omni-Path Fabric Suite FastFabric User Guide* for more information.

```
opareport -o vfinfo
Index: 0   Name: Networking  …..
-------------------------------------------------------------------------
Index: 1   Name: Default   …..
-------------------------------------------------------------------------
Index: 2   Name: Admin   …..
-------------------------------------------------------------------------
Index: 3   Name: Storage …..
-------------------------------------------------------------------------
Index: 4   Name: CheckPoint …..
-------------------------------------------------------------------------
Index: 5   Name: ComputeLow
ServiceId: 0x0000000000000000  MGID: 0x0000000000000000:0x0000000000000000
PKey: 0x3   SL: 0  Select: 0x3: PKEY SL   PktLifeTimeMult: 2
MaxMtu:  8192  MaxRate: unlimited   Options: 0x03: Security QoS
QOS: Bandwidth:  20%  PreemptionRank: 1  HoQLife:    8 ms
-------------------------------------------------------------------------
Index: 6   Name: ComputeHigh
ServiceId: 0x0000000000000000  MGID: 0x0000000000000000:0x0000000000000000
PKey: 0x6   SL: 1  Select: 0x3: PKEY SL   PktLifeTimeMult: 2
MaxMtu:  8192  MaxRate: unlimited   Options: 0x03: Security QoS
QOS: HighPriority  PreemptionRank: 2  HoQLife:    8 ms
-------------------------------------------------------------------------
```

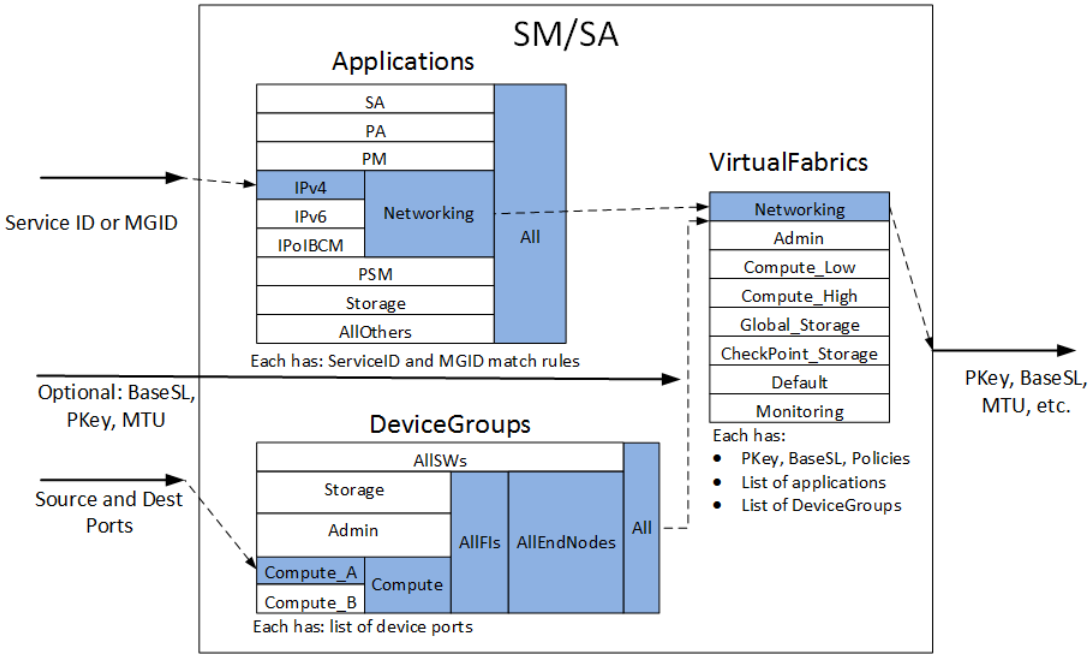
# Appendix D Integrating Other Service Applications with Virtual Fabrics

## D.1 Unicast Applications

Virtual Fabrics are designed to integrate automatically within the open fabrics alliance stack. Applications that take advantage of standard connection establishment and address resolution (a.k.a. Path Record resolution) mechanisms such as RDMA CM, IB CM, and ibacm automatically make the necessary SA queries. These SA queries allow the fabric manager to provide Path Records with the appropriate settings for Service Level (SL), Partition Key (PKey), Maximum Transfer Unit (MTU) and other parameters used for fabric communications.

The standard flow for Path Records and virtual fabric address resolution is as shown in the following figure:

**Figure 16. Address Resolution and vFabrics**



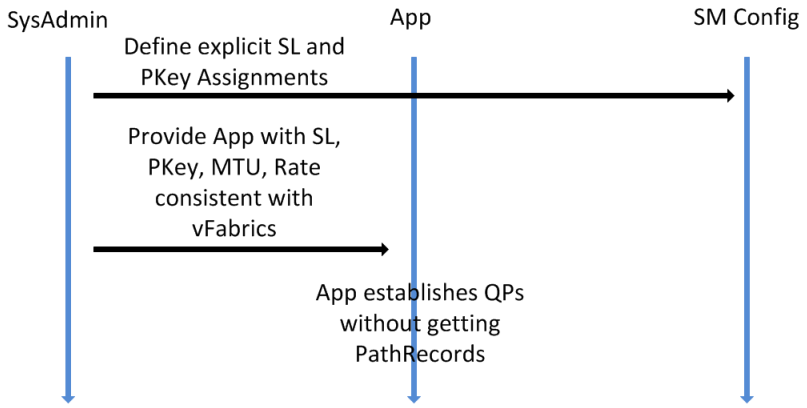

As shown in the flow diagram, the SA makes use of the Service Identifier (Service Id) supplied in the `PathRecord` query to identify the application making the request. The source and destination nodes specified in the query imply the possible set of `DeviceGroups` that are relevant, and then the SA finds the virtual fabric(s) that represent the intersection of the given application with the possible device groups. To ensure this process works smoothly and correctly, it's important that the FM

configuration of Virtual Fabrics specifies the appropriate set of Service Ids for the application. Some applications may provide direct configuration of the Service Id as a 64-bit number. This is especially true of those using the IB CM or ibacm directly. Applications using the RDMA CM specify a protocol and port number that are used to compose a 64-bit service ID.

RDMA Service IDs take the format 0x0000000001NNPPPP where N is the IANA protocol number and P is the port number the application needs to bind on, both in hexadecimal. For example, Lustre is known to use the TCP protocol (0x06) on port 987 (0x03DB) so its Service ID is expected to be 0x00000000010603DB. The `opafm.xml` file has a predefined rule for Lustre, iSER and general RDMA apps as well as a few examples of non-RDMA applications.

In some cases, the application may use non-standard, out-of-band, or ad-hoc mechanisms to establish connections. In that case, the key connection parameters of `PKey`, `BaseSL`, and `MTU` will need to be specified *a priori*. This approach is shown in the following figure.

**Figure 17. vFabric Cheats for QoS and Security**




This process may be manual. In which case the sysadmin (a human) must make explicit choices of `BaseSL`, `PKey`, and `MTU` in the FM configuration and then must provide those exact same values to the application. If mistakes are made, the application could end up running on the wrong QoS level or perhaps even fail to start up due to being unable to communicate with the desired nodes in the fabric.

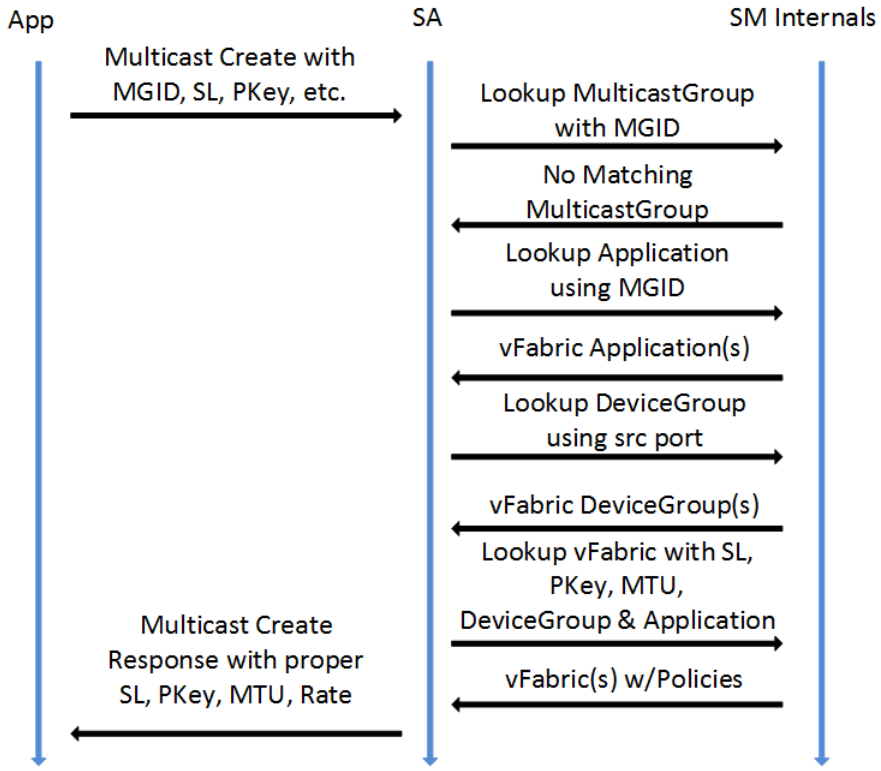
As discussed in Integrating Job Schedulers with Virtual Fabrics on page 179, Intel® Omni-Path provides assorted tools that may be used to identify the `PKey`, `BaseSL`, and `MTU` associated with a given virtual fabric. In some cases, those tools may be used to automate the discovery of the `BaseSL`, `PKey`, and `MTU` and then provide them directly to the application, thus reducing the risk of human mistakes and permitting future starts of the application to correctly obtain any changes to the virtual fabrics configuration.

## D.2 Multicast Applications

Applications that take advantage of standard multicast mechanisms (a.k.a. Multicast Member Records) either directly, through IPoFabric (a.k.a. IPoIB), or ibacm, automatically receive the proper `SL`, `PKey`, and `MTU` as part of the multicast group parameters.

The standard flow for Multicast Member Records when creating a new multicast group and virtual fabric address resolution is shown in the following figure.

**Figure 18. Multicast Create and vFabrics**




As shown in the flow diagram in the previous figure, the SA uses the Multicast Group ID (MGID) supplied in the Multicast Member Record to identify the application making the request. The source nodes specified in the query imply the possible set of DeviceGroups that are relevant, and then the SA find the virtual fabric(s) that represent the intersection of the given application with the possible device groups. To ensure this process works smoothly and correctly, it is important that the FM configuration of Virtual Fabrics specifies the appropriate set of MGIDs for the application. Some applications may provide direct configuration of the MGID as a 128-bit number. This is especially true of those using the IB SA or ibacm directly. Applications using IPoFabric (a.k.a. IPoIB) specify an IP multicast group and these are used to compose a 128-bit MGID.

For IPv4, MGIDs are composed by IPoFabric as follows:

```
MGID = 0xffFS401bPPPP0000:00000000G
```

where F=flags, S=scope, P=PKey, and G=IP Multicast Group

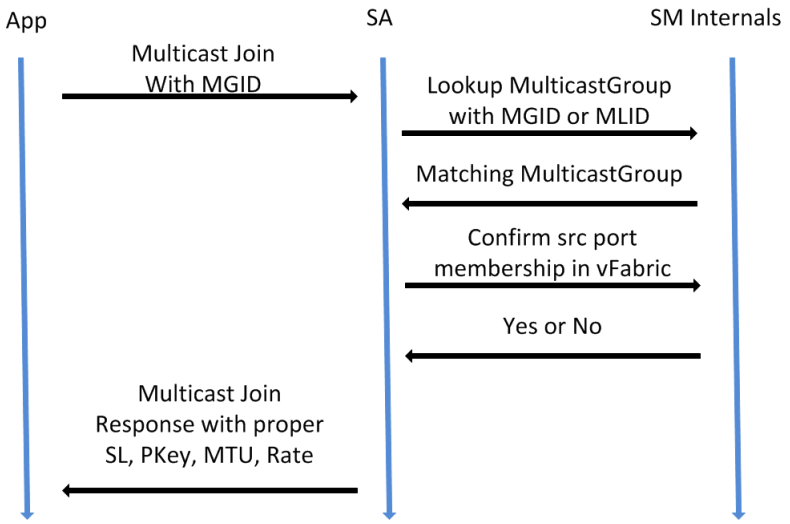
For IPv6, MGIDs are composed by IPoFabric as follows:

```
MGID = 0xffFS601bPPPPGGGG:GGGGGGGGGGGGGGGG
```

where F=flags, S=scope, P=PKey and G=IP Multicast Group

When an application joins an existing multicast group through the Multicast Member Record mechanisms, the steps in the following figure occur:

**Figure 19. Multicast Join and vFabrics**




In this case the pre-existing multicast group will already have an assigned `SL`, `PKey`, and `MTU`. The new join request will merely need to confirm its ability to communicate with that `PKey`.

There is limited support for non-standard participation in multicast groups. While groups may be pre-created in the `opafm.xml` configuration file (by specifying an appropriate MulticastGroup section with all the relevant parameters for the multicast group), the act of joining the group requires the use of Multicast Member Records. Such joins may be performed using the proxy join mechanism, whereby one node issues Multicast Member Record join or leave requests on behalf of another node. In that case, out-of-band mechanisms may be used by the application to communicate or discover the address (for example, Multicast LID) and parameters assigned to the multicast group by the FM.

The current set of multicast groups, along with their `SL`, `PKey`, and `MTU` and members may be listed using the `opashowmc` command. See the *Intel® Omni-Path Fabric Suite FastFabric User Guide* for more information on `opashowmc`.

# Appendix E Node Naming Recommendations

Many of the features in Intel® Omni-Path are designed to work easiest when all the nodes in the fabric are assigned unique `NodeDesc` values. For servers, the `NodeDesc` is automatically assigned by the OFA stack to be consistent with the hostname of the server. For switches, all models of Intel® Omni-Path switches permit the sysadmin to set the node description of the switch. See the *Intel® Omni-Path Fabric Suite FastFabric User Guide* for information on setting the node description of externally managed switches and see the relevant switch CLI guide for setting the node description of internally managed switches.

Within the FM, the `DeviceGroup` feature permits a set of nodes to be identified by wild-carded name comparison. `DeviceGroups` may be used as part of virtual fabrics, PM device groups, routing algorithm optimizations (such as dgshortestpath's `RoutingOrder` and faftrees's `RouteLast` and `CoreSwitches`). To make these features the easiest to use, Intel recommends using carefully selected names for the various switches and servers that can be easily matched with `DeviceGroup` wildcard patterns.

Using names such as the examples in the first table below permits the simple specification of wildcards for device groups with name examples shown in the second table.

**Table 42. Device Group Names Examples**

| Hostname/node description | Purpose |
|---|---|
| OSS01 | Lustre Object Store server |
| OSS02 | |
| OSS03 | |
| MDS01 | Lustre metadata server |
| Compute001 | Compute node |
| Compute002 | |
| Compute900 | |
| FM01 | Fabric manager |
| FM02 | |
| Boot01 | Boot server` |
| Boot02 | |
| Login01 | Login server |
| Login02 | |
| SWEdge001 | First edge switch |

***continued...***

| Hostname/node description | Purpose |
|---|---|
| SWEdge002 | |
| SWCore001 | First core switch |
| SWCore002 | |

**Table 43. Wildcard Device Group Examples**

| Wildcard | Purpose |
|---|---|
| OSS* | All Object Store servers |
| Compute* | All compute nodes |
| FM* | All FMs |
| Compute[001-200] | First 200 compute nodes |
| SW* | All switches |
| SWEdge* | All edge switches |
| SWCore* | All core switches |

In general the recommendation is to consider what device groups you may want to use for your cluster and then pick names such that each device group can consist of a small number of wildcards that will match all the desired nodes. This recommendation can make creation, management, and the performance of your device group configuration in the `opafm.xml` configuration file easier and faster.

# Appendix F FM Log Messages

## F.1 FM Event Messages

The host-based and embedded FM both log significant fabric events in a standard machine-readable format. The format for these special event messages provides information not only about the event, but information regarding what nodes in the fabric are causing the event.

### F.1.1 FM Event Message Format

The format of these messages is as follows:

```
<prefix>;MSG:<msgType>|SM:<sm_node_desc>:port <sm_port_number>|
COND:<condition>|NODE:<node_desc>:port <port_number>:<node_guid>|
LINKEDTO:<linked_desc>:port <linked_port>:<linked_guid>| DETAIL:<details>
```

Where:

- `<prefix>` – Includes the date and time information of the event along with either the slot number OR hostname and IP address of the FM reporting the message.
- `<msgType>` – Is one of the following values:
  - `ERROR`
  - `WARNING`
  - `NOTICE`
  - `INFORMATION`
- `<sm_node_desc>` and `<sm_port_number>` – Indicate the node name and port number of the SM that is reporting the message, prefixed with the word '`port`'. For the embedded version of the SM, the port number will be 0. Any pipes (`|`) or colons (`:`) in the node description will be converted to spaces in the log message.
- `<condition>` – Is one of the conditions from the event SM Reporting Table that are detailed in the section FM Event Descriptions on page 188. The condition text includes a unique identification number. The possible conditions are as follows:
  1. Redundancy Lost
  2. Redundancy Restored
  3. Appearance in Fabric
  4. Disappearance from Fabric
  5. SM State Change to Master
  6. SM State Change to Standby
  7. SM Shutdown

8. Fabric Initialization Error
9. Link Integrity Error
10. Security Error
11. Other Exception
12. Fabric Summary
13. SM State Change to Inactive
14. SM standby configuration inconsistency
15. SM standby virtual fabric configuration inconsistency
16. Reserved for Future Use
17. PM secondary configuration inconsistency

- `<node_desc>`, `<port_number>`, and `<node_guid>` are the node description, port number and node GUID of the port and node that are primarily responsible for the event. Any pipes (`|`) or colons (`:`) in the node description will be converted to spaces in the log message.
- `<linked_desc>`, `<linked_port>` and `<linked_guid>` are optional fields describing the other end of the link. These fields and the '`LINKEDTO`' keyword will only be shown in applicable messages. Any pipes (`|`) or colons (`:`) in the node description will be converted to spaces in the log message.
- `<details>` is an optional free-form field detailing additional information useful in diagnosing the log message cause.

## F.1.2 FM Event Descriptions

The following are the FM event messages, their severity, an explanation, possible causes for the event.

### F.1.2.1 #1 Redundancy Lost

#### Severity

Warning

#### Explanation

The subnet manager emits this message when it is the only running Subnet Manager on a given subnet.

#### Causes

No redundant SM exists on the subnet.

A user shutdown a redundant SM or possibly disconnected or shutdown the node on which the SM was running.

#### Action

If running redundant SMs on a fabric, verify health of each host or switch running an SM.

### F.1.2.2 #2 Redundancy Restored

#### Severity

Notice

#### Explanation

The Master SM for the subnet detected that another SM has come online.

#### Causes

A user started a redundant SM on another host or switch.

A user just connected two separate subnets together.

#### Action

None

### F.1.2.3 #3 Appearance in Fabric

#### Severity

Notice

#### Explanation

A new HFI port, switch, inter-switch link or Subnet Manager was detected by the master Subnet Manager.

#### Causes

User action

#### Action

None

### F.1.2.4 #4 Disappearance from Fabric

#### Severity

Notice

#### Explanation

An HFI port, switch, inter-switch link or Subnet Manager has disappeared from fabric. This encompasses system shutdowns, and loss of connectivity.

#### Action

The administrator should validate whether or not the components have disappeared from the fabric due to user action or not. Nodes will typically disappear from the fabric when they are rebooted, re-cabled, or if their Intel® Omni-Path Fabric stacks are stopped.

### F.1.2.5 #5 SM State Change to Master

#### Severity

Notice

#### Explanation

Subnet manager transitioned into the master state from one of the 'standby', 'discovering' or 'not active' states.

#### Action

The administrator should check the state of the machine (or chassis) that was providing the master SM service to determine if it has failed and needs to be replaced, or whether the state change occurred due to user action.

#### Example

```
Nov 28 17:45:25 sample-host fm0_sm[29326]:
;MSG:NOTICE|SM:sample-host.sample-domain.com:port 1|COND:#5 SM state to
master|NODE:sample-host.sample-domain.com:port
1:0x0x00066a00a0000405|DETAIL:transition from DISCOVERING to MASTER
```

### F.1.2.6 #6 SM State Change to Standby

#### Severity

Notice

#### Explanation

Subnet manager transitioned from 'master' into 'standby' state.

#### Action

The administrator should validate that this was due to a modification in the Intel® Omni-Path Fabric network configuration. If not, then this issue should be reported to customer support.

#### Example

```
Nov 29 12:15:28 sample-host fm0_sm[31247]:
;MSG:NOTICE|SM:sample-host.sample-domain.com:port 1|COND:#6 SM state to
standby|NODE:sample-host.sample-domain.com:port
1:0x0x00066a00a0000405|DETAIL:transition from MASTER to STANDBY
```

### F.1.2.7 #7 SM Shutdown

#### Severity

Notice

#### Explanation

The master subnet manager is shutting down.

#### Action

The administrator should check the state of the machine (or chassis) that was providing the master SM service, or whether the state change occurred due to user action.

#### Example

```
;MSG:NOTICE|SM:sample-host.sample-domain.com:port 1|COND:#7 SM
shutdown|NODE:sample-host.sample-domain.com:port 1:0x0x00066a00a0000405|DETAIL:
```

### F.1.2.8 #8 Fabric Initialization Error

#### Severity

Notice

#### Explanation

Some form of error occurred during fabric initialization. Examples of possible errors include:

- Link could not be activated in 4x mode.
- Subnet manager could not initialize a port or node with proper configuration.

#### Action

The administrator should perform the fabric troubleshooting procedure to isolate and repair the faulty component. The faulty component could be the SM platform itself (for example, its own HFI) or a component in the Intel® Omni-Path Fabric network.

#### Example

```
Apr  6 22:48:42 sample-host fm0_sm[21458]: sample-host; MSG:NOTICE|SM:sample-
host:port 2|COND:#8
Fabric initialization error|NODE:sample-host2:port
1:0x0011750000ffd7af|LINKEDTO:Intel OPA Switch:port
18:0x00066a00d9000108|DETAIL:Failed to set portinfo for node
```

### F.1.2.9 #9 Link Integrity Error

#### Severity

Notice

#### Explanation

The SM received an asynchronous trap from a switch or end-port indicating a link integrity problem.

#### Action

The administrator should perform the fabric troubleshooting procedure to isolate and repair the faulty component. This is typically due to a bad cable, an incorrect cable being used for the signaling rate and cable length (for example, too small a wire gauge), or a hardware failure on one of the two HFI ports.

### F.1.2.10 #10 Security Error

#### Severity

Notice

#### Explanation

The SM received an asynchronous trap from a switch or end-port indicating a management key violation.

#### Action

The administrator should validate that the software configuration has not changed, because this issue is most likely due to a configuration issue. However, this event could also indicate a more serious issue such as a hacking attempt.

### F.1.2.11 #11 Other Exception

#### Severity

Notice

#### Explanation

The subnet manager encountered an error at some time after fabric initialization. Examples of possible errors are:

- The SM received an invalid request for information.
- The SM could not perform action requested by another fabric entity such as a request to create or join a multicast group with an unrealizable MTU or rate.

#### Action

The administrator should check to see if other SM related problems have occurred and perform the corrective actions for those items. If these other exceptions continue to persist, then customer support should be contacted.

### F.1.2.12 #12 Fabric Summary

#### Severity

Notice

#### Explanation

A brief message describing the number of changes that the SM detected on its last subnet sweep. This message will include totals for the number of switches, HFIs, end-ports, total physical ports and SMs that have appeared or disappeared from the fabric. This message will only be logged at the end of a subnet sweep if the SM had detected changes.

#### Action

As this is only a summary of events detected during a fabric sweep, the administrator should examine the logs for preceding messages that describe the fabric changes in detail.

#### Example

```
Apr  8 15:31:36 sample-host fm0_sm[21458]: sample-host; MSG:NOTICE|SM:sample-
host:port
2|COND:#12 Fabric Summary|NODE:sample-host:port 2:0x00066a01a0000405|
DETAIL:Change
Summary: 1 SWs disappeared, 0 HFIs appeared, 1 end ports disappeared, 3 total
ports
disappeared, 0 SMs appeared
```

### F.1.2.13 #13 SM State Change to Inactive

#### Severity

Notice

#### Explanation

Subnet manager transitioned from *standby* into *inactive* state.

#### Action

The administrator should check for inconsistencies in XML configurations between the master SM and this SM.

#### Example

```
Nov 29 12:15:28 sample-host fm0_sm[31247]:
;MSG:NOTICE|SM:sample-host.sample-domain.com:port 1|COND:#13 SM state to
inactive|NODE:sample-host.sample-domain.com:port
1:0x0x00066a00a0000405|DETAIL:transition from STANDBY to NOTACTIVE
```

### F.1.2.14 #14 SM Inconsistency

#### Severity

Warning

#### Explanation

Deactivating the Standby Subnet Manager and Secondary Performance Manager due to inconsistent Subnet Manager XML configuration on Standby.

#### Action

If the condition persists, compare the XML configuration files between the master and standby SM for inconsistencies.

#### Example

```
Oct 22 12:49:06 shaggy fm0_sm[31032]: shaggy; MSG:WARNING|SM:shaggy:port 1|
COND:#14
SM standby configuration inconsistency|NODE:i9k118:port
0:0x00066a00d8000118|DETAIL:Deactivating standby SM i9k118 : 0x00066a00d8000118
which has a SM configuration inconsistency with master! The secondary PM will
also
be deactivated.
```

### F.1.2.15 #15 SM Virtual Fabric Inconsistency

#### Severity

Warning

#### Explanation

Deactivating the Standby Subnet Manager and Secondary Performance Manager due to inconsistent Subnet Manager Virtual Fabrics XML configuration on Standby.

#### Action

If the condition persists, compare the XML configuration files between the master and standby SM for inconsistencies.

#### Example

```
Oct 22 12:23:41 shaggy fm0_sm[30778]: shaggy; MSG:WARNING|SM:shaggy:port 1|
COND:#15
SM standby virtual fabric configuration inconsistency|NODE:i9k118:port
0:0x00066a00d8000118|DETAIL:Deactivating standby SM i9k118 : 0x00066a00d8000118
which has a Virtual Fabric configuration inconsistency with master! The secondary
PM will also be deactivated.
```

### F.1.2.16 #16 Reserved for Future Use

### F.1.2.17 #17 PM Inconsistency

#### Severity

Warning

#### Explanation

Deactivating the Secondary Performance Manager and Standby Subnet Manager due to inconsistent Performance Manager XML configuration on Secondary.

#### Action

If the condition persists, compare the XML configuration files between the primary and secondary PM for inconsistencies.

#### Example

```
Oct 22 12:51:42 shaggy fm0_sm[31173]: shaggy; MSG:WARNING|SM:shaggy:port 1|
COND:#17
PM secondary configuration inconsistency|NODE:i9k118:port
0:0x00066a00d8000118|DETAIL:Attempting to deactivate secondary PM which has a
configuration inconsistency with primary! The standby SM will also be deactivated.
```

## F.2 Other Log Messages

In addition to the FM Event messages detailed in the previous section, the FM software suite may emit other log messages that provide extra detail for use by technical personnel in troubleshooting fabric issues.

Log messages generally follow this format:

```
<prefix>: <severity>[<module>]: <component>: <function>:
<message>
```

Where:

- `prefix` - Includes time and date followed by the hostname and/or IP of the FM reporting the message, the instance name and a Process ID (PID) number.
- `severity` - One of the following:

  FATAL, ERROR, WARN, NOTIC, INFO, PROGR, VBOSE, DBG[1-4], ENTER or EXIT.
- `module` - Program module that generated the message. Typically the name of the sub-component or library that saw the event.
- `component` - Name of the FM process that owns the module.
- `function` - Part of the sub-module where the event occurred. This is probably only useful for developers but might give insight to what the FM is currently doing.
- `message` - Free form message text giving more details or explaining the event.

*Note:* Some of the listed components of the formatting may be omitted.

#### Example

```
Jan 20 14:40:22 phgppriv36 fm0_sm[4082]: PROGR[topology]: SM:
topology_main: DISCOVERY CYCLE END. 0 SWs, 2 HFIs, 2 end ports,
2 total ports, 1 SM(s), 26 packets, 0 retries, 0.004 sec sweep
```

### F.2.1 Information (INFO)

#### F.2.1.1 Last full member of multicast group GID 0xff12401bffff0000:00000000ffffffff is no longer in fabric, deleting all members

##### SM Area

Discovery

##### Meaning

The last full member of the group has left. The group is removed from the fabric.

##### Action

None.

#### F.2.1.2 topology_discovery: now running as a STANDBY SM

##### SM Area

Discovery

##### Meaning

SM has transitioned to STANDBY mode.

##### Action

None.

#### F.2.1.3 TT: DISCOVERY CYCLE START

##### SM Area

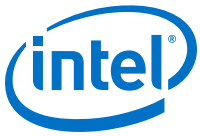
Discovery

##### Meaning

Discovery sweep has started.

##### Action

None.



### F.2.1.4 TT: DISCOVERY CYCLE END

#### SM Area

Discovery

#### Meaning

Discovery sweep has ended.

#### Action

None.

### F.2.1.5 Port x of node [y] HFI1 belongs to another SM [0x0001]; Marking port as NOT MINE!

#### SM Area

Discovery

#### Meaning

Usually happens during the merging of two fabrics.

#### Action

None.

### F.2.1.6 sa_PathRecord: requested source GUID/LID not found/active in current topology

#### SM Area

Administrator.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric.

#### Action

Check the health of the requester and the connected port if the message persists.

### F.2.1.7 sa_PathRecord: requested destination GUID not an active port nor a Multicast Group

#### SM Area

Administrator.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric or the destination has dropped from fabric.

#### Action

None.

### F.2.1.8 sa_XXXXXXXX: Cannot find source lid of 0x0001 in topology in request to subscribe/unsubscribe...

#### SM Area

Administrator.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric or request received from the node that the SM has dropped from the fabric due to non-response to SMA queries.

#### Action

Check health of the node at lid 0x0001 if the fabric is stable.

### F.2.1.9 sa_XXXXXXXX: requested source Lid/GUID not found/active in current topology

#### SM Area

Administrator.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric or request received from the node that the SM has dropped from the fabric due to non-response to SMA queries.

#### Action

Check the health of node at lid 0x0001 if the fabric is stable.

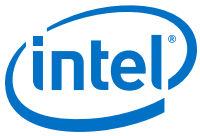
### F.2.1.10 sa_McMemberRecord_Set: Port GID in request (0xFE80000000000000:00066a00d9000143) from HFI1, Port 0x00066a00d9000143, LID 0x0001, for group 0xFF12401BFFFF0000:00000000FFFFFFFF can't be found or not active in current topology, returning status 0x0001/0x0200

#### SM Area

Administrator.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric or request received from the node that the SM has dropped from the fabric due to the non-response to SMA queries.



#### Action

Check health of node at lid 0x0001 if fabric is stable.

### F.2.1.11 sa_McMemberRecord_Set: Last full member left multicast group GID 0xFF12401BFFFF0000:00000000FFFFFFFF, deleting group and all members

#### SM Area

Administrator.

#### Meaning

Group is cleaned out when last the FULL member leaves.

#### Action

None.

## F.2.2 Warning (WARN)

### F.2.2.1 failed to send reply [status=x] to SMInfo GET request from node HFI1 guid 0x00066a00d9000143, TID=0x811E796027000000

#### SM Area

SM to SM Communication.

#### Meaning

Lost communication path to other SM on node HFI1.

#### Action

Check the health of the node described in the message and status of the SM node.

### F.2.2.2 failed to send reply [status=x] to SMInfo SET request from node HFI1 guid 0x00066a00d9000143, TID=0x811E796027000000

#### SM Area

SM to SM Communication.

#### Meaning

Lost communication path to the other SM on node HFI1.

#### Action

Check the health of the node described in the message and the status of the SM node.

### F.2.2.3 SmInfo SET control packet not from a Master SM on node HFI1, lid [0x1], guid 0x00066a00d9000143, TID=0x811E796027000000

#### SM Area

SM to SM Communication.

#### Meaning

The SM on node HFI1 is violating the protocol.

#### Action

If the condition persists, turn off the SM on node HFI1.

### F.2.2.4 Standby SM received invalid AMOD[1-5] from SM node HFI1, LID [0x1], guid [0x00066a00d9000143], TID=0x811E796027000000

#### SM Area

SM to SM Communication.

#### Meaning

SM on node HFI1 is violating the protocol specification.

#### Action

If the condition persists, turn off the SM on node HFI1.

### F.2.2.5 MASTER SM did not receive response to Handover Acknowledgement from SM node HFI1, LID [0x1], guid [0x00066a00d9000143]

#### SM Area

SM to SM Communication.

#### Meaning

The SM on node HFI1 is incompatible or lost the communication path.

#### Action

Remove the incompatible SM from the fabric or check the health of node HFI1.

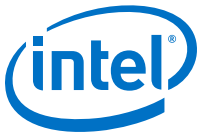
### F.2.2.6 INACTIVE SM received invalid STANDBY transition request from SM node HFI1, LID [0x1], guid [0x00066a00d9000143], TID=0x811E796027000000

#### SM Area

SM to SM Communication.

#### Meaning

The SM on node HFI1 is violating the protocol specification.



#### Action

If the condition persists, turn off the SM on node HFI1.

### F.2.2.7 Master SM received invalid Handover Ack from remote SM HFI1, LID [0x1], guid [0x00066a00d9000143], TID=0x811E796027000000; remote not in STANDBY state [Discovering]

#### SM Area

SM to SM Communication.

#### Meaning

The SM on node HFI1 is violating the protocol specification.

#### Action

If the condition persists, turn off the SM on node HFI1.

### F.2.2.8 Master SM received invalid MASTER transition [requested state] from remote [remote state] SM HFI1, LID [0x1], guid [0x00066a00d9000143], TID=0x811E796027000000

#### SM Area

SM to SM Communication.

#### Meaning

The SM on node HFI1 is violating the protocol specification.

#### Action

If the condition persists, turn off the SM on node HFI1.

### F.2.2.9 Master SM did not receive response to Handover Acknowledgment from [remote state] SM node HFI1, LID [0x1], guid [0x00066a00d9000143]

#### SM Area

SM to SM Communication.

#### Meaning

Lost communication path to the other SM on node HFI1.

#### Action

Check the health of node described in the message and the status of the SM node.

### F.2.2.10 SM at shaggy HFI-1, portGuid=0x0011750000ff8f4d has a different SM configuration consistency checksum [418863] from us [417845]

#### SM Area

SM to SM Communication.

#### Meaning

The SM on node HFI1 configuration does not match master.

#### Action

Verify that the XML configuration between master and standby SM is consistent.

### F.2.2.11 No transitions allowed from DISCOVERING state; Got (ANY) request from [state] SM node HFI1, LID [0x1], guid [0x00066a00d9000143]

#### SM Area

SM to SM Communication.

#### Meaning

The SM on node HFI1 is violating the protocol specification.

#### Action

If the condition persist, turn off the SM on node HFI1.

### F.2.2.12 SmInfo from SM at SMLID[0x1] indicates SM is no longer master, switching to DISCOVERY state

#### SM Area

SM to SM Communication.

#### Meaning

Remote SM may have handed over to another SM on the fabric.

#### Action

None.

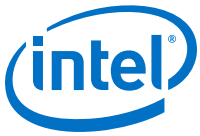
### F.2.2.13 Switching to DISCOVERY state; Failed to get SmInfo from master SM at LID 0x1

#### SM Area

SM to SM Communication.

#### Meaning

Lost the communication path to the other SM at lid 0x1.



#### Action

Check the health of the node described in the message and the status of the SM node.

### F.2.2.14 too many errors during sweep, will re-sweep in a few seconds

#### SM Area

Discovery.

#### Meaning

Multiple cable pulls or chassis removal/insertion event.

#### Action

Check links with high error count and reseat or replace cable. If the condition persists, capture the log information and call support.

### F.2.2.15 unable to setup port [x] of node Sw1/HFI1, nodeGuid 0x00066a00d9000143, ignoring port!

#### SM Area

Discovery.

#### Meaning

Lost communication path to node HFI1.

#### Action

Check the health of the node port described in the message.

### F.2.2.16 Get NodeInfo failed for node off Port x of Node 0x00066a00d9000143:HFI1, status=7

#### SM Area

Discovery.

#### Meaning

The node connected to port x of HFI1 is not responding.

#### Action

Check the health of the node connected to the port.

### F.2.2.17 Get NodeDesc failed for node off Port X of Node 0x00066a00d9000143:HFI1, status = 7

#### SM Area

Discovery.

#### Meaning

The node connected to port x of HFI1 is not responding.

#### Action

Check the health of the node connected to the port.

### F.2.2.18 Failed to get Switchinfo for node sw1 guid 0x00066a00d9000143: status = 7

#### SM Area

Discovery.

#### Meaning

Switch node 1 is not responding.

#### Action

If the condition persists, check the health of the switch and capture health data if possible.

### F.2.2.19 Failed to set Switchinfo for node sw1 nodeGuid 0x00066a00d9000143: status = 7

#### SM Area

Discovery.

#### Meaning

Switch node 1 not responding.

#### Action

If the condition persists, check the health of the switch and capture health data if possible.

### F.2.2.20 Failed to get PortInfo from NodeGUID x [Hfi1] Port 1; Ignoring port!

#### SM Area

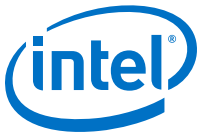
Discovery.

#### Meaning

Port x of HFI1 not responding.

#### Action

Check the health of node HFI1.



### F.2.2.21 port on other side of node sw1 index x port X is not active

#### SM Area

Discovery.

#### Meaning

The node may have been marked down if it did not respond to SMA queries.

#### Action

Check the health of the node connected to switch 1 port X.

### F.2.2.22 Node Hfi1 [0x00066a00d9000143] port[x] returned MKEY[0x1] when MKEY[0x0] was requested!

#### SM Area

Discovery.

#### Meaning

Another SM with a different Mkey configured.

#### Action

Stop one of the subnet managers and make the configuration consistent.

### F.2.2.23 Cannot get PORTINFO for node HFI1 nodeGuid 0x00066a00d9000143 port X status=Y

#### SM Area

Discovery.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric. Persistence indicates that the node may be having problems.

#### Action

Check the health of the node HFI1/SW1 if the fabric was idle or persistent condition.

### F.2.2.24 Cannot set PORTINFO for node HFI1 nodeGuid 0x00066a00d9000143 port X status=Y

#### SM Area

Discovery.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric. Persistence indicates the node may be having problems.

#### Action

Check the health of node HFI1/SW1 if the fabric was idle or persistent condition.

### F.2.2.25 Could not find neighbor for NodeGUID x Sw1] (neighbor idx x, port y) in new topology; spanning tree not up to date

#### SM Area

Discovery.

#### Meaning

Caused by simultaneous removal/insertion events in the fabric.

#### Action

None.

### F.2.2.26 sa_NodeRecord_GetTable: Invalid node type[~1-3] in request from lid 0x1

#### SM Area

Administrator.

#### Meaning

Invalid data in the SA request.

#### Action

Check the health of the requester at lid 0x1.

### F.2.2.27 sa_PathRecord_Set: Cannot find path to port 0x00066a00d9000144 from port 0x00066a00d9000143: failing src/dst pkey 0x800d validation

#### SM Area

Administrator.

#### Meaning

The source and destination do not share a partition with the given PKey

#### Action

Configuration change required if they should have access.

### F.2.2.28 sa_PathRecord_Set: Cannot find path to port 0x00066a00d9000144 from port 0x00066a00d9000143: failing req/dst pkey validation

#### SM Area

Administrator.

#### Meaning

The requesting node and destination do not share a PKey.

#### Action

Configuration change required if they should have access.

### F.2.2.29 sa_PathRecord_Set: Cannot find path to port 0x00066a00d9000144 from port 0x00066a00d9000143: failing vFabric rate validation (mtu=2,rate=3)

#### SM Area

Administrator.

#### Meaning

The source and destination do not share a path in a vFabric that contains limitations on max mtu and rate.

#### Action

Configuration change required if path is valid.

### F.2.2.30 sa_PathRecord/SA_TraceRecord: Failed PKey check for source x and destination y for PKey 0x800d

#### SM Area

Administrator.

#### Meaning

A request was for given pkey, but source of query is not a member of the same partition.

#### Action

Configuration change may be necessary.

### F.2.2.31 sa_PathRecord/SA_TraceRecord: Failed pairwise PKey check for request

#### SM Area

Administrator.

#### Meaning

A query request failed pairwise pkey checks.

#### Action

Configuration change may be necessary.

### F.2.2.32 sm_resolve_pkeys_for_vfs: VFabric has undefined pkey. Assigning pkey 0x3.

#### SM Area

Configuration.

#### Meaning

A vFabric with an undefined pkey has been assigned a pkey.

#### Action

None.

### F.2.2.33 sa_ServiceRecord_GetTable: Filter serviced record ID=0x1000000000003531 from lid 0x4 due to pkey mismatch from request port

#### SM Area

Administrator.

#### Meaning

PKey validation failed for service record, request node does not have valid pkey.

#### Action

Configuration change required if request should be valid.

### F.2.2.34 sa_xxxxxx: too many records for SA_CM_GET

#### SM Area

Administrator.

#### Meaning

May have duplicate data in the fabric.

#### Action

Check topology data for duplicate GUIDs.

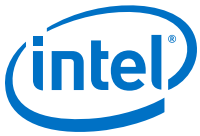
### F.2.2.35 sa_TraceRecord/Pathrecord_set: Cannot find path to port 0x00066a00d9000144 from port 0x00066a00d9000144: LFT entry for destination is 255 from switch Sw1 (nodeGuid 0x00066a00d9000999)

#### SM Area

Administrator.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric.



#### Action

Check SW1 for a bad port and health of the destination node if the condition persists.

### F.2.2.36 sa_TraceRecord/Pathrecord_set: Cannot find path to destination port 0x00066a00d9000144 from source port 0x00066a00d9000143; INVALID TOPOLOGY, next/last_nodep is NULL

#### SM Area

Administrator.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric.

#### Action

Check SW1 for a bad port and health of the destination node if the condition persists.

### F.2.2.37 sa_updateMcDeleteCountForPort: MC Dos threshold exceeded for: Node= HFI1, GUID=0x00066a00d9000143, PortIndex=1; bouncing port.

#### SM Area

Administrator.

#### Meaning

SM Multicast denial of service configured and threshold has been reached. Bouncing the port in attempt to clear issue.

#### Action

If multiple occurrence, check health of node HFI1.

### F.2.2.38 sa_updateMcDeleteCountForPort: MC Dos threshold exceeded for: Node= HFI1, GUID=0x00066a00d9000143, PortIndex=1; disabling port.

#### SM Area

Administrator.

#### Meaning

SM Multicast denial of service configured and threshold has been reached. disabling the port.

#### Action

Check the health of the node HFI1.

#### F.2.2.39 sa_updateMcDeleteCountForPort: MC Dos threshold exceeded for: Node= HFI1, GUID=0x00066a00d9000143, PortIndex=1; disabling port.

**SM Area**

Administrator.

**Meaning**

SM Multicast denial of service configured and threshold has been reached. disabling the port.

**Action**

Check the health of the node HFI1.

### F.2.3 Error

#### F.2.3.1 could not perform HANDOVER to remote SM HFI1: 0x00066a00d9000143

**SM Area**

SM to SM communication.

**Meaning**

Lost communication path to the other SM on node guid 0x00066a00d9000143.

**Action**

Check the health of the node HFI1 and status of the SM node.

#### F.2.3.2 can't get PortInfo, sleeping

**SM Area**

Discovery.

**Meaning**

topology_initialize: cannot get PortInfo; sleeping.

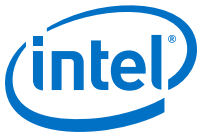
**Action**

Make sure the stack is running. Restart the SM node and stack.

#### F.2.3.3 port state < INIT, sleeping

**SM Area**

Discovery.



#### Meaning

The node port of the SM is down.

#### Action

Be certain the host cable is connected to a switch (host SM only).

### F.2.3.4 can't get/set isSM, sleeping

#### SM Area

Discovery.

#### Meaning

SM cannot communicate with the stack.

#### Action

Make sure the stack is running. Restart the SM node and stack.

### F.2.3.5 can't set up my port, sleeping

#### SM Area

Discovery.

#### Meaning

The SM cannot communicate with stack.

#### Action

Make sure the stack is running. Restart the SM node and stack.

### F.2.3.6 Get NodeInfo failed for local node. status 7

#### SM Area

Discovery.

#### Meaning

The SM cannot communicate with the stack.

#### Action

If condition persist, restart the SM node.

### F.2.3.7 sm_setup_node: Get NodeDesc failed for local node, status 7

#### SM Area

Discovery.

#### Meaning

The SM cannot communicate with the stack.

#### Action

If the condition persists, restart the SM node.

### F.2.3.8 Error adding Node GUID: 0x00066a00d9000143 to tree. Already in tree!

#### SM Area

Discovery.

#### Meaning

Duplicate Node GUID in fabric.

#### Action

Using fabric tools, locate the device with the duplicate node GUID and remove it.

### F.2.3.9 Error adding Port GUID: 0x00066a00d9000143 to tree. Already in tree!

#### SM Area

Discovery.

#### Meaning

Duplicate Port GUID in the fabric.

#### Action

Using fabric tools, locate the device with the duplicate port GUID and remove it.

### F.2.3.10 Duplicate NodeGuid for Node HFI1 nodeType[1-3] guid 0x00066a00d9000143 and existing node[x] nodeType=1-3, HFI2, guid 0x00066a00d9000143

#### SM Area

Discovery.

#### Meaning

A duplicate Node Guid in fabric.

#### Action

Using fabric tools, locate the device with the duplicate node GUID and remove it.

### F.2.3.11 Marking port[x] of node[x] HFI1 guid 0x00066a00d9000143 DOWN in the topology

#### SM Area

Discovery.

#### Meaning

port x of HFI1 is not responding.

#### Action

Check the health of node HFI1.

### F.2.3.12 Failed to init SL2SC/SC2SL Map (setting port down) on node HFI1/sw1 nodeGuid 0x00066a00d9000143 node index X port index Y

#### SM Area

Discovery.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric. Persistence indicates that the node may be having problems.

#### Action

Check the health of node HFI1/Sw1 if the fabric was idle or persistent condition.

### F.2.3.13 Failed to init VL Arb (setting port down) on node HFI1/Sw1 nodeGuid 0x00066a00d9000143 node index X port index Y

#### SM Area

Discovery.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric. Persistence indicates that the node may be having problems.

#### Action

Check the health of node HFI1/Sw1 if the fabric was idle or persistent condition.

### F.2.3.14 TT(ta): can't ARM/ACTIVATE node HFI1/sw1 guid 0x00066a00d9000143 node index X port index Y

#### SM Area

Discovery.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric. Persistence indicates that the node may be having problems.

#### Action

Check the health of node HFI1/Sw1 if the fabric was idle or persistent condition.

### F.2.3.15 sa_xxxxx: Reached size limit at X records

#### SM Area

Administrator.

#### Meaning

The response buffer is too large.

#### Action

Contact support.

### F.2.3.16 sa_NodeRecord_Set: NULL PORTGUID for Node Guid[0x00066a00d9000143], HFI1, Lid 0x1

#### SM Area

Administrator.

#### Meaning

Possible data corruption.

#### Action

Contact support.

### F.2.3.17 sa_TraceRecord: destination port is not in active state; port LID: 0x1 (port GUID 0x00066a00d9000144)

#### SM Area

Administrator.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric.

#### Action

Check the health of the destination if the condition persists.

### F.2.3.18 sa_TraceRecord: Cannot find path to port LID 0x2 (port guid 0x00066a00d9000144) from port LID 0x1 (port guid 0x00066a00d9000143)

#### SM Area

Administrator.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric.

#### Action

Check for the next 3 messages.

### F.2.3.19 sa_TraceRecord_Fill: Reached size limit while processing TRACE_RECORD request

#### SM Area

Administrator.

#### Meaning

The response buffer is too large.

#### Action

Contact support.

### F.2.3.20 sa_PathRecord: NULL PORTGUID in Source/Destination Gid 0xFE80000000000000:0000000000000000 of PATH request from Lid 0x1

#### SM Area

Administrator.

#### Meaning

Invalid data in the SA request.

#### Action

Check the health of the requester at lid 0x1.

### F.2.3.21 sa_PathRecord: Cannot find path to port LID 0x2 (port guid 0x00066a00d9000144) from port LID 0x1 (port guid 0x00066a00d9000143)

#### SM Area

Administrator.

#### Meaning

May be caused by simultaneous removal/insertion events in the fabric.

#### Action

Check the health of the destination lid 0x2.

### F.2.3.22 sa_PathRecord: Cannot find path to port LID 0x2 (port guid 0x00066a00d9000144) from port LID 0x1 (port guid 0x00066a00d9000143) with pkey 0x800d

#### SM Area

Administrator.

#### Meaning

A path does not exist in the partition with the given PKey between the given source and destination.

#### Action

Check configuration to determine if path should exist in given pkey. Check the health of the destination lid 0x2 if configuration is valid.

### F.2.3.23 sa_PathRecord: port LID 0x1 (port guid 0x00066a00d9000143) not a member of multicast group 0xff12401bffff0000:00000000ffffffff

#### SM Area

Administrator.

#### Meaning

Group may have just been deleted or the requester is not a member of the group.

#### Action

None.

### F.2.3.24 sa_McMemberRecord_Set: Port GID in request (0x0080000000000000:0x0000000000000000)from HFI1, Port 0x00066a00d9000143, LID 0x1 has a NULL GUID/invalid prefix, returning status 0x0500

#### SM Area

Administrator.

#### Meaning

Invalid data in the SA request.

#### Action

Check the health of HFI1.

### F.2.3.25 sa_McMemberRecord_Set: MTU selector of 2 with MTU of 4 does not work with realizable MTU of 1 for request from compute-0-24, Port 0x00066A00A00005C5, LID 0x009C, returning status 0x0200

#### SM Area

Administrator.

#### Meaning

The requester port data is not compatible with the group data.

#### Action

Create group at lowest common denominator or host should join with rate selector of "less than" rather than "exactly".

### F.2.3.26 sa_McMemberRecord_Set: Rate selector of 2 with Rate of 3 does not work with realizable Rate of 2 for request from compute-0-24, Port 0x00066A00A00005C5, LID 0x009C, returning status 0x0200

#### SM Area

Administrator.

#### Meaning

Node compute-0-24 has requested a port rate that is incompatible with the group rate.

#### Action

Check that the requester port is not running at 1X width or that the multicast group was not created with a rate greater than what some the host ports can support.

### F.2.3.27 sa_McMemberRecord_Set: Component mask (0x00000000000XXXXX) does not have bits required to create (0x00000000000130C6) a group for new MGID of 0xFF12401BFFFF0000:00000000FFFFFFFF for request from HFI1, Port 0x00066a00d9000143

#### SM Area

Administrator.

#### Meaning

End node may be trying to join a group that does not exist.

#### Action

OpenIB and Sun stacks require that the broadcast group be pre-created by the SM.

### F.2.3.28 sa_McMemberRecord_Set: Component mask of 0x0000000000010083 does not have bits required

### (0x00000000000130C6) to CREATE a new group in request from HFI1, Port 0x00066a00d9000143

#### SM Area

Administrator.

#### Meaning

Specific bits must be set in a CREATE group request.

#### Action

The requester is violating the protocol specification.

## F.2.3.29 sa_McMemberRecord_Set: Bad (limited member) PKey of 0x1234 for request from ibhollab54 HFI-1, Port 0x00066a00d9000143, LID 0x1, returning status 0x200

#### SM Area

Administrator.

#### Meaning

The PKey specified in request was limited, it should be full.

#### Action

Check configuration.

## F.2.3.30 sa_McMemberRecord_Set: MC group create request denied for node ibhollab54 HFI-1, port 0x00066a00d9000144 from lid 0x2, failed VF validation (mgid=0xFF12401BFFFF0000:0x0000000000000016, sl=x, pkey=0xnnnn)

#### SM Area

Administrator.

#### Meaning

An attempt to create an mcast group failed due to validation failures.

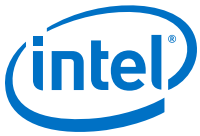
#### Action

Check configuration if create by source should be valid.

## F.2.3.31 sa_McMemberRecord_Set: Invalid MGID (0xFF270000FFFF0000:00000000FFFFFFFF) in CREATE/JOIN request from HFI1, Port 0x00066a00d9000143, LID 0x0001, returning status 0x0500

#### SM Area

Administrator.



#### Meaning

MGID requested violating the protocol specification.

#### Action

The requester is violating the protocol specification.

### F.2.3.32 sa_McMemberRecord_Set: Join state of 0x1-2 not full member for NULL/NEW GID request from HFI1, Port 0x00066a00d9000143, LID 0x0001, returning status 0x0200

#### SM Area

Administrator.

#### Meaning

Creation of a Multicast Group requires FULL membership.

#### Action

The requester is violating the protocol specification.

### F.2.3.33 sa_McMemberRecord_Set: Join state of ~0x1 not full member for request to CREATE existing MGID of 0xFF12401BFFFF0000:00000000FFFFFFFF

#### SM Area

Administrator.

#### Meaning

Creation of a Multicast Group requires FULL membership.

#### Action

The requester is violating the protocol specification.

### F.2.3.34 sa_McMemberRecord_Set: Component mask of 0x00000000000XXXXX does not have bits required (0x0000000000010083) to JOIN group with MGID 0xFF12401BFFFF0000:00000000FFFFFFFF in request from %s, Port 0x%.16"CS64"X, LID 0x%.4X, returning status 0x%.4X

#### SM Area

Administrator.

#### Meaning

Specific bits must be set in a JOIN group request.

#### Action

The requester is violating the protocol specification.

### F.2.3.35 sa_McMemberRecord_Set: Maximum number groups reached (1000), failing CREATE request from HFI1, Port 0x00066a00d9000143, LID 0x0011, returning status 0x0100

#### SM Area

Administrator.

#### Meaning

No resources.

#### Action

Delete some of the multicast groups or configure the SM to overload MLIDs during a group creation.

### F.2.3.36 sa_McMemberRecord_Set: Failed to assign GID for CREATE request from HFI1, Port 0x00066a00d9000143, LID 0x0001, returning status 0x0100

#### SM Area

Administrator.

#### Meaning

No resources.

#### Action

Delete some of the multicast groups or configure the SM to overload the MLIDs during group creation.

### F.2.3.37 sa_McMemberRecord_Set: No multicast LIDs available for request from HFI1, Port 0x00066a00d9000143, LID 0x0001, returning status 0x0100

#### SM Area

Administrator.

#### Meaning

No resources.

#### Action

Delete some of the multicast groups or configure the SM to overload the MLIDs during group creation.

### F.2.3.38 sa_McMemberRecord_Set: MGID 0xFF12401BFFFF0000:00000000FFFFFFFF does not exist; Failing JOIN request from HFI1, Port 0x00066a00d9000143, LID 0x0001, returning status 0x0200

#### SM Area

Administrator.

#### Meaning

JOIN of a group that does not exist.

#### Action

OpenIB and Sun stacks require that the broadcast group be pre-created by SM.

### F.2.3.39 sa_McMemberRecord_Set: Qkey/Pkey of 0x1234 does not match group QKey of 0x4321 for group 0xFF12401BFFFF0000:00000000FFFFFFFF for request from HFI1, Port 0x00066a00d9000143, LID 0x0001, returning status 0x0200

#### SM Area

Administrator.

#### Meaning

SM may have been set to create the default broadcast group with parameters not valid for the fabric.

#### Action

Reconfigure the default broadcast group with the proper parameters.

### F.2.3.40 sa_McMemberRecord_Set: Group MTU of 5 greater than requester port mtu of 2/4 for group 0xFF12401BFFFF0000:00000000FFFFFFFF for request from HFI1, Port 0x00066a00d9000143, LID 0x0001, returning status 0x0200

#### SM Area

Administrator.

#### Meaning

SM may have been set to create the default broadcast group with parameters not valid for fabric.

#### Action

Reconfigure the default broadcast group with the proper parameters.

### F.2.3.41 sa_McMemberRecord_Set: Group Rate/MTU of X is too low/high for requested rate/mtu of Y, rate/mtu selector of 2, and port rate/mtu of

### Z for group 0xFF12401BFFFF0000:00000000FFFFFFFF in request from HFI1

#### SM Area

Administrator.

#### Meaning

The SM may have been set to create a default broadcast group with parameters not valid for a fabric or a host has created the group at a RATE not supported by other hosts.

#### Action

Create the group at lowest common denominator or the host should join with rate selector of "less than" rather "exactly".

## F.2.3.42 sa_InformInfo: Subscription for security trap not from trusted source[lid=0x0001], smkey=0x0, returning status 0x0200

#### SM Area

Administrator.

#### Meaning

Requester using wrong smkey.

#### Action

Make sure to use the same SMkey as what is configured in the SM.

## F.2.3.43 sm_resolve_pkeys_for_vfs: VFabric has application SA selected, bad PKey configured 0x1, must use Default PKey.",

#### SM Area

Administrator.

#### Meaning

The SA select is limited to Virtual Fabrics using the Default PKey 0x7fff.

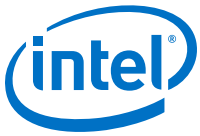
#### Action

Configuration change needed for SA Select.

## F.2.3.44 sm_resolve_pkeys_for_vfs: Virtual Fabric VF0013 MulticastGroup configuration error, MGID does not match app, disabling Default Group

#### SM Area

Administrator.



#### Meaning

The Multicast Group has an MGID configured that does not match any application that is part of this Virtual Fabric.

#### Action

Configuration change needed for mcast group creation.

### F.2.3.45 sm_resolve_pkeys_for_vfs: Virtual Fabric VF0013 MulticastGroup configuration error, mismatch on pkey. Disabling Default Group

#### SM Area

Administrator.

#### Meaning

The MulticastGroup linked to this Virtual Fabric do not share a common pkey. Disabling the mcast group for this vFabric.

#### Action

Configuration change required if mcast group default creation is needed.

### F.2.3.46 sm_initialize_port/sm_dbsync: cannot refresh sm pkeys

#### SM Area

Administrator.

#### Meaning

An internal error occurred when attempting to refresh the SM Pkeys.

#### Action

Contact customer support if condition persists.

### F.2.3.47 sa_ServiceRecord_Add: Failed to ADD serviced record ID=0x1000000000003531 from lid 0x2 due to invalid pkey

#### SM Area

Administrator.

#### Meaning

Serviced record add failure due to request with invalid PKey

#### Action

Configuration change required if request should be granted.

### F.2.3.48 sa_ServiceRecord_Add: Failed to ADD serviced record ID=0x1000000000003531 from lid 0x2 due to pkey

### F.2.3.49 mismatch from request port

#### SM Area

Administrator.

#### Meaning

Serviced record add failure due to request with PKey not shared by requestor.

#### Action

Configuration change required if request should be granted.

### F.2.3.50 sa_ServiceRecord_Add: Failed to ADD serviced record ID=0x1000000000003531 from lid 0x2 due to pkey mismatch from service port

#### SM Area

Administrator.

#### Meaning

Serviced record add failure due to request with PKey not shared by requestor.

#### Action

Configuration change required if request should be granted.

# Appendix G INSTALL Command

## Syntax

```
./INSTALL [-r root] [-v|-vv] [-a|-n|-U|-u|-s|-O|-N|- i comp|-e comp] [-E comp]
[-D comp] [--user_configure_options 'options']
[--kernel_configure_options 'options'] [--prefix dir] [--without-depcheck]
[--rebuild] [--force] [--answer keyword=value]
```

or

```
./INSTALL -C
```

or

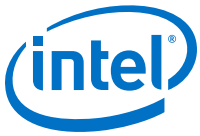
```
./INSTALL -V
```

*Note:* To access help for this command, type `./INSTALL -?` and press **Enter**.

## Options

| | |
|---|---|
| `-a` | Install all ULPs and drivers with default options. |
| `-n` | Install all ULPs and drivers with default options but with no change to autostart options. |
| `-U` | Upgrade/re-install all presently installed ULPs and drivers with default options and no change to autostart options. |
| `-i` *comp* | Install the given component with default options. Can appear more than once on command line. |
| `--user_configure_options` '*options*' | Specify additional OpenFabrics Alliance* (OFA) build options for user space srpms. Causes rebuild of all user srpms. |
| `--kernel_configure_options` '*options*' | Specify additional OFA build options for driver srpms. Causes rebuild of all driver srpms. |
| `--prefix` *dir* | Specify alternate directory prefix for install of OFA. Default is `/usr`. Causes rebuild of needed srpms. |
| `--without-depcheck` | Disable check of OS dependencies. |

| | |
|---|---|
| `--rebuild` | Force OFA Delta rebuild. |
| `--force` | Force installation even if distributions do not match.<br><br>*Note:* Using this option can result in undefined behaviors. |
| `-O` | Keep current modified rpm config file. |
| `-N` | Use new default rpm config file. |
| `-u` | Uninstall all ULPs and drivers with the default options. |
| `-s` | Enable autostart for all installed drivers. |
| `-r` | Specify alternate root directory. Default is `/`. |
| `-e` *comp* | Uninstall the given component with default options. Can appear more than once on command line. |
| `-E` *comp* | Enable autostart of given component. This option can appear with `-D` or multiple times on a command line. |
| `-D` *comp* | Disable autostart of given component. This option can appear with `-E` or multiple times on command line. |
| `-v` | Outputs with verbose logging. |
| `-vv` | Outputs with very verbose debug logging. |
| `-C` | Outputs the list of supported component names.<br><br>Supported components include: `opa_stack ibacm mpi_selector intel_hfi oftools opa_stack_dev fastfabric delta_ipoib opafm mvapich2_gcc_hfi mvapich2_intel_hfi openmpi_gcc_hfi openmpi_intel_hfi gasnet openshmem sandiashmem mvapich2 openmpi delta_mpisrc delta_debug`<br><br>Supported component name aliases include: `opa ipoib mpi psm_mpi verbs_mpi pgas mpisrc opadev` |



| | |
|---|---|
| `-V` | Outputs the version number of the software. |
| `--user_queries` | Permit non-root users to query the fabric. (Default) |
| `--no_user_queries` | Non-root users cannot query the fabric. |
| `--answer` *`keyword=value`* | Provides an answer to a question that might occur during the operation. Answers to questions that are not asked are ignored. For interactive installs, invalid answers result in prompts. For non-interactive installs, the default is used. Possible questions include:<br><br>`UserQueries` – Permit non-root users to query the fabric. |

# Appendix H Setting up Pre-Defined Topology Verification Security

This section describes how to set up the Pre-Defined Topology Verification security feature, and the two basic operating modes.

In general, Intel recommends using the FastFabric TUI and the punch-list verification process to administer and verify a cluster ***before enabling*** Pre-Defined Topology Verification feature. The FastFabric TUI is described in *Verifying the Host* section in the *Intel® Omni-Path Fabric Suite FastFabric User Guide*.

After the cluster connectivity has been verified, you can create the Pre-Defined topology input configuration file. There are multiple FastFabric tools to assist in creating a topology input file. See *Intel® Omni-Path Fabric Suite FastFabric User Guide* (opaxlattopology, opagentopology, opareport).

There are two types of input configuration files:

- Based on node GUIDs and port numbers
- Based on node Descriptions and port numbers.

## H.1 Pre-Defined Topology Verification Based on Node GUIDs and Port Numbers

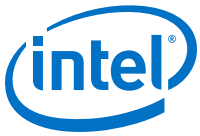
Pre-Defined Topology Verification based on node GUIDs and port numbers is considered "more secure" because node GUIDs cannot be modified by software. For network bootable nodes, this topology configuration is highly recommended. The main issue with a network boot is that the node descriptions may change as the OS is booted, and unless the node description remains constant through the booting process, the use of constant node GUID is required so nodes are not incorrectly quarantined.

Node replacement may be more tedious as the node GUID would need to be updated in the topology file and the FM restarted before new nodes would be permitted to join the cluster.

## H.2 Pre-Defined Topology Verification based on Node Descriptions and Port Numbers

Pre-Defined Topology Verification based on node descriptions is considered "less secure" because node Descriptions can be modified by software. But using node Descriptions in topology definitions facilitates easy node replacement as the new node only needs to have its node description updated; and no FM restart is required.

Node/link lookup and validation is performed with Node Description and Port Number only if `NodeGUID` field enforcement level is Disabled. Node/link lookup by Node Description, however, is independent of the `NodeDescription` field enforcement level.



Intel recommends unique node descriptions so that the link resolves exactly to one pair. Otherwise, it is possible that a link may resolved to a similarly named link if node descriptions are replicated.

## H.3 Node Replacement with Pre-Defined Topology Verification

This section describes how to replace an HFI when using the Pre-Defined Topology Verification feature.

For topology files based on node descriptions, the HFI replacement process is straightforward: Update the node description of the replacement node prior to adding to the cluster. No restart of FM is required, and the node will pass verification.

For topology files based on node GUIDs and port numbers (see Pre-Defined Topology Verification based on Node Descriptions and Port Numbers on page 228), an FM Restart is required.

1. The node to be replaced should have its node description updated to the correct name prior to adding the node to the cluster.
2. Modify the input topology configuration file, replacing the old node GUID with the new node GUID. An alternative way to find the correct GUIDs to be replaced is to search the topology file for the replacement node description (assuming unique node descriptions).

   *Note:* This modification must be done on every topology configuration file on each FM in the cluster. Otherwise the FMs will detect this inconsistency upon restart, and will not participate in FM redundancy.
3. Add the replacement node back to the cluster and restart all FMs. The newly replaced node should activate. If it does not, examine the logs for warnings indicating why the node did not activate; it may have been quarantined due to other reasons.

# Appendix I Core-Level Public Key Infrastructure (PKI) Best Practices Guidelines

## I.1 Overview

The usage of the TLS protocol via OpenSSL by the Fabric Manager, FastFabric, and Fabric Manager GUI application components of the Intel® Omni-Path Fabric Suite software package requires that a private key, public key, and certificate be obtained for each of these components.

The guidelines presented in this section are intended to provide network administrators and other qualified personnel a reference for the basic administration tasks required to support private keys, public keys, and certificates using the OpenSSL command set.

### Assumptions

- Target audience - network administrators and other qualified personnel with root users access
- OS environment - Linux
- OpenSSL software supporting TLSv1.2 has been installed
- Intel® Omni-Path Fabric Suite software package has been installed

### Abbreviations and Definitions

*<intel-opa-comp>*: The Fabric Manager, FastFabric, or Fabric Manager GUI application component

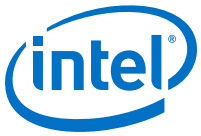
## I.2 Applying for a Certificate for an Intel® Omni-Path Fabric Suite Software Package Component

The Fabric Manager, FastFabric, and Fabric Manager GUI application components of the Intel® Omni-Path Fabric Suite software package use the Ephemeral Elliptic Curve Diffie-Hellman (ECDHE) and Digital Signature Algorithm (DSA) algorithms to perform mutual authentication of each during the authentication and key exchange/agreement handshake phase of TLS.

This requires that a Private/Public key pair and certificate are obtained for each of these components. This section describes the steps required for each component.

### I.2.1 Step 1: Set up the OPA Component OpenSSL Configuration File

1. Create a work directory for the *<intel-opa-comp>*.
2. Move to the *<intel-opa-comp>* work directory.



3. OpenSSL has a default configuration file (`openssl.cnf`) that is used for OPA component certificate requests and other related tasks. The Intel® Omni-Path Fabric Suite software package provides a sample `openssl.cnf` configuration file (`opa_comp_openssl.cnf-sample`, installed in directories `/usr/lib/opa/samples` and `/usr/share/opa-fm/samples`) for an Intel® Omni-Path component. This configuration file can be modified for the unique requirements of the customer. Copy this sample configuration file to the work directory:

```
cp /usr/share/opa/samples/opa_comp_openssl.cnf-sample opa_comp_openssl.cnf
```

4. Use your favorite editor to edit the `opa_comp_openssl.cnf` configuration file. Modify at least the following fields marked in bold to meet your unique requirements.

```
#
# Intel OPA FM/FF Component OpenSSL example configuration file.
# This is mostly being used for generation of certificate requests.
#

# This definition stops the following lines choking if HOME isn't
# defined.
HOME            = .
RANDFILE        = $ENV::HOME/.rnd

# Extra OBJECT IDENTIFIER info:
#oid_file        = $ENV::HOME/.oid
##oid_section        = new_oids

# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions        =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)

[ req ]
default_bits        = 2048
default_md        = sha256
distinguished_name    = opa_comp_distinguished_name
attributes        = req_attributes
prompt                        = no

# Passwords for private keys if not present they will be prompted for
# input_password = secret
# output_password = secret

# This sets a mask for permitted string types. There are several options.
# default: PrintableString, T61String, BMPString.
# pkix     : PrintableString, BMPString (PKIX recommendation before 2004)
# utf8only: only UTF8Strings (PKIX recommendation after 2004).
# nombstr : PrintableString, T61String (no BMPStrings or UTF8Strings).
# MASK:XXXX a literal mask value.
# WARNING: ancient versions of Netscape crash on BMPStrings or UTF8Strings.
string_mask = utf8only

req_extensions = v3_req # The extensions to add to a certificate request

[ opa_comp_distinguished_name ]
countryName            = Country Name (2 letter code)
#countryName_default        = US
#countryName_min        = 2
#countryName_max        = 2

stateOrProvinceName        = State or Province Name (full name)
#stateOrProvinceName_default    = Default Province
```

```
localityName            = Locality Name (eg, city)
#localityName_default        = Default City

organizationName        = Organization Name (eg, company)
#organizationName_default    = Default Company Ltd

organizationalUnitName        = Organizational Unit Name (eg, section)
#organizationalUnitName_default    =

commonName            = Common Name (eg, your name or your server\'s hostname)
#commonName_max            = 64

emailAddress            = Email Address
#emailAddress_max        = 64

# SET-ex3            = SET extension number 3

[ req_attributes ]
#challengePassword        = A challenge password
#challengePassword_min        = 4
#challengePassword_max        = 20

unstructuredName        = An optional company name

[ v3_req ]

# Extensions to add to a certificate request

basicConstraints        = CA:FALSE
keyUsage                = nonRepudiation, digitalSignature, keyEncipherment
###############################################################################
```

### I.2.2 Step 2: Create Private Key

#### I.2.2.1 HSM - Elliptic Curve (EC) Private Key

1. Direct the OpenSSL environment to use the customized `opa_comp_openssl.cnf` configuration file used for an Intel® OP Software component:

```
export OPENSSL_CONF/<path>~/<intel-opa-comp>/opa_comp_openssl.cnf
```

2. Execute the following OpenSSL command to generate the EC parameters for the component:

```
openssl ecparam -out <intel-opa-comp>-ec-param.pem -name prime256v1
```

3. Execute the following OpenSSL command to validate the content of the EC parameters:

```
openssl ecparam -in <intel-opa-comp>-ec-param.pem -noout -text
```

Output:

```
ASN1 OID: prime256v1
```

4. Execute the following OpenSSL command to generate the EC Private Key for the component (note, without Pass Phrase):

```
openssl ecparam -in <intel-opa-comp>-ec-param.pem -genkey -out <intel-opa-
comp-ecdsa-priv-key>.pem
```

5. Execute the following OpenSSL command to validate the content of the EC Private Key:

```
openssl ecparam -in <intel-opa-comp-ecdsa-priv-key>.pem -noout -text
```

Output:

```
ASN1 OID: prime256v1
```

### I.2.2.2 ESM - DSA Private Key

1. Direct the OpenSSL environment to use the customized `opa_comp_openssl.cnf` configuration file used for an Intel® OP Software component:

```
export OPENSSL_CONF=~/opaCA/opa_comp_openssl.cnf
```

2. Execute the following OpenSSL command to generate the DSA parameters for the component:

```
openssl dsaparam -out <intel-opa-comp-dsa-param>.pem 2048
```

Output:

```
Generating DSA parameters, 2048 bit long prime
This could take some time
.+.+..........................+.................+............+......+.+.....
+.....+..................++++++++++++++++++++++++++++++++++++++++++++++++++++++*
.+..+.....................+.......+..................+.+..+........
+..............+...+...........+.+.......................+..........
+...............+.+..+.....+.......................
+..........................+......+.......................
+.................+..+..+.............+.....+...+..............................
+......+...+..................+..............+..................+........
+..............+..+...+........+.......+.+...........+..+...........+..
+............+.................+.+.....................+.+.......+..+.......+.
+..................+...............+...+..+.......+..............+...+........
+......+..+...+.....................+.......+...+.+..................
+..........+.........+..+..+....+.+...............+...........+...
+............+..+.................+.....+.....+.+..+...
+.......................+.....................+........
+....................+...+..+.....+.........+.....+..........+...+...........
+...+.......................+...+..+..+.................+...
+..................+..............+.............................+.....
+..............+...+.........+.+.......+..+.............................+..
+...................+......+.+..............+.+............+......+.+....+..
+......+......+......+..+.....+...+.........+.....+.....+......+....+.+........
+.....+.......................+.................+..........+..............
+...........+.........+.+.+.............+..+.+...+......+.+.....................
+..........+.+.+.........................+................................+.....
+..............+.....+.+.....................................................+...
+.........+...........+............+......+.....+.....+..+.........+..
+............+...+........+............+...+..............+........
+..........+..+.+.....+..+....+......+.....+............+....+.....
```

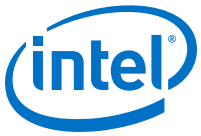
```
+............+........+.............+.........+...........
+.......................+....+........+...........+..+.....
+...........................+......+.............................
+....................+.....+.............+.....+.....+......+.+.
+............+..+................................+..+........+...+.+......+..
+.....................................+.+.....+............
+.....................+.+........+..........+...................+.......+.....
+...........+..............+.........+.................+....+......
+...................+.....................................+.............+....
+...+.........+.........................................+....+..+.............
+...........+......+..........+.....................+.........+...
+...................+....+..+...+..+......+............+............
+............+.+..............+........+............+.....+...+.+........
+..............+...+..........+.....................+.........+.........
+..............+..+....+.............+....................+...+.
+.....................................+...++++++++++++++++++++++++++++++++++++
++++++++++++++*
```

3. Execute the following OpenSSL command to validate the content of the DSA parameters:

```
openssl dsaparam -noout -text -in <intel-opa-comp-dsa-param>.pem
```

Output:

```
P:
        00:ff:51:07:30:85:b8:54:32:09:43:4b:85:e0:d3:
        00:60:23:7a:61:f6:7d:dd:5a:d4:09:9b:c4:89:da:
        dc:35:59:47:a6:5f:03:4c:f9:42:cc:f6:0c:50:32:
        4e:87:c1:1b:8d:8a:44:57:53:50:24:73:c7:25:bf:
        5b:c9:e4:d6:fe:8d:3f:f0:1a:c2:2b:1a:53:14:44:
        48:1b:dc:9d:b8:b0:84:54:4f:5d:92:35:e8:00:27:
        2b:1c:06:96:34:e0:09:60:05:94:d6:e5:06:2c:7e:
        2a:cf:be:fc:e6:65:c7:c0:87:12:f1:07:22:c4:e4:
        d6:a7:9c:71:e1:58:40:61:23:9d:a7:d3:93:b4:a9:
        ab:5c:a0:e8:cd:e0:d1:f3:6f:cb:1d:5a:ad:3a:dc:
        ee:ce:80:bb:c0:59:12:ad:70:89:bf:58:6c:5e:f1:
        41:ee:0c:f3:0c:49:7f:99:dc:11:9c:25:d9:23:a0:
        f6:c4:74:c5:7f:80:df:fd:70:2a:ac:63:dc:d7:b3:
        3e:07:24:b3:67:44:45:e5:6b:a9:f0:90:ea:13:3c:
        32:90:8e:5c:ac:fb:05:1b:1c:01:5a:62:ae:dd:0e:
        cc:ff:74:35:11:8c:b9:d7:40:aa:7b:46:59:2f:45:
        b8:59:b9:4d:85:a5:8a:62:4f:2e:cf:b2:16:62:be:
        ae:c9
    Q:
        00:95:06:74:47:f2:8f:aa:ab:c8:d3:6c:9d:6e:ef:
        cf:3f:ba:a7:97:9b:2c:4d:dc:1c:7e:04:78:78:87:
        ad:63:cd
    G:
        00:cc:0b:af:53:7c:5f:5c:2f:77:7d:ca:ec:d3:42:
        df:f4:79:6e:2e:43:52:59:f5:c4:25:bf:46:ef:b5:
        b5:7b:67:00:f9:64:ff:d8:ac:28:69:2f:99:b6:40:
        ce:2c:86:ac:8b:c8:0c:70:52:f0:d7:32:de:eb:68:
        01:8f:bc:89:fe:e4:83:55:4a:27:ba:63:3d:51:1c:
        ec:dc:a7:73:0b:9c:83:4e:4f:00:5d:17:7f:9f:a0:
        3c:c4:1c:85:6c:b4:90:15:85:a2:c3:43:c1:19:3a:
        5b:18:82:4d:8f:15:9b:1a:fd:d7:c7:32:a8:6e:07:
        7f:5d:a7:7f:b1:e8:9b:78:4d:5f:3a:96:66:e0:d5:
        a2:fa:ca:c3:56:09:23:1e:fe:0e:e0:41:43:d7:a1:
        b3:9a:e7:08:90:86:82:56:79:4d:b4:ed:5d:27:e8:
        49:d2:6d:86:ca:78:65:96:5e:2f:72:b7:f9:20:44:
        c2:37:54:c0:14:b6:4f:67:10:29:50:04:34:01:e1:
        1f:bd:ee:48:79:21:44:b2:8e:cb:5f:9e:09:74:fa:
        29:aa:45:49:b8:a7:2a:ed:81:9c:ff:d1:fc:27:2f:
        86:db:25:59:7c:fc:57:8b:e7:0f:e3:f0:f5:70:49:
        20:0c:06:1b:d5:aa:52:a1:93:38:d6:34:03:99:a7:
        c2:32
```



4. Execute the following OpenSSL command to generate the DSA Private Key for the component (note, without Pass Phrase):

```
openssl gendsa -out <intel-opa-comp-dsa-priv-key>.pem <intel-opa-comp-dsa-
param>.pem
```

Output:

```
Generating DSA key, 2048 bits
```

5. Execute the following OpenSSL command to validate the content of the DSA Private Key:

```
openssl dsa -in <intel-opa-comp-dsa-priv-key>.pem -noout -text
```

Output:

```
read DSA key
Private-Key: (2048 bit)
priv:
    43:b1:a6:1d:e4:fa:73:a4:db:18:0f:48:68:9e:29:
    f4:c5:06:9b:66:8c:b8:d1:a3:b0:f8:e9:59:e0:44:
    af:a3
pub:
    00:bc:98:c5:bd:ac:e5:a7:ff:05:c1:d3:a3:28:c0:
    30:fd:d1:79:af:ba:e4:94:88:f4:5b:3e:49:04:68:
    21:2e:64:8b:05:d7:c9:48:ac:3b:b9:d5:ad:29:f1:
    90:d5:9e:8e:bd:b4:26:c1:90:f8:ad:1e:af:63:2c:
    9b:de:68:57:fc:6b:ee:1d:7c:99:68:98:70:15:dc:
    14:02:e5:c2:64:58:e4:aa:2b:2c:60:41:49:6d:c9:
    e3:fe:49:83:36:a1:b9:fd:6f:4a:16:1b:7a:99:2d:
    1b:2f:78:7d:e9:a5:fa:ad:6e:27:e6:2c:8b:db:01:
    3e:d6:8f:e5:d8:fc:26:26:6b:81:b7:0d:1e:fd:a5:
    ba:fe:e8:eb:02:4f:31:95:05:36:c0:c4:8e:01:10:
    65:85:a8:bb:5d:7b:22:0d:d1:f6:2d:41:f1:16:1d:
    57:84:4d:a2:ed:9b:da:d7:2c:db:ee:bc:2c:fa:6c:
    2e:03:30:e5:a8:f5:3e:d2:5c:4f:e2:0b:9f:eb:15:
    0e:1c:8b:f5:4e:2f:c4:eb:15:b8:aa:83:85:54:9a:
    0e:54:38:68:9a:19:7e:1a:98:a8:d0:46:06:29:69:
    7b:0c:40:a8:4d:b3:38:b5:70:2b:67:fd:a4:cf:a4:
    14:f3:08:0c:7d:78:4d:a0:71:eb:17:c8:53:ba:62:
    b9:33
P:
    00:d3:33:23:28:7f:87:d2:db:29:14:94:18:66:0f:
    a9:e8:c6:0c:53:18:3f:a7:bb:af:0d:e5:76:5a:6b:
    20:23:ea:3a:62:bd:b5:43:d5:13:f8:4c:40:00:c4:
    b3:2b:5a:20:ad:45:52:16:9b:c2:11:df:ad:83:9c:
    f2:5c:9c:94:73:fe:31:08:a7:b1:cb:42:0b:67:48:
    26:14:4c:c3:f9:3c:2f:99:e7:c3:ef:2f:07:94:d4:
    ac:c1:00:34:f8:74:71:8d:fb:62:65:39:fa:9c:13:
    5b:a6:12:f0:0f:9b:75:6d:fa:ca:1e:01:c7:7a:89:
    9f:d4:bc:88:9d:e4:6e:b9:f6:6d:ae:03:64:59:4b:
    c4:56:90:68:57:91:06:c5:20:3e:a1:c7:c9:85:e7:
    52:81:6d:bd:2d:29:2b:f2:46:db:5b:4d:1e:cf:61:
    d7:df:88:82:85:8c:5b:5e:01:c2:e3:3f:f7:23:00:
    92:84:af:8e:c1:06:2d:e2:c9:60:ab:26:23:82:a6:
    66:9e:1e:33:ec:37:7e:89:3f:fd:41:38:c9:cb:20:
    10:67:64:ff:72:8e:00:ef:50:51:b4:66:3c:1c:46:
    61:11:58:19:f3:bb:8a:1e:03:b8:27:13:2b:6a:6c:
    1f:07:df:40:31:8e:30:c6:71:33:4e:de:5d:e4:05:
    01:c7
Q:
    00:96:05:f8:54:97:82:ff:1d:18:31:1e:cc:86:c2:
    c0:76:fe:2e:8b:15:57:bb:c8:61:d0:ae:f8:ef:f0:
    78:b7:2b
```

```
G:
    4c:8a:b5:71:cc:8b:9b:b0:b2:a8:3e:3f:ac:3d:21:
    e1:4a:ff:47:40:33:2a:ed:de:5c:f6:40:db:ee:60:
    02:77:57:53:9b:9e:50:b5:94:7b:c1:c9:54:2e:0f:
    e0:29:68:0c:15:20:df:41:ee:e0:0c:af:61:11:8b:
    bb:ef:8c:91:6f:c4:70:d1:1d:52:1c:f7:ac:dd:af:
    a3:fe:86:38:35:b9:6c:78:0e:5d:c6:58:9e:97:10:
    45:d6:b4:0e:26:24:93:8d:4e:4f:9e:e2:e5:12:d4:
    8a:5c:54:fc:7c:05:49:9b:5e:0b:09:b7:84:4d:ba:
    22:4c:54:9b:0c:3c:47:79:6e:85:c0:55:ac:e5:8b:
    22:1a:ad:62:31:ed:dc:64:35:79:b0:05:19:dc:17:
    b9:a4:93:bc:ac:a1:e6:e3:31:ed:e7:e3:ff:57:fb:
    07:a5:87:50:16:bd:01:98:b8:82:8d:c9:09:83:0c:
    e4:a1:24:08:ba:03:a9:d7:04:01:84:d3:22:9d:da:
    87:88:db:ec:7f:77:e1:f9:18:6b:20:01:22:2b:44:
    28:3f:d9:50:28:20:3e:fc:f1:4e:ac:92:68:e7:c1:
    76:42:1d:47:6a:5a:48:bd:6c:71:14:ac:3b:0d:e0:
    5d:11:34:e5:0d:9c:8a:35:fa:78:77:7f:f6:77:a6:
    d9
```

6. Execute the following OpenSSL command to generate the corresponding DSA Public Key for the component:

```
openssl dsa -in <intel-opa-comp-dsa-priv-key>.pem -pubout -out <intel-opa-
comp-dsa-pub-key>.pem
```

Output:

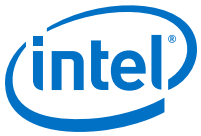
```
read DSA key
writing DSA key
```

7. Execute the following OpenSSL command to validate the content of the DSA Public Key:

```
openssl
dsa -pubin -in <intel-opa-comp-dsa-pub-key>.pem -noout
-text
```

Output:

```
read DSA key
pub:
    00:bc:98:c5:bd:ac:e5:a7:ff:05:c1:d3:a3:28:c0:
    30:fd:d1:79:af:ba:e4:94:88:f4:5b:3e:49:04:68:
    21:2e:64:8b:05:d7:c9:48:ac:3b:b9:d5:ad:29:f1:
    90:d5:9e:8e:bd:b4:26:c1:90:f8:ad:1e:af:63:2c:
    9b:de:68:57:fc:6b:ee:1d:7c:99:68:98:70:15:dc:
    14:02:e5:c2:64:58:e4:aa:2b:2c:60:41:49:6d:c9:
    e3:fe:49:83:36:a1:b9:fd:6f:4a:16:1b:7a:99:2d:
    1b:2f:78:7d:e9:a5:fa:ad:6e:27:e6:2c:8b:db:01:
    3e:d6:8f:e5:d8:fc:26:26:6b:81:b7:0d:1e:fd:a5:
    ba:fe:e8:eb:02:4f:31:95:05:36:c0:c4:8e:01:10:
    65:85:a8:bb:5d:7b:22:0d:d1:f6:2d:41:f1:16:1d:
    57:84:4d:a2:ed:9b:da:d7:2c:db:ee:bc:2c:fa:6c:
    2e:03:30:e5:a8:f5:3e:d2:5c:4f:e2:0b:9f:eb:15:
    0e:1c:8b:f5:4e:2f:c4:eb:15:b8:aa:83:85:54:9a:
    0e:54:38:68:9a:19:7e:1a:98:a8:d0:46:06:29:69:
    7b:0c:40:a8:4d:b3:38:b5:70:2b:67:fd:a4:cf:a4:
    14:f3:08:0c:7d:78:4d:a0:71:eb:17:c8:53:ba:62:
    b9:33
P:
    00:d3:33:23:28:7f:87:d2:db:29:14:94:18:66:0f:
    a9:e8:c6:0c:53:18:3f:a7:bb:af:0d:e5:76:5a:6b:
    20:23:ea:3a:62:bd:b5:43:d5:13:f8:4c:40:00:c4:
    b3:2b:5a:20:ad:45:52:16:9b:c2:11:df:ad:83:9c:
```



```
    f2:5c:9c:94:73:fe:31:08:a7:b1:cb:42:0b:67:48:
    26:14:4c:c3:f9:3c:2f:99:e7:c3:ef:2f:07:94:d4:
    ac:c1:00:34:f8:74:71:8d:fb:62:65:39:fa:9c:13:
    5b:a6:12:f0:0f:9b:75:6d:fa:ca:1e:01:c7:7a:89:
    9f:d4:bc:88:9d:e4:6e:b9:f6:6d:ae:03:64:59:4b:
    c4:56:90:68:57:91:06:c5:20:3e:a1:c7:c9:85:e7:
    52:81:6d:bd:2d:29:2b:f2:46:db:5b:4d:1e:cf:61:
    d7:df:88:82:85:8c:5b:5e:01:c2:e3:3f:f7:23:00:
    92:84:af:8e:c1:06:2d:e2:c9:60:ab:26:23:82:a6:
    66:9e:1e:33:ec:37:7e:89:3f:fd:41:38:c9:cb:20:
    10:67:64:ff:72:8e:00:ef:50:51:b4:66:3c:1c:46:
    61:11:58:19:f3:bb:8a:1e:03:b8:27:13:2b:6a:6c:
    1f:07:df:40:31:8e:30:c6:71:33:4e:de:5d:e4:05:
    01:c7
Q:
    00:96:05:f8:54:97:82:ff:1d:18:31:1e:cc:86:c2:
    c0:76:fe:2e:8b:15:57:bb:c8:61:d0:ae:f8:ef:f0:
    78:b7:2b
G:
    4c:8a:b5:71:cc:8b:9b:b0:b2:a8:3e:3f:ac:3d:21:
    e1:4a:ff:47:40:33:2a:ed:de:5c:f6:40:db:ee:60:
    02:77:57:53:9b:9e:50:b5:94:7b:c1:c9:54:2e:0f:
    e0:29:68:0c:15:20:df:41:ee:e0:0c:af:61:11:8b:
    bb:ef:8c:91:6f:c4:70:d1:1d:52:1c:f7:ac:dd:af:
    a3:fe:86:38:35:b9:6c:78:0e:5d:c6:58:9e:97:10:
    45:d6:b4:0e:26:24:93:8d:4e:4f:9e:e2:e5:12:d4:
    8a:5c:54:fc:7c:05:49:9b:5e:0b:09:b7:84:4d:ba:
    22:4c:54:9b:0c:3c:47:79:6e:85:c0:55:ac:e5:8b:
    22:1a:ad:62:31:ed:dc:64:35:79:b0:05:19:dc:17:
    b9:a4:93:bc:ac:a1:e6:e3:31:ed:e7:e3:ff:57:fb:
    07:a5:87:50:16:bd:01:98:b8:82:8d:c9:09:83:0c:
    e4:a1:24:08:ba:03:a9:d7:04:01:84:d3:22:9d:da:
    87:88:db:ec:7f:77:e1:f9:18:6b:20:01:22:2b:44:
    28:3f:d9:50:28:20:3e:fc:f1:4e:ac:92:68:e7:c1:
    76:42:1d:47:6a:5a:48:bd:6c:71:14:ac:3b:0d:e0:
    5d:11:34:e5:0d:9c:8a:35:fa:78:77:7f:f6:77:a6:
    d9
```

### I.2.3 Step 3: Create a Certificate Request (CSR)

1. Execute the following OpenSSL command to generate a CSR for the component (note, the Public Key will be generated and will be inserted into the CSR):

```
openssl req -new -key <intel-opa-comp-priv-key>.pem -out <intel-opa-comp-
csr>.pem
```

2. Execute the following OpenSSL command to validate the content of the CSR (note, the Public Key is in the CSR):

```
openssl req -in <intel-opa-comp-csr>.pem -text -noout
```

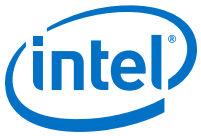
#### I.2.3.1 HSM - EC Signed CSR Example

```
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=US, ST=Pennsylvania, L=King of Prussia, O=OPA Component Sample
Corporation, OU=OPA Component Sample Division, CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:8e:85:df:04:ac:36:81:7f:6a:09:58:8a:de:a1:
```

```
                    7a:ab:b2:00:34:e0:d3:9f:92:2a:da:d9:6c:40:58:
                    00:46:b0:55:15:4c:60:dd:55:40:46:23:13:5c:65:
                    30:26:01:3d:a8:73:86:6c:38:fc:9b:e4:45:14:70:
                    20:e4:2c:b1:84
                ASN1 OID: prime256v1
        Attributes:
            unstructuredName         :unable to print attribute
        Requested Extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage:
                Digital Signature, Non Repudiation, Key Encipherment
    Signature Algorithm: ecdsa-with-SHA256
         30:45:02:21:00:c7:1e:cc:4f:b9:97:fb:8d:a5:4b:c3:fd:ea:
         68:23:3b:43:45:88:b4:3f:63:7e:e3:2e:0a:90:02:07:2d:c0:
         43:02:20:11:6f:b3:dc:33:53:a3:7c:14:d9:11:7e:9f:eb:dc:
         2a:27:b8:00:cf:76:69:5b:ef:62:95:d1:c5:10:d9:8d:4e
```

### I.2.3.2 ESM - DSA Signed CSR Example

```
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=US, ST=Pennsylvania, L=King of Prussia, O=OPA Component Sample
Corporation, OU=OPA Component Sample Division, CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Subject Public Key Info:
            Public Key Algorithm: dsaEncryption
                pub:
                    20:54:44:6a:53:70:f3:11:3e:1f:a1:1d:bb:9e:c0:
                    5e:1d:49:bd:17:b5:ba:2b:e5:6d:36:41:8c:c6:3e:
                    32:47:f6:ed:81:8f:d3:2b:1a:bb:ff:63:e1:8d:19:
                    1d:df:93:f4:a0:32:69:25:4c:09:7a:89:87:01:4b:
                    db:7b:ee:2f:49:f0:f4:98:70:df:f0:a1:61:71:b5:
                    29:cd:7c:35:4b:4c:0c:eb:48:ad:2b:04:fc:d2:fc:
                    e5:f1:6e:2e:55:74:0f:5d:b6:54:15:04:bd:17:4e:
                    76:24:e8:8a:40:20:dd:df:4c:60:93:8a:ab:ee:02:
                    ed:eb:9b:82:4a:8a:85:4c
                P:
                    00:dd:0f:41:f9:58:46:21:7c:2b:a1:36:76:1a:d5:
                    d0:ae:57:5f:67:2f:b3:a7:30:f0:f2:e3:0a:c8:04:
                    0f:cc:97:01:5f:fa:67:f3:38:83:44:72:16:69:fe:
                    30:5d:e1:18:b5:ff:a3:93:f3:0b:11:ab:26:57:53:
                    e3:8c:1f:d7:0f:e5:fb:ec:89:9f:02:be:a3:dc:c3:
                    26:c4:af:c2:c6:24:e8:76:a5:c1:ec:d9:18:50:31:
                    a1:61:4f:0e:2e:67:05:e9:20:4b:07:13:d7:eb:d9:
                    21:81:f5:50:a0:2b:cc:18:a2:47:79:73:01:ae:de:
                    3c:0c:a0:73:31:51:be:bb:e9
                Q:
                    00:dd:eb:0e:b6:34:28:8f:d8:78:da:fa:a2:ca:25:
                    6f:af:08:27:fc:81
                G:

                    00:90:ee:ee:e2:e7:59:b9:30:7f:ab:d5:b9:8f:75:
                    20:d7:04:33:ef:72:6d:3c:c2:70:c4:6e:a5:ec:26:


ed:e9:a3:66:b3:b1:54:49:0d:b0:50:7f:f2:51:bc:
                    31:58:f8:ae:a2:cb:fc:9d:6b:42:59:01:0d:5f:7b:
                    77:d8:fd:2e:fb:88:b4:b5:42:60:a4:e9:60:88:3f:
                    df:c4:fb:bc:4e:3e:72:c1:8e:49:44:93:2e:64:d0:
                    42:0a:a7:37:1b:c4:d6:69:50:13:41:2a:58:39:64:
                    3f:45:47:f7:86:ca:b1:b7:66:61:9f:03:05:4e:c5:
                    4d:42:bf:6a:8e:43:e8:6b:75
        Attributes:
            unstructuredName         :unable to print attribute
        Requested Extensions:
            X509v3 Basic Constraints:
```



```
                CA:FALSE
            X509v3 Key Usage:
                Digital Signature, Non Repudiation, Key Encipherment
    Signature Algorithm: dsa_with_SHA256
         r:
             00:ab:96:34:ae:4e:36:4f:50:91:b8:c9:3d:b7:33:
             bd:32:93:bf:5c:f7
         s:
             6d:ef:71:e6:60:53:f8:c1:29:39:7e:35:62:76:1d:
             16:4e:62:c2:a6
```

### I.2.4 Step 4: Submit CSR to Third-party CA

1. Send CSR fee payment and the *<intel-opa-comp-csr>*.csr to the CA.
2. CA processes the *<intel-opa-comp-csr>*.csr and creates a X.509 certificate *<intel-opa-comp-cert>*.pem in PEM format.
3. CA signs and sends you the X.509 certificate *<intel-opa-comp-cert>*.pem.

### I.2.5 Step 5: Receive Certificate (X.509) from Third-party CA

1. Upon reception of the X.509 certificate for the component, execute the following OpenSSL command to validate the content of the certificate:

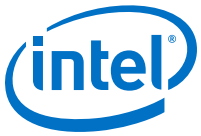
```
openssl x509 in <intel-opa-comp-cert>.pem noout text
```

#### I.2.5.1 HSM - EC Signed Certificate (X.509) Example

```
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=US, ST=Pennsylvania, L=King of Prussia, O=OPA Component Sample
Corporation, OU=OPA Component Sample Division, CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:8e:85:df:04:ac:36:81:7f:6a:09:58:8a:de:a1:
                    7a:ab:b2:00:34:e0:d3:9f:92:2a:da:d9:6c:40:58:
                    00:46:b0:55:15:4c:60:dd:55:40:46:23:13:5c:65:
                    30:26:01:3d:a8:73:86:6c:38:fc:9b:e4:45:14:70:
                    20:e4:2c:b1:84
                ASN1 OID: prime256v1
        Attributes:
            unstructuredName         :unable to print attribute
        Requested Extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage:
                Digital Signature, Non Repudiation, Key Encipherment
    Signature Algorithm: ecdsa-with-SHA256
         30:45:02:21:00:c7:1e:cc:4f:b9:97:fb:8d:a5:4b:c3:fd:ea:
         68:23:3b:43:45:88:b4:3f:63:7e:e3:2e:0a:90:02:07:2d:c0:
         43:02:20:11:6f:b3:dc:33:53:a3:7c:14:d9:11:7e:9f:eb:dc:
         2a:27:b8:00:cf:76:69:5b:ef:62:95:d1:c5:10:d9:8d:4e
```

### I.2.5.2 ESM - DSA Signed Certificate (X.509) Example

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 2 (0x2)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: CN=phgppriv03.ph.intel.com, ST=PA, C=US/
emailAddress=root@tradeshowhell.com, O=Intel STL FE TLS Prototype, OU=Intel TCG
        Validity
            Not Before: Oct 29 16:39:54 2014 GMT
            Not After : Oct 28 16:39:54 2019 GMT
        Subject: CN=server.sample.com, ST=Pennsylvania, C=US/
emailAddress=john.doe@sample.com, O=Sample Corporation, OU=Sample Division
        Subject Public Key Info:
            Public Key Algorithm: dsaEncryption
                pub:
                    5b:5c:ab:5d:28:0a:2c:6e:16:cc:ab:f0:a6:c7:f0:
                    36:fa:8b:e1:0f:27:2f:c0:7d:ba:7d:5c:b6:f3:61:
                    f6:e8:cc:7b:29:e3:f9:a0:38:88:07:04:36:d7:b7:
                    9d:4a:2d:38:19:f0:c5:e2:f4:24:59:43:91:2d:a7:
                    c2:0d:22:ed:80:a6:cd:60:ea:23:b9:0c:13:60:04:
                    6b:32:8c:6a:bf:c4:83:a3:25:6d:d5:6a:f6:3b:4e:
                    55:29:bd:af:ef:e5:cd:04:52:f4:e4:59:bf:33:86:
                    2b:86:40:6c:08:9a:e5:17:8a:6a:f4:d6:07:86:0e:
                    65:13:10:75:e1:61:5f:c5:b8:42:34:10:eb:bc:ad:
                    12:d7:a8:62:a4:e2:31:1d:b3:51:aa:59:08:24:8f:
                    ea:6f:ac:29:f3:40:d0:b5:4a:e9:42:5a:f8:39:b1:
                    3f:04:d9:db:a4:c1:84:58:16:59:a1:93:81:3b:f8:
                    4e:89:9f:9b:dd:11:76:40:da:99:65:5e:02:d1:ce:
                    8b:a6:fe:fe:38:9e:69:36:e3:45:ed:db:14:d5:52:
                    0d:8c:34:bf:e2:3c:a9:9f:d5:6f:5c:78:a7:91:fb:
                    59:16:bf:71:dc:29:cd:de:95:cc:bf:be:6a:8c:dc:
                    4a:79:03:7e:b7:7c:71:ee:31:cf:7c:27:42:4e:6a:
                    24
                P:
                    00:ff:51:07:30:85:b8:54:32:09:43:4b:85:e0:d3:
                    00:60:23:7a:61:f6:7d:dd:5a:d4:09:9b:c4:89:da:
                    dc:35:59:47:a6:5f:03:4c:f9:42:cc:f6:0c:50:32:
                    4e:87:c1:1b:8d:8a:44:57:53:50:24:73:c7:25:bf:
                    5b:c9:e4:d6:fe:8d:3f:f0:1a:c2:2b:1a:53:14:44:
                    48:1b:dc:9d:b8:b0:84:54:4f:5d:92:35:e8:00:27:
                    2b:1c:06:96:34:e0:09:60:05:94:d6:e5:06:2c:7e:
                    2a:cf:be:fc:e6:65:c7:c0:87:12:f1:07:22:c4:e4:
                    d6:a7:9c:71:e1:58:40:61:23:9d:a7:d3:93:b4:a9:
                    ab:5c:a0:e8:cd:e0:d1:f3:6f:cb:1d:5a:ad:3a:dc:
                    ee:ce:80:bb:c0:59:12:ad:70:89:bf:58:6c:5e:f1:
                    41:ee:0c:f3:0c:49:7f:99:dc:11:9c:25:d9:23:a0:
                    f6:c4:74:c5:7f:80:df:fd:70:2a:ac:63:dc:d7:b3:
                    3e:07:24:b3:67:44:45:e5:6b:a9:f0:90:ea:13:3c:
                    32:90:8e:5c:ac:fb:05:1b:1c:01:5a:62:ae:dd:0e:
                    cc:ff:74:35:11:8c:b9:d7:40:aa:7b:46:59:2f:45:
                    b8:59:b9:4d:85:a5:8a:62:4f:2e:cf:b2:16:62:be:
                    ae:c9
                Q:
                    00:95:06:74:47:f2:8f:aa:ab:c8:d3:6c:9d:6e:ef:
                    cf:3f:ba:a7:97:9b:2c:4d:dc:1c:7e:04:78:78:87:
                    ad:63:cd
                G:
                    00:cc:0b:af:53:7c:5f:5c:2f:77:7d:ca:ec:d3:42:
                    df:f4:79:6e:2e:43:52:59:f5:c4:25:bf:46:ef:b5:
                    b5:7b:67:00:f9:64:ff:d8:ac:28:69:2f:99:b6:40:
                    ce:2c:86:ac:8b:c8:0c:70:52:f0:d7:32:de:eb:68:
                    01:8f:bc:89:fe:e4:83:55:4a:27:ba:63:3d:51:1c:
                    ec:dc:a7:73:0b:9c:83:4e:4f:00:5d:17:7f:9f:a0:
                    3c:c4:1c:85:6c:b4:90:15:85:a2:c3:43:c1:19:3a:
                    5b:18:82:4d:8f:15:9b:1a:fd:d7:c7:32:a8:6e:07:
                    7f:5d:a7:7f:b1:e8:9b:78:4d:5f:3a:96:66:e0:d5:
                    a2:fa:ca:c3:56:09:23:1e:fe:0e:e0:41:43:d7:a1:
                    b3:9a:e7:08:90:86:82:56:79:4d:b4:ed:5d:27:e8:
```



```
                    49:d2:6d:86:ca:78:65:96:5e:2f:72:b7:f9:20:44:
                    c2:37:54:c0:14:b6:4f:67:10:29:50:04:34:01:e1:
                    1f:bd:ee:48:79:21:44:b2:8e:cb:5f:9e:09:74:fa:
                    29:aa:45:49:b8:a7:2a:ed:81:9c:ff:d1:fc:27:2f:
                    86:db:25:59:7c:fc:57:8b:e7:0f:e3:f0:f5:70:49:
                    20:0c:06:1b:d5:aa:52:a1:93:38:d6:34:03:99:a7:
                    c2:32
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
    Signature Algorithm: sha1WithRSAEncryption
        76:55:e1:b1:db:c5:95:4b:38:5d:8e:90:a0:6b:a4:a6:e5:8f:
        91:15:02:b8:a8:2b:8f:f6:2b:ba:9e:ba:bc:bd:db:c8:04:98:
        9c:69:45:cb:50:ea:39:f2:b3:68:bf:39:24:4c:1f:25:96:38:
        7f:2c:78:6f:8b:5d:05:58:80:11:70:50:a1:37:f0:58:cd:62:
        e2:36:9b:1b:29:9d:f2:c9:97:52:7d:1b:a6:28:76:2b:c6:1d:
        96:e3:bc:34:a9:09:6a:1c:e2:27:cf:0f:a1:d4:2c:49:db:b7:
        84:e7:79:59:a3:92:5a:40:aa:ae:73:b5:fa:ba:6a:fd:99:be:
        c2:ff:09:68:e1:51:68:48:e9:05:cb:74:6e:cb:2a:ca:54:8f:
        97:22:cc:57:11:81:8c:3e:ff:71:b3:fb:84:fc:29:db:67:ba:
        34:a3:98:24:8e:1e:dc:7e:82:9d:72:a7:c5:e3:48:2e:2f:e6:
        aa:81:27:7f:f2:42:94:b4:9c:fc:11:85:5b:a3:d9:2b:f1:f7:
        55:5d:21:2f:ac:e1:25:62:85:45:a0:44:39:1a:62:37:16:40:
        b4:92:24:8a:0e:0f:08:f5:1a:77:10:30:08:02:34:91:3d:f7:
        e5:6f:00:e3:8c:2a:11:ec:36:2d:41:2f:ae:8a:e2:10:6a:74:
        57:cf:86:31
```

## I.2.6 Step 6: Install Private Key and Certificate

### I.2.6.1 FM Component

#### I.2.6.1.1 HSM

1. Rename the Private-Key *<intel-opa-comp-priv-key>*.pem of the FM component to the name configured for the opafm.xml parameter <SslSecurityFmPrivateKey> (default parameter value: fm_key.pem):

```
mv <intel-opa-comp-priv-key>.pem fm_key.pem
```

2. Copy the Private Key fm_key.pem to the directory configured for the opafm.xml parameter <SslSecurityDir> (default parameter value: /usr/local/ssl/opafm/)

```
cp fm_key.pem /usr/local/ssl/opafm/
```

3. Rename the certificate *<intel-opa-comp-cert>*.pem of the FM to the name configured for the opafm.xml parameter <SslSecurityFmCertificate> (default parameter value: fm_cert.pem):

```
mv <intel-opa-comp-cert>.pem fm_cert.pem
```

4. Copy the certificate fm_cert.pem to the directory configured for the opafm.xml parameter <SslSecurityDir>:

```
cp fm_cert.pem /usr/local/ssl/opafm/
```

#### I.2.6.1.2 ESM

1. Establish a SFTP session to the managed chassis:

```
sftp admin@<managed-chassis-ip-address>
```

2. Rename the Private-Key *<intel-opa-comp-priv-key>*.pem of the FM component to the name configured for the opafm.xml parameter <SslSecurityFmPrivateKey> (default parameter value: fm_key.pem):

```
mv <intel-opa-comp-priv-key>.pem  fm_key.pem
```

3. SFTP the Private-Key fm_key.pem to the directory configured for the opafm.xml parameter <SslSecurityDir> (default parameter value: /mmc0:4/)

```
sftp> put fm_key.pem /mmc0:4/
```

4. Rename the certificate *<intel-opa-comp-cert>*.pem of the FM to the name configured for the opafm.xml parameter <SslSecurityFmCertificate> (default parameter value: fm_cert.pem):

```
mv <intel-opa-comp-cert>.pem  fm_cert.pem
```

5. SFTP the certificate fm_cert.pem to the directory configured for the opafm.xml parameter <SslSecurityDir>:

```
sftp> put fm_cert.pem /mmc0:4/
```

6. Exit the SFTP session:

```
sftp> exit
```

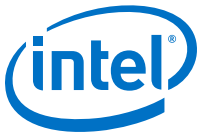
### I.2.6.2 FF Component

1. Rename the Private Key *<intel-opa-comp-priv-key>*.pem of the FF component to the name configured for the opaff.xml parameter <SslSecurityFFPrivateKey> (default parameter value: ff_key.pem):

```
mv <intel-opa-comp-priv-key>.pem  ff_key.pem
```

2. Copy the Private Key ff_key.pem to the directory configured for the opaff.xml parameter <SslSecurityDir> (default parameter value: /usr/local/ssl/opafm/)

```
cp ff_key.pem /usr/local/ssl/opafm/
```



3. Rename the certificate `<intel-opa-comp-cert>.pem` of the FF to the name configured for the `opaff.xml` parameter `<SslSecurityFFCertificate>` (default parameter value: `ff_cert.pem`):

```
mv <intel-opa-comp-cert>.pem  ff_cert.pem
```

4. Copy the certificate `ff_cert.pem` to the directory configured for the `opaff.xml` parameter `<SslSecurityDir>`:

```
cp ff_cert.pem /usr/local/ssl/opafm/
```

#### I.2.6.3 Fabric Manager GUI Component

Installation of the Private Key and certificate for the Fabric Manager GUI component is not addressed in these Guidelines. Reference the PKI Best Practices Guidelines for the Fabric Manager GUI document.

## I.3 Third-party CA Certificate for an Intel® Omni-Path Fabric Suite Software Package Component

The third-party CA used to obtain certificates for the Fabric Manager, FastFabric, or Fabric Manager GUI components of the Intel® Omni-Path Fabric Suite software package publishes its own certificate. The certificate of the CA is required to verify the certificates issued by that CA.

The Fabric Manager, FastFabric, or Fabric Manager GUI components do not support downloading the CA certificate.

It is the responsibility of the administrator(s) to perform this task prior to starting the Fabric Manager, FastFabric, or Fabric Manager GUI components. The administrator must do the following procedures to set up the CA certificate.

*Note:* To minimize the administrative effort, this task could be done once and the same CA certificate used for all components.

### I.3.1 Step 1: Download Third-party CA Certificate

1. Execute the appropriate steps published by the third-party CA to download the CA certificate `<intel-opa-comp-ca-cert>.pem` to the local Linux server environment.
2. Execute the following OpenSSL command to validate the content of the CA certificate `<intel-opa-comp-ca-cert>.pem`:

```
openssl x509 -in <intel-opa-comp-ca-cert>.pem -noout -text
```

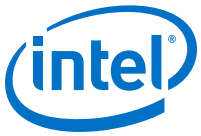
#### I.3.1.1 HSM - EC Signed Certificate Example

```
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=US, ST=Pennsylvania, L=King of Prussia, O=OPA Component Sample
```

```
Corporation, OU=OPA Component Sample Division, CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:8e:85:df:04:ac:36:81:7f:6a:09:58:8a:de:a1:
                    7a:ab:b2:00:34:e0:d3:9f:92:2a:da:d9:6c:40:58:
                    00:46:b0:55:15:4c:60:dd:55:40:46:23:13:5c:65:
                    30:26:01:3d:a8:73:86:6c:38:fc:9b:e4:45:14:70:
                    20:e4:2c:b1:84
                ASN1 OID: prime256v1
        Attributes:
            unstructuredName         :unable to print attribute
        Requested Extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage:
                Digital Signature, Non Repudiation, Key Encipherment
    Signature Algorithm: ecdsa-with-SHA256
         30:45:02:21:00:c7:1e:cc:4f:b9:97:fb:8d:a5:4b:c3:fd:ea:
         68:23:3b:43:45:88:b4:3f:63:7e:e3:2e:0a:90:02:07:2d:c0:
         43:02:20:11:6f:b3:dc:33:53:a3:7c:14:d9:11:7e:9f:eb:dc:
         2a:27:b8:00:cf:76:69:5b:ef:62:95:d1:c5:10:d9:8d:4e
```

### I.3.1.2 ESM - DSA Signed Certificate Example

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 2 (0x2)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: CN=phgppriv03.ph.intel.com, ST=PA, C=US/
emailAddress=root@tradeshowhell.com, O=Intel STL FE TLS Prototype, OU=Intel TCG
        Validity
            Not Before: Oct 29 16:39:54 2014 GMT
            Not After : Oct 28 16:39:54 2019 GMT
        Subject: CN=server.sample.com, ST=Pennsylvania, C=US/
emailAddress=john.doe@sample.com, O=Sample Corporation, OU=Sample Division
        Subject Public Key Info:
            Public Key Algorithm: dsaEncryption
                pub:
                    5b:5c:ab:5d:28:0a:2c:6e:16:cc:ab:f0:a6:c7:f0:
                    36:fa:8b:e1:0f:27:2f:c0:7d:ba:7d:5c:b6:f3:61:
                    f6:e8:cc:7b:29:e3:f9:a0:38:88:07:04:36:d7:b7:
                    9d:4a:2d:38:19:f0:c5:e2:f4:24:59:43:91:2d:a7:
                    c2:0d:22:ed:80:a6:cd:60:ea:23:b9:0c:13:60:04:
                    6b:32:8c:6a:bf:c4:83:a3:25:6d:d5:6a:f6:3b:4e:
                    55:29:bd:af:ef:e5:cd:04:52:f4:e4:59:bf:33:86:
                    2b:86:40:6c:08:9a:e5:17:8a:6a:f4:d6:07:86:0e:
                    65:13:10:75:e1:61:5f:c5:b8:42:34:10:eb:bc:ad:
                    12:d7:a8:62:a4:e2:31:1d:b3:51:aa:59:08:24:8f:
                    ea:6f:ac:29:f3:40:d0:b5:4a:e9:42:5a:f8:39:b1:
                    3f:04:d9:db:a4:c1:84:58:16:59:a1:93:81:3b:f8:
                    4e:89:9f:9b:dd:11:76:40:da:99:65:5e:02:d1:ce:
                    8b:a6:fe:fe:38:9e:69:36:e3:45:ed:db:14:d5:52:
                    0d:8c:34:bf:e2:3c:a9:9f:d5:6f:5c:78:a7:91:fb:
                    59:16:bf:71:dc:29:cd:de:95:cc:bf:be:6a:8c:dc:
                    4a:79:03:7e:b7:7c:71:ee:31:cf:7c:27:42:4e:6a:
                    24
                P:
                    00:ff:51:07:30:85:b8:54:32:09:43:4b:85:e0:d3:
                    00:60:23:7a:61:f6:7d:dd:5a:d4:09:9b:c4:89:da:
                    dc:35:59:47:a6:5f:03:4c:f9:42:cc:f6:0c:50:32:
                    4e:87:c1:1b:8d:8a:44:57:53:50:24:73:c7:25:bf:
                    5b:c9:e4:d6:fe:8d:3f:f0:1a:c2:2b:1a:53:14:44:
                    48:1b:dc:9d:b8:b0:84:54:4f:5d:92:35:e8:00:27:
                    2b:1c:06:96:34:e0:09:60:05:94:d6:e5:06:2c:7e:
```



```
                    2a:cf:be:fc:e6:65:c7:c0:87:12:f1:07:22:c4:e4:
                    d6:a7:9c:71:e1:58:40:61:23:9d:a7:d3:93:b4:a9:
                    ab:5c:a0:e8:cd:e0:d1:f3:6f:cb:1d:5a:ad:3a:dc:
                    ee:ce:80:bb:c0:59:12:ad:70:89:bf:58:6c:5e:f1:
                    41:ee:0c:f3:0c:49:7f:99:dc:11:9c:25:d9:23:a0:
                    f6:c4:74:c5:7f:80:df:fd:70:2a:ac:63:dc:d7:b3:
                    3e:07:24:b3:67:44:45:e5:6b:a9:f0:90:ea:13:3c:
                    32:90:8e:5c:ac:fb:05:1b:1c:01:5a:62:ae:dd:0e:
                    cc:ff:74:35:11:8c:b9:d7:40:aa:7b:46:59:2f:45:
                    b8:59:b9:4d:85:a5:8a:62:4f:2e:cf:b2:16:62:be:
                    ae:c9
                Q:
                    00:95:06:74:47:f2:8f:aa:ab:c8:d3:6c:9d:6e:ef:
                    cf:3f:ba:a7:97:9b:2c:4d:dc:1c:7e:04:78:78:87:
                    ad:63:cd
                G:
                    00:cc:0b:af:53:7c:5f:5c:2f:77:7d:ca:ec:d3:42:
                    df:f4:79:6e:2e:43:52:59:f5:c4:25:bf:46:ef:b5:
                    b5:7b:67:00:f9:64:ff:d8:ac:28:69:2f:99:b6:40:
                    ce:2c:86:ac:8b:c8:0c:70:52:f0:d7:32:de:eb:68:
                    01:8f:bc:89:fe:e4:83:55:4a:27:ba:63:3d:51:1c:
                    ec:dc:a7:73:0b:9c:83:4e:4f:00:5d:17:7f:9f:a0:
                    3c:c4:1c:85:6c:b4:90:15:85:a2:c3:43:c1:19:3a:
                    5b:18:82:4d:8f:15:9b:1a:fd:d7:c7:32:a8:6e:07:
                    7f:5d:a7:7f:b1:e8:9b:78:4d:5f:3a:96:66:e0:d5:
                    a2:fa:ca:c3:56:09:23:1e:fe:0e:e0:41:43:d7:a1:
                    b3:9a:e7:08:90:86:82:56:79:4d:b4:ed:5d:27:e8:
                    49:d2:6d:86:ca:78:65:96:5e:2f:72:b7:

f9:
20:44:
                    c2:37:54:c0:14:b6:4f:67:10:29:50:04:34:01:e1:
                    1f:bd:ee:48:79:21:44:b2:8e:cb:5f:9e:09:74:fa:
                    29:aa:45:49:b8:a7:2a:ed:81:9c:ff:d1:fc:27:2f:
                    86:db:25:59:7c:fc:57:8b:e7:0f:e3:f0:f5:70:49:
                    20:0c:06:1b:d5:aa:52:a1:93:38:d6:34:03:99:a7:
                    c2:32
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE


    Signature Algorithm: sha1WithRSAEncryption
        76:55:e1:b1:db:c5:95:4b:38:5d:8e:90:a0:6b:a4:a6:e5:8f:
        91:15:02:b8:a8:2b:8f:f6:2b:ba:9e:ba:bc:bd:db:c8:04:98:
        9c:69:45:cb:50:ea:39:f2:b3:68:bf:39:24:4c:1f:25:96:38:
        7f:2c:78:6f:8b:5d:05:58:80:11:70:50:a1:37:f0:58:cd:62:
        e2:36:9b:1b:29:9d:f2:c9:97:52:7d:1b:a6:28:76:2b:c6:1d:
        96:e3:bc:34:a9:09:6a:1c:e2:27:cf:0f:a1:d4:2c:49:db:b7:
        84:e7:79:59:a3:92:5a:40:aa:ae:73:b5:fa:ba:6a:fd:99:be:
        c2:ff:09:68:e1:51:68:48:e9:05:cb:74:6e:cb:2a:ca:54:8f:
        97:22:cc:57:11:81:8c:3e:ff:71:b3:fb:84:fc:29:db:67:ba:
        34:a3:98:24:8e:1e:dc:7e:82:9d:72:a7:c5:e3:48:2e:2f:e6:
        aa:81:27:7f:f2:42:94:b4:9c:fc:11:85:5b:a3:d9:2b:f1:f7:
        55:5d:21:2f:ac:e1:25:62:85:45:a0:44:39:1a:62:37:16:40:
        b4:92:24:8a:0e:0f:08:f5:1a:77:10:30:08:02:34:91:3d:f7:
        e5:6f:00:e3:8c:2a:11:ec:36:2d:41:2f:ae:8a:e2:10:6a:74:
        57:cf:86:31
```

### I.3.2 Step 2: Install Third-party CA Certificate

#### I.3.2.1 FM Component

#### I.3.2.1.1 HSM

1. Copy the CA certificate *<intel-opa-comp-ca-cert>*.pem to the name configured for the opafm.xml parameter <SslSecurityFmCaCertificate> (default parameter value: fm_ca_cert.pem):

```
cp <intel-opa-comp-ca-cert>.pem  fm_ca_cert.pem
```

2. Copy the CA certificate fm_ca_cert.pem to the directory configured for the opafm.xml parameter <SslSecurityDir> (default parameter value: /usr/local/ssl/opafm/):

```
cp fm_ca_cert.pem /usr/local/ssl/opafm/
```

#### I.3.2.1.2 ESM

1. Establish a SFTP session to the managed chassis:

```
sftp admin@<managed-chassis-ip-address>
```

2. Copy the CA certificate *<intel-opa-comp-ca-cert>*.pem to the name configured for the opafm.xml parameter <SslSecurityFmCaCertificate> (default parameter value: fm_ca_cert.pem):

```
cp <intel-opa-comp-ca-cert>.pem  fm_ca_cert.pem
```

3. SFTP the CA certificate fm_ca_cert.pem to the directory configured for the opafm.xml parameter <SslSecurityDir> (default parameter value: /mmc0:4/)

```
sftp> put fm_ca_cert.pem /mmc0:4/
```

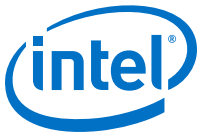
4. Exit the SFTP session:

```
sftp> exit
```

### I.3.2.2 FF Component

1. Copy the CA certificate *<intel-opa-comp-ca-cert>*.pem to the name configured for the opaff.xml parameter <SslSecurityFFCaCertificate> (default parameter value: ff_ca_cert.pem):

```
cp <intel-opa-comp-ca-cert>.pem  ff_ca_cert.pem
```



2. Copy the CA certificate `ff_ca_cert.pem` to the directory configured for the `opafm.xml` parameter `<SslSecurityDir>`:

```
cp ff_ca_cert.pem /usr/local/ssl/opafm/
```

### I.3.2.3 Fabric Manager GUI Component

For instructions for installing the third-party certificate for the Fabric Manager GUI component, refer to SSL Key Creation on page 274.

## I.4 Diffie-Hellman Parameters for an Intel® Omni-Path Fabric Suite Software Package Component

The Fabric Manager, FastFabric, and Fabric Manager GUI components of the Intel® Omni-Path Fabric Suite software package use the Ephemeral Diffie-Hellman (DHE), Ephemeral Elliptic Curve Diffie-Hellman (ECDHE), and Digital Signature Algorithm (DSA) algorithms to perform mutual authentication of each during the authentication and key-exchange/agreement handshake phase of TLS.

The ECDHE algorithm requires the generation of a set of parameters that are used during the key-exchange/agreement handshake phase of TLS. The generation of these parameters can be time consuming during runtime. For performance reasons, the Fabric Manager, FastFabric, and Fabric Manager GUI components of the Intel® Omni-Path Fabric Suite software package do not support the generation of these parameters during the key-exchange/agreement handshake phase of TLS.

It is the responsibility of the administrator(s) to perform this task prior to starting the Fabric Manager, FastFabric, and Fabric Manager GUI components. The administrator must do the following procedures to set up the Diffie-Hellman(DH) parameters.

*Note:* To minimize the administrative effort, this task could be done once and the same DH parameters used for all components.

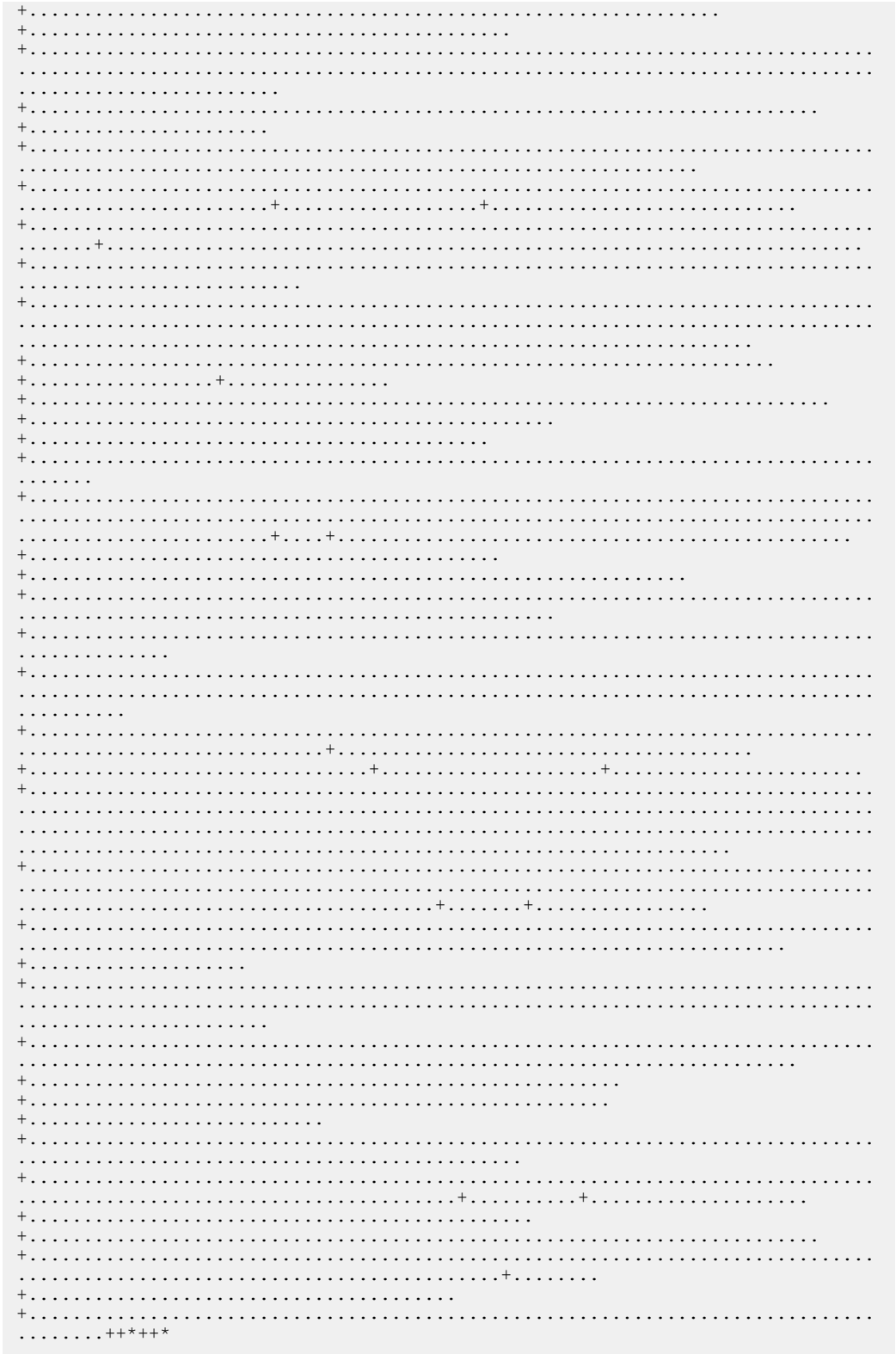
### I.4.1 Step 1: Generate Diffie-Hellman Parameters

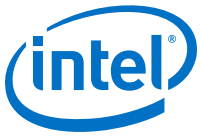
1. Execute the following OpenSSL command to generate the Diffie-Hellman parameters:

```
openssl dhparam -out <intel-opa-comp-dh-param>.pem 2048
```

Output:

```
Generating DH parameters, 2048 bit long safe prime, generator 2
This is going to take a long time
.....................................................................
.....................................................................
.....................................................
+.....................................................................
.....................................................................
.....................................................
+.....................................................................
.....................................................................
.....................................................................
.....................................................
```





2. Execute the following OpenSSL command to validate the Diffie-Hellman parameters:

```
openssl dhparam -in <intel-opa-comp-dh-param>.pem -noout -text
```

Output:

```
PKCS#3 DH Parameters: (2048 bit)
        prime:
            00:d6:8f:0d:9f:f8:eb:ea:65:d8:01:44:c3:31:22:
            52:98:16:4c:39:af:f3:2a:5f:58:6e:96:68:0e:68:
            3f:fc:14:0c:72:db:15:bd:22:73:de:6d:d7:6b:91:
            18:fc:43:d7:8b:fa:a5:e9:46:b0:4e:cd:1e:1a:93:
            01:ec:87:89:93:4a:95:9a:83:d0:bc:b4:75:af:2b:
            a3:d2:64:93:8e:93:05:36:c6:a7:87:c0:90:02:c8:
            eb:aa:39:81:7f:cf:fe:57:33:ab:8e:1e:1a:db:93:
            d7:c9:3e:4a:0f:e4:d4:0b:bd:0e:8e:f6:57:b8:94:
            30:ae:5f:3e:92:3a:97:b6:72:13:e7:ee:7e:d2:4c:
            62:88:ca:19:b4:8a:ef:2f:ff:24:6f:ad:e6:d2:98:
            ec:a9:54:38:1a:15:12:da:25:d7:0e:33:3c:7d:d5:
            41:79:11:49:ce:99:9e:2f:2d:10:fc:58:9e:c2:74:
            f7:5c:ba:e7:49:90:bc:f8:ab:af:f4:40:53:4e:ab:
            63:b6:65:39:f7:d4:7b:8b:a7:5f:90:70:c4:f2:7d:
            4b:66:20:55:59:19:bd:9d:55:3c:48:26:d9:fc:72:
            38:a7:ed:ef:53:cb:db:eb:0e:4a:0c:73:4e:c5:93:
            aa:15:9c:71:ea:fc:e7:c0:7d:cc:24:f4:36:db:4e:
            61:eb
        generator: 2 (0x2)
```

## I.4.2 Step 2: Install Diffie-Hellman Parameters

### I.4.2.1 FM Component

#### I.4.2.1.1 HSM

1. Copy the Diffie-Hellman parameters *<intel-opa-comp-dh-param>*.pem to the name configured for the opafm.xml parameter <SslSecurityFmDHParameters> (default parameter value: fm_dh_parms.pem):

```
cp <intel-opa-comp-dh-param>.pem  fm_dh_parms.pem
```

2. Copy the Diffie-Hellman parameters fm_dh_parms.pem to the directory configured for the opafm.xml parameter <SslSecurityDir> (default parameter value: /usr/local/ssl/opafm/).

```
cp fm_dh_parms.pem /usr/local/ssl/opafm/
```

#### I.4.2.1.2 ESM

1. Establish a SFTP session to the managed chassis:

```
sftp admin@<managed-chassis-ip-address>
```

2. Copy the Diffie-Hellman parameters `<intel-opa-comp-dh-param>.pem` to the name configured for the `opafm.xml` parameter `<SslSecurityFmDHParameters>` (default parameter value: `fm_dh_parms.pem`):

```
cp fm_dh_parms.pem /usr/local/ssl/opafm/
```

3. SFTP the Diffie-Hellman parameters `fm_dh_parms.pem` to the directory configured for the `opafm.xml` parameter `<SslSecurityDir>` (default parameter value: `/mmc0:4/`)

```
sftp> fm_dh_parms.pem /mmc0:4/
```

4. Exit the SFTP session:

```
sftp> exit
```

### I.4.2.2 FF Component

1. Copy the Diffie-Hellman parameters `<intel-opa-comp-dh-param>.pem` to the name configured for the `opaff.xml` parameter `<SslSecurityFFDHParameters>` (default parameter value: `ff_dh_parms.pem`):

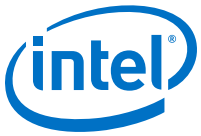
```
cp <intel-opa-comp-dh-param>.pem  ff_dh_parms.pem
```

2. Copy the Diffie-Hellman parameters `ff_dh_parms.pem` to the directory configured for the `opaff.xml` parameter `<SslSecurityDir>`:

```
cp ff_dh_parms.pem /usr/local/ssl/opafm/
```

### I.4.2.3 Fabric Manager GUI Component

Installation of the Diffie-Hellman parameters for the Fabric Manager GUI component is not addressed in these Guidelines. Reference the PKI Best Practices Guidelines for the Fabric Manager GUI document.



## I.5 Optional - Install Certificate Revocation List (CRL) for an Intel® Omni-Path Fabric Suite Software Package Component

Key exchange related protocols such as SSL/TLS utilize a CRL to determine whether a certificate has been compromised. The Fabric Manager, FastFabric, and Fabric Manager GUI components of the Intel® Omni-Path Fabric Suite components of the Intel® Omni-Path Fabric Suite Fabric Manager software package do not support an automated method for periodically checking for CRL updates by the CA issuer.

It is the responsibility of the administrator(s) to perform this task prior to starting the Fabric Manager, FastFabric, and Fabric Manager GUI components of the Intel® Omni-Path Fabric Suite components. When CRL updates occurs, the administrator must do the following to setup the CRL.

*Note:* To minimize the administrative effort, this task could be done once and the same CRL used for all components.

### I.5.1 Step 1: Download CRL from Third-party CA

1. Execute the appropriate steps published by the third-party CA to download latest CRL *<intel-opa-comp-crl >*.pem to the work directory.
2. Execute the following OpenSSL command to validate the content of the CRL *<intel-opa-comp-crl >*.pem:

```
openssl crl -in <intel-opa-comp-crl>.pem -noout -text
```

HSM - EC Signed CRL Example

```
Certificate Revocation List (CRL):
        Version 2 (0x1)
    Signature Algorithm: ecdsa-with-SHA256
        Issuer: /C=US/ST=Pennsylvania/L=King of Prussia/O=OPA Sample
Corporation/OU=OPA Sample Division/CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Last Update: Nov 15 07:10:22 2014 GMT
        Next Update: Dec 15 07:10:22 2014 GMT
        CRL extensions:
            X509v3 CRL Number:
                1
No Revoked Certificates.
    Signature Algorithm: ecdsa-with-SHA256
         30:45:02:20:6d:37:e5:5d:89:f0:03:72:c5:da:12:f7:06:44:
         d7:eb:1d:90:e9:08:b6:f7:e2:0e:e1:43:78:fd:1d:97:1b:4d:
         02:21:00:c3:c9:db:53:d2:0f:69:21:2d:c5:a1:b7:71:48:9d:
         b6:d5:e4:1f:52:df:0b:c0:ea:bc:97:e4:8c:64:ca:ff:ff
```

ESM - DSA Signed CRL Example

```
Certificate Revocation List (CRL):
        Version 2 (0x1)
    Signature Algorithm: dsa_with_SHA256
        Issuer: /C=US/ST=Pennsylvania/L=King of Prussia/O=OPA Sample
Corporation/OU=OPA Sample Division/CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Last Update: Nov  7 13:47:03 2014 GMT
        Next Update: Dec  7 13:47:03 2014 GMT
```

```
        CRL extensions:
            X509v3 CRL Number:
                1
No Revoked Certificates.
    Signature Algorithm: dsa_with_SHA256
         r:
             3a:b9:b8:cc:b0:22:05:cf:3f:4b:2e:62:63:b1:d3:
             23:11:02:ea:5d:06:57:3b:ae:d9:02:e2:60:26:fe:
             82:13
         s:
             58:36:1f:3e:3b:1b:e1:b4:df:c1:60:22:0b:61:91:
             24:c9:72:de:1e:40:4c:36:6c:72:6e:3d:a8:12:fc:
             b5:85
```

3. Copy the CRL *<intel-opa-comp-crl >*.pem to the directory configured for the ifs_fm.xml parameter <SslSecurityDir> (default value is /usr/local/ssl/ifs_fm/)

## I.5.2 Step 2: Install CRL from Third-party CA

### I.5.2.1 FM Component

#### I.5.2.1.1 HSM

1. Copy the CRL *<intel-opa-comp-crl >*.pem to the name configured for the opafm.xml parameter <SslSecurityFmCaCRL> (default parameter value: fm_ca_crl.pem):

```
cp <intel-opa-comp-crl >.pem  fm_ca_crl.pem
```

2. Copy the CRL fm_ca_crl.pem to the directory configured for the opafm.xml parameter <SslSecurityDir> (default parameter value: /usr/local/ssl/opafm/):

```
cp fm_ca_crl.pem /usr/local/ssl/opafm/
```

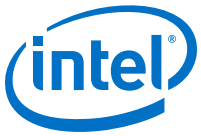
#### I.5.2.1.2 ESM

1. Establish a SFTP session to the managed chassis:

```
sftp admin@<managed-chassis-ip-address>
```

2. Copy the CRL *<intel-opa-comp-crl >*.pem to the name configured for the opafm.xml parameter < SslSecurityFmDHParameters> (default parameter value: fm_dh_parms.pem):

```
cp <intel-opa-comp-crl >.pem  fm_ca_crl.pem
```



3. SFTP the CRL `fm_ca_crl.pem` to the directory configured for the `opafm.xml` parameter `<SslSecurityDir>` (default parameter value: `/mmc0:4/`)

```
sftp> put fm_ca_crl.pem /mmc0:4/
```

4. Exit the SFTP session:

```
sftp> exit
```

### I.5.2.2 FF Component

1. Copy the CRL *<intel-opa-comp-crl >*`.pem` to the name configured for the `ifs_ff.xml` parameter `<SslSecurityFFCaCRL>` (default parameter value: `ff_ca_crl.pem`):

```
cp <intel-opa-comp-crl >.pem  ff_ca_crl.pem
```

2. Copy the CRL `ff_ca_crl.pem` to the directory configured for the `ifs_ff.xml` parameter `<SslSecurityDir>`:

```
cp ff_ca_crl.pem /usr/local/ssl/ifs_fm/
```

### I.5.2.3 Intel® Omni-Path Fabric Suite Fabric Manager Component

Installation of the CRL from the third-party CA for the Intel® Omni-Path Fabric Suite Fabric Manager component is not addressed in these Guidelines. Reference the PKI Best Practices Guidelines for the Intel® Omni-Path Fabric Suite Fabric Manager document.

# Appendix J Advanced-Level Public Key Infrastructure Best Practices Guidelines

## J.1 Overview

The guidelines in this section are intended to provide a reference for the advance administration tasks required to support a self-signed Certificate Authority (CA) using the OpenSSL command set.

*Note:* Instructions in Core-Level Public Key Infrastructure (PKI) Best Practices Guidelines on page 230 must be completed first in order to utilize this section effectively.

### J.1.1 Self-Signed Certificate Authority (CA)

#### J.1.1.1 Setting up the Work Environment

##### J.1.1.1.1 Set Up Work Directory

1. Create a work directory for the self-signed CA:

```
mkdir opaCA
```

2. Move to the work directory:

```
cd opaCA/
```

3. Create a directory to retain issued certificates:

```
mkdir certs
```

4. Create a directory to retain issued Certificate Revocation List (CRL):

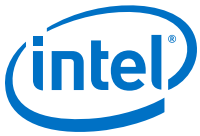
```
mkdir crl
```

5. Create a directory to retain new OPA component certificates:

```
mkdir newcerts
```

6. Create a directory to retain the Private Key of the OPA CA:

```
mkdir private
```



#### J.1.1.1.2 Set Up Certificate Database

1. OpenSSL requires the creation of a file called `serial`. This file is used to keep track of the last serial number used to issue a certificate. Create the `serial` file:

```
echo '01' > serial
```

2. OpenSSL requires the creation of a file called `crlnumber`. This file is used to keep track of the last serial number used to create a Certificate Revocation List (CRL). Create the `crlnumber` file:

```
echo '01' > crlnumber
```

3. OpenSSL requires the creation of a file called `index.txt`. This file is used to keep track of the certificates that have been issued by the CA. Create the `index.txt` file:

```
touch index.txt
```

#### J.1.1.1.3 Validate Layout of Work Environment

1. Enter this command:

```
ls
```

Output:

```
certs  crl crlnumber index.txt  newcerts  private  serial
```

### J.1.1.2 Setting Up the Self-Signed CA OpenSSL Configuration File

1. Move to the work directory:

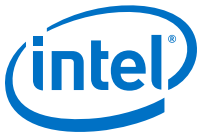
```
cd opaCA/
```

2. OpenSSL has a default configuration file (`openssl.cnf`) that is used for certificate requests and self-signed CA related tasks. The Intel® Omni-Path Fabric Suite Fabric Manager software package provides a sample `openssl.cnf` configuration file (`opa_ca_openssl.cnf-sample`, installed in directories `/usr/lib/opa/samples` and `/usr/share/opa-fm/samples`) for the CA. This configuration file can be modified for the unique requirements of the customer. Copy this sample configuration file to the work directory:

```
cp /usr/share/opa/samples/opa_ca_openssl.cnf-sample opa_ca_openssl.cnf
```

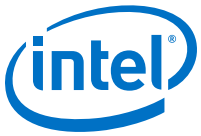
3. Use your favorite editor to edit the opa_ca_openssl.cnf configuration file. Modify at least the following fields marked in **bold** to meet your unique requirements:

```
#
# Intel OPA Self-Signed CA OpenSSL example configuration file.
# This is mostly being used for generation of certificate requests.
#

# This definition stops the following lines choking if HOME isn't
# defined.
HOME            = .
RANDFILE        = $ENV::HOME/.rnd

# Extra OBJECT IDENTIFIER info:
#oid_file        = $ENV::HOME/.oid
oid_section        = new_oids

# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions        =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)

[ new_oids ]

# We can add new OIDs in here for use by 'ca', 'req' and 'ts'.
# Add a simple OID like this:
# testoid1=1.2.3.4
# Or use config file substitution like this:
# testoid2=${testoid1}.5.6

# Policies used by the TSA examples.
tsa_policy1 = 1.2.3.4.1
tsa_policy2 = 1.2.3.4.5.6
tsa_policy3 = 1.2.3.4.5.7
####################################################################
[ ca ]
default_ca    = opa_ca        # The default ca section

####################################################################
[ opa_ca ]

dir        = /home/<username>/opaCA        # Where everything is kept
certs        = $dir/certs        # Where the issued certs are kept
crl_dir        = $dir/crl        # Where the issued crl are kept
database    = $dir/index.txt    # database index file.
#unique_subject    = no            # Set to 'no' to allow creation of
                    # several ctificates with same subject.
new_certs_dir    = $dir/newcerts        # default place for new certs.

certificate    = $dir/cacert.pem     # The CA certificate
serial        = $dir/serial         # The current serial number
crlnumber    = $dir/crlnumber    # the current crl number
                    # must be commented out to leave a V1 CRL
crl        = $dir/crl.pem         # The current CRL
private_key    = $dir/private/cakey.pem# The private key
RANDFILE    = $dir/private/.rand    # private random number file

x509_extensions    = opa_ca_cert_extensions        # The extentions to add to
the cert

# Comment out the following two lines for the "traditional"
# (and highly broken) format.
#name_opt     = opa_ca        # Subject Name options
#cert_opt     = opa_ca        # Certificate field options

# Extension copying option: use with caution.
# copy_extensions = copy
```



```
# Extensions to add to a CRL. Note: Netscape communicator chokes on V2 CRLs
# so this is commented out by default to leave a V1 CRL.
# crlnumber must also be commented out to leave a V1 CRL.
# crl_extensions    = crl_ext

default_days    = 365            # how long to certify for
default_crl_days= 30            # how long before next CRL
default_md    = sha256        # default MD
preserve    = no            # keep passed DN ordering

# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy        = opa_ca_policy_match

# For the CA policy
[ opa_ca_policy_match ]
commonName        = supplied
countryName        = supplied
stateOrProvinceName    = supplied
organizationName    = supplied
emailAddress        = supplied
organizationalUnitName    = optional

#################################################################

[ req ]
default_bits        = 2048
default_md        = sha256
default_keyfile     = /home/<username>/opaCA/private/cakey.pem
distinguished_name    = opa_ca_distinguished_name
attributes        = req_attributes
x509_extensions    = opa_ca_extensions    # The extentions to add to the self
signed cert
prompt                    = no

# Passwords for private keys if not present they will be prompted for
# input_password = secret
# output_password = secret

# This sets a mask for permitted string types. There are several options.
# default: PrintableString, T61String, BMPString.
# pkix     : PrintableString, BMPString (PKIX recommendation before 2004)
# utf8only: only UTF8Strings (PKIX recommendation after 2004).
# nombstr : PrintableString, T61String (no BMPStrings or UTF8Strings).
# MASK:XXXX a literal mask value.
# WARNING: ancient versions of Netscape crash on BMPStrings or UTF8Strings.
string_mask = utf8only

[ opa_ca_distinguished_name ]
countryName            = Country Name (2 letter code)
#countryName_default        = XX
#countryName_min        = 2
#countryName_max        = 2

stateOrProvinceName        = State or Province Name (full name)
#stateOrProvinceName_default    = Default Province

localityName            = Locality Name (eg, city)
#localityName_default        = Default City

organizationName        = Organization Name (eg, company)
#organizationName_default    = Default Company Ltd

organizationalUnitName        = Organizational Unit Name (eg, section)
#organizationalUnitName_default    =

commonName            = Common Name (eg, your name or your server\'s hostname)

#commonName_max        = 64
```

```
emailAddress            = Email Address
#emailAddress_max        = 64

# SET-ex3            = SET extension number 3

[ req_attributes ]
challengePassword        = A challenge password
challengePassword_min        = 4
challengePassword_max        = 20

unstructuredName        = An optional company name

[ opa_ca_cert_extensions ]

# These extensions are added when 'ca' signs a request.

# This goes against PKIX guidelines but some CAs do it and some software
# requires this to avoid interpreting an end user certificate as a CA.

basicConstraints=CA:FALSE

# Here are some examples of the usage of nsCertType. If it is omitted
# the certificate can be used for anything *except* object signing.

# This is OK for an SSL server.
# nsCertType            = server

# For an object signing certificate this would be used.
# nsCertType = objsign

# For normal client use this is typical
# nsCertType = client, email

# and for everything including object signing:
# nsCertType = client, email, objsign

# This is typical in keyUsage for a client certificate.
# keyUsage = nonRepudiation, digitalSignature, keyEncipherment

# This will be displayed in Netscape's comment listbox.
nsComment            = "OpenSSL Generated Certificate"

# PKIX recommendations harmless if included in all certificates.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer

# This stuff is for subjectAltName and issuerAltname.
# Import the email address.
# subjectAltName=email:copy
# An alternative to produce certificates that aren't
# deprecated according to PKIX.
# subjectAltName=email:move

# Copy subject details
# issuerAltName=issuer:copy

#nsCaRevocationUrl        = http://www.domain.dom/ca-crl.pem
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName

# This is required for TSA certificates.
# extendedKeyUsage = critical,timeStamping

[ opa_ca_extensions ]

# Extensions for a typical CA
```



```
# PKIX recommendation.

subjectKeyIdentifier=hash

authorityKeyIdentifier=keyid:always,issuer

# This is what PKIX recommends but some broken software chokes on critical
# extensions.
#basicConstraints = critical,CA:true
# So we do this instead.
basicConstraints = CA:true

# Key usage: this is typical for a CA certificate. However since it will
# prevent it being used as an test self-signed certificate it is best
# left out by default.
# keyUsage = cRLSign, keyCertSign

# Some might want this also
# nsCertType = sslCA, emailCA

# Include email address in subject alt name: another PKIX recommendation
# subjectAltName=email:copy
# Copy issuer details
# issuerAltName=issuer:copy

# DER hex encoding of an extension: beware experts only!
# obj=DER:02:03
# Where 'obj' is a standard or added object
# You can even override a supported extension:
# basicConstraints= critical, DER:30:03:01:01:FF

[ crl_ext ]

# CRL extensions.
# Only issuerAltName and authorityKeyIdentifier make any sense in a CRL.

# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always
```

### J.1.1.3 Setting Up a Self-Signed Root Certificate

Execute the instructions of the following section that is applicable to the OPA SM component.

#### J.1.1.3.1 Self-Signed Root Certificate to Sign HSM-related PEM Files

1. Move to the OPA work directory:

```
cd opaCA/
```

2. Direct the OpenSSL environment to use the customized `opa_ca_openssl.cnf` configuration file:

```
export OPENSSL_CONF=/<path>/opaCA/opa_ca_openssl.cnf
```

3. Execute the following OpenSSL command to generate the EC parameters and the EC Private Key of the self-signed CA:

```
openssl ecparam -name prime256v1 -genkey -out ./private/cakey.pem
```

4. Execute the following OpenSSL command to generate the Root certificate of the self-signed CA:

```
openssl req -x509 -new -key ./private/cakey.pem -out cacert.pem
```

5. Execute the following OpenSSL command to validate the content of the EC Private Key of the self-signed CA:

```
# openssl ecparam -text -in ./private/cakey.pem -noout
```

Output:

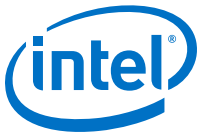
```
ASN1 OID: prime256v1
```

6. Execute the following OpenSSL command to validate the content of the Root certificate of the self-signed CA:

```
openssl x509 -in cacert.pem -noout -text
```

Output:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
    Signature Algorithm: ecdsa-with-SHA256
        Issuer: C=US, ST=Pennsylvania, L=King of Prussia, O=OPA Sample
Corporation, OU=OPA Sample Division, CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Validity
            Not Before: Nov 15 07:06:04 2014 GMT
            Not After : Nov 15 07:06:04 2015 GMT
        Subject: CN=phgppriv03.ph.intel.com, C=US, ST=Pennsylvania, O=OPA
Component Sample Corporation/emailAddress=root@phgppriv03.ph.intel.com,
OU=OPA Component Sample Division
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:8e:85:df:04:ac:36:81:7f:6a:09:58:8a:de:a1:
                    7a:ab:b2:00:34:e0:d3:9f:92:2a:da:d9:6c:40:58:
                    00:46:b0:55:15:4c:60:dd:55:40:46:23:13:5c:65:
                    30:26:01:3d:a8:73:86:6c:38:fc:9b:e4:45:14:70:
                    20:e4:2c:b1:84
                ASN1 OID: prime256v1
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                39:74:80:E3:03:55:AC:F6:6F:2C:69:48:FF:5E:78:DF:FF:48:99:BF
            X509v3 Authority Key Identifier:
                keyid:D4:54:0A:34:1A:9F:1E:9C:4C:60:2E:B0:CB:
89:57:80:B1:1F:E3:23

    Signature Algorithm: ecdsa-with-SHA256
         30:45:02:20:28:5f:23:ca:ed:9c:42:f8:5b:57:b7:03:44:24:
         8b:e9:20:88:13:c1:d9:c4:67:52:6b:eb:47:ca:31:95:77:5a:
         02:21:00:ad:4b:b0:cf:4d:ba:35:0e:9d:aa:1b:20:44:52:4c:
         18:3c:a7:11:7f:7e:a7:75:10:7f:da:78:30:d6:f7:f5:47
```



#### J.1.1.3.2 Self-Signed Root Certificate to Sign ESM-related PEM Files

1. Move to the OPA work directory:

```
cd opaCA/
```

2. Direct the OpenSSL environment to use the customized `opa_ca_openssl.cnf` configuration file:

```
export OPENSSL_CONF=/<path>/opaCA/opa_ca_openssl.cnf
```

3. Execute the following OpenSSL command to generate the DSA parameters and the DSA Private Key of the self-signed CA:

```
openssl dsaparam -genkey 2048 -out ./private/cakey.pem
```

Output:

```
Generating DSA parameters, 2048 bit long prime
This could take some time
.......+.....................+.+.............+.....................
+.........+.............+.+...+....+..+.............+.....+.........
+.........................+.....+...........+..+.....+...........+.+++++++
++++++++++++++++++++++++++++++++++++++++++*
.+.+.....................+........+...........+..+..
+.........................+...+.......+...+.+....+...+..+....+.+...
+...................+......+...+........+..+........+....+................
+............+.............+................................
+.............+......+.+.................+...+.........+..+.......
+.............+..+...........+..........+..+.....................+......
+........+.............+....+..+.+.....+......+..+....+..+...+......++++++++++
++++++++++++++++++++++++++++++++++++++++*
```

4. Execute the following OpenSSL command to generate the Root certificate of the self-signed CA:

```
openssl req -x509 -new -key ./private/cakey.pem -out cacert.pem
```

5. Execute the following OpenSSL command to validate the content of the DSA Private Key of the self-signed CA:

```
openssl dsa -in ./private/cakey.pem -noout -text
```

Output:

```
read DSA key
Private-Key: (2048 bit)
priv:
    27:5e:52:52:b0:08:69:41:f9:52:85:60:f4:f7:6c:
    b0:6b:ff:c5:c4:8a:93:70:fe:3c:20:f9:70:c8:dd:
    18:eb
pub:
    3e:fa:61:ba:a3:74:67:a7:8e:c2:1f:60:77:65:75:
    ff:d7:3d:2e:a2:75:a0:ae:59:3a:2c:c4:8e:93:a9:
    ec:46:28:e6:af:6c:47:4a:fd:f5:8c:83:4f:4b:b5:
    a0:60:e5:c5:8d:22:5a:4b:8c:79:a4:d8:25:fb:ab:
    5c:99:9a:aa:c1:3a:0b:5b:2e:87:55:d7:11:37:13:
    00:be:bb:29:a3:14:eb:58:81:19:03:a8:20:ef:dc:
```

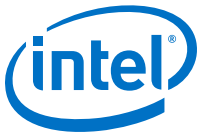
```
    f4:7e:66:4e:a9:c6:6b:26:26:9d:fd:44:c8:2f:58:
    bb:9f:84:a5:a4:a1:a8:53:53:10:6a:a0:7f:24:2c:
    03:3e:64:15:42:44:dc:9f:6d:74:bc:a4:83:fc:ed:
    1a:1d:e3:10:a9:f5:86:e5:f3:2b:16:6f:86:75:ed:
    a0:72:66:77:0c:da:70:9f:c8:d9:91:91:67:57:b3:
    d4:2d:c7:68:b0:2f:2a:ea:d6:fa:2f:f6:9a:03:e5:
    4a:41:c0:34:5f:f4:28:b0:ca:84:07:22:d6:d5:ef:
    23:cc:52:a1:57:b7:ac:90:c6:9a:e7:46:3a:1f:f9:
    9a:8d:cc:b8:67:7d:61:e2:19:ed:79:0f:ee:d7:22:
    c9:7d:f2:d4:dd:df:e0:a8:e8:ec:17:94:8c:51:ea:
    f4:eb:f8:05:21:54:0a:3e:94:03:00:92:0c:dd:e8:
    58
P:
    00:8d:20:86:d7:6a:6e:73:e4:6d:c1:dc:ea:b3:38:
    fc:5a:12:8e:c4:4b:d9:b1:93:f6:f5:35:c3:8a:39:
    8e:23:10:79:51:b2:d9:bc:a9:f0:b5:e3:85:6c:d8:
    74:92:34:a6:0f:fb:b4:f2:a6:51:06:9a:6d:f8:65:
    8d:d6:12:35:dd:a8:df:f9:aa:53:c3:65:ea:0b:62:
    c0:36:6e:dc:15:6f:c4:03:1f:e9:9f:d7:e9:b2:7c:
    32:db:82:5e:69:81:d3:08:97:9c:ea:82:ed:ae:7d:
    2b:e7:5a:40:97:0d:b2:62:7b:1f:f9:c4:7e:b2:e4:
    a1:e4:22:b3:1c:61:5b:eb:d0:9f:1e:9b:f1:7e:b1:
    b8:7a:8e:b3:ea:66:78:f7:92:f9:5e:5c:02:e5:f9:
    a6:cc:c2:ce:c5:63:01:fd:bf:91:96:01:8d:e6:fb:
    e4:e4:6f:55:d8:0d:06:0b:fd:92:38:67:7b:8a:ff:
    1a:d8:6b:45:bb:b7:06:e2:98:53:13:fb:c0:5d:e7:
    06:69:28:85:ab:10:fd:65:f8:ae:b3:7e:67:f6:8f:
    5b:b1:23:02:14:c2:ab:55:7f:11:ab:42:c9:a7:14:
    2b:42:a5:fe:0f:54:7d:d1:7a:64:91:95:b7:26:e8:
    19:8b:7c:18:c4:c8:18:b9:18:2a:c5:b9:22:f9:86:
    2a:cb
Q:
    00:a4:bd:a6:a1:be:b8:f5:52:f8:56:06:a1:bb:88:
    7a:f9:f2:42:a8:21:d8:6d:9a:40:e6:50:65:2b:92:
    a1:5b:67
G:
    00:82:80:ad:15:64:f0:69:f4:90:c1:f7:84:a1:36:
    a8:34:bf:ff:b2:8c:5f:2d:d3:d8:b4:3d:a7:cc:0a:
    5f:1e:c3:96:2b:f8:ea:0f:9a:77:2e:22:bd:0b:08:
    fb:11:7e:57:b0:f5:6f:73:3e:34:d2:c4:95:67:cf:
    8b:43:52:a7:76:44:09:41:79:d4:0d:da:b3:bc:b2:
    eb:63:5f:97:c5:e6:5c:44:13:4d:dd:81:20:ea:a6:
    c8:2e:a6:c7:f5:ac:a3:0b:e7:b1:6c:43:36:78:30:
    13:86:53:39:9f:e8:f6:be:a5:96:bd:e2:de:2b:57:
    37:26:3a:d2:36:1e:2f:8c:36:b9:7b:78:3b:77:e6:
    02:41:cb:ec:d2:02:9c:00:52:49:c8:3f:f0:be:b8:
    80:3a:df:bf:11:47:28:b6:c9:3d:44:ff:1a:c8:fe:
    1e:e7:61:70:9a:71:f7:1d:50:1e:fa:d4:8c:aa:7a:
    6d:7f:e8:04:66:fe:f3:90:2b:f0:15:32:8e:80:de:
    ed:6d:9c:7f:b0:f6:ee:72:c1:cd:a4:94:c0:ee:25:
    50:15:15:b1:e4:a4:0a:c2:ef:f6:88:0d:84:ab:36:
    c7:81:7b:34:84:bf:6c:6c:55:24:4b:4f:6c:45:b9:
    9d:96:db:9f:a1:6c:59:9e:8f:ea:68:4f:11:75:e0:
    3d:23
```

6. Execute the following OpenSSL command to validate the content of the Root certificate of the self-signed CA:

```
openssl x509 -in cacert.pem -noout -text
```

Output:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            c6:4c:2f:23:0c:69:40:35
        Signature Algorithm: dsa_with_SHA256
```



```
        Issuer: C=US, ST=Pennsylvania, L=King of Prussia, O=OPA Sample
Corporation, OU=OPA Sample Division, CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Validity
            Not Before: Nov  7 15:22:01 2014 GMT
            Not After : Dec  7 15:22:01 2014 GMT
        Subject: C=US, ST=Pennsylvania, L=King of Prussia, O=OPA Sample
Corporation, OU=OPA Sample Division, CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Subject Public Key Info:
            Public Key Algorithm: dsaEncryption
                pub:
                    3e:fa:61:ba:a3:74:67:a7:8e:c2:1f:60:77:65:75:
                    ff:d7:3d:2e:a2:75:a0:ae:59:3a:2c:c4:8e:93:a9:
                    ec:46:28:e6:af:6c:47:4a:fd:f5:8c:83:4f:4b:b5:
                    a0:60:e5:c5:8d:22:5a:4b:8c:79:a4:d8:25:fb:ab:
                    5c:99:9a:aa:c1:3a:0b:5b:2e:87:55:d7:11:37:13:
                    00:be:bb:29:a3:14:eb:58:81:19:03:a8:20:ef:dc:
                    f4:7e:66:4e:a9:c6:6b:26:26:9d:fd:44:c8:2f:58:
                    bb:9f:84:a5:a4:a1:a8:53:53:10:6a:a0:7f:24:2c:
                    03:3e:64:15:42:44:dc:9f:6d:74:bc:a4:83:fc:ed:
                    1a:1d:e3:10:a9:f5:86:e5:f3:2b:16:6f:86:75:ed:
                    a0:72:66:77:0c:da:70:9f:c8:d9:91:91:67:57:b3:
                    d4:2d:c7:68:b0:2f:2a:ea:d6:fa:2f:f6:9a:03:e5:
                    4a:41:c0:34:5f:f4:28:b0:ca:84:07:22:d6:d5:ef:
                    23:cc:52:a1:57:b7:ac:90:c6:9a:e7:46:3a:1f:f9:
                    9a:8d:cc:b8:67:7d:61:e2:19:ed:79:0f:ee:d7:22:
                    c9:7d:f2:d4:dd:df:e0:a8:e8:ec:17:94:8c:51:ea:
                    f4:eb:f8:05:21:54:0a:3e:94:03:00:92:0c:dd:e8:
                    58
                P:
                    00:8d:20:86:d7:6a:6e:73:e4:6d:c1:dc:ea:b3:38:
                    fc:5a:12:8e:c4:4b:d9:b1:93:f6:f5:35:c3:8a:39:
                    8e:23:10:79:51:b2:d9:bc:a9:f0:b5:e3:85:6c:d8:
                    74:92:34:a6:0f:fb:b4:f2:a6:51:06:9a:6d:f8:65:
                    8d:d6:12:35:dd:a8:df:f9:aa:53:c3:65:ea:0b:62:
                    c0:36:6e:dc:15:6f:c4:03:1f:e9:9f:d7:e9:b2:7c:
                    32:db:82:5e:69:81:d3:08:97:9c:ea:82:ed:ae:7d:
                    2b:e7:5a:40:97:0d:b2:62:7b:1f:f9:c4:7e:b2:e4:
                    a1:e4:22:b3:1c:61:5b:eb:d0:9f:1e:9b:f1:7e:b1:
                    b8:7a:8e:b3:ea:66:78:f7:92:f9:5e:5c:02:e5:f9:
                    a6:cc:c2:ce:c5:63:01:fd:bf:91:96:01:8d:e6:fb:
                    e4:e4:6f:55:d8:0d:06:0b:fd:92:38:67:7b:8a:ff:
                    1a:d8:6b:45:bb:b7:06:e2:98:53:13:fb:c0:5d:e7:
                    06:69:28:85:ab:10:fd:65:f8:ae:b3:7e:67:f6:8f:
                    5b:b1:23:02:14:c2:ab:55:7f:11:ab:42:c9:a7:14:
                    2b:42:a5:fe:0f:54:7d:d1:7a:64:91:95:b7:26:e8:
                    19:8b:7c:18:c4:c8:18:b9:18:2a:c5:b9:22:f9:86:
                    2a:cb
                Q:
                    00:a4:bd:a6:a1:be:b8:f5:52:f8:56:06:a1:bb:88:
                    7a:f9:f2:42:a8:21:d8:6d:9a:40:e6:50:65:2b:92:
                    a1:5b:67
                G:
                    00:82:80:ad:15:64:f0:69:f4:90:c1:f7:84:a1:36:
                    a8:34:bf:ff:b2:8c:5f:2d:d3:d8:b4:3d:a7:cc:0a:
                    5f:1e:c3:96:2b:f8:ea:0f:9a:77:2e:22:bd:0b:08:
                    fb:11:7e:57:b0:f5:6f:73:3e:34:d2:c4:95:67:cf:
                    8b:43:52:a7:76:44:09:41:79:d4:0d:da:b3:bc:b2:
                    eb:63:5f:97:c5:e6:5c:44:13:4d:dd:81:20:ea:a6:
                    c8:2e:a6:c7:f5:ac:a3:0b:e7:b1:6c:43:36:78:30:
                    13:86:53:39:9f:e8:f6:be:a5:96:bd:e2:de:2b:57:
                    37:26:3a:d2:36:1e:2f:8c:36:b9:7b:78:3b:77:e6:
                    02:41:cb:ec:d2:02:9c:00:52:49:c8:3f:f0:be:b8:
                    80:3a:df:bf:11:47:28:b6:c9:3d:44:ff:1a:c8:fe:
                    1e:e7:61:70:9a:71:f7:1d:50:1e:fa:d4:8c:aa:7a:
                    6d:7f:e8:04:66:fe:f3:90:2b:f0:15:32:8e:80:de:
                    ed:6d:9c:7f:b0:f6:ee:72:c1:cd:a4:94:c0:ee:25:
                    50:15:15:b1:e4:a4:0a:c2:ef:f6:88:0d:84:ab:36:
                    c7:81:7b:34:84:bf:6c:6c:55:24:4b:4f:6c:45:b9:
                    9d:96:db:9f:a1:6c:59:9e:8f:ea:68:4f:11:75:e0:
```

```
                    3d:23
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                B4:0B:2E:4E:EB:E1:91:CE:40:09:F7:AD:AE:AB:B7:64:F2:13:96:E4
            X509v3 Authority Key Identifier:
                keyid:B4:0B:2E:4E:EB:E1:91:CE:
40:09:F7:AD:AE:AB:B7:64:F2:13:96:E4

            X509v3 Basic Constraints:
                CA:TRUE
    Signature Algorithm: dsa_with_SHA256
        30:44:02:20:25:62:33:4d:4d:a4:25:07:71:f8:0b:4f:e8:be:
        ec:6c:41:0c:9e:2c:4a:53:97:4e:bb:e9:09:45:3c:a6:7e:8e:
        02:20:65:40:d5:14:e7:cf:f2:8c:ea:a4:32:13:68:55:06:6c:
        97:8b:68:e9:e7:00:58:1a:ee:ca:d5:55:a7:42:10:15
```

### J.1.1.4 Issuing Certificates

1. Move to the OPA work directory:

```
cd opaCA/
```

2. Direct the OpenSSL environment to use the customized `opa_comp_openssl.cnf` configuration file used for an OPA component:

```
export OPENSSL_CONF=/<path>/opaCA/opa_comp_openssl.cnf
```

3. If the Private Key and/or the CSR of the component does not exist, then execute Step 1: Set up the OPA Component OpenSSL Configuration File on page 230 and Step 2: Create Private Key on page 232 that describe how to create the Private Key and a CSR for a OPA component.
4. Direct the OpenSSL environment to use the customized `opa_ca_openssl.cnf` configuration file of the self-signed CA:

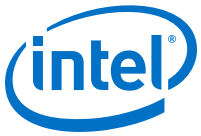
```
export OPENSSL_CONF=/<path>/opaCA/opa_ca_openssl.cnf
```

5. Sign the certificate of the OPA component:

```
openssl ca -in <intel-opa-comp-csr>.pem -out <intel-opa-comp-cert>.pem
```

Output:

```
Using configuration from /home/kingchar/opaCA/opa_ca_openssl.cnf
Enter pass phrase for /home/kingchar/opaCA/private/cakey.pem:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'US'
stateOrProvinceName   :ASN.1 12:'Pennsylvania'
localityName          :ASN.1 12:'King of Prussia'
organizationName      :ASN.1 12:'OPA Sample Corporation'
organizationalUnitName:ASN.1 12:'OPA Sample Division'
commonName            :ASN.1 12:'phgppriv03.ph.intel.com'
emailAddress          :IA5STRING:'root@phgppriv03.ph.intel.com'
Certificate is to be certified until Nov  4 19:38:48 2015 GMT (365 days)
Sign the certificate? [y/n]:y
```



```
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

6. Execute the following OpenSSL command to validate the content of the signed certificate of the OPA component:

```
openssl x509 -in <intel-opa-comp-cert>.pem -noout -text
```

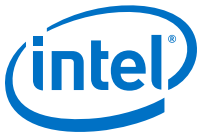
HSM – EC Signed Certificate Example

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
    Signature Algorithm: ecdsa-with-SHA256
        Issuer: C=US, ST=Pennsylvania, L=King of Prussia, O=OPA Sample
Corporation, OU=OPA Sample Division, CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Validity
            Not Before: Nov 15 07:06:04 2014 GMT
            Not After : Nov 15 07:06:04 2015 GMT
        Subject: CN=phgppriv03.ph.intel.com, C=US, ST=Pennsylvania, O=OPA
Component Sample Corporation/emailAddress=root@phgppriv03.ph.intel.com,
OU=OPA Component Sample Division
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:8e:85:df:04:ac:36:81:7f:6a:09:58:8a:de:a1:
                    7a:ab:b2:00:34:e0:d3:9f:92:2a:da:d9:6c:40:58:
                    00:46:b0:55:15:4c:60:dd:55:40:46:23:13:5c:65:
                    30:26:01:3d:a8:73:86:6c:38:fc:9b:e4:45:14:70:
                    20:e4:2c:b1:84
                ASN1 OID: prime256v1
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                39:74:80:E3:03:55:AC:F6:6F:2C:69:48:FF:5E:78:DF:FF:48:99:BF
            X509v3 Authority Key Identifier:
                keyid:D4:54:0A:34:1A:9F:1E:9C:4C:60:2E:B0:CB:
89:57:80:B1:1F:E3:23

    Signature Algorithm: ecdsa-with-SHA256
         30:45:02:20:28:5f:23:ca:ed:9c:42:f8:5b:57:b7:03:44:24:
         8b:e9:20:88:13:c1:d9:c4:67:52:6b:eb:47:ca:31:95:77:5a:
         02:21:00:ad:4b:b0:cf:4d:ba:35:0e:9d:aa:1b:20:44:52:4c:
         18:3c:a7:11:7f:7e:a7:75:10:7f:da:78:30:d6:f7:f5:47
```

ESM – DSA Signed Certificate Example

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
        Signature Algorithm: dsa_with_SHA256
        Issuer: C=US, ST=Pennsylvania, L=King of Prussia, O=OPA Sample
Corporation, OU=OPA Sample Division, CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Validity
            Not Before: Nov  4 19:38:48 2014 GMT
            Not After : Nov  4 19:38:48 2015 GMT
        Subject: CN=phgppriv03.ph.intel.com, C=US, ST=Pennsylvania, O=OPA
```

```
Sample Corporation/emailAddress=root@phgppriv03.ph.intel.com, OU=OPA Sample
Division
        Subject Public Key Info:
            Public Key Algorithm: dsaEncryption
                pub:
                    13:45:a0:d7:68:e5:03:7d:92:9f:2c:76:f5:64:7e:
                    5d:b5:89:22:50:05:09:70:a6:5e:77:d6:5a:e8:a7:
                    ef:8c:eb:bf:22:12:ca:34:03:4d:dd:50:6a:4d:20:
                    14:94:fd:eb:5c:c3:64:59:de:b4:0b:9a:a9:c7:48:
                    4c:1d:88:4a:ad:e4:d0:ca:22:9f:58:69:fc:5a:7d:
                    a9:16:80:c1:81:8f:24:0e:75:35:f0:1d:eb:3a:03:
                    0f:36:fd:40:56:96:a9:05:72:f5:a4:35:83:78:18:
                    8c:e5:57:94:6f:a1:7b:e5:9c:a8:4b:47:31:15:39:
                    86:94:0a:de:50:e1:03:d7:83:3c:cd:b8:23:79:87:
                    0d:3d:02:77:0c:6b:7e:2e:ba:4d:e2:ee:f9:ad:83:
                    47:c2:2f:e1:1f:5e:0c:81:45:85:59:72:08:b3:9a:
                    67:28:0a:95:71:92:85:cf:ac:e2:21:a8:3b:f5:51:
                    59:24:e2:d9:77:9a:31:e3:5f:71:5f:86:4b:98:31:
                    90:32:1d:86:a0:6e:57:01:fa:a4:a1:d6:8f:9f:6d:
                    b5:00:0b:6b:f0:f8:23:18:8d:56:43:72:f4:ad:68:
                    ae:a1:20:95:d9:e6:ea:de:db:c4:5b:0f:f0:dd:4f:
                    d8:db:00:05:f7:17:43:1e:24:54:30:b0:55:1f:db:
                    3b
                P:
                    00:d6:2c:7b:d6:1f:66:b1:f7:cd:b4:23:eb:bc:ab:
                    ca:0a:3f:d4:26:d8:00:22:5b:56:66:42:54:95:d8:
                    79:f3:6c:c2:9a:75:22:09:35:60:2c:d4:56:e3:44:
                    b5:4c:72:c3:26:2a:3c:f9:8c:64:a7:a8:16:0b:2e:
                    0a:17:48:24:8c:ed:15:98:1a:e0:29:0d:2e:48:42:
                    32:e2:f8:47:ec:35:57:48:c8:68:3d:e3:70:b9:c8:
                    94:00:88:95:2d:10:79:6f:8b:86:bb:a2:8f:34:4b:
                    c8:cf:f5:12:10:ff:06:e5:39:52:9a:3b:fc:e5:f3:
                    77:0f:03:be:9d:11:9c:0b:ec:0a:a1:69:77:86:49:
                    68:d0:50:a1:d1:76:97:e0:84:7e:c7:0d:7a:72:f9:
                    5f:f9:6c:9d:57:5c:28:e1:0a:31:b8:6f:b8:11:bf:
                    ee:a0:d4:ee:46:4d:f7:bd:4e:f8:07:a3:72:26:c3:
                    50:41:ee:f6:ff:b8:c4:f2:80:79:4f:38:f5:3e:f9:
                    f8:3a:4d:58:af:3f:3a:92:64:48:54:da:a6:e9:00:
                    08:f6:63:e9:a5:15:41:a6:95:e0:19:fb:8e:1c:f8:
                    98:87:ee:97:5c:f8:ea:8f:d4:d8:16:15:61:5c:e0:
                    57:bd:a0:10:72:66:06:04:a2:8d:36:db:d3:9a:59:
                    8e:03
                Q:
                    00:d8:71:93:e1:34:1c:63:73:76:1e:9e:c2:f4:84:
                    41:bf:de:89:5e:f4:b8:a4:84:16:d5:9f:0f:bd:19:
                    29:8c:d5
                G:
                    28:80:f0:f5:d7:b3:09:c6:dc:f1:4a:c0:b8:3d:56:
                    f4:8d:a7:e7:68:6f:1a:99:4a:7d:5f:a2:d0:eb:76:
                    f8:c3:d6:42:80:ce:89:64:af:bf:18:03:df:b3:4a:
                    49:a9:3f:21:37:5e:35:85:d0:96:3f:c7:f7:43:eb:
                    9f:b7:2a:a4:c1:62:63:6c:2d:e6:72:26:f1:c7:40:
                    d4:35:14:4e:b8:1d:ef:6e:72:2d:18:29:d7:11:2b:
                    d6:f4:11:ef:0b:4f:e4:4a:d2:e1:37:bb:d8:fd:ee:
                    e2:4a:3e:b1:3b:f1:24:21:0b:77:9b:5c:c3:42:26:
                    bd:67:d1:95:1e:fd:b0:c8:ec:67:02:b1:99:d0:4e:
                    72:af:33:eb:36:fa:4b:28:c3:00:7f:e0:55:11:d0:
                    5e:0d:7d:3d:be:b1:51:c5:ec:1c:02:fe:52:6e:56:
                    ea:8e:ec:0f:bf:0e:f5:e7:2d:1b:f6:c9:73:3f:d5:
                    8a:4e:45:6d:dc:91:e0:5e:a3:d1:65:b4:f4:f4:20:
                    07:f5:71:9e:e6:7f:0b:dc:62:41:62:5f:7a:3b:7b:
                    f7:81:bb:dc:00:de:0b:4d:38:33:cf:da:00:21:c3:
                    5c:85:8d:0b:67:bc:fd:4f:f1:29:c1:7f:36:3a:42:
                    bc:16:73:8e:86:8c:9c:7e:29:e3:82:1b:9f:1a:1b:
                    3c
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
```



```
                E5:07:4C:A2:A5:B2:0A:3A:84:40:CE:12:8C:F1:27:D8:19:2B:AE:46
            X509v3 Authority Key Identifier:
                keyid:B4:2B:EC:40:08:CD:AD:57:31:E8:0B:3F:
1B:D4:B8:31:B3:AD:B6:D9

    Signature Algorithm: dsa_with_SHA256
        30:45:02:20:4d:22:b9:93:5a:da:5c:03:e2:b9:be:94:2e:eb:
        af:89:fa:42:42:a2:c1:e8:bb:d5:a5:cd:48:43:10:6c:ab:ff:
        02:21:00:9e:80:2c:54:14:46:fd:53:ca:04:25:9e:4a:a2:ca:
        3c:74:09:35:75:83:aa:01:22:27:3c:94:65:64:18:9d:df
```

7. Install the self-signed CA and component certificates and the self-signed CRL. Reference Core-Level Public Key Infrastructure (PKI) Best Practices Guidelines on page 230 for detailed instructions on the installation of these files.

### J.1.1.5 Revoking Certificates

1. Move to the OPA work directory:

```
cd ~/opaCA/
```

2. Direct the OpenSSL environment to use the customized `opa_ca_openssl.cnf` configuration file of the self-signed CA:

```
export OPENSSL_CONF=~/opaCA/opa_ca_openssl.cnf
```

3. Execute the following OpenSSL command to revoke the signed certificate of the OPA component:

```
openssl ca -revoke <intel-opa-comp-cert>.pem
```

Output:

```
Using configuration from /home/kingchar/opaCA/opa_ca_openssl.cnf
Enter pass phrase for /home/kingchar/opaCA/private/cakey.pem:
Revoking Certificate 01.
Data Base Updated
```

### J.1.1.6 Generating a Certificate Revocation List (CRL)

1. Move to the OPA work directory:

```
cd ~/opaCA/
```

2. Direct the OpenSSL environment to use the customized `opa_ca_openssl.cnf` configuration file of the self-signed CA:

```
export OPENSSL_CONF=~/opaCA/opa_ca_openssl.cnf
```

3. Execute the following OpenSSL command to generate a new CRL:

```
openssl ca -gencrl -out crl/ca.crl
```

Output:

```
Using configuration from /home/kingchar/opaCA/opa_ca_openssl.cnf
Enter pass phrase for /home/kingchar/opaCA/private/cakey.pem:
```

4. Execute the following OpenSSL command to validate the content of the new CRL:

```
openssl crl -in crl/ca.crl -noout -text
```

#### J.1.1.6.1 HSM – EC Signed CRL Example

```
Certificate Revocation List (CRL):
        Version 2 (0x1)
    Signature Algorithm: ecdsa-with-SHA256
        Issuer: /C=US/ST=Pennsylvania/L=King of Prussia/O=OPA Sample Corporation/
OU=OPA Sample Division/CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Last Update: Nov 15 07:10:22 2014 GMT
        Next Update: Dec 15 07:10:22 2014 GMT
        CRL extensions:
            X509v3 CRL Number:
                1
No Revoked Certificates.
    Signature Algorithm: ecdsa-with-SHA256
         30:45:02:20:6d:37:e5:5d:89:f0:03:72:c5:da:12:f7:06:44:
         d7:eb:1d:90:e9:08:b6:f7:e2:0e:e1:43:78:fd:1d:97:1b:4d:
         02:21:00:c3:c9:db:53:d2:0f:69:21:2d:c5:a1:b7:71:48:9d:
         b6:d5:e4:1f:52:df:0b:c0:ea:bc:97:e4:8c:64:ca:ff:ff
```

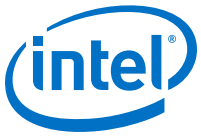
#### J.1.1.6.2 ESM - DSA Signed CRL Example

```
Certificate Revocation List (CRL):
        Version 2 (0x1)
        Signature Algorithm: dsa_with_SHA256
        Issuer: /C=US/ST=Pennsylvania/L=King of Prussia/O=OPA Sample Corporation/
OU=OPA Sample Division/CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
        Last Update: Nov  4 20:48:36 2014 GMT
        Next Update: Dec  4 20:48:36 2014 GMT
        CRL extensions:
            X509v3 CRL Number:
                1
Revoked Certificates:
    Serial Number: 01
        Revocation Date: Nov  4 20:33:02 2014 GMT
    Signature Algorithm: dsa_with_SHA256
        30:46:02:21:00:ab:56:8f:d6:22:7f:6e:2b:2a:21:14:6d:32:
        8d:87:87:c2:39:a7:c4:f9:2f:b9:b4:ef:43:2f:b8:d8:6a:18:
        d6:02:21:00:d5:13:d1:08:45:0c:34:3c:f8:84:af:5a:b0:14:
        7c:b1:92:2b:0e:fb:40:95:b1:54:9b:15:b8:f3:92:1a:cc:f4
```

### J.1.2 Diagnostics

Openssl provides a generic server and client command-line diagnostic utilities that could be used to verify/debug SSL/TLS connections. The following sections provide steps on how to use these tools.

#### J.1.2.1 OpenSSL s_server



1. Log onto the Linux/UNIX server root access that is to be used as the OpenSSL server.
2. Create a work directory for the OpenSSL server:

```
mkdir opaServer
```

3. Move to the server work directory:

```
cd ~/ opaServer/
```

4. Copy all OPA FM related PEM files to the server work directory:

```
cp /usr/local/ssl/ifs_fm/fm*.pem
```

5. Execute the following OpenSSL command to start the openssl server:

```
openssl s_server -cert fm_cert.pem -key fm_key.pem -dhparam fm_dh_parms.pem -
CAfile fm_ca_cert.pem -bugs -tls1_2 -debug -cipher ECDHE-ECDSA-AES128-GCM-
SHA256
```

Output:

```
Setting temp DH parameters
Using default temp ECDH parameters
ACCEPT
```

### J.1.2.2 OpenSSL s_client

1. Log onto the Linux/UNIX server root access that is to be used as the OpenSSL client
2. Create a work directory for the openssl client:

```
mkdir opaClient
```

3. Move to the client work directory:

```
cd ~/ opaClient/
```

4. Copy all OPA FF related PEM files to the client work directory:

```
cp /usr/local/ssl/ifs_fm/ff*.pem
```

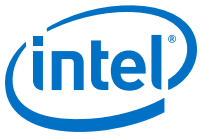
5. Execute the following OpenSSL command to start the openssl client:

```
openssl s_client -cert ff_cert.pem -key ff_key.pem -CAfile ff_ca_cert.pem -
bugs -debug -tls1_2 -cipher ECDHE-ECDSA-AES128-GCM-SHA256
```

Output:

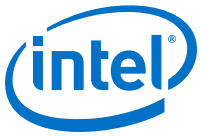
```
SHA256
CONNECTED(00000003)
write to 0x27c3710 [0x27d1243] (119 bytes => 119 (0x77))
```

```
0000 - 16 03 01 00 72 01 00 00-6e 03 03 54 6f 8c 37 be   ....r...n..To.7.
0010 - 2c 3e 5c 98 67 93 cc a5-59 67 89 83 13 f5 d2 22   ,>\.g...Yg....."
0020 - be 50 6e 00 26 2c 79 a5-aa 0a 30 00 00 04 c0 2b   .Pn.&,y...0....+
0030 - 00 ff 01 00 00 41 00 0b-00 04 03 00 01 02 00 0a   .....A..........
0040 - 00 06 00 04 00 18 00 17-00 23 00 00 00 0d 00 22   .........#.....".
0050 - 00 20 06 01 06 02 06 03-05 01 05 02 05 03 04 01   . ..............
0060 - 04 02 04 03 03 01 03 02-03 03 02 01 02 02 02 03   ................
0070 - 01 01 00 0f 00 01 01                              .......
read from 0x27c3710 [0x27c8cf3] (5 bytes => 5 (0x5))
0000 - 16 03 03 00 42                                    ....B
read from 0x27c3710 [0x27c8cf8] (66 bytes => 66 (0x42))
0000 - 02 00 00 3e 03 03 54 6f-8c 37 00 d9 93 0f d0 95   ...>..To.7......
0010 - c2 ed c3 45 e5 fd a6 94-5b df 7e 73 fd 85 0f 4e   ...E....[.~s...N
0020 - cf 77 10 e6 14 75 00 c0-2b 00 00 16 ff 01 00 01   .w...u..+.......
0030 - 00 00 0b 00 04 03 00 01-02 00 23 00 00 00 0f 00   ..........#.....
0040 - 01 01                                             ..
read from 0x27c3710 [0x27c8cf3] (5 bytes => 5 (0x5))
0000 - 16 03 03 05 f4                                    .....
read from 0x27c3710 [0x27c8cf8] (1524 bytes => 1524 (0x5F4))
0000 - 0b 00 05 f0 00 05 ed 00-03 02 30 82 02 fe 30 82   ..........0...0.
0010 - 02 a4 a0 03 02 01 02 02-01 01 30 0a 06 08 2a 86   ..........0...*.
0020 - 48 ce 3d 04 03 02 30 81-cc 31 0b 30 09 06 03 55   H.=...0..1.0...U
0030 - 04 06 13 02 55 53 31 15-30 13 06 03 55 04 08 0c   ....US1.0...U...
0040 - 0c 50 65 6e 6e 73 79 6c-76 61 6e 69 61 31 18 30   .Pennsylvania1.0
0050 - 16 06 03 55 04 07 0c 0f-4b 69 6e 67 20 6f 66 20   ...U....King of
0060 - 50 72 75 73 73 69 61 31-1f 30 1d 06 03 55 04 0a   Prussia1.0...U..
0070 - 0c 16 4f 50 41 20 53 61-6d 70 6c 65 20 43 6f 72   ..OPA Sample Cor
0080 - 70 6f 72 61 74 69 6f 6e-31 1c 30 1a 06 03 55 04   poration1.0...U.
0090 - 0b 0c 13 4f 50 41 20 53-61 6d 70 6c 65 20 44 69   ...OPA Sample Di
00a0 - 76 69 73 69 6f 6e 31 20-30 1e 06 03 55 04 03 0c   vision1 0...U...
00b0 - 17 70 68 67 70 70 72 69-76 30 33 2e 70 68 2e 69   .phgppriv03.ph.i
00c0 - 6e 74 65 6c 2e 63 6f 6d-31 2b 30 29 06 09 2a 86   ntel.com1+0)..*.
00d0 - 48 86 f7 0d 01 09 01 16-1c 72 6f 6f 74 40 70 68   H........root@ph
00e0 - 67 70 70 72 69 76 30 33-2e 70 68 2e 69 6e 74 65   gppriv03.ph.inte
00f0 - 6c 2e 63 6f 6d 30 1e 17-0d 31 34 31 31 31 35 30   l.com0...1411150
0100 - 37 30 36 30 34 5a 17 0d-31 35 31 31 31 35 30 37   70604Z..15111507
0110 - 30 36 30 34 5a 30 81 c6-31 20 30 1e 06 03 55 04   0604Z0..1 0...U.
0120 - 03 0c 17 70 68 67 70 70-72 69 76 30 33 2e 70 68   ...phgppriv03.ph
0130 - 2e 69 6e 74 65 6c 2e 63-6f 6d 31 0b 30 09 06 03   .intel.com1.0...
0140 - 55 04 06 13 02 55 53 31-15 30 13 06 03 55 04 08   U....US1.0...U..
0150 - 0c 0c 50 65 6e 6e 73 79-6c 76 61 6e 69 61 31 29   ..Pennsylvania1)
0160 - 30 27 06 03 55 04 0a 0c-20 4f 50 41 20 43 6f 6d   0'..U... OPA Com
0170 - 70 6f 6e 65 6e 74 20 53-61 6d 70 6c 65 20 43 6f   ponent Sample Co
0180 - 72 70 6f 72 61 74 69 6f-6e 31 2b 30 29 06 09 2a   rporation1+0)..*
0190 - 86 48 86 f7 0d 01 09 01-16 1c 72 6f 6f 74 40 70   .H........root@p
01a0 - 68 67 70 70 72 69 76 30-33 2e 70 68 2e 69 6e 74   hgppriv03.ph.int
01b0 - 65 6c 2e 63 6f 6d 31 26-30 24 06 03 55 04 0b 0c   el.com1&0$..U...
01c0 - 1d 4f 50 41 20 43 6f 6d-70 6f 6e 65 6e 74 20 53   .OPA Component S
01d0 - 61 6d 70 6c 65 20 44 69-76 69 73 69 6f 6e 30 59   ample Division0Y
01e0 - 30 13 06 07 2a 86 48 ce-3d 02 01 06 08 2a 86 48   0...*.H.=....*.H
01f0 - ce 3d 03 01 07 03 42 00-04 8e 85 df 04 ac 36 81   .=....B.......6.
0200 - 7f 6a 09 58 8a de a1 7a-ab b2 00 34 e0 d3 9f 92   .j.X...z...4....
0210 - 2a da d9 6c 40 58 00 46-b0 55 15 4c 60 dd 55 40   *..l@X.F.U.L`.U@
0220 - 46 23 13 5c 65 30 26 01-3d a8 73 86 6c 38 fc 9b   F#.\e0&.=.s.l8..
0230 - e4 45 14 70 20 e4 2c b1-84 a3 7b 30 79 30 09 06   .E.p .,...{0y0..
0240 - 03 55 1d 13 04 02 30 00-30 2c 06 09 60 86 48 01   .U....0.0,..`.H.
0250 - 86 f8 42 01 0d 04 1f 16-1d 4f 70 65 6e 53 53 4c   ..B......OpenSSL
0260 - 20 47 65 6e 65 72 61 74-65 64 20 43 65 72 74 69    Generated Certi
0270 - 66 69 63 61 74 65 30 1d-06 03 55 1d 0e 04 16 04   ficate0...U.....
0280 - 14 39 74 80 e3 03 55 ac-f6 6f 2c 69 48 ff 5e 78   .9t...U..o,iH.^x
0290 - df ff 48 99 bf 30 1f 06-03 55 1d 23 04 18 30 16   ..H..0...U.#..0.
02a0 - 80 14 d4 54 0a 34 1a 9f-1e 9c 4c 60 2e b0 cb 89   ...T.4....L`....
02b0 - 57 80 b1 1f e3 23 30 0a-06 08 2a 86 48 ce 3d 04   W....#0...*.H.=.
02c0 - 03 02 03 48 00 30 45 02-20 28 5f 23 ca ed 9c 42   ...H.0E. (_#...B
02d0 - f8 5b 57 b7 03 44 24 8b-e9 20 88 13 c1 d9 c4 67   .[W..D$.. .....g
02e0 - 52 6b eb 47 ca 31 95 77-5a 02 21 00 ad 4b b0 cf   Rk.G.1.wZ.!..K..
02f0 - 4d ba 35 0e 9d aa 1b 20-44 52 4c 18 3c a7 11 7f   M.5.... DRL.<...
0300 - 7e a7 75 10 7f da 78 30-d6 f7 f5 47 00 02 e5 30   ~.u...x0...G...0
0310 - 82 02 e1 30 82 02 87 a0-03 02 01 02 02 09 00 d3   ...0............
0320 - 2a a4 2e 20 15 51 aa 30-0a 06 08 2a 86 48 ce 3d   *.. .Q.0...*.H.=
0330 - 04 03 02 30 81 cc 31 0b-30 09 06 03 55 04 06 13   ...0..1.0...U...
```



```
0340 - 02 55 53 31 15 30 13 06-03 55 04 08 0c 0c 50 65   .US1.0...U....Pe
0350 - 6e 6e 73 79 6c 76 61 6e-69 61 31 18 30 16 06 03   nnsylvania1.0...
0360 - 55 04 07 0c 0f 4b 69 6e-67 20 6f 66 20 50 72 75   U....King of Pru
0370 - 73 73 69 61 31 1f 30 1d-06 03 55 04 0a 0c 16 4f   ssia1.0...U....O
0380 - 50 41 20 53 61 6d 70 6c-65 20 43 6f 72 70 6f 72   PA Sample Corpor
0390 - 61 74 69 6f 6e 31 1c 30-1a 06 03 55 04 0b 0c 13   ation1.0...U....
03a0 - 4f 50 41 20 53 61 6d 70-6c 65 20 44 69 76 69 73   OPA Sample Divis
03b0 - 69 6f 6e 31 20 30 1e 06-03 55 04 03 0c 17 70 68   ion1 0...U....ph
03c0 - 67 70 70 72 69 76 30 33-2e 70 68 2e 69 6e 74 65   gppriv03.ph.inte
03d0 - 6c 2e 63 6f 6d 31 2b 30-29 06 09 2a 86 48 86 f7   l.com1+0)..*.H..
03e0 - 0d 01 09 01 16 1c 72 6f-6f 74 40 70 68 67 70 70   ......root@phgpp
03f0 - 72 69 76 30 33 2e 70 68-2e 69 6e 74 65 6c 2e 63   riv03.ph.intel.c
0400 - 6f 6d 30 1e 17 0d 31 34-31 31 31 35 30 36 35 30   om0...1411150650
0410 - 31 34 5a 17 0d 31 34 31-32 31 35 30 36 35 30 31   14Z..14121506501
0420 - 34 5a 30 81 cc 31 0b 30-09 06 03 55 04 06 13 02   4Z0..1.0...U....
0430 - 55 53 31 15 30 13 06 03-55 04 08 0c 0c 50 65 6e   US1.0...U....Pen
0440 - 6e 73 79 6c 76 61 6e 69-61 31 18 30 16 06 03 55   nsylvania1.0...U
0450 - 04 07 0c 0f 4b 69 6e 67-20 6f 66 20 50 72 75 73   ....King of Prus
0460 - 73 69 61 31 1f 30 1d 06-03 55 04 0a 0c 16 4f 50   sia1.0...U....OP
0470 - 41 20 53 61 6d 70 6c 65-20 43 6f 72 70 6f 72 61   A Sample Corpora
0480 - 74 69 6f 6e 31 1c 30 1a-06 03 55 04 0b 0c 13 4f   tion1.0...U....O
0490 - 50 41 20 53 61 6d 70 6c-65 20 44 69 76 69 73 69   PA Sample Divisi
04a0 - 6f 6e 31 20 30 1e 06 03-55 04 03 0c 17 70 68 67   on1 0...U....phg
04b0 - 70 70 72 69 76 30 33 2e-70 68 2e 69 6e 74 65 6c   ppriv03.ph.intel
04c0 - 2e 63 6f 6d 31 2b 30 29-06 09 2a 86 48 86 f7 0d   .com1+0)..*.H...
04d0 - 01 09 01 16 1c 72 6f 6f-74 40 70 68 67 70 70 72   .....root@phgppr
04e0 - 69 76 30 33 2e 70 68 2e-69 6e 74 65 6c 2e 63 6f   iv03.ph.intel.co
04f0 - 6d 30 59 30 13 06 07 2a-86 48 ce 3d 02 01 06 08   m0Y0...*.H.=....
0500 - 2a 86 48 ce 3d 03 01 07-03 42 00 04 8e f0 a3 df   *.H.=....B......
0510 - 62 05 58 ec 32 3f d9 6b-83 6d 41 23 ed 2e 20 c4   b.X.2?.k.mA#.. .
0520 - 61 f5 0b 24 74 1b 49 51-59 2e dc d2 f7 63 89 d4   a..$t.IQY....c..
0530 - be c7 bf a9 e0 ff da 0b-33 f0 3a 89 21 22 3a 7b   ........3.:.!":{
0540 - e6 94 9b c3 1c 0b cc 7a-f2 f5 be c5 a3 50 30 4e   .......z.....P0N
0550 - 30 1d 06 03 55 1d 0e 04-16 04 14 d4 54 0a 34 1a   0...U.......T.4.
0560 - 9f 1e 9c 4c 60 2e b0 cb-89 57 80 b1 1f e3 23 30   ...L`....W....#0
0570 - 1f 06 03 55 1d 23 04 18-30 16 80 14 d4 54 0a 34   ...U.#..0....T.4
0580 - 1a 9f 1e 9c 4c 60 2e b0-cb 89 57 80 b1 1f e3 23   ....L`....W....#
0590 - 30 0c 06 03 55 1d 13 04-05 30 03 01 01 ff 30 0a   0...U....0....0.
05a0 - 06 08 2a 86 48 ce 3d 04-03 02 03 48 00 30 45 02   ..*.H.=....H.0E.
05b0 - 21 00 a3 d7 db e9 a9 20-c5 ef c8 ca 40 9e c9 00   !...... ....@...
05c0 - 60 51 eb 08 9f 09 56 7c-4a 10 03 26 3e 61 f9 0c   `Q....V|J..&>a..
05d0 - 29 1b 02 20 01 14 32 6f-d6 44 20 44 b2 ff 50 b1   ).. ..2o.D D..P.
05e0 - 2f 68 f5 72 7e 80 53 15-04 10 f3 4b 0b 6f ac 84   /h.r~.S....K.o..
05f0 - 7c e9 95 df                                       |...
depth=1 C = US, ST = Pennsylvania, L = King of Prussia, O = OPA Sample
Corporation, OU = OPA Sample Division, CN = phgppriv03.ph.intel.com,
emailAddress = root@phgppriv03.ph.intel.com
verify return:1
depth=0 CN = phgppriv03.ph.intel.com, C = US, ST = Pennsylvania, O = OPA
Component Sample Corporation, emailAddress = root@phgppriv03.ph.intel.com, OU
= OPA Component Sample Division
verify return:1
read from 0x27c3710 [0x27c8cf3] (5 bytes => 5 (0x5))
0000 - 16 03 03 00 93                                    .....
read from 0x27c3710 [0x27c8cf8] (147 bytes => 147 (0x93))
0000 - 0c 00 00 8f 03 00 17 41-04 5a a1 1d d0 42 99 9f   .......A.Z...B..
0010 - ee 94 ee d0 a7 b6 7b be-46 40 27 8c ab df 28 3a   ......{.F@'...(:
0020 - bf 5d 04 53 7f d3 63 f9-fa 77 93 a0 f9 b1 90 b5   .].S..c..w......
0030 - 42 0f ec e4 25 62 9a 0d-22 28 87 1f 96 6f b2 a6   B...%b.."(...o..
0040 - 4f ca e0 01 e4 74 f6 fb-84 06 03 00 46 30 44 02   O....t......F0D.
0050 - 20 57 d8 e0 38 d6 c4 c4-61 f6 6b bc b2 43 94 a2    W..8...a.k..C..
0060 - 82 94 27 d2 8d 09 49 a7-62 bc 7c 86 ee db c6 fb   ..'...I.b.|.....
0070 - be 02 20 68 0e ca 64 38-77 b8 d4 ad 3b 26 3e 09   .. h..d8w...;&>.
0080 - 8a 26 1b 78 4e 2e f3 b2-3c 0b cf e3 dd 8a c0 88   .&.xN...<.......
0090 - c1 4e ea                                          .N.
read from 0x27c3710 [0x27c8cf3] (5 bytes => 5 (0x5))
0000 - 16 03 03 00 04                                    .....
read from 0x27c3710 [0x27c8cf8] (4 bytes => 4 (0x4))
0000 - 0e                                                .
0004 - <SPACES/NULS>
write to 0x27c3710 [0x27d66b0] (75 bytes => 75 (0x4B))
```

```
0000 - 16 03 03 00 46 10 00 00-42 41 04 d3 83 3d e0 f8   ....F...BA...=..
0010 - a7 dd cb 62 f3 cc 2d d7-7d 45 db cb 9d 04 0f 74   ...b..-.}E.....t
0020 - a4 f6 84 87 27 6c 34 67-38 e4 ca 48 42 bc 46 ba   ....'l4g8..HB.F.
0030 - 7b 80 61 85 3d cf c3 d2-f8 e1 67 a2 e1 1e 41 e3   {.a.=.....g...A.
0040 - d5 f4 ae 0d b7 0f 41 a9-59 14 b3                  ......A.Y..
write to 0x27c3710 [0x27d66b0] (6 bytes => 6 (0x6))
0000 - 14 03 03 00 01 01                                 ......
write to 0x27c3710 [0x27d66b0] (45 bytes => 45 (0x2D))
0000 - 16 03 03 00 28 f1 9c ad-6e ab 23 c2 22 30 79 b0   ....(...n.#."0y.
0010 - 93 90 74 0c 9b 98 f7 d4-39 ef 0f 8f 58 de 59 90   ..t.....9...X.Y.
0020 - f6 3f 2e b0 93 93 7d e7-a4 bf fd d5 9d            .?....}......
read from 0x27c3710 [0x27c8cf3] (5 bytes => 5 (0x5))
0000 - 16 03 03 00 aa                                    .....
read from 0x27c3710 [0x27c8cf8] (170 bytes => 170 (0xAA))
0000 - 04 00 00 a6 00 00 1c 20-00 a0 27 d9 a1 36 f2 e9   ....... ..'..6..
0010 - df aa de d6 0d 49 cd d2-10 be 7f 08 4e 3c 5f 12   .....I......N<_.
0020 - 70 02 f4 e8 bb bb 79 23-51 40 e0 45 cb c1 cd a4   p.....y#Q@.E....
0030 - 98 8f 6f c9 d2 0d b1 a1-36 f1 d9 9a 3a bb 22 a8   ..o.....6...:.".
0040 - f8 2b 26 76 18 68 b5 c4-ee 69 07 19 27 67 61 04   .+&v.h...i..'ga.
0050 - 61 b8 af 69 42 32 9f d3-3f fa 28 63 f4 c3 cd e5   a..iB2..?.(c....
0060 - 66 a4 4e b8 69 b9 fe 8f-df 20 74 43 4d 1b 30 9f   f.N.i.... tCM.0.
0070 - ec 1b b2 8c f6 52 2d c1-e2 0f 3b e3 36 a8 3a 76   .....R-...;.6.:v
0080 - 32 42 50 78 c8 aa e2 e5-98 45 73 9b b3 38 41 9a   2BPx.....Es..8A.
0090 - 5f 6c 17 c9 3b a6 8a b5-1e cf f7 99 26 f9 51 0d   _l..;.......&.Q.
00a0 - cd 10 cb 6d 15 b2 01 7d-46 a6                     ...m...}F.
read from 0x27c3710 [0x27c8cf3] (5 bytes => 5 (0x5))
0000 - 14 03 03 00 01                                    .....
read from 0x27c3710 [0x27c8cf8] (1 bytes => 1 (0x1))
0000 - 01                                                .
read from 0x27c3710 [0x27c8cf3] (5 bytes => 5 (0x5))
0000 - 16 03 03 00 28                                    ....(
read from 0x27c3710 [0x27c8cf8] (40 bytes => 40 (0x28))
0000 - c4 dc db 3f 9c 9d 4d 7e-bb 82 14 c5 b0 2b 13 35   ...?..M~.....+.5
0010 - 49 2a 35 31 a8 d1 20 8c-06 ef de 6d cd 42 bb 7f   I*51.. ....m.B..
0020 - 4a 1d 96 63 a8 6e b3 6c-                          J..c.n.l
---
Certificate chain
 0 s:/CN=phgppriv03.ph.intel.com/C=US/ST=Pennsylvania/O=OPA Component Sample
Corporation/emailAddress=root@phgppriv03.ph.intel.com/OU=OPA Component Sample
Division
   i:/C=US/ST=Pennsylvania/L=King of Prussia/O=OPA Sample Corporation/OU=OPA
Sample Division/CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
 1 s:/C=US/ST=Pennsylvania/L=King of Prussia/O=OPA Sample Corporation/OU=OPA
Sample Division/CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
   i:/C=US/ST=Pennsylvania/L=King of Prussia/O=OPA Sample Corporation/OU=OPA
Sample Division/CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIC/jCCAqSgAwIBAgIBATAKBggqhkjOPQQDAjCBzDELMAkGA1UEBhMCVVMxFTAT
BgNVBAgMDFBlbm5zeWx2YW5pYTEYMBYGA1UEBwwPS2luZyBvZiBQcnVzc2lhMR8w
HQYDVQQKDBZPUEEgU2FtcGxlIENvcnBvcmF0aW9uMRwwGgYDVQQLDBNPUEEgU2Ft
cGxlIERpdmlzaW9uMSAwHgYDVQQDDBdwaGdwcHJpdjAzLnBoLmludGVsLmNvbTEr
MCkGCSqGSIb3DQEJARYccm9vdEBwaGdwcHJpdjAzLnBoLmludGVsLmNvbTAeFw0x
NDExMTUwNzA2MDRaFw0xNTExMTUwNzA2MDRaMIHGMSAwHgYDVQQDDBdwaGdwcHJp
djAzLnBoLmludGVsLmNvbTELMAkGA1UEBhMCVVMxFTATBgNVBAgMDFBlbm5zeWx2
YW5pYTEpMCcGA1UECgwgT1BBIENvbXBvbmVudCBTYW1wbGUgQ29ycG9yYXRpb24x
KzApBgkqhkiG9w0BCQEWHHJvb3RAcGhncHByaXYwMy5waC5pbnRlbC5jb20xJjAk
BgNVBAsMHU9QQSBDb21wb25lbnQgU2FtcGxlIERpdmlzaW9uMFkwEwYHKoZIzj0C
AQYIKoZIzj0DAQcDQgAEjoXfBKw2gX9qCViK3qF6q7IANODTn5Iq2tlsQFgARrBV
FUxg3VVARiMTXGUwJgE9qHOGbDj8m+RFFHAg5CyxhKN7MHkwCQYDVR0TBAIwADAs
BglghkgBhvhCAQ0EHxYdT3BlblNTTCBHZW5lcmF0ZWQgQ2VydGlmaWNhdGUwHQYD
VR0OBBYEFDl0gOMDVaz2byxpSP9eeN//SJm/MB8GA1UdIwQYMBaAFNRUCjQanx6c
TGAusMuJV4CxH+MjMAoGCCqGSM49BAMCA0gAMEUCIChfI8rtnEL4W1e3A0Qki+kg
iBPB2cRnUmvrR8oxlXdaAiEArUuwz026NQ6dqhsgRFJMGDynEX9+p3UQf9p4MNb3
9Uc=
-----END CERTIFICATE-----
subject=/CN=phgppriv03.ph.intel.com/C=US/ST=Pennsylvania/O=OPA Component
```



```
Sample Corporation/emailAddress=root@phgppriv03.ph.intel.com/OU=OPA Component
Sample Division
issuer=/C=US/ST=Pennsylvania/L=King of Prussia/O=OPA Sample Corporation/
OU=OPA Sample Division/CN=phgppriv03.ph.intel.com/
emailAddress=root@phgppriv03.ph.intel.com
---
No client certificate CA names sent
---
SSL handshake has read 1987 bytes and written 245 bytes
---
New, TLSv1/SSLv3, Cipher is ECDHE-ECDSA-AES128-GCM-SHA256
Server public key is 256 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : TLSv1.2
    Cipher    : ECDHE-ECDSA-AES128-GCM-SHA256
    Session-ID:
4C96AD2276C116E9E4FE3686B1E09B61C581662C3717C91C2C7F29BC91FF5DDF
    Session-ID-ctx:
    Master-Key:
0CB866CE453B30CF41B9F967973F64FEE44EDA3982B30C30B7088A6458DB760DF14F689A5A7134
9E21D2A6BDFDB2AA4B
    Key-Arg   : None
    Krb5 Principal: None
    PSK identity: None
    PSK identity hint: None
    TLS session ticket lifetime hint: 7200 (seconds)
    TLS session ticket:
    0000 - 27 d9 a1 36 f2 e9 df aa-de d6 0d 49 cd d2 10 be   '..6.......I....
    0010 - 7f 08 4e 3c 5f 12 70 02-f4 e8 bb bb 79 23 51 40   ..N<_.p.....y#Q@
    0020 - e0 45 cb c1 cd a4 98 8f-6f c9 d2 0d b1 a1 36 f1   .E......o.....6.
    0030 - d9 9a 3a bb 22 a8 f8 2b-26 76 18 68 b5 c4 ee 69   ..:."..+&v.h...i
    0040 - 07 19 27 67 61 04 61 b8-af 69 42 32 9f d3 3f fa   ..'ga.a..iB2..?.
    0050 - 28 63 f4 c3 cd e5 66 a4-4e b8 69 b9 fe 8f df 20   (c....f.N.i.... 
    0060 - 74 43 4d 1b 30 9f ec 1b-b2 8c f6 52 2d c1 e2 0f   tCM.0......R-...
    0070 - 3b e3 36 a8 3a 76 32 42-50 78 c8 aa e2 e5 98 45   ;.6.:v2BPx.....E
    0080 - 73 9b b3 38 41 9a 5f 6c-17 c9 3b a6 8a b5 1e cf   s..8A._l..;.....
    0090 - f7 99 26 f9 51 0d cd 10-cb 6d 15 b2 01 7d 46 a6   ..&.Q....m...}F.

    Start Time: 1416596535
    Timeout   : 7200 (sec)
    Verify return code: 0 (ok)
---
```

Now you can begin to type any message on the client window or server window to send messages over the secure TLS session.

# Appendix K SSL Key Creation for Fabric Manager GUI

## K.1 Overview

This section is intended to guide the user in the creation and installation of SSL keys to allow the Fabric Manager GUI to interact with the FM. On the FM Linux host, PEM files are generated for the FM. P12 and PEM files are generated for the Fabric Manager GUI and transferred to the Fabric Manager GUI host (a Microsoft Windows host in the example below). Lastly, the Fabric Manager GUI key and trust stores are updated.

## K.2 SSL Key Creation

### K.2.1 Generate and Install Certificates for FM

Refer to Core-Level Public Key Infrastructure (PKI) Best Practices Guidelines on page 230 for detailed instructions on generating and installing certificates.

### K.2.2 Generate and Transfer Certificates for Fabric Manager GUI

1. Refer to Core-Level Public Key Infrastructure (PKI) Best Practices Guidelines on page 230 for detailed instructions on generating and installing certificates.
2. Logged in as "root" on the Linux host, create the .p12 file:

```
[root@linuxhost]# openssl pkcs12 -export -in ifs_ff_cert.pem -inkey
ifs_ff_priv_key.pem > fv_cert.p12

Enter Export Password:
Verifying - Enter Export Password:
```

3. Transfer the `fv_cert.p12` and `opaca_cert.pem` files to the Fabric Manager GUI host.

### K.2.3 Import Certificates to Fabric Manager GUI Key Store and Trust Store

1. Create the key store and enter new password:

```
C:\temp\ssl>keytool -importkeystore -srckeystore fv_cert.p12 -destkeystore
fv.jks
-srcstoretype pkcs12

Picked up _JAVA_OPTIONS: -Djava.net.preferIPv4Stack=true
Enter destination keystore password:
Re-enter new password:
```

```
Enter source keystore password:

Entry for alias 1 successfully imported.
Import command completed:  1 entries successfully imported, 0 entries failed
or
Cancelled
```

2. Create the trust store and enter the new password:

```
C:\temp\ssl>keytool -import -keystore truststore.jks -alias ca_cert -file
ifs_ca_cert.pem

Picked up _JAVA_OPTIONS: -Djava.net.preferIPv4Stack=true
Enter keystore password:
Re-enter new password:

Owner: EMAILADDRESS=root@phgppriv24.ph.intel.com, CN=phgppriv24.ph.intel.com,
OU
=OPA Sample Division, O=OPA Sample Corporation, L=King of Prussia,
ST=Pennsylvania, C=US
Issuer: EMAILADDRESS=root@phgppriv24.ph.intel.com,
CN=phgppriv24.ph.intel.com, O
U=OPA Sample Division, O=OPA Sample Corporation, L=King of Prussia,
ST=Pennsylvania, C=US
Serial number: ff32afe7ef4b442b
Valid from: Mon Jan 05 16:15:51 EST 2015 until: Tue Jan 05 16:15:51 EST 2016
Certificate fingerprints:
         MD5:  D8:29:F3:FA:68:8C:52:52:35:0F:D9:B8:62:F9:8F:29
         SHA1: 26:FF:27:1C:70:13:6D:ED:11:6E:34:3F:21:AD:9B:DE:38:CE:92:63
         SHA256: 5E:96:E0:15:95:1D:37:AC:E2:DD:6D:07:20:B6:07:9D:
7E:AD:E8:97:E1:
F1:F6:8E:31:EA:7C:A8:01:DC:90:3A
         Signature algorithm name: SHA256withECDSA
         Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: 64 42 56 B0 26 94 16 C5   AA 3D E3 0B F4 4A 43 0C  dBV.&....=...JC.
0010: 98 07 07 21                                        ...!
]
]

#2: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 64 42 56 B0 26 94 16 C5   AA 3D E3 0B F4 4A 43 0C  dBV.&....=...JC.
0010: 98 07 07 21                                        ...!
]
]
```

3. Review certificate and enter “y” to add certificate to key store:

```
Trust this certificate? [no]:  y
Certificate was added to keystore

C:\temp\ssl>
```

# Appendix L PM Counters Calculations

This section discusses the manipulation, weighting, and grouping of counters from both sides of the link within the PM for summarizing the state of the individual port.

## L.1 Categories

The PM bundles counters in several categories:

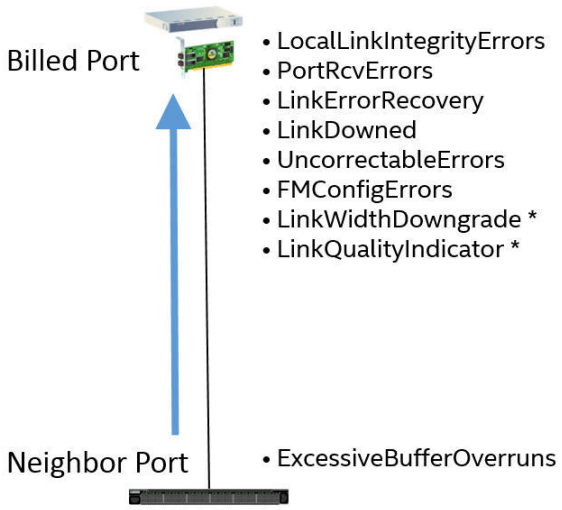
- Integrity
- Congestion
- Bubble
- Routing
- Security
- SMA Congestion

Each subsection contains a list of counters and their definitions. Each counter is gathered by the PM during its sweep. The counters are then grouped and passed to equations, which are described in each of the following subsections.

For a few counters that could not be bundled into the above categories, the PM also has a set of Calculated Counters to provide extra content.

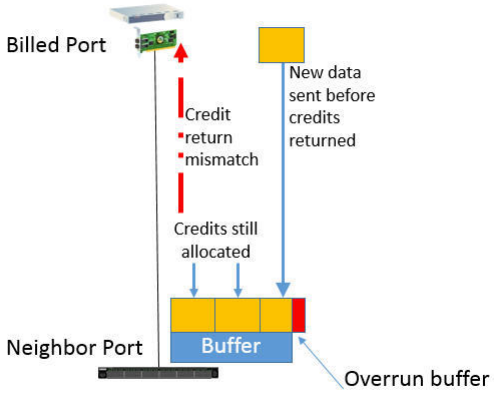
## L.2 Integrity Category Calculations

The integrity category calculation applies weights to each of the following counters before summing them for the final value:




*Note:* All Counters will be the delta value from the previous sweep unless otherwise noted (*). See LinkQualityIndicator Equation and Rationale on page 280 and LinkWidthDowngrade Equation and Rationale on page 280 for the equations and the rationale.

During calculation of the Integrity category, the `ExcessiveBufferOverruns` counter from the remote (neighbor) port is used for the local (billed) ports Integrity calculation because this counter is an indication that the billed port sent more data than it should have and the neighbor's port over ran its buffer. This event is extremely rare, and one likely cause is that the billed port received an inaccurate credit return from the neighbor, so it is treated as an Integrity issue on the receive side of the billed port.




## L.2.1 Integrity Counters

### L.2.1.1 Link Integrity

These counters reflect errors in the Physical (PHY) and Link Layers, as well as errors in firmware. In some cases, these errors are benign and can be ignored. However in other cases, excessive link integrity errors can indicate a hardware problem such as a poor connection, marginal cable, incorrect length/model cable for signal rate, or damaged/broken hardware, such as bad connectors.

When a bad packet is detected, one of these counters is incremented and the Link Layer may either discard or replay the packet.

During the link training sequence, assorted errors may be observed. This is a normal part of the link training and clock synchronization process. Hence, errors observed as part of rebooting nodes or moving cables should not be considered a problem.

The category is calculated as a weighted sum of the counters in the group. With the exception of ExcessiveBufferOverrunErrors, the counters in this group report on the receive side of the link. However, the counter can indicate a problem on either side of the link.

### L.2.1.2 Link Quality Indicator (LQI)

This is a status indicator, similar to the signal strength bar display on a mobile phone, that enumerates link quality as a range of 0-5, with 5 being very good. Values in the lower part of the range may indicate hardware problems with components such as ports and cables that surface as signal integrity issues, leading to performance and other problems. The LQI gives you an instantaneous view of a link's quality on every hardware port.

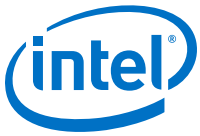
**Table 44. Link Quality Values and Description**

| Link Quality Value | Description |
|---|---|
| 5 | Working at or above preferred link quality, no action needed. |
| 3 | Working on low end of acceptable link quality, recommended corrective action on next maintenance window. |
| 2 | Working below acceptable link quality, recommend timely corrective action. |
| 1 | Working far below acceptable link quality, recommend immediate corrective action. |
| 0 | Link down |

### L.2.1.3 LocalLinkIntegrityErrors (LLI) Counter

This counter indicates the number of retries initiated by a link transfer layer receiver.

The retry rate is represented by the Link Quality Indicator. A link that is meeting performance requirements has a Link Quality of 5, which corresponds to 1000 or fewer replays per second.



#### L.2.1.4 PortRcvErrors (RxE) Counter

This counter indicates the total number of packets containing an error that were received by the port, including Link Layer protocol violations and malformed packets. It indicates possible misconfiguration of a port, either by the SM or by user intervention. It can also indicate hardware issues or extremely poor link signal integrity.

#### L.2.1.5 ExcessiveBufferOverrunErrors (EBO) Counter

This counter, associated with credit management, indicates an input buffer overrun. It indicates possible misconfiguration of a port, either by the SM or by user intervention. It can also indicate hardware issues or extremely poor link signal integrity.

#### L.2.1.6 LinkErrorRecovery (LER) Counter

This counter indicates the number of times the link has successfully completed the link error recovery process.

Link Quality Indicator is the primary indicator for link quality to use. This counter is factored into the value reported for Link Quality Indicator. This counter may be non-zero for a properly functioning link.

#### L.2.1.7 LinkDowned (LD) Counter

This counter indicates the total number of times the port has failed the link error recovery process and downed the link. These events can cause disruptions to fabric traffic.

#### L.2.1.8 UncorrectableErrors (Unc) Counter

This counter indicates the number of unrecoverable device errors. This may indicate a defect in the reporting device.

#### L.2.1.9 FMConfigErrors Counter (FMC)

This counter reports inconsistent configurations of the low-level Subnet Management Agent (SMA) on either side of the link. It indicates possible misconfiguration of a port, either by the SM or by user intervention.

### L.2.2 Integrity Weights Rationale

The default weights for most of the above counters is 100 except for `LinkDowned`, `LinkQualityIndicator`, `LocalLinkIntegrityErrors`, and `LinkErrorRecovery`, which are 25, 40, 0, and 0 respectively. Most of the counters have a weight of 100 because even one error can be significant indication of an Integrity issue at the port. The Threshold for Integrity events defaults to 100, so a weighted sum of 100 or more will trigger indications in the user interfaces that an Integrity issue is occurring.

The `LinkDowned` counter has a weight of 25 because while the link going down seems like a significant Integrity issue, the majority of the times this counter will increment is during routine maintenance such as rebooting servers.

The LinkQualityIndicator value passes to a unique equation (See LinkQualityIndicator Equation and Rationale on page 280) as this value is not a counter and instead an indication of signal quality similar to the bar display on a mobile phone. The indicator has a weight of 40 because this allows a value of 4 ("Very Good") or 5 ("Excellent") to not exceed the threshold.

Additionally, LocalLinkIntegrityErrors and LinkErrorRecovery have default weights of zero, meaning they are not included by default in the calculation of the category. These counters are already included when calculating the LinkQualityIndicator and it would be redundant to include them twice.

### L.2.3 LinkQualityIndicator Equation and Rationale

LinkQualityIndicator value is used in an exponential equation to allow lower values of the field to have more impact of the calculation of the billed port's Integrity value.

The Equation is defined as:

$= 2^{(5\text{-}LQI)} - 1$

### L.2.4 LinkWidthDowngrade Equation and Rationale

LinkWidthDowngrade value is passed into the below equation to indicate the number of lanes down for the billed port.

*= LinkWidth.Active – LinkWidthDowngrade.RxActive*

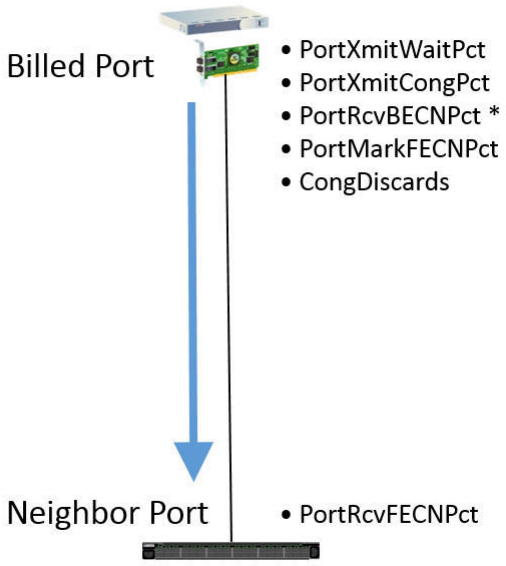
By default, all links normally are active at 4x (or 4 lane widths). Each lane is bidirectional, meaning that each lane has Tx (Transmit) and Rx (Receive) directions. In the event a LinkWidthDowngrade occurs, this will indicate one of the four lanes went down for one of the two directions.

Only one direction was chosen instead of both because a port's transmit lane count must be equal to the neighbor port's receive lane count. This means that the Rx lane is the same as a Tx lane on the neighbor port.

The Rx lanes were chosen in this calculation as the Integrity issue is billed to the receiving port.

## L.3 Congestion Category Calculations

The congestion category calculation applies weights to the each of the following counters before summing them for the final value.




* `PortRcvBECNPct` is included only when the billed port is an HFI.

*Note:* Counters that end in "Pct" are intermediate values computed by the PM to make the congestion counter more useful to the user (see Pct10 Counter Calculations on page 289 for equations).

All counters except `CongDiscards` (formerly `SwPortCongestion`) are the result of intermediate calculation to give more context for more accurate indications of which ports are actually in a congested state.

For example, `PortXmitWaitPct` is calculated using `PortXmitWait` physical counters at its base. `PortXmitWait` is the number of times a flit (unit of data) could be sent, but was unable to send because there was not enough credits. The counter value of 10 times means nothing without knowing the total number of attempted flits sent. If the total is also 20, then this is very bad. However, if the total attempted flits sent is 1,000,000 (1 million), then this value is less than 0.001% and therefore the port is not really in a congested state. See Congestion on page 282 for more details.

`PortRcvBECNPct` is included only when the billed port is an HFI because of the nature of the BECN flag that is a part of CCA (see Section 2.8 Congestion Control Architecture). The BECN flag is set on a packet returning to the sender to inform the sender that the sending path is congested because the receiving endpoint has received FECNs. The FECN flag is marked on a packet when that packet passes through a congestion link. However, due to the nature of routing, the returning packet with the BECN does not have to take the same route through the fabric and therefore does not indicate anything about congestion for any switch ports it passes through.

`PortRcvFECNPct` is included using the neighbor's value because the base counter `PortRcvFECN` increments whenever the port receives a packet. This means that the congested link is usually earlier in the path of the packet.

### L.3.1 Congestion

These counters reflect possible errors that indicate traffic congestion in the fabric.

When congestion or a packet that has seen congestion is detected, one of these counters is incremented and then depending on the issue reported, the packet must wait. In an extreme case, the packet may time out and be dropped.

The category is calculated as a weighted sum of the counters in the context of the utilization counters. With the exception of PortRcvFECN, the counters are all reported on the transmit side of the link. In addition, PortRcvBECN is only taken if the local node is an HFI. However, the counter could indicate a problem on either side of the link.

#### L.3.1.1 CongDiscards (CD) Counter

*Note:* Formerly known as "SwPortCongestion".

This switch-only counter indicates the number of packets that were discarded as unable to transmit due to timeouts.

#### L.3.1.2 PortRcvFECN (RxF) Counter

When a device receives a packet with the Forward Explicit Congestion Notification (FECN) bit set to one, this counter is incremented.

#### L.3.1.3 PortRcvBECN (RxB) Counter

When a device receives a packet with the Backward Explicit Congestion Notification (BECN) bit set to one, this counter is incremented.

#### L.3.1.4 PortMarkFECN (MkF) Counter

This counter indicates the total number of packets that were marked Forward Explicit Congestion Notification (FECN) by the transmitter due to congestion.

#### L.3.1.5 PortXmitTimeCong (TxTC) Counter

This counter indicates the total number of *flit times* that the port was in a congested state for any data VL.

#### L.3.1.6 PortXmitWait (TxW) Counter

This counter indicates the amount of time (in *flit times*) any virtual lane had data but was unable to transmit due to no credits available.

### L.3.2 Congestion Category Intermediate Calculations

The majority of the congestion related counters are relevant only when the associated data transfer counter is used to normalize its value. Below are the intermediate values that came from normalizing the congestion counters:

- `PortXmitWaitPct`: the port had to wait to send a flit relative to time it was sending or waiting. Range is 0.01% to 1% yielding input to the threshold equation between 1 and 100.

$$= \frac{PortXmitWait * 10000}{PortXmitWait + PortXmitData}$$

- `PortXmitCongPct`: Percentage of the time the port was in a congested state. Range is 0.1 % to 10% yielding input to the threshold equation between 1 and 100. (Possible Risk of Double counting in the denominator)

$$= \frac{PortXmitTimeCongestion * 1000}{PortXmitTimeCongestion + PortXmitData}$$

- `PortRcvBECNPct`: Percentage of packets an HFI port received with a BECN bit set. Range is 0.1% to 10% yielding input to the threshold equation between 1 and 100.

$$= \frac{PortRcvBECN * 1000}{PortRcvPkts}$$

- `PortMarkFECNPct`: Percentage of packets a port has marked with a FECN bit. Range is 0.1% to 10% yielding input to the threshold equation between 1 and 100.

$$= \frac{PortMarkFECN * 1000}{PortXmitPkts}$$

- `PortRcvFECNPct`: Percentage of packets a port has received with a FECN bit set. Range is 0.1% to 10% yielding input to the threshold equation between 1 and 100.

$$= \frac{Neighbor\ PortRcvFECN * 1000}{Neighbor's\ PortRcvPkts}$$

*Note:* The "*1000" operation allows a tenth of a percentage point to be express in an integer. 24.6% would equal 246. Also, if counter(s) in the divisor is '0' then calculation is returned as '0.'

### L.3.2.1 Mathematical Rationale

The following elements explain the mathematical rationale behind the Pct values used in calculations:

- 100gbs —> 1.6B flits/sec = 1.6M fits/msec
- Hoq timeout = 8ms

  12M flits/8msec

- Therefore, 1 `CongDiscards`= 12M flits of `XmitWait`/ sec

  12M flits/s / 1.6B flits/s ≈ 0.6%

In conclusion, 0.6% is a reportable value of congestion, so the range for certain Pct counters is 0.01% to 1%, for example, `XmitWaitPct` and `BubblePct`.

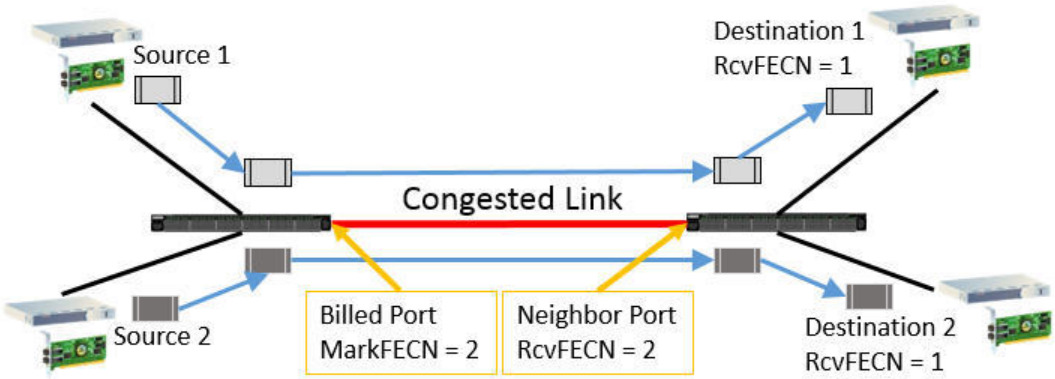
### L.3.3 Congestion Weight Rationale

Each counter's weight is what is believed to be values that would accurately indicate a congested port. Using a combination of the value used to normalize the "Pct" intermediate counters and their importance, the following values were selected:

- `CongDiscards` is weighted at 100 to indicate that even one discard due to congestion is enough to exceed a threshold.
- `PortXmitWaitPct` is weighted at 10 to indicate that a port with a congested percentage exceeding 0.1% is enough to exceed the threshold. This means that one in a thousand flits had to wait because not enough credits were returned and the port is therefore congested.
- `PortXmitTimeCongPct` is weighted at 25 to indicate that a port at a congested percentage exceeding 0.4% is enough to exceed the threshold. This means that one in 250 flits had to wait because the port was in a temporary congested state and therefore congested.

The remaining three counters are in use only when Congestion Control Architecture (CCA) is enabled. `PortMarkFECNPct` is weighted at 25 to indicate that a port marked over 0.4% should exceed the congestion category threshold. This means that the link was congested for at least one in 250 packets.

- `PortRcvFECNPct` is weighted only 5 because every port that receives this packet increments this counter. The weight is so low because the congested link might have been earlier in the route through the fabric, but the FECN bit continues to be counted on this packet until it reaches its destination. This means that it is possible for ports/links that are not congested to increment this counter. However, the fact that most ports have several different routes that pass through it, allows the congested link to have the highest value.

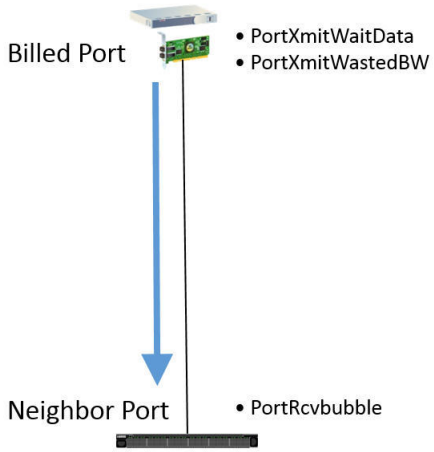


Additionally, PortMarkFECNPct is a stronger indicator of a congested link; other links on the route may be possible contributors to the congestion. However, if the cause of congestion is a destination HFI, the HFI will receive many FECNs.

- PortRcvBECNPct is weighted only 1 because this counter indicates an HFI that is a possible cause or contributor to congestion elsewhere in the fabric. While this means the congested link may not be attached to this port, it may still play a small role in the overall congestion in the fabric.

## L.4 Bubble Category Calculations

The bubble category calculation takes the greater value between the transmitted bubbles and receive bubbles.




*Note:* Counters that end in "Pct" are intermediate values computed by the PM to make the bubble counter more useful to the user (See Pct10 Counter Calculations on page 289 for equations and rationale).

A Bubble occurs when an unexpected idle flit is transmitted or received.

Similar to the counters within the congestion category, the counters in the bubble category also have more meaning when in the context of the total number of flits sent.

### L.4.1 Bubble

These counters occur when an unexpected idle flit is transmitted or received.

The transmit port sends idle flits until it can continue sending the rest of the packet. The category is calculated as follows:

1. The maximum value between the sum of the XmitWastedBW and XmitWaitData or the neighbor's PortRcvBubble.
2. Then divide the previous value by the port's utilization to provide context.

### L.4.1.1 PortXmitWastedBW (WBW) Counter

This counter indicates the number of *flit times* where one or more packets have been started but the transmitters are forced to send idles due to bubbles in the ingress stream. Also, the VLs that have data to be sent are not permitted to preempt the currently transmitting VL.

### L.4.1.2 PortXmitWaitData (TxWD) Counter

This counter indicates the number of *flit times* where one or more packets have been started but interrupted due to bubbles in the ingress stream.

### L.4.1.3 PortRcvBubble (RxBb) Counter

This counter indicates the total number of *flit times* where one or more packets have started to be received, but the receiver received idle flits from the wire.

## L.4.2 Bubble Category Intermediate Calculations

The bubble related counters are only relevant when the associated data transfer counter is used to normalize it's value. Below are the intermediate values that came from normalizing the bubble counters:

- `PortXmitBubblePct`: Percentage of time an idle flit was sent in the middle of a packet.

$$= \frac{(PortXmitWastedBW + PortXmitWaitData) * 10000}{(PortXmitWastedBW + PortXmitWaitData) + PortXmitData}$$

- `PortRcvBubblePct`: Percentage of time an idle flit was received in the middle of a packet.

$$= \frac{PortRcvBubble * 10000}{PortRcvBubble + PortRcvData}$$

- Bubble Category: (`BubblePct`)

  `= MAX(PortXmitBubblePct,Neighbor's PortRcvBubblePct)`

Range is 0.01% to 2.5% yielding input to the threshold between 1 and 255.

# L.5 Routing Category Calculations

Currently there is only one counter associated with the routing category, `PortRcvSwitchRelayErrors`. We use the delta value from the previous sweep to calculate the billed port's routing category.

## L.5.1 Routing

These counters reflect possible routing issues. When a routing issue occurs, the offending packet is dropped.

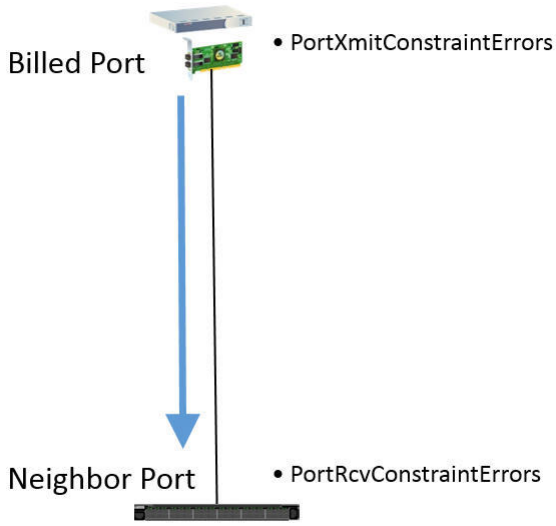
A typical cause of this error is the routing to a wrong egress port or an improper Service Channel (SC) mapping. These errors can be a side effect of a port or device going down while traffic was still in flight to or through the given port or device.

### L.5.1.1 PortRcvSwitchRelayErrors (RxSR)

This counter indicates the number of packets that were dropped due to internal routing errors. It indicates possible misconfiguration of a switch by the SM.

## L.6 Security Category Calculations

The Security category calculation is just the sum of the following counters:




Both counters represent the number of security violation at either egress (Xmit) or ingress (Rcv) of a port. Possible security violation include Partition Key or Source LID violations.

For the billed port, we use the sum of the local port's `PortXmitConstraintErrors` and its neighbor's `PortRcvConstraintErrors`.

### L.6.1 Security

These counters reflect possible security problems in the fabric.

Security problems can occur if a PKey or SLID violation occurs at the port during the ingress or egress of a packet.

The category is calculated as the sum of the neighbor's PortRcvConstraintErrors and the local port's PortXmitConstraintErrors.

### L.6.1.1 PortRcvConstraintErrors (RxCE)

This counter is incremented when partition key or source LID violations are detected in a received packet, indicating a possible security issue or misconfiguration of device security settings.

### L.6.1.2 PortXmitConstraintErrors (TxCE)

This counter is incremented when partition key violations are detected in a packet attempting to be transmitted, indicating a possible security issue or misconfiguration of device security settings.

## L.7 SMA Congestion Calculations

SMA Congestions Calculations are a subset of the normal Congestion category and use the same weights and intermediate calculations, except we replace the use of the port level counters with their equivalent VL 15 (Management VL) counters.

### L.7.1 PortVLXmitWait[15] (VLTxW[15]) Counter

This counter behaves the same as PortXmitWait, but it is restricted to VL 15, which carries only SM traffic.

### L.7.2 VLCongDiscards[15] (VLCD[15]) Counter

*Note:* Formerly known as "SwPortVLCongestion".

This counter behaves the same as CongDiscards, but it is restricted to VL 15, which carries only SM traffic.

### L.7.3 PortVLRcvFECN[15] (VLRxF[15]) Counter

This counter behaves the same as PortRcvFECN, but it is restricted to VL 15, which carries only SM traffic.

### L.7.4 PortVLRcvBECN[15] (VLRxB[15]) Counter

This counter behaves the same as PortRcvBECN, but it is restricted to VL 15, which carries only SM traffic.

### L.7.5 PortVLXmitTimeCong[15] (VLTxTC[15]) Counter

This counter behaves the same as PortXmitTimeCong, but it is restricted to VL 15, which carries only SM traffic.

### L.7.6 PortVLMarkFECN[15] (VLMkF[15]) Counter

This counter behaves the same as PortMarkFECN, but it is restricted to VL 15, which carries only SM traffic.

## L.8 Pct10 Counter Calculations

In addition to the Category calculations, the PM has two additional computed values. These values are provided to give you additional information that does not fall into the above categories.

The name Pct10 comes from the fact that these values are percentage values that are in units of 0.1%.

These values are calculated at a per-port level. Then when a user makes a PA request, such as GroupInfo, the maximum value for the internal and external subgroups are included. This allows you to see the largest values from within the group.

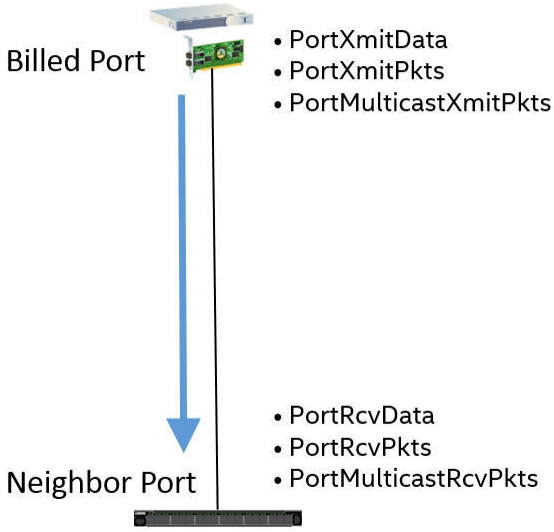
### L.8.1 UtilizationPct10 Calculation

UtilizationPct10 value is the percentage of utilization compared to the max theoretical value a port should be able to support. The Max theoretical value is computed using the Lane Width, the Lane Speed, and the sweep interval. The calculation for Utilization is given as the number of data flits passed between the two ports divided by the max possible value.

$$= \frac{MAX(PortXmitData, neighbor's\ PortRcvData) * 1000}{Rate\ (in\ MBps) * \frac{10^6\ flits}{8\ MB} * interval\ (in\ \text{sec})}$$

We multiply the value by 1000 to convert the value to units of 0.1%.

Both `PortXmitData` and the neighbor's `PortRcvData` are used for this calculation so that if PMA counters are not fetched from one side of the link, the value can still be calculated. Typically `PortXmitData` will match the neighbor `PortRcvData`. See Utilization on page 290 for descriptions of the Utilization counters provided by the PMA.




*Note:* All Counters will be the delta value from the previous sweep.

### L.8.2 PortXmitDiscardsPct10 Calculation

PortXmitDiscardsPct10 is the percentage value of total packets dropped over the total packet that were attempted to be sent:

$$= \frac{PortXmitDiscards * 1000}{PortXmitDiscards + PortXmitPkts}$$

`PortXmitDiscards` is the total number of packets discarded before they could be sent while `PortXmitPkts` is the total number of packets that were successfully sent. Therefore, their sum should be the total number of packets attempted to be sent.

We choose to use only `PortXmitPkts` during the calculation of PortXmitDiscardsPct10, as `PortXmitDiscards` is only a transmit counter and is not bidirectional.

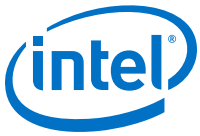
### L.8.3 PortXmitDiscards (TxDc)

This counter indicates the number of packets dropped due to several reasons including timeouts and improper packet lengths.

*Note:* This counter is a super set that includes Congestion Discards counter.

## L.9 Utilization

These counters reflect the normal utilization of the port and Virtual Lane when present.



Several of these counters are used during the calculation of Congestion, SMA Congestion, and the Bubble Categories. The Utilization metrics provide a way of giving some of the other counters context by comparing them to the amount of data or packets that were transmitted or received.

### L.9.1 PortXmitData (TxD) and PortVLXmitData[n]

These counters indicate the total number of fabric packet flits transmitted. This does not include idle nor other LF command flits.

### L.9.2 PortRcvData (RxD) and PortVLRcvData[n]

These counters indicate the total number of fabric packet flits received.

### L.9.3 PortMulticastXmitPkts (MTxP)

This counter indicates the number of multicast and collective packets transmitted.

### L.9.4 PortMulticastRcvPkts (MRxP)

This counter indicates the number of multicast and collective packets received.

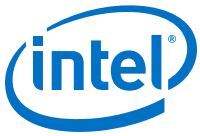
## L.10 Other

These counters do not fit into any of the previous categories.

### L.10.1 PortRcvRemotePhysicalErrors (RxRP)

This counter indicates the number of downstream effects of signal integrity (SI) problems. It indicates an SI issue in the upstream path.

This counter was not included as it does not directly indicate the link that had the issue, so it can be misleading.





# Appendix M Best Practices for ESM Logging

Before running an ESM, there are several changes Intel recommends that you make to the default switch log levels. This will ensure that important ESM specific logging is enabled and written to the log. To modify the log levels, use the `logConfigure` CLI command, which is documented in the *Intel® Omni-Path Fabric Switches Command Line Interface Reference Guide*, "logConfigure" section.

Intel recommends the default log levels be set to enable the following levels: Dump, Fatal, Error, Alarm, Warning, Partial, Config, Info, Periodic, and Notice. This should be done for the Ram and Syslog.

The CLI command `logShow` can be used to dump the contents of the ESM logs.

Intel recommends that the user redirect the ESM logging to a syslog server, such as a Linux server running the `rsyslogd` service. This can be done using the `logSyslogConfig CLI` command or configured through the Chassis Viewer (see *Intel® Omni-Path Fabric Switches Command Line Interface Reference Guide*).