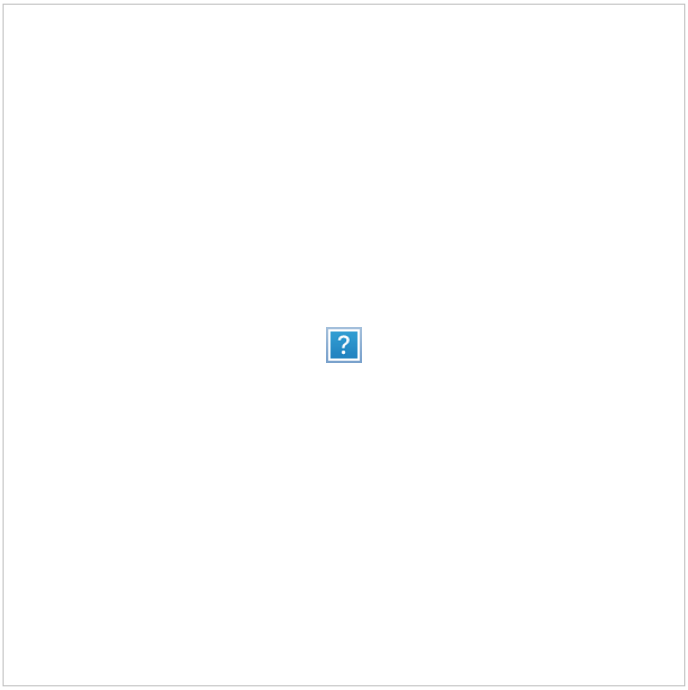
GigaIO™ FabreX™ Software (FXS) v2.7.0 User Manual



www.gigaio.com

are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. GigaIO shall not be liable for technical or editorial errors or omissions contained herein.

For the latest information published by GigaIO, see the company website (http://www.gigaio.com).

Contents

Revision History

Rev.	Date	Description
1.11	10 Mar 2023	• Global: Changed FabreX Restricted Shell (rShell) to FabreX Shell (fabshell). • Global: CentOS to CentOS/Rocky where applicable. • Added terms to the "Glossary" section. • Updated "FabreX Manager Switch Utilities" to capture fmtool upgrades. • Updated "fabshell Options" section for v2.7.0. • Updated "Gathering FabreX Switch Logs" section. • Added a step to reboot the switch in the "FabreX Switch Web User Interface" 'Update Page' subsection. • Updated steps in the "Update the FabreX Switch using the Web UI" section. • Added chapter/appendix now "Deprecated - FabreX Manager v2.6 and Earlier Tool Features."
1.10 and earlier	—	For other document versions, visit **Archived Releases** on the Service

revisions		Desk Documentation page.

About this User's Manual

This user manual is intended for users who are responsible for configuring and maintaining the GigaIO™ FabreX™ Software (FXS).

Related Documentation

Documentation available for FabreX Software (FXS), associated software tools, and hardware resources may be found on the GigaIO Service Desk site (login and password are required to access). Its Documentation page includes the following items.

Related Documentation

Document Name	Description
GigaIO™ FabreX™ System (PCIe Gen 4) Deployment Guide	FabreX System component installation and configuration between Hosts, FabreX Switch and available resources
GigaIO™ Redfish API, Release 2.7.0, User's Manual	FabreX-customized RESTful interface commands that conform to the DMTF Redfish® standard
GigaIO™ FabreX™ Fabric Manager Developer Guide	FabreX software how-to guide on performing device discovery using composition examples from the GigaIO Redfish API
Software Release Notes - FabreX™	The latest FabreX software changes, fixes, updates, and workarounds. FabreX is a PCIe-based fabric solution providing composability of IO resources and host-to-host communication
GigaIO™ 24-Port FabreX™ Switch User's Manual: Models RS4024/RS3024	24 Port, 1U, PCIe Switch installation and management interface instructions
GigaIO™ 4-	Adapter card installation and configuration

Port FabreX™ Network Adapter Card User's Manual: Models FA4004/FA3004	instructions
GigaIO™ Storage Pooling Appliance User's Manual: Model RB3032	PCIe Gen 3 Storage Pooling Appliance (SPA) installation and management interface instructions
GigaIO™ Accelerator Pooling Appliance Hardware User's Manual	PCIe Gen 4 Accelerator Pooling Appliance (APA) hardware overview, configuration, installation, and troubleshooting instructions.
GigaIO™ Accelerator Pooling Appliance Management System User's Manual	PCIe Gen 4 Accelerator Pooling Appliance (APA) management system GUI overview, setup, and usage.
GigaIO™ Accelerator Pooling Appliance Redfish API, Version 1.6.0, User's Manual	PCIe Gen 4 Accelerator Pooling Appliance (APA) Redfish details the RESTful interface used to manage the APA from third-party applications.

All the documents listed are available on the GigaIO Service Desk Documentation page (login and password are required to access).

For login registration, visit the GigaIO Service Desk. For other support assistance, email support@gigaio.com or call 1 (760) 487-8395.

Glossary

Glossary of Terms

Term	Definition
ACS	Access Control Services are used to control which devices are allowed to communicate with one another, in order to avoid improper routing of packets.
APA	GigaIO Accelerator Pooling Appliance (formerly Add-in Card Resource Box)
BDF	Bus:Device.Function is a notation used to describe PCI and

	PCIe devices in a system.
CMI	CMI communications may be managed by a processor on the FabreX Network Adapter Card or by a processor at the opposing cable end.
compose/decompose	Composing binds disaggregated compute and storage physical resources and collectively manages these resources as a high-perfomance computing system (software fabric), without physically moving any hardware components. Decomposing unbinds aggregated compute and storage resources from the software fabric.
composition-only mode	Composition mode is the ability to move PCIe resources (GPUs, accelerator cards, NVMe™ drives) from one host to another, meaning no host-to-host PCIe communication (like MPI, NVMe-oF, GDR, or IP over NTB).
Crosslink	A link between two upstream ports or two downstream ports, where the ports go through a process to determine which will act as the USP and which will act as the DSP. The crosslink is a physical layer option that works around the need for a defined upstream/downstream direction in the link training process. Another crosslink used is non-transparent endpoint (NT EP) to NT EP, where the crosslink is two USPs.
DHM	Dynamic Host Mode (or dynahost) is a conditional flag that determines whether host-tags are included or excluded when fetching topology templates via Redfish. The flag is available in the FabreX Shell interface as a command option.
DSP	Downstream port is the port facing toward PCIe leaf segments (upstream port or endpoint).
Endpoint (or EP)	An Endpoint is a device, such as an NVMe SSD or GPU/FPGA Accelerator, with an upstream connection for the PCIe root complex. Endpoints can be considered as a leaf at the end of a branch, in a PCIe tree topology, with no

	downstream or bridge connections to other devices. PCI-SIG defines an "Endpoint" as "A Function that has a Type 00h Configuration Space header."
fabshell	The FabreX shell interface enables users to interact with the FabreX Switch on a limited basis. The fabshell permits users to configure a switch for the local network and perform minimal maintenance duties not supported elsewhere. Users can also use fabshell to power off the switch, reboot the switch, set the system time, and set the hostname, along with other actions.
fmtool	The FabreX Manager Tool is used to configure the topology. It is also use to check the fabric, as well as manager and worker switch statuses.
FXS	FabreX Software is a software engine that drives the performance and dynamic composability of GigaIO software-defined infrastructure (SDI) for enterprise data centers and high-performance computing environments.
GDR	GPUDirect® RDMA enables a GPU application to be run over multiple servers with GPUs attached to them.
GUID	Global Unique IDentity is essentially the serial number of the device(s) connected on the FabreX network. GUIDs are used to uniquely identify each device and port connection, in-band.
Host Bridge (or HB)	The host bridge (or root complex) is the interface between the top-level PCI bus and the CPU (the host).
I/O Switch	A PCIe switch with at least one upstream and one or more downstream ports, connecting one or more hosts to other PCIe switches or endpoints.
I/O zone	An I/O zone supports whole-endpoint-device composition in a standard PCIe tree structure. Within an I/O Zone, any endpoint device may be composed to any single host. There is no device sharing with FabreX Switch Models

	RS3024/RS4024. Device sharing is possible with RS4124 switch.
IPN	Internal port number is a 'logical' representation of one, two, or four physical switch ports used by FabreX Manager Tools and Redfish for configuring downstream ports.
KLPP	A kernel module that supports the userspace LPP library. *See LPP*.
Lane	A "lane" is a PCI Express® term defined as "A set of differential signal pairs, a differential pair for transmission and a differential pair for reception."
Link	A "link" is a PCI Express term defined as "A collection of two Ports and their interconnecting Lanes. "A by-n Link is composed of n Lanes.
LPP	Libfabric PCIe Provider. A libfabric provider that runs on FabreX.
LTR	A loadable topology record is a file that has the names for server hosts and designates the switch and port to which each host connects.
MPICH	Message Passing Interface CHameleon is a high performance and widely portable implementation of the Message Passing Interface (MPI) standard. GigaIO currently uses the MPICH 3.3.x release.
NCCL	The NVIDIA Collective Communications Library (NCCL) implements multi-GPU and multi-node collective communication primitives. NCCL provides routines that are that are performance optimized for NVIDIA GPUs to achieve high bandwidth over PCIe.
NT zone	An NT Zone supports direct host-to-host, non-cached memory access, as well as some forms of endpoint device lending via the FabreX non-transparent fabric. All hosts within an NT Zone provision per-host private memory

	spaces and may inter-operate using that memory. Certain over FabreX-enabled features, such as MPI/Libfabric, NVMe-oF, and GDR, can only be used between hosts within a common NT Zone.
NVMe-oF™	Non-Volatile Memory over Fabrics uses a message-based model to communicate between the host computer and system over a network.
Port	Port is typically used to reference a PCIe connection with four lanes. When ports are connected between upstream and down stream devices on the network, links can be established for PCIe communication. Four ports connected in parallel between two devices make a x16 lane PCIe connection.
Root Complex (or RC)	A root complex device connects the processor and memory subsystem to the PCI Express switch fabric composed of one or more switch devices.
SPA	GigaIO Storage Pooling Appliance (formerly U.2 Resource Box)
Switch Port	Physical connectors numbered 1-24 on the switch. Each switch port represents a x4 PCIe Gen 3 or Gen 4 link. The switch ports can be grouped into x4, x8, or x16 PCIe for either Gen 3 or Gen 4 links.
USP	Upstream port is the port facing toward the PCIe Root.
Zone	A grouping of hosts, switch ports, and endpoint devices supporting certain inter-operational characteristics, depending on the type of zone. See also, IO zone or NT zone.

Introduction

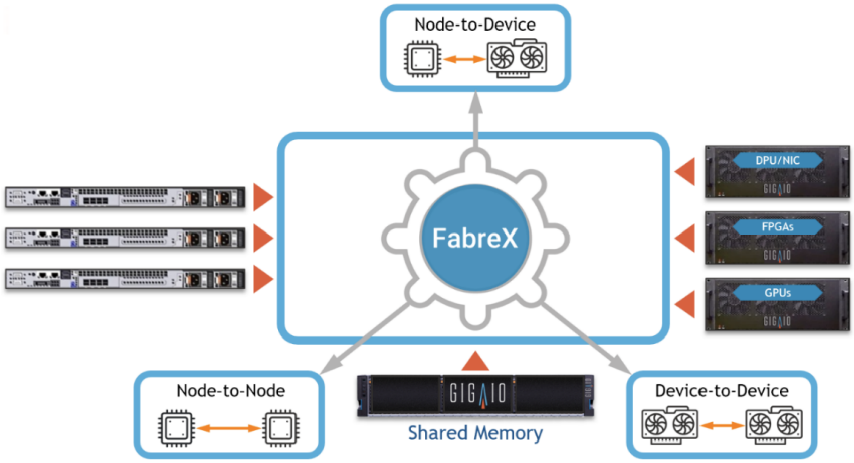
FabreX™ Software (FXS) is a fundamental building block for composable memory

driving performance and dynamic composability of GigaIO's Software-Defined Hardware™ (SDH) for enterprise data centers and high-performance computing environments. The FabreX Fabric Manager (FM) is the central building block and software engine. This Linux-based, resource-efficient software layers onto FabreX hardware for easy-to-use composing of computing clusters on-the-fly. Designed to be an open environment with robust APIs, FabreX FM eases integration and encourages third-party developer to enhance the usability, flexibility, and performance of the fabric. You can continue to use the tools you know and trust — with an entirely new layer of capability.

Workloads no longer need to be limited by what fits in in a server chassis. GigaIO's FabreX software provides the flexibility to compose bare metal devices such as GPUs, FPGAs, NVMe, and even DRAM to a server. This allows the servers' resources to scale up or down dynamically to meet workloads needs. FabreX can even connect multiple servers together so they share resources previously confined to a server's sheet metal, or share the workload with a remote server. All this is accomplished using the same Direct Memory Access (DMA) method that a server uses to access resources residing on its motherboard.

Software tools have been fully integrated with FabreX software. GigaIO's FabreX software includes host-based drivers to enable these advanced features, ranging from device discovery assistance to enabling TCP traffic across FabreX. Since FabreX can connect servers using DMA, the possibilities for host optimizations are endless. Some of the current host features include: compatibility assistant, Message Passing Interface (MPI), LibFabric PCI Provider, NVMeOF Initiator / Target, GPU Direct DMA, NCCL Driver, and TCP/IP Driver.

GigaIO also provides an integration guide and an easy to use CLI to allow any customer to build playbooks, blueprints, or just straight API calls for personalized integration and seamless integration with Ansible, Chef, Puppet and other commonly used DevOps tool sets.



GigaIO's memory fabric serves as the foundation for PCIe switching infrastructure, enabling native protocol communications from server-to-server, server-to-device, or even device-to-device communications. FabreX software creates and manages these communication paths. Its DMTF open-source Redfish APIs provide unprecedented integration with a range of third-party applications for fabric automation and ochestration.Today's IT environments have many needs, but what they don't want is another management pane of glass. GigaIO's software integrates with our customers' existing tools, and thanks to our open ecosystem platform, an ever-growing list of pre-integrated partners.

FabreX Server Software

Hardware and Software Requirements

FabreX System Requirements

Requirement	Description
FXS Version 2.7.0	Latest FabreX Host server software release
Platforms	Xeon (x86/x64) based servers with PCIe Gen 4 or Gen 3 x16 support
Required disk space for installation	50 GB free space available

Operating System	Centos7, Rocky8, or Ubuntu20.04 Refer to the GigaIO Service Desk for the latest customer release notes listing changes and updates to supported operating system distributions and kernels.
Installer Privileges	Administrator privileges are required on the target machine.

FabreX Server Package Functions and Features

The following table lists each FabreX Server package available for installation, along with whether it is required or optional and a brief functional description.

> ⓘ Filenames applicable to CentOS, Rocky or Ubuntu installations are listed in the following table using parentheses (CentOS/Rocky or Ubuntu).

FXS Server Packages

Packages	Description
• kernel (CentOS/Rocky) • kernel-devel (CentOS/Rocky) • kernel-headers (CentOS/Rocky) • linux-image (Ubuntu) • linux-headers (Ubuntu	**Required**: The *Kernel* package includes the Linux operating system's core (CentOS/Rocky or Ubuntu) for the host server.
• gigaio-fabrexfm-host-worker • gigaio-support	**Required**: The *FabreX* package includes: GigaIO FabreX Fabric Manager hostside daemon and GigaIO scripts and configuration file required by FabreX users.
	Optional: The high-performance computing (*HPC*)

• gigaio-libfabric • gigaio-libfabric-devel (CentOS/Rocky) • gigaio-libfabric-dev (Ubuntu) • gigaio-mpich • gigaio-mpich-devel (CentOS/Rocky) • gigaio-mpich-dev (Ubuntu)	package includes: Libfabric PCIe Provider (LPP) that runs on FabreX PCIe networks. FabreX offers high-performance RDMA functions to form the foundation of the LPP provider. Higher level primitives are implemented at the libfabric and kernel LPP module (KLPP) layers. GigaIO Message Passing Interface CHameleon (*MPICH*), a high performance and widely portable implementation of the Message Passing Interface (MPI) standard. The MPICH 3.3 release is currently used.
• gigaio-nvme-cli • nvmetcli	**Optional**: The Non-Volatile Memory over Fabrics (*NVMe-oF*) package includes: NVMe management command line interface with a command set and a tool available to configure the NVMe.
• gigaio-fabrex-nccl • gigaio-nv-klpp-peer-dkms	⚠ Only perform the GPUDirect RDMA (*GDR*) package install only after the GPUs and relevant drivers are set up on the host system. Otherwise, this install may fail and report errors. **Optional**: The GPUDirect RDMA (*GDR*) package that enables a GPU application to be run over multiple servers with GPUs attached to them. This package includes: GigaIO FabreX NVIDIA Collective Communications Library (NCCL) to support multi-GPU collective communication primitives that are topology-aware and can easily be integrated into applications to enhance GPU performance.

	ⓘ Only required when NVIDIA GPUs are installed for NVIDIA A10, 24GB; A30, 24GB; A40, 48GB; and A100, 80GB; as well as, XIlinx U55C, FPGA KLPP, a peer module required to support GDR functionality.

Installation Instructions

Downloading the FabreX Server Package

You can download the FabreX Server Package from the Software Releases site on the GigaIO Service Desk (login required).

ⓘ The next section applies to CentOS/Rocky only. If you are using Ubuntu, go to the section "**Running the FabreX Installation Script to Install the FabreX Server Package**."

Installing Development Tools for CentOS only

For CentOS only: Install "Development Tools" for CentOS 7.0 or later using the following command:

```
sudo yum groupinstall "Development Tools"
```

Running the FabreX Installation Script to Install the FabreX Server Package

The FabreX installation script (fabrex_install.sh) is a convenient and easy-to-use CLI to install modules included in the FabreX Server Package.

To run the FabreX installation script, perform the following steps, as applicable to your

CentOS, Rocky, or Ubuntu-based system:

1. Copy the downloaded FabreX software release tarball to the host.
2. Go to the directory where the software package was downloaded. To untar the files, perform step **a** for CentOS and step **b** for Ubuntu:

   > The tarball takes approximately 30 seconds to a minute to unpack.

   a. For CentOS, enter the following command:

   ```
   tar -xjvf FabreX_<VERSION>_CentOS7.tar.bz2
   ```

   b. For Ubuntu, enter the following command:

   ```
   tar -xjvf FabreX_<VERSION>_Ubuntu20.tar.bz2
   ```

3. Enter the extracted directory.
   a. For CentOS, enter:

   ```
   cd centos7
   ```

   b. For Ubuntu, enter:

   ```
   cd ubuntu20
   ```

4. During package installations, the following options that may be used, unless otherwise indicated.

   > Option usage is interchangeable and may include flags (prepended with one dash) and/or names (prepended with two dashes).

   FabreX installation script command options

Option	Description

Flag	Name	
-k	--kernel	Installs **kernel** packages for CentOS or Ubuntu on host
-x	--fabrex	Installs f**abrex** packages on host
-l	--libfabric	Installs **libfabric** packages on host
-n	--nvme	Installs **nvme** packages on host
-g	--gdr	Installs **gdr** packages on host
-f	--full	Use the **full** option to install all packages on the host
-d	--downgrade	Use the **downgrade** option when downgrading a package. Use with an option flag to specify which package to downgrade ⓘ Not supported for Ubuntu.
-a	--airgapped	Use the **airgapped** option when an airgapped installation is intended. Use with an option flag to specify the package ⓘ Not supported for Ubuntu.
-h	--help	Use the **help** option to display a list of command options like the one shown here
-u	--uninstall	Passes a flag to **uninstall** the flag passed it.

5. Run the install script from the extracted directory using the basic command and

options from the preceding table. For example:

```
sudo ./fabrex_install.sh -options --options
```

The following examples show two use cases that demonstrate a CentOS host server installation. The first use case shows how to downgrade a full package install to the prior version using the option names.The second use case shows how to install two modules, fabrex and libfabric using option flags.

> **Important!** The --downgrade option is not required. Also, the --airgapped option is not supported for Ubuntu host servers.

```
sudo ./fabrex_install.sh --downgrade --full   ## for a comple
sudo ./fabrex_install.sh -xl                  ## installing f
```

The script uses a pre-defined hierarchy to install packages in the correct order. The install script displays the results from each package installed in one of the colors below:

- **green** = Successful package install
- **yellow** = Prompts and instructs users on actions to perform
- **red**= Indicates where the installation failed and directs the user to the log file for details

Each time the installation script runs it creates a log file, `install.log.` Each new log file is appended one after another. The installation script also compares the packages installed against those needed, and only installs the packages needed.

6. Once you have finished running the script to install FabreX Server Package modules, continue to the next section to verify the kernel version, grub settings, and fabrexfm-host service.

Verifying Kernel Version, Grub Settings, and fabrexfm-host.service

After a host server reboot, confirm that the kernel version and grub settings have taken effect.

1. Check that you are running the GigaIO kernel by logging into the server and issuing the following command:

   ```
   uname -r
   ```

   The output should be the**<VERSION>** from the Grub configuration step.

2. Verify that IOMMU argument has been applied for the connected CPU server, using step **a** for Intel and step **b** for AMD.

   > ⓘ The BOOT_IMAGE output string should include an argument, **intel_iommu=on** or **amd_iommu=on**, as applicable.

   a. If using an Intel CPU server, issue the following:

      ```
      cat /proc/cmdline | grep intel_iommu=on
      ```

   b. If using an AMD CPU server, issue the following:

      ```
      cat /proc/cmdline | grep amd_iommu=on
      ```

3. Verify that the fabrexfm-host.service is active.

   ```
   $ sudo systemctl status fxworker.service
   ● fxworker.service - FabreX Host Worker Service
      Loaded: loaded (/etc/systemd/system/fxworker.service; enab
      Active: active (running) since Mon 2021-08-23 11:26:00 PDT
    Main PID: 1741 (fxwd)
      CGroup: /system.slice/fxworker.service
              ├─1741 /opt/gigaio-fabrexfm-host-worker/fxwd
              └─1742 /opt/gigaio-fabrexfm-host-worker/fxwd
   ```

```
Aug 24 17:17:37 seaturtle fxwd[1742]:   nfp_peer_ccs_msi_off:
Aug 24 17:17:37 seaturtle fxwd[1742]:   nfp_ccs_msi_data: 0x0
Aug 24 17:17:37 seaturtle fxwd[1742]:   nfp_xlate_addr: 0x0
Aug 24 17:17:37 seaturtle fxwd[1742]:   nfp_xlate_size: 0x0
Aug 24 17:17:37 seaturtle fxwd[1742]:   nfp_tgt_partition: 0x
Aug 24 17:17:37 seaturtle fxwd[1742]: *** *** *** ***
Aug 24 17:17:37 seaturtle fxwd[1742]: ms:<32> INFO       [md:
Aug 24 17:17:37 seaturtle fxwd[1742]: ms:<32> INFO       [md:
Aug 24 17:17:37 seaturtle fxwd[1742]: clear_nic(): - CALLING
Aug 24 17:17:37 seaturtle fxwd[1742]: clear_nic(): status: 0,
Hint: Some lines were ellipsized, use -l to show in full.
```

4. Power down the host server. All servers connected to FabreX Switches need to be powered off prior to installing a topology configuration.

> When power cycling a switch, add 30 seconds to allow the switch power supplies to fully discharge.

Disabling the nouveau driver

The nouveau driver will interfere with FabreX switch settings which sometimes causes GPUs not to be seen by the host server during enumeration process. It is recommended to disable the nouveau to prevent interference with switch settings from occurring.

1. Create a file at `/etc/modprobe.d/blacklist-nouveau.conf` with the following contents:

```
blacklist nouveau
options nouveau modeset=0
```

2. Create a backup copy of the current `initramfs`:

```
sudo cp -p /boot/initramfs-$(uname -r).img /boot/initramfs-$(
```

3. Now create the `initramfs` for the current kernel:

```
sudo dracut -f
```

> The dracut command can take several minutes to complete, depending on the server configuration.

4. Reboot the host server. To ensure the nouveau driver is not loaded, run the following command:

```
lsmod | grep nouveau
```

Checking the fxworker log

The host log output is captured in /var/log/fabrex/fxworker.log on the host server. You can use this log to view the status of the host and its peers. Here is an example of when server1 is connected to server2.

```
Apr 2  13:30:42 server1 fxwd[1970]: update_peer: guid: 500e004a000
...

Apr 2  13:30:42 server1 fxwd[1970]: nfp_peer_id: 'server2'
```

You will need to have the fabric configured to enable the hosts to be connected together.

Verifying the Fabric is Loaded on the Host Server

To verify the fabric is loaded, run the following on each host server:

```
sudo cat /sys/kernel/debug/ntb_hw_switchtec/linkstats
```

This command will produce something similar to the following code example:

```
Switchtec NTB Control Plane Link Status
Hostname "server1": FLID: 0x8FDE0731, NIC IDX: 0, NTB NIC GUID: 0x

PNUM   STATE   PCCS   LLINK   POLL   PSTATE   PLINK   FLID         PEER ID
   0   rdy     cfg    up      slow   rdy      up      0xCAD59D29   server2
```

You will want to see **"up"** for the LLINK and PLINK and **"rdy"** for the STATE and PSTATE, as shown above, for each of the peers of the respective host. The last two lines will only be present if you have more than one host server running and connected to the fabric.

FabreX Manager Switch Utilities

Prerequisites

Prerequisite steps 1 through 5 help determine whether you are ready to bring up the FXS. When ready, follow steps 6 through 11 to bring up the FabreX network.

FabreX Pre-Bringup Checklist

Step	Task
1	Ensure that the equipment (switches and all hosts) participating in the fabric are connected via a management network that supports TCP/IP (typically Ethernet).
2	Ensure that all hosts servers are connected to FabreX Switches designated by the topology of the fabric being loaded. LTR files have the names for server hosts and designate which switch and port each host connects.
3	Ensure that all GigaIO software packages are installed and configured. For installation instructions, see the *GigaIO FabreX System Deployment Guide* on the GigaIO Support Documentation page.
4	Ensure that the host servers, IO resource boxes, and FabreX

	switches are powered down. When power cycling a switch, allow the switch power supplies to fully power down (approximately 30 seconds).
5	Ensure that you have installed FabreX Manager on the management server. If you need to install it, use the section, "**FabreX Manager Tool Setup Guide**," then proceed with step 6 below.
6	Power up all IO resource boxes first. Wait until all the IO boxes fully power up to an operational state.
7	Power up all the FabreX switches. Wait until the switches fully power up and to an operational state.
8	Configure the fabric using the instructions in the section, "**Initializing a Topology on the Fabric**."
9	**Note**: This step applies only to topologies with IO resources. Validate that the DSP (connected to the IO resource boxes) LTSSM are at L0. For instructions, see '**Port Status**' in the "**FabreX Manager Tool Usage**452789192section.
10	Power up all the servers. Wait until the servers fully power up to an operational state.
11	Validate the server links LTSSM are at L0. For instructions, see '**Port Status**' in the "**FabreX Manager Tool Usage**452789192section.

FabreX Manager Tool Setup Guide

The FabreX Manager Tool is an optional command line interface. Its usage is not required when Redfish or other orchestration software tools are available.

Note! The Fabric Manager should only be installed if the user desires to use fmtool to access and configure the FabreX switch. When installing the FabreX Manager Tool it is necessary to use a system that is outside the fabric and run it from a CentOS 7.x (or later), Rocky8.x (or later), or Ubuntu 16.04.x (or later) LTS-based management workstation

The following packages are available for FabreX Manager Tool:

- Centos/Rocky: gigaio-fabrexfm-tool-*.x86_64.rpm
- Ubuntu: gigaio-fabrexfm-tool_*_amd64.deb

These files are downloadable from the Service Desk. Visit Software Releases to download the latest version.

The topology directories will install in the directory `/opt/gigaio-fabrexfm-tool/topologies/release/<topology>` and contain the files needed to initialize the fabric.

CentOS/Rocky Installation

1. Download the latest **CentOS 7.x/Rocky8.x** software release.
2. Install **gigaio-fabrexfm-tool-*.x86_64.rpm** package.

```
sudo yum localinstall -y gigaio-fabrexfm-tool-*.x86_64.rpm
```

Ubuntu Installation

1. Download the latest **Ubuntu 16.04.x** software release.
2. Install **gigaio-fabrexfm-tool_*_amd64.deb** package.

```
sudo apt install ./gigaio-fabrexfm-tool_*_amd64.deb
```

Verifying the fmtool package installation

Using the applicable package (CentOS/Rocky, or Ubuntu), verify the proper fmtool version is available:

1. For CentOS/Rocky, verify that the latest package version was installed:

```
$ sudo yum list installed | grep fabrexfm-tool
gigaio-fabrexfm-tool.x86_64             2.7.*.release   @/gig
$
```

2. For Ubuntu, verify that the latest package version was installed:

```
$ sudo apt list | grep fabrexfm-tool
WARNING: apt does not have a stable CLI interface. Use with c
gigaio-fabrexfm-tool/* 2.7.*.release amd64 [upgradable from:
$
```

3. Verify that fmtool is available:

```
[user@workstation ~]$ which fmtool
/usr/bin/fmtool
[user@workstation ~]$
```

4. Help information is also available by running:

```
[user@workstation ~]$ man fmtool
```

5. For more information about fmtool options, see the section, **FabreX Manager Tool Usage**.

FabreX Manager Tool Files

Once the FabreX Manager Tool (fmtool) package has been installed on a management workstation, the fabric is ready to be configured. To create a configuration, perform the

following steps:

> It is recommended that updates to fmtool be kept in-sync with the FabreX software running on the switch.

1. Create an authorization file.
2. Select a FabreX Loadable Topology Record (LTR) from the list of supported topologies (for details visit Topologies on the Service Desk). See the section, "**FabreX Loadable Topology Record**" for information about LTRs.
3. Edit the accompanying host_list.json file that identifies the site-specific component names in the topology.
4. Configure the topology using the LTR and host_list.json file.

The following sections describe the various files used by the FabreX Manager Tool.

Authorization File

You must provide HTTP Basic authorization credentials to fmtool. The credentials should be stored in a file, the default is $HOME/.fabrex_auth. To create the credentials file, run the following command:

```
$ echo -n admin@gigaio.com:password1 | base64 > ~/.fabrex_auth
$ chmod 600 ~/.fabrex_auth
$ cat ~/.fabrex_auth
YWRtaW5AZ2lnYWlvLmNvbTpwYXNzd29yZDE=
```

> **Important!** If you have multiple switches in your fabric, keep the username and password *exactly* the same for all switches. If you have changed the admin user password (on the manager switch) via the FabreX shell interface, you should change the password on all other worker switches to match. You will also need change the password used to create ~/.fabrex_auth file to match.
>
> If you would like to provide an alternate location for the authorization file, you can direct fmtool to use it by providing the arguments -a <path_to_auth_file> to fmtool. With the default authorization file in place, you should now be able to run

the fmtool command.

```
$  fmtool --status <switch_name>

fmtool: 2023-02-07 at 09:41:21
Requesting status from <switch_name>...
Request:
GET https:// <switch_name>/fabrex/v2/status
Response:
HTTP/1.1 200 OK
++++++++++ Status ++++++++++
fabric_state   : CONFIGURED
fabric_id      : fabrex0
hostlist_state : CONFIGURED

--- Switches ---
product :: RS4024
switch_id :: <switch_name>
switch_state :: CONFIGURED
sw_serial_number ::
sw_uuid ::
sw_info :: {'fabrex': '2.7.0', 'firmware': '3.90 B062', 'fpga': '7
GUID :: 8ad85e00512c000a
config_state :: RUNNING
config_file :: sj5-r-c8c1.ylb.2B32G.4B512G.I16G.X4x128G-1.4--062
location :: {}
...

SUCCESS: status from <switch_name>.
```

If you do not have your authorization information correct or in the default location, you will see the following error returned by fmtool:

```
$ cat ~/.fabrex_auth_bad
```

```
Zm1hZ2VudEBnaWdhaW8uY29tOkhhbmcxMFRheWxvck1hZA==
$ fmtool -a ~/.fabrex_auth_bad -s <switch_name>

fmtool: 2023-02-25 at 13:25:30
Requesting status from <switch_name>...
Response:
{
    "status_code": 401,
    "body": {
        "msg": "The server could not verify that you are authorize
    }
}
Error: Cannot process request
use --verbose for more information

$ fmtool -s <switch_name>

fmtool: 2023-02-25 at 13:22:32
Requesting status from <switch_name>...
Error: No such file or directory - /home/user/.fabrex_auth
```

FabreX Loadable Topology Records

FabreX Loadable Topology Records (LTRs) describe a specific topology consisting of one or more FabreX switches and one or more servers.The supported LTR files are delivered in the FabreX FM Tool package and can be found on the management server. Each of these directories also has a host_list.json file that needs to be modified for components within the topology installed onto the fabric.

```
user@manager_switch:~$ fmtool --topos <switch_name>

fmtool: 2023-02-06 at 16:54:11
Requesting topology list from <switch_name>...
Request:
GET https:// <switch_name>/redfish/v1/Fabrics/Oem/GigaIO/Topologie
Response:
```

```
HTTP/1.1 200 OK
topologies:
2S-8x4-X2
1S-4x2-1
1S-6x0-1
1S-2x4-1
GC3P.7S-18x24-X1
GDK.1S-3x4-X1
1S-8x2-X1
2S-8x4-X1
2S-8x4-1
GP.2S-6x8-X1
2S-4x8-X1
GC2P.5S-12x16-X1
1S-4x2-X1
1S-2x4-X1
current topology: UNCONFIGURED
SUCCESS: topologies from <switch_name>
$
```

Users should never edit or modify a topology file.

The host_list.json file

An example host_list.json file is found in the topology directory and is used to identify required, optional, and restricted components:

- The unique names of FabreX switch(es).
- The unique aliases of any IO devices connected to the "dsp" ports on the switch(es).
- The resource "zone" identifier name(s). A resource zone is a collection of IO devices connect to "dsp" ports (ResourceBlocks) which can be composed together. For a single hybrid switch configuration, the entire switch is one zone. For multiple switch configurations, you may have multiple zones, and each should have a unique name. Redfish uses these zones to keep track of which IO

resources can be connected to which "host" (USP) ports.

- The unique host names of servers connected to the "host" ports of the switch(es).

The initialization PATCH command to `redfish/v1/Fabrics/Oem/GigaIO/Topologies/<id>` should follow these rules:

Rules for Entities

REQUIRED	switch-tags, zone-tags, dsp tags	Switch values must equal the DNS hostname of the switch
OPTIONAL	host-tags	Whether or not supplied by the Redfish template
RESTRICTED	Everything else	

> All switches and server host names must resolve to an IP address.

If the connection is to be blank, you can use any unique name to identify that the connection will not be used. You can change this unique name at a later date when you add another server. All default tag-values MUST be replaced. For any device that is not connected, a dummy name is still required for proper Redfish functionality.

> Host-tags are no longer (since FabreX 2.3) part of the host_list.json content. Any host may join the fabric if connected to an appropriate USP host port. These port connections are in every topology map (tmap.json) and diagram. The Dynamic Host Mode (DHM or dynahost) conditional flag available in the fabshell can be set to include host-tags or exclude them when fetching topology templates via Redfish. The FabreX Manager (FM) behaves *dynamically* with host addition or removal, so long as the host-tags were excluded from the initialization PATCH for the topology. The FM behaves *statically* to host addition and removal if host-tags were included during topology initialization. For more information on 'dynahost' usage, see the fabshell or "**FabreX Shell**" section.

The following example shows the content of the host_list.json file from the GDK.1S-3x4-X1 topology.

Note the absence of a comma for the last entry in the example. You leave out the comma In the last entry, which is the required syntax. Be sure to use the exact syntax shown when configuring your host JSON file. Only change the parameters on the right side of the colon between the quotes.

```
$ cat host_list.json
{
    "s1p7dsp": "<SWITCH1_PORT7_NAME>",
    "s1p9dsp": "<SWITCH1_PORT9_NAME>",
    "s1p11dsp": "<SWITCH1_PORT11_NAME>",
    "s1p13dsp": "<SWITCH1_PORT13_NAME>",
    "switch1": "<SWITCH1_NAME>",
    "zone1": "<ZONE1_NAME>"
}
```

If you need to change the server after you have a fabric up and running, see the section, "**Changing a Server Hostname in an Existing Fabric**."

Each data line in the template is a TYPE, VALUE pair. Elements on the left (before each colon) are TYPE elements that give the generic identifier for the topological element. Elements to the right of each colon are the VALUE elements, which must be substituted with user provided data.

For example, the tag, switch1, is the interface for switch1's DNS name. If your switch is named, frodo as shown in the following example, you would replace the angle-bracket-portion with "frodo" in a PATCH command to the redfish/v1/Fabrics/Oem/GigaIO/Topologies/1S-2x4-1 resource.

In addition, other non-switch-oriented tags in the template must be substituted as well, and the substituted values become <id> portions of other Redfish resources.

```
$ cat host_list.json
{
```

```
    "s1p9dsp": "JBOG_PORT_9",
    "s1p13dsp": "UNUSED_PORT_13",
    "s1p17dsp": "JBOF_PORT_17",
    "s1p21dsp": "UNUSED_PORT_21",
    "switch1": "frodo",
    "zone1": "zone1"
}
```

Quite simply, the DSP port tags (and zone tags) will need some kind of alias, which is used in referencing Redfish information queried later on. So, choose an alias that makes sense and use Linux-like hostname rules for those aliases.

FabreX Manager Tool Usage

The FabreX Manager Tool (fmtool), along with the files in the **FabreX Manager Tool Files** section, are used to configure a topology. The fmtool executable installs in the /usr/bin directory.

Its usage is given below:

```
usage: fmtool [-h] [--help]
       fmtool [-V] [--version]
       fmtool [-a <AUTH_FILE>] [-v] <HOST_FILE> <TOPO_ID> <FABRIC_
       fmtool [-a <AUTH_FILE>] [-v] -B switch:<SWITCH_NAME>,part_i
       fmtool [-a <AUTH_FILE>] [-v] -l <LOGLEVEL> <FABRIC_MANAGER>
       fmtool [-a <AUTH_FILE>] [-v] -L <WORKER1>:<LOGLEVEL1>,...,<
       fmtool [-a <AUTH_FILE>] [-v] [-j] -s <FABRIC_MANAGER>
       fmtool [-a <AUTH_FILE>] [-v] [-j] -t <FABRIC_MANAGER>
       fmtool [-a <AUTH_FILE>] [-v] [-j] -T <TOPO_ID> <FABRIC_MANA
       fmtool [-a <AUTH_FILE>] [-v] -u <FABRIC_MANAGER>
       fmtool [-a <AUTH_FILE>] [-v] -U switch:<SWITCH_NAME>,port_i
       fmtool write-config
```

```
        Deprecated. To be removed by 2.8:
        fmtool[-a <AUTH_FILE>][-v][-f] <HOST_FILE> <LTR_FILE> <FABR

...
```

⚠ The fabric_manager is the same as the manager_switch.

General options

FM Tool General Options

Option	Usage
man fmtool	Brings up the fmtool manual. Similar output as fmtool -h
-a --auth	Path to base64 encoded auth file, such as 'echo -n email:password | base64 > ~/.fabrex_auth' (default: /home/user/.fabrex_auth)
-B --bind	Allows a user to bind downstream port(s) to a partition on a switch. switch:<SWITCH_NAME>,part_id:<PARTITION_ID>,port_id:<PORT_ID>, --bind switch:<SWITCH_NAME>,part_id:<PARTITION_ID>,port_id:<PORT_ID>
-f --force	Forces a reset of switch(es) and iniitializes the fabric. Meaningful only when loading a topology with deprecated form
<fabric_manager>	This is the name of the switch identified as the <manager_switch> for the topology. It must be one of the switch names identified in the host_list.json file and must be used exclusively for subsequent fmtool commands. See the section, "**Initializing a Topology on the Fabric**."

	All other switches in the topology will need to have their "showmgr" setting be set to the manager switch name using the "setmgr" command. By default, each switch has sw-rs4024 or sw-rs3024 as the manager switch. In multi-switch topologies, no two switches in topology can have the exact same name.
<host_file>	Provides tags for components within the topology being installed onto the fabric. It provides the path to host_list.json file. Usually delivered along with a ltr.yml file. See the section, "**Initializing a Topology on the Fabric**." A host list identifies: • Hostname of FabreX switch(es) • Hostname of server(s) connected to the port(s) on the switch(es) • Aliases of any I/O/drive(s) connected to the ports on the switch(es) • Zone identifier name(s). For unused port(s), use any arbitrary string such as 'blank1', 'blank2,' and so on.
-j --json	Suppress transaction info and output JSON data to console where appropriatej.
-l , --loglevel <loglevel>	Changes the loglevel of the <fabric_manager>. Allowed values are: <INT>:[0, DEBUG, 1, INFO, 2, WARNING, 3, ERROR, 4, CRITICAL]
-L --worker-loglevel	Change the loglevel of worker(s). Allowed values are: <INT>:[0, DEBUG, 1, INFO, 2, WARNING, 3, ERROR, 4, CRITICAL] Example: <WORKERi>:<LOGLEVELi>[,<WORKERj>:<LOGLEVELj>...], --worker-loglevel <WORKERi>:<LOGLEVELi>[,<WORKERj>:<LOGLEVELj>...]

<ltr_file|topo_id>	Path to ltr.yml file or a topo_id. FabreX Topology Records (LTRs) describe a specific topology consisting of FabreX switch(es) and server(s). A topo_id is the explicit name of a topology as retrieved from the --topos command.
-s --status	Provides fabric status information - initialized/uninitialized; connected and configured hosts and switches (port, partition, configuration, and binding information)
-t, --topos	Provides list of topologies installed on the fabric manager.
-T --template	Provides the topology host list file template JSON output for the requested topology.
-u --uninitialize	Uninitializes (dismantles) a fabric.
-U --unbind	Unbind DSP (downstream port) command. Allows a user to unbind port(s) from a partition on a switch. Example: -U switch:<switch_name>,port_id:<port_id>, --unbind switch:<switch_name>,port_id:<port_id>
-v --verbose	Increases the verbosity of the fmtool output
-V --version	Show the program's version number and exit.
--no-prompts	Never prompt for confirmation.
--write-config	Write a default fmtool config file to the user's FabreX directory.

Displaying the fmtool man page

The following example shows the current usage to display the FabreX Manager Tool 'fmtool' Manual.

```
user@manager_switch:~$ man fmtool
FMTOOL(1)
```

```
NAME
       fmtool - fmtool 2.7.0

SYNOPSYS
       fmtool [-h] [--help]
       fmtool [-V] [--version]
       fmtool [-a <AUTH_FILE>] [-v] <HOST_FILE> <TOPO_ID> <FABRIC_
       fmtool [-a <AUTH_FILE>] [-v] -B switch:<SWITCH_NAME>,part_i
       fmtool [-a <AUTH_FILE>] [-v] -l <LOGLEVEL> <FABRIC_MANAGER>
       fmtool [-a <AUTH_FILE>] [-v] -L <WORKER1>:<LOGLEVEL1>,...,<
       fmtool [-a <AUTH_FILE>] [-v] [-j] -s <FABRIC_MANAGER>
       fmtool [-a <AUTH_FILE>] [-v] [-j] -t <FABRIC_MANAGER>
       fmtool [-a <AUTH_FILE>] [-v] [-j] -T <TOPO_ID> <FABRIC_MANA
       fmtool [-a <AUTH_FILE>] [-v] -u <FABRIC_MANAGER>
       fmtool [-a <AUTH_FILE>] [-v] -U switch:<SWITCH_NAME>,port_i
       fmtool write-config

       Deprecated. To be removed by 2.8:
       fmtool[-a <AUTH_FILE>][-v][-f] <HOST_FILE> <LTR_FILE> <FABR

DESCRIPTION
       Client for FabreX Fabric Management; allows a user to initi

       Supported Version(s): FabreX 2.7.0

OPTIONS
       Following argument list describes all supported arguments,

   POSITIONAL ARGUMENTS
        <HOST_FILE>          Path to host_list.json file. Usually
                         A host list is needed to identify:
                         (a) The hostname of FabreX switch(es)
                         (b) The hostname of server(s) connected to
                         (c) The aliases of any IO/drive(s) connect
                         (d) The zone identifier name(s).
                         For unused port(s), use any arbitrary stri
   <LTR_FILE|TOPO_ID>    Path to ltr.yml file, or a topo_id. FabreX
```

```
                        a specific topology consisting of FabreX s
                        A topo_id is the explicit name of a topolc
   <FABRIC_MANAGER>     This is the hostname of the switch identif
                        the topology. It must be used exclusively

  ...
```

Uninitializing a Topology on the Fabric

To remove the currently established topology, use the uninitialize command. The following example shows the stucture from the help file.

The `fmtool` usage for uninitializing a fabric is given by:

```
$fmtool [-a <AUTH_FILE>] [-v] -u <FABRIC_MANAGER>
```

The expected output is shown in the following example:

```
$ fmtool --uninitialize <switch_name>

fmtool: 2023-02-06 at 11:20:06
You are about to UNINITIALIZE (dismantle) a fabric! Ensure that al
        Proceed? [Y/N]:y
Sending uninit fabric to <switch_name>...
Request:
PUT https:// <switch_name>/fabrex/v2/uninit
Response:
HTTP/1.1 200 OK
SUCCESS: uninitialize fabric complete on <switch_name>
$
```

Initializing a Topology on the Fabric

The process for FXS v2.7.0 differs from earlier versions in the following ways:

1. Determine the topology of interest:
    a. Run command fmtool -t or --topos, and
    b. Run fmtool -T or --template <topo_id> to gather the topology of interest in a topo_id template
2. Save the template, modifying it as needed to prepare the host_list.json file.

The following example shows how the selecting the topology process has changed for FabreX 2.7.0:

```
fmtool [-a <AUTH_FILE>] [-v] <HOST_FILE> <TOPO_ID> <FABRIC_MANAGER
```

Note that previous FXS versions used a [-f] option and identified a <ltr_file> instead of a <TOPO_ID> in the fmtool command line. The former command structure is shown in the example below.

```
fmtool [-a <AUTH_FILE>>] [-v] [-f] <HOST_FILE> <ltr_file> <FABRIC_
```

So too, has the configuration process changed, the current release uses a template provided by the master switch. The master switch communicates with the switch that is being configured with a topology. The template provides the topology host_list file template in a JSON output for the requested topology.

The next FabreX release will deprecate the command structure style previously used in earlier FXS versions. For now, users can apply the new process or continue to use the process listed in previous FXS versions. For reference, the earlier process is

The following procedure shows how fmtool is used to select a topology, gather the topology into a template and use it to prepare the host_list.json.file.

> ⓘ Before you proceed, make sure to unintialize a fabric before initializing it. See section on **Uninitializing a Topology on the Fabric**.

1. Determine the topology of interest by running fmtool -t or --topos.

    ```
    fmtool --topos <switch_name>
    ```

```
fmtool: 2023-02-06 at 16:54:11
Requesting topology list from <switch_name>...
Request:
GET https:// <switch_name>/redfish/v1/Fabrics/Oem/GigaIO/Topo
Response:
HTTP/1.1 200 OK
topologies:
  2S-8x4-X2
  1S-4x2-1
  1S-6x0-1
  1S-2x4-1
  GC3P.7S-18x24-X1
  GDK.1S-3x4-X1
  1S-8x2-X1
  2S-8x4-X1
  2S-8x4-1
  GP.2S-6x8-X1
  2S-4x8-X1
  GC2P.5S-12x16-X1
  1S-4x2-X1
  1S-2x4-X1
current topology: GDK.1S-3x4-X1
SUCCESS: topologies from <switch_name>
$
```

2. Gather the topology of interest in a topo_id template by running fmtool -T or --template <topo_id>. The following example uses GDK.1S-3x4-X1 as the topology of interest.

```
$ fmtool -T GDK.1S-3x4-X1 <switch_name>

fmtool: 2023-02-07 at 08:41:23
Request:
GET https:// <switch_name>/redfish/v1/Fabrics/Oem/GigaIO/Topo
Response:
HTTP/1.1 200 OK
```

```
host list template json:

{
    "s1p7dsp": "<SWITCH1_PORT7_NAME>",
    "s1p9dsp": "<SWITCH1_PORT9_NAME>",
    "s1p11dsp": "<SWITCH1_PORT11_NAME>",
    "s1p13dsp": "<SWITCH1_PORT13_NAME>",
    "switch1": "<SWITCH1_NAME>",
    "zone1": "<ZONE1_NAME>"
}
$
```

3. Next, is to prepare the host_list.json file by snipping out the appropriate section of the template shown in step 2 to create a host_list.json file, then save the template using a unique name. The following example, shows the modified host_list.json file, named **GDK.host_list.json**.

```
$ cat GDK.host_list.json
{
    "s1p7dsp": "JBOF-1",
    "s1p9dsp": "JBOF-2",
    "s1p11dsp": "JBOF-3",
    "s1p13dsp": "JBOF-4",
    "switch1": "<switch_name>",
    "zone1": "IOZone1"
}
$
```

4. Once you have a new topology, initialize it using the modified host_list.json created from the template.

```
$ fmtool -v GDK.host_list.json GDK.1S-3x4-X1 <switch_name>


fmtool: 2023-02-07 at 09:13:22
cfg-file: CFG FILE '/home/user/fabrex/fmtool.ini' NOT FOUND
Configuring fabric...
```

```
Redfish INIT for topo GDK.1S-3x4-X1
using authorization file: /home/user/.fabrex_auth
mode: PATCH; url: https:// <switch_name>/redfish/v1/Fabrics/O
Request:
PATCH https:// <switch_name>/redfish/v1/Fabrics/Oem/GigaIO/To
Response:
HTTP/1.1 200 OK
{
    "@odata.id": "/redfish/v1/Fabrics/Oem/GigaIO/Topologies/G
    "@odata.type": "#GigaIO.Oem.Topology",
    "Entities": {
        "s1p7dsp": "JBOF-1",
        "s1p9dsp": "JBOF-2",
        "s1p11dsp": "JBOF-3",
        "s1p13dsp": "JBOF-4",
        "switch1": "<switch_name>",
        "zone1": "IOZone1"
    },
    "Id": "GDK.1S-3x4-X1",
    "Name": "GDK.1S-3x4-X1"
}
SUCCESS: sent GDK.1S-3x4-X1 to <switch_name>
NOTE: Initializing fabric asynchronously...
Run 'fmtool -s <switch_name>' to check host and/or switch con
$
```

5. At this point, fmtool --status can be run to verify the host/switch states, as shown in the following example:

```
$ fmtool --status <switch_name>


fmtool: 2023-02-07 at 09:41:21
Requesting status from <switch_name>...
Request:
GET https:// <switch_name>/fabrex/v2/status
Response:
```

```
HTTP/1.1 200 OK
++++++++++ Status ++++++++++
fabric_state   : CONFIGURED
fabric_id      : fabrex0
hostlist_state : CONFIGURED

--- Switches ---
product :: RS4024
switch_id :: <switch_name>
switch_state :: CONFIGURED
sw_serial_number ::
sw_uuid ::
sw_info :: {'fabrex': '2.7.0', 'firmware': '3.90 B062', 'fpga
GUID :: 8ad85e00512c000a
config_state :: RUNNING
config_file :: sj5-r-c8c1.ylb.2B32G.4B512G.I16G.X4x128G-1.4--
location :: {}
...
```

6. Lastly, you can also use fmtool --topos if you wish to see or confirm which topology is running. Look for **current topology** at the end of the output:

```
$ fmtool --topos <switch_name>


fmtool: 2023-02-07 at 09:48:38
Requesting topology list from <switch_name>...
Request:
GET https:// <switch_name>/redfish/v1/Fabrics/Oem/GigaIO/Topo
Response:
HTTP/1.1 200 OK
topologies:
  2S-8x4-X2
  1S-4x2-1
  1S-6x0-1
  1S-2x4-1
  GC3P.7S-18x24-X1
```

```
  GDK.1S-3x4-X1
  1S-8x2-X1
  2S-8x4-X1
  2S-8x4-1
  GP.2S-6x8-X1
  2S-4x8-X1
  GC2P.5S-12x16-X1
  1S-4x2-X1
  1S-2x4-X1
current topology: GDK.1S-3x4-X1
SUCCESS: topologies from <switch_name>
```

Checking the Status of the Fabric

The `fmtool` usage to obtain the status of a fabric is given by:

```
fmtool [-a <AUTH_FILE>] [-v] [-j] -s <FABRIC_MANAGER>
```

The status information is returned in JSON form and is quite long. It consists of the following structure:

- Fabric
- Switches
- Hosts

Fabric Status

The following example shows obtaining status for the single switch topology 1S-2x4-1. Here the **Fabric** state is `CONFIGURED`, meaning a topology and host list file have been used to configure the topology:

```
$ fmtool --status <switch_name>


fmtool: 2023-02-07 at 09:41:21
```

```
Requesting status from <switch_name>...
Request:
GET https:// <switch_name>/fabrex/v2/status
Response:
HTTP/1.1 200 OK
++++++++++ Status ++++++++++
fabric_state   : CONFIGURED
fabric_id      : fabrex0
hostlist_state : CONFIGURED

--- Switches ---
product :: RS4024
switch_id :: <switch_name>
switch_state :: CONFIGURED
sw_serial_number ::
sw_uuid ::
sw_info :: {'fabrex': '2.7.0', 'firmware': '3.90 B062', 'fpga': '7
GUID :: 8ad85e00512c000a
config_state :: RUNNING
config_file :: sj5-r-c8c1.ylb.2B32G.4B512G.I16G.X4x128G-1.4--062
location :: {}
...
```

Also note that the final `status_code` for the `fmtool -s` command was 200, an indication that the command is complete.

Here is example output for `fmtool -s` for a switch (or topology) that is NOT `CONFIGURED`:

```
[user@fmtool-server 1S-2x4-1] $ fmtool -s <switch_name>

fmtool: 2023-01-19 at 11:57:32
Requesting status from <switch_name>...
Request:
GET https:// <switch_name>/fabrex/v2/status
Response:
```

```
Response:
HTTP/1.1 200 OK
++++++++++ Status ++++++++++
fabric_state   : NOTCONFIGURED
fabric_id      : None
hostlist_state : NOTCONFIGURED

--- Switches ---
product :: RS4024
switch_id :: <switch_name>
switch_state :: CONFIGURED
sw_serial_number ::
sw_uuid ::
sw_info :: {'fabrex': '2.7.0', 'firmware': '3.9x', 'fpga': '7a9.30
GUID :: 0000050110200006
config_state :: RUNNING
config_file :: sj5-r-c043.ylb.2B16G.4B256G.I16G.X4x64G-1.4--062
location :: {'Id': 'drawer4', 'ParentId': 'rack2'}
==============================

...

================================
  Binding info:
bound       : [9, 13, 17, 21]
can_unbind  : True
unbound     : []
can_bind    : False
bind_dest   : [0, 1, 2]
open_slots  : True
================================

--- Hosts ---
Success: Status from <switch_name>
```

Host Server State and Status

The following table lists host states.

Host Server States

Field	Host state indicates if the host is...
DISCONNECTED	Disconnected from fabric management communicating with the host server. This is usually because the host server is powered off.
NOTCONFIGURED	In the initial state of a host server before it connects to or configures with fabric management. If the host server is powered, but still in this state, the host driver may not be installed.
CONNECTED	Up and communicating with fabric management, with the host server connected and ready to accept requests from the manager.
CONFIGURED	Configured by fabric manager; the host server is now ready to communicate with other peer servers that are also CONFIGURED.

The following sample output shows two hosts that are `CONNECTED` and `CONFIGURED,` while the third host is `DISCONNECTED.`

The peers server status indicates if the host server's peers are:

- NOTCONFIGURED - Fabric manager has not configured the peer host for communication with the host server.
- CONFIGURED - Fabric manager has configured the peer server for communication with the host server.

```
...

            "Host List State": "CONFIGURED",
            "hosts": [
```

```
{
    "pc_peer_id": "HOST1",
    "peers": [
        {
            "pc_peer_id": "HOST2",
            "state": "CONFIGURED"
        },
        {
            "pc_peer_id": "HOST3",
            "state": "NOTCONFIGURED"
        }
    ],
    "state": "CONFIGURED"
},
{
    "pc_peer_id": "HOST2",
    "peers": [
        {
            "pc_peer_id": "HOST1",
            "state": "CONFIGURED"
        },
        {
            "pc_peer_id": "HOST3",
            "state": "NOTCONFIGURED"
        }
    ],
    "state": "CONFIGURED"
}
{
    "pc_peer_id": "HOST3",
    "peers": [
        {
            "pc_peer_id": "HOST1",
            "state": "NOTCONFIGURED"
        },
        {
            "pc_peer_id": "HOST2",
            "state": "NOTCONFIGURED"
```

```
                        }
                    ],
                    "state": "DISCONNECTED"
                }
            ]
...
```

Switch State and Status

The following output shows switch_state details for a worker switch state in the topology. The switch state returns with the following fields:

Worker Switch States

Field	Fabric Management state is...
DISCONNECTED	Disconnected from communicating with the worker switch (in a multi-switch topology). Usually this occurs because the worker switch is powered off.
NOTCONFIGURED	In the initial state of a worker switch before it connects to or configures with fabric management. If the worker switch is in this state it may be due to: • setmgr having the wrong manager_switch set • installed version has a mismatch
CONNECTED	Up and communicating, with the worker switch connected and ready to accept requests from the manager switch.
CONFIGURED	Configured for the worker switch with the correct configuration file.

The following output shows status details for a manager or worker switch status in the topology. The switch status returns with the following fields:

Switch State Status

Field	Status
switch_state	See switch states in the preceding table
binding	Shows the binding status for all DSPs
GUID	Switch GUID
switch_id	Name of the manager or worker switch
config_state	Lists the configuration state of the topology the fmtool command loaded
ports	Lists the port details
config_file	Lists the file that is configured for the manager or worker switches

```
...
            "Switches": [
                {
                    "switch_state": "CONFIGURED",
                    "binding": {
...
                    },
                    "GUID": "0000010105190003",
                    "switch_id": "<switch_name>",
                    "config_state": "RUNNING",
                    "ports": [
...
                    "config_file": "sj1-r-c043.yls.2B32G.I32G-1.3-
                }
            ]
...
```

Switch Configuration State and File

A topology consists of one or more switches, where each switch is running a particular configuration. The `config_state` field indicates one of the following states:

Switch Configuration States

State	Configuration file in a switch stanza...
UNAVAILABLE	Is invalid; no action is taken
LOADABLE	Is valid; can be loaded, activated, and run after a switch reset
ACTIVATED	Is loaded and configured/activated; a switch reset will run this file
RUNNING	Is active and running

The `config_file` field indicates which configuration has been loaded on the switch and is running.

```
...
                "config_state": "RUNNING",
...
                "config_file": "sj1-r-c043.yls.2B32G.I32G-1.3--08c
...
```

Switch Partitions

Switch ports are grouped into partitions. Each partition contains one upstream port (USP) and zero or more downstream ports (DSPs).

Port Numbering

Use the lowest numbered Switch Port Number to designate which port connections to bind or unbind. All x4 and x8 or x16 grouped ports are referenced by the lowest numbered Switch Port Number.

Port Binding Status

Switch ports are `bound` to a partition. DSPs can be unbound from a partition and bound to another partition. The `binding` field indicates the current port binding configuration for the given switch.

Port Binding Status Port Fields

Port Field	Usage
`can_unbind`	True if there are any DSPs in the bound list
`bind_dest`	Valid partition numbers
`unbound`	List of DSPs currently not bound to any partition
`bound`	List of DSPs currently bound to a partition
`open_slots`	True if there is a DSP that can be bound/unbound from a bind_destination
`can_bind`	True if there are any DSPs in the unbound list

The following example shows how the port fields are used to bind or unbind a partition.

```
...
                    "binding": {
                        "can_unbind": true,
                        "bind_dest": [
                            0,
                            1
                        ],
                        "unbound": [],
                        "bound": [
                            9,
                            13,
                            17,
                            21
                        ],
```

```
                        "open_slots": true,
                        "can_bind": false
                    },
...
```

Port Status

A switch port represents a PCIe port on the switch. The **ports** field indicates the current PCIe port configuration for the given switch.

Port Configuration Fields

Port Field	Usage
`Dir`	Indicates whether the port is upstream (USP) or downstream (DSP).
`LexPort`	The physical port number(s) corresponding to this PCIe port, with the range of switch ports comprising an x16, x8, or x4 link. In fmtools, it references the lowest port number in the range shown for the port grouping.
`Par`	The partition of the port is currently bound to on the switch
`Max`	The maximum lane count possible for this port or group of ports: x16, x8, or x4
`Neg`	The negotiated lane count of this port or group of ports: x16, x8, or x4
`Status`	The PCIe physical link status of the port: UP or DOWN
`Rate`	The negotiated PCIe link rate of the port, with the following possible values: • 16G: 16 Gbits/sec per lane, Gen 4 PCIe • 8G: 8 Gbits/sec per lane, Gen 3 PCIe • 5G: 5 Gbits/sec per lane, Gen 2 PCIe • 2.5G: 2.5 Gbits/sec per lane, Gen 1 PCIe

The following example shows how the ports field is used to indicate the current PCIe port configuration for the given switch. The ports are grouped by the partition they are bound to.

```
...
                    "ports": [
                        {
                            "Neg": "x16",
                            "Rate": "8G",
                            "Status": "UP",
                            "LexPort": "1..4",
                            "Dir": "USP",
                            "Max": "x16",
                            "Par": "0"
                        },
                        {
                            "Neg": "x0",
                            "Rate": "2.5G",
                            "Status": "DOWN",
                            "LexPort": "9..12",
                            "Dir": "DSP",
                            "Max": "x16",
                            "Par": "0"
                        },
...
```

Stop the Fabric

The key reasons to run this command:

- To add a new or different host server/switch
- To change the physical connection to any switch in the fabric
- To reboot or power cycle a worker switch (non-manager)

⚠ If rebooting the switch, the host servers should be powered down; otherwise, they may experience a kernel panic due to PCIe sub-tree disappearing while the switch is rebooting.

When power cycling a switch, allow the switch power supplies to fully power down (approximately 30 seconds).

The `fmtool usage to` remove the current topology of a fabric is given by:

```
fmtool [-v] -u <manager_switch>
```

The following example de-configures the single switch topology configured above, where the switch host name is ***<manager_switch>***:

```
$ fmtool -u <manager_switch>
Sending uninit: <manager_switch>...Done
```

Unbind and Bind IO Ports in a Running Topology

ⓘ Power down the host server prior to running a bind or unbind operation on IO ports.

The `fmtool` usage for binding and unbinding IO in a fabric is given by:

```
fmtool [-v] -U switch:switch_name,port_id:<port ID> <manager_switc
fmtool [-v] -B switch:switch_name,part_id:<partition ID>,port_id:<
```

The following example unbinds the DSP, identified by `port_id` 13, from its current partition on switch `switch_name`. Note that if you only have one switch, `switch_name`

is equal to <manager_switch>.

```
$ fmtool -U switch:<switch_name>,port_id:13 <switch_name>
Sending unbind: <switch_name>...Done
```

If you have multiple switches, use switch_name to indicate the switch from which to unbind/bind the DSP. The following example shows how to bind switch_name port 13 to partition *1* on <manager_switch>:

```
$ fmtool -B switch:<switch_name>,part_id:1,port_id:13 <manager_swi
Sending bind: <manager_switch>...Done
```

Bind Error Report

In bind and unbind operations, the correct partition and port numbers must be specified. Errors include attempts to:

- bind a bound port
- unbind an unbound port
- unbind a USP
- specify an invalid partition
- specify an invalid partition/port combination

The error output will list valid partitions and ports. The following example shows an attempt to bind switch port 13, which fails because switch port 13 is not in the unbound ports list.

```
$ fmtool -B switch:switch name,part_id:1,port_id:13 <manager_switc
Sending bind: <manager_switch>...Error 500, bind failed
valid paritions: [0,1]
bound ports: ["9","13","17","21"]
unbound ports: []
```

Set Switch log level

Each switch produces a log output to a file. The `fmtool` usage to set the log level for a switch is given by:

```
fmtool [-v] [-l <loglevel>] <manager_switch>
```

Valid values for `loglevel` are:

Value	loglevel
0	DEBUG
1	INFO
2	WARNING
3	ERROR
4	CRITICAL

The default loglevel is 1 (INFO). Note that loglevel 0 includes 1-4, loglevel 1 includes 2-4, and so on.

The following example sets the `loglevel` for switch `<manager_switch>` to 0.

```
$ fmtool -l 0 <manager_switch>
Sending loglevel: <manager_switch>...Done
```

Set fxwd log level

Each switch produces output to the fxworker logging file. Using fmtool command you can set the logging level independently on each host.

```
fmtool [-v] -L server1_hostname:<loglevel1>,...,serverN_hostname:<
```

The following example sets the `loglevel` for hosts `server1_hostname` and `server2_hostname` to 0 (DEBUG):

```
$ fmtool -L server1_hostname:0,uppers:0 <manager_switch>
Sending host loglevel: <manager_switch>...Done
```

The host log output is captured in /var/log/fabrex/fwworker.log on `server2_hostname`. In looking at the log after running the command above, notice the additional *DEBUG* output in the log,

```
Feb  3 21:59:18 host1 fxwd: [INFO    ] SET LOGLEVEL: 0
Feb  3 21:59:29 host1 fxwd: [INFO    ] REMOVE PEER: "server1_hostn
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_peer_id = server1_h
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_peer_flid = 1
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_host_index = 0
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_ntb_guid = 0x500e00
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_dtf_bar: 2
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_dtf_sz: 0x800000000
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_dtf_off: 0
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_mtf_bar: 2
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_mtf_sz: 0x100000
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_mtf_off: 0xfee00000
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_peer_ccs_off = 0
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_peer_ccs_msi_off 0
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_peer_ccs_msi_data 0
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_stat = 0x2
```

This `fxwd loglevel` persists until changed by `fmtool`, host is rebooted, or the `fxworker` service is restarted on the host.

Changing a Server in an Existing Fabric

It is highly recommended not to change the server hostname since it requires the running fabric be stopped to include the new server. If you must change the server, use the following procedure:

1. Stop all host applications on the host you are trying to replaced that uses the fabric for communication.
2. Power down the server.
3. Connect to the new server and power on the server.

FabreX Shell (fabshell) Interface

The FabreX shell, called fabshell interface enables users to interact with the FabreX Switch on a limited basis. The fabshell permits users to configure a switch for the local network and perform minimal maintenance duties not supported elsewhere. Users can also use fabshell to power off the switch, reboot the switch, set the system time, set the hostname, and other actions.

Users have two options for logging into fabshell:

1. You can ssh using an IP address (default), but only if you have the switch connected to the network and know the Switch IP address or DNS name.
   If you are connected to a network, go to the section, "**Logging into fabshell using an IP Address or DNS Name**."

2. Or, you can connect to the FabreX Switch to configure the IP address. Go to the section, "**Logging into fabshell Using the Console Setup**."

Choose the option applicable to your network configuration. Each option provides you with the same fabshell information.

Logging into fabshell Using the Console Setup

For the console setup, connect a keyboard, mouse, and monitor to the associated ports on the FabreX Switch. You must connect to the FabreX Switch to configure the IP address.

1. Once you are connected to the switch, log in and enter your password using the following screen.

> **Important!** If you have multiple switches in your fabric, the username and password must be kept *exactly* the same for all switches.

```
GIGAIO FabreX (TM) 24 Port PCIe Switch
fabrex2 login: admin
password: password1

WELCOME to FABREX (TM)
Last login: Thu Oct 13 06:13:57 2022 from <switch IP address>
FabreX Shell version: 2.x.x
```

> After the "Welcome to FabreX" message, the fabshell switch information (swinfo) screen displays switch details. These details include current port and LED statuses, and package versions, and services. This information can be viewed anytime by running `swinfo` from the fabshell. To learn more about various options, see the section "**fabshell Options**."

2. Congratulations! You are now in the FabreX Shell.
   You will see the same switch information shown for the other login option, "**Logging into fabshell using an IP Address or DNS Name**." With either option, you may run the **`swinfo`** command at any time to view switch details again.
3. To run various fabshell commands go to the section, "**fabshell Options**."

> Once you have finished setting up the switch, be sure to note the switch <hostname>. You will need the hostname to initialize the fabric with fmtool.

Logging into fabshell Using an IP Address or DNS Name

You must have the switch connected to the network and know the switch IP address or DNS name to run DHCP setup.

The 'dhcp' command enables DHCP, which is the default for the switch.

1. Type `ssh admin@`<IP address or the DNS switch name>.
2. Enter the password as `password1`.

   Important! If you have multiple switches in your fabric, the username and password must be kept *exactly* the same for all switches.

3. Congratulations! You are now in the FabreX fabshell.
   You will see the same switch information shown in earlier section,"**Logging into fabshell Using the Console Setup**." With either option, you may run the **`swinfo`** command anytime to view the switch information again.
4. To run various fabshell commands go to the section, "**fabshell Options**."

   Once you have finished setting up the switch, note the switch <hostname>. You will need the hostname to initialize the fabric with fmtool.

Fabshell Options

The fabshell has several commands that enable a user to interact with the FabreX Switch on a limited basis. For example, configuring a switch for the local network or perform minimal maintenance duties not supported elsewhere. The following sections briefly describe the most commonly used commands and demonstrate their use.

- Use the **Shift** and **Page Up** keys together if you need to scroll up the display, such as when the screen's content scrolls out of view during fabshell initialization or when running the swinfo command.
- Use the **Tab** key to autocomplete commands entered on the command line. The fabshell user interface supports command auto-completion. Autocomplete should work once the completion engine can locate a uniquely matching command.

> These keyboard shortcuts are useful when running fabshell while physically connected to the switch's management port using a monitor and keyboard.

Help

The help command displays all the available help topics for the switch. To see a list of fabshell command options, type **`help`**.

```
fx>help

Commands (type help <topic>):
.............................
autoflash    esverify  htable   ping     reboot   shutdown     swr
checkpoints  fbank     inspkg   pwrs     rmpkg    shutdownlog  tai
delhst       flashfw   logs     q        service  static       tim
dhcp         help      netinfo  quit     setmgr   summary      top
dynahost     hostname  passwd   readcmi  showmgr  swinfo       upl

fx>
```

Use help with an individual command to display the options available. For example, if you wish to see a brief description of every command offered, use `help summary`. Each of the following sections includes a help topic example.

Summary

The summary command displays categories listing relevant commands, along with a brief description. For instance, if you want to interact with firmware, summary displays the applicable commands under Firmware Maintenance. Type **`'summary'`** to see each category, along with applicable commands.

```
fx>help summary
summary: List a command summary

fx>summary
Firmware Maintenance Commands --
```

```
  autoflash   : Query the auto recovery mode, or set it to ON or C
  fbank       : Query or toggle the current flash bank.
  flashfw     : Reprogram the flash with the current FabreX packag
  swreset     : Reset the PCIe switch.
Logging Commands --
  logs        : Copy current logs into ftp area for retrieval.
  tail        : Display the last lines of the switch worker logfil
Network Commands --
  dhcp        : Set the switch to use dhcp. <Default>
  htable      : Update the static host table for network name resc
  netinfo     : Display network information.
  ping        : Issue a ping command.
  static      : Set a static IP for the switch.
Switch Software Package Commands --
  inspkg      : Install an available FabreX software package on th
  rmpkg       : Delete an available package file from switch stora
  uplgio      : Upload a .gio package file from a USB device to sw
Other Commands --
  checkpoints : Check status and manage fxmanager checkpoints.
  delhst      : Delete all history items.
  dynahost    : Query, enable, or disable state of Dynamic Host Mc
  esverify    : Set/Clear event sender ssl certificate verificatic
  help        : Show this screen.
  hostname    : Set the hostname and optional domain name for the
  passwd      : Set the admin password.
  pwrs        : Query the password restoration mode, or set it to
  readcmi     : Query cables and/or connected interface cards over
  reboot      : Restart the switch computer without a PCIe switch
  service     : Stop, start, or restart one or more services.
  setmgr      : Sets the manager of the fabric in which this switc
  shutdown    : Power OFF the switch system.
  shutdownlog : Display the emergency shutdown log.
  showmgr     : Displays the current fabric manager.
  swinfo      : Display switch information.
  summary     : Display this screen.
  time        : Display current system time.
  topos       : Maintain Custom Topologies installed.
  quit, q     : Exit the shell.
```

```
fx>
```

Swinfo

You may run the **`swinfo`**command at any time once you log into the FabreX switch.

Switch Login

```
fx>swinfo

GIGAIO FabreX (TM) 24 Port PCIe Switch
fabrex2 login: admin
password: password1

WELCOME to FABREX (TM)
Last login: Thu Mar 10 06:13:57 2023 from 192.168.xx.xx
FabreX Shell version: 2.x.x
================================================
```

Anytime you invoke the `swinfo` command, you'll see a display about the FabreX Switch divided into the following sections:

- **Switch Details**
- **Port Status** information
- **LED Status** information
- **Cable IDs** connected to switch ports
- **GigaIO Services** current status

> For details on port status or LED status fields, go to the section, "**Displaying Port Status and Displaying LED Status**1822589314 respectively.

The following examples show each portion of the **`swinfo`** display portions in the order presented:

Switch Login and Details

```
           Switch Details
 - Hostname.............: <switch_name>
 - Manager..............: localhost
 - IP...................: <switch IP address>
 - GUID.................: <globally unique identifier>
 - FabreX Version.......: 2.x.x
 - Platform.............: RS4024
 - FPGA.................: 799.3002
 - Firmware.............: 3.90 B062
 - Uptime...............: 4days,
 - CFG File.............: <configuration file>
============================================================
```

Port Status Details

```
                        Port Status
------PORT:
 -                     1          5          9         13         17
 - Binding info          usp 0.0    usp 1.0    dsp 1.1    dsp 1.2    d
 - LTSSM State           L0         DETECT     DETECT     DETECT     L
 - Config Width          x16        x16        x16        x16        x
 - Negotiate Width       x16        x0         x0         x0         x
 - Initial Speed         16G        16G        16G        16G        1
 - Config Speed          16G        16G        16G        16G        1
 - Force Link Speed      Inv        Inv        Inv        Inv        I
 - Negotiate Speed       16G        2.5G       2.5G       2.5G       8
 - BW Timestamp          17:6:27    17:6:27    17:6:27    17:6:27    1
 - BW Interval (uS)      5000256    5000256    5000256    5000256    5
 - Computed BandWidth    000.000 B  000.000 B  000.000 B  000.000 B  0
 - Total BandWidth       000.000 B  000.000 B  000.000 B  000.000 B  0

============================================================
```

LED Status Details

```
                    LED Status
 - Portn[01] U P Up 00  00  8/ 8 gen4 N0 B blue       blue
 - Portn[02]   P    ff                N1 B blue       blue
 - Portn[03] U P    01  00  0/ 8 gen1 N0   red      B blue
 - Portn[04]   P    ff                N0   red      B blue
 - Portn[05] U      02  00  0/ 8 gen1 N0   black    B blue
 - Portn[06]        ff                N0   black    B blue
 - Portn[07] U      03  00  0/ 8 gen1 N0   black    B blue
 - Portn[08]        ff                N0   black    B blue
 - Portn[09] D P Up 00 R15  4/16 gen3 N0   red      B yellow
 - Portn[10]   P    ff                N0   red      B yellow
 - Portn[11]   P    ff                N0   red      B yellow
 - Portn[12]   P    ff                N3 B green      yellow
 - Portn[13] D P Up 01  00 16/16 gen3 N0 B green      yellow
 - Portn[14]   P    ff                N1 B green      yellow
 - Portn[15]   P    ff                N2 B green      yellow
 - Portn[16]   P    ff                N3 B green      yellow
 - Portn[17] U P Up 04  00 16/16 gen4 N0 B blue       green
 - Portn[18]   P    ff                N1 B blue       green
 - Portn[19]   P    ff                N2 B blue       green
 - Portn[20]   P    ff                N3 B blue       green
 - Portn[21] U P Up 05  00 16/16 gen4 N0 B blue       green
 - Portn[22]   P    ff                N1 B blue       green
 - Portn[23]   P    ff                N2 B blue       green
 - Portn[24]   P    ff                N3 B blue       green
============================================================
```

Cable IDs (serial numbers)

```
                      Cable IDs
 - Port[01]   No Cable
 - Port[02]   No Cable
 - Port[03]   No Cable
 - Port[04]   No Cable
 - Port[05]   'APF193900172VF'
 - Port[06]   'APF193900172TY'
 - Port[07]   'APF193900172UN'
```

```
 - Port[08]   'APF193900172UV'
 - Port[09]   'APF193900172NF'
 - Port[10]   'APF1929001467B'
 - Port[11]   'APF193900172PT'
 - Port[12]   'APF193900172TT'
 - Port[13]   'APF193900172R0'
 - Port[14]   'APF193900172U1'
 - Port[15]   'APF193900172PM'
 - Port[16]   'APF193900172VY'
 - Port[17]   'APF193900172VK'
 - Port[18]   'APF193900172V2'
 - Port[19]   'APF193900172NH'
 - Port[20]   'APF193900172MM'
 - Port[21]   'APF193900172W2'
 - Port[22]   'APF193900172UY'
 - Port[23]   'APF193900172U7'
 - Port[24]   'APF193900172VC'
============================================================
```

GigaIO Services

```
                        GigaIO Services
 - collector.service             RUNNING
 - cmi.service                   RUNNING
 - emabridge.service             RUNNING
 - fbrxagentd.service            RUNNING
 - fxapi.service                 RUNNING
 - fxema.service                 RUNNING
 - fxmanager.service             RUNNING
 - fxworker.service              RUNNING
 - hotplugmonitor.service        RUNNING
 - ledctrl.service               RUNNING
 - lexfwdr.service               RUNNING
 - lexsw.service                 RUNNING
 - nginx.service                 RUNNING
 - redfish.service               RUNNING
 - redfish-ssdp.service          RUNNING
```

```
 - ser2net.service                RUNNING
============================================================
```

Static

The static command allows the user to set a static IP for the switch. To see the command arguments available, run **`help static`**. The following example uses the `static` command with the help option.

```
fx>help static
usage: static  -i IPADDR -d DNS [-m SNMASK] [-g GATEWAY] [-n NETWR

    Arguments:
      -i  IPADDR            IP address in IPv4 form: 192.168.1.0
      -d  DNS               DNS ip address in IPv4 form. Quote and
                            multiple addresses with spaces, eg. "1
      -g  GATEWAY           Gateway in IPv4 form. Defaults to 192.
      -m  SNMASK            Subnet mask in IPv4 form. Defaults to

    Optional arguments:
      -b  BCAST             Broadcast address in inet4 form. Defau
                            three octets of network>.255
      -n  NETWRK            Network mask in IPv4 form. Defaults to


fx>static  -i IPADDR -d DNS [-m SNMASK] [-g GATEWAY] [-n NETWRK] [
```

Hostname

Hostname is used to set the switch hostname and domain (optional). For example, type, `hostname` <hostname> [-d <domain_name>].

```
fx>help hostname
hostname: Set the hostname for the switch.
          usage: hostname <hostname> [-d <domain_name>]
```

```
fx>hostname <switch_name> -d datacenter
```

Htable

Htable allows a user to clear, list, or restore the hosts table from a previous configuration, as well as to add and delete entries. Up to 1024 entries may be added. Entries can be added one at a time or in a batch. One or more batch operations can be performed by separating individual entries with commas within the entry_string. When adding entries, the user must supply at minimum an IP address and a hostname. The hostname is viewed as a fully qualified domain name, as shown in the help file example provided. In addition, the user may supply ONE optional alias, which is usually the short form of the hostname.

The deletion (del) command allows a user to identify the target record by IP address, index, alias, or (host) name. Options must be used to select the type of target selector being used unless the selector is an IP address. In other words, 'del' defaults to an IP address argument. Multiple selectors can be placed on the command line, separated by spaces.

Each command issued to htable runs to completion. In the case of multiple arguments, all arguments must be valid in order for the operation to complete successfully. In the case of the "index" selector, it is assumed that the target index is derived from a previous "list" command.

Several indices may be included on the command line at once. However, if individual arguments are given for successive del commands, indices may change for each successive command.

Be sure to use a list command before each del invocation to ensure use of the proper target index.

ⓘ Htable should only be used if no DNS/DHCP server is present on the network.

```
fx>help htable
htable: Update the static host table for network name resolution.
          usage: htable <add> <entry_string>
```

```
                htable <del> [options] <entry_selector> [additic
                htable <clear|list|restore>

            Sub-commands:
              add     -- add one or more lookup entries to the host
              del     -- delete one or more lookup entries from the
              clear   -- remove all lookup entries from the hosts t
              list    -- list (with indices) all lookup entries in
              restore -- restore lookup table to its previous state

            Add Operation Arguments:
              entry_string -- "<host_record>[, [host_record][[, ...
                 host_record -- "<ip_address> <fqdn> [alias]"
                    ip_address -- example: 192.168.1.4
                    fqdn -- example: host1.yourdomain.com
                    alias -- short form of hostname. example: host1

              The alias component is optional. The entry_string may
              records separated by commas.

            Delete Operation Arguments:
              OPTIONS (Mutually Exclusive)
                 -x delete by index
                 -a delete by alias
                 -n delete by name
                 Defaults to deletion by ip (i.e. 102.34.2.1)
              entry_selector -- Terms listed as arguments matching
                                The Selector must match the term wi
                                a successful removal.

fx>htable add "<ip_address> <host-name>.<your-domain>.com <host-na

Examples:
htable add "192.168.99.200 GigaIO-Switch1.gigaio.com  GigaIO-Switc
htable add "192.168.99.142 host1.gigaio.com  host1"
```

Dynahost

Dynamic Host Mode (DHM) or dynahost is a conditional flag that determines whether or not host-tags are included when fetching topology templates via Redfish. If the dynahost flag is true (ON or enabled), host-tags are excluded from the returned template and, if the dynahost flag is false (OFF or disabled), host-tags are included. Note that host-tags, whether included or excluded while fetching topology templates via Redfish, are no longer (since FabreX 2.3) part of the host_list.json content. Any host may join the fabric if connected to an appropriate USP host port. These port connections are in every topology map (tmap.json) and diagram.

The dynahost flag's default condition is OFF (false). Users may wish to keep the Dynahost flag OFF so host-tags will be returned in a template for host-port identification, or choose to identify host-ports using the tmap or through other means. The Fabex Manager (FM) behaves *dynamically* with host addition or removal, so long as the host-tags were excluded from the initialization PATCH for the topology. The FM behaves *statically* to host addition and removal if host-tags were included during topology initialization.

The following table defines Dynamic Host behavior with respect to the Fabric Manager. For details on including or excluding one or more host-tags in the initialization PATCH command, see the *GigaIO™ Redfish API User's Manual*.

Dynamic Host Behavior

DHM	host-tags	FM behavior
Dynamic	excluded	Accepts any valid hostname from hosts attempting to join the fabric
Static	included	Accepts only the valid hostname provided via host-tag during the topology initialization PATCH command.

The following example shows the dynahost sub-commands and command usage.

```
fx> help dynahost
dynahost: Check status or enable/disable Dynamic Host Mode.

          usage: dynahost [ON|OFF|enable|disable]
```

```
            Sub-commands:
              ON or enable   -- enable Dynamic Host Mode
              OFF or disable -- disable Dynamic Host Mode

fx> dynahost
Dynamic Host Mode:  OFF
fx> dynahost enable
Dynamic Host Mode:  ON

fx>
```

Esverify

Esverify is a toggle command used to set or clear (enable or disable) SSL verificiation. Using esverify without an option shows the current condition. The following example shows the command options and usage.

```
fx>help esverify
esverify: Set/Clear event sender ssl certificate verification.

    usage: esverify [ON|OFF|1|0]

             Current state is reported if no argument is provided.
                Arguments:
                 ON, 1 : Enable SSL verification.
                 OFF, 0: Disable SSL verification.

fx> esverify
Verification DISABLED

fx> esverify 1
Verification ON

fx> esverify OFF
```

```
Verification OFF
```

Ping

Issue a ping to debug or check conditions related to the fabric network. The following example shows the many command options available with ping.

```
fx>help ping
Usage: ping [-aAbDrR] [-c count] [-i interval] [-I interface] [-m
            [-p pattern] [-s packetsize] [-w deadline] [-W timeout

    Use "help ping_options" for option details."

fx>help ping_options
OPTIONS
       -a     Audible ping.

       -A     Adaptive ping. Interpacket interval adapts to round-
              present in the network. Minimal interval is 200msec

       -b     Allow pinging a broadcast address.

       -c count
              Stop after sending count ECHO_REQUEST packets. With

       -D     Print timestamp (unix time + microseconds as in gett

       -i interval
              Wait interval seconds between sending each packet.
              Only super-user may set interval to values less 0.2

       -I interface
              interface is either an address, or an interface name
              interface is an interface name, it sets source inter
              specification (by the '%'-notation in destination, c
```

```
       -m mark
              use mark to tag the packets going out. This is usefu
              bound processing.

       -p pattern
              You  may  specify  up  to 16 ``pad'' bytes to fill o
              example, -p ff will cause the sent packet to be fill

       -r     Bypass the normal routing tables and send directly t
              is returned.  This option can be used to ping a loca

       -R     ping  only.  Record route.  Includes the RECORD_ROUT
              the IP header is only large enough for nine such rou

       -s packetsize
              Specifies the number of data bytes to be sent.  The
              header data.

       -w deadline
              Specify a timeout, in seconds, before ping exits reg
              count packet are sent, it waits either for deadline

       -W timeout
              Time to wait for a response, in seconds. The option
```

Readcmi

Readcmi is a command used to read the cable management interface (CMI) on the server host adapter card or on the FabreX Switch. Using readcmi,with or without the id option, will list the FabreX Switch cable IDs. The same cable ID information appears when running the swinfo command. The following examples show the readcmi command, its options and usage.

```
fx> help readcmi
```

```
readcmi: Read cable cmi data.

           usage: readcmi [id|card|port <n>|all|debug] [-h|cmd-opt

           Sub-commands:
              id -- read cable ids from all ports. (default)
              card -- read data from ports through which interface
              port <n> -- read cmi data from port <n>, where <n> i
              all -- read cmi data from all ports.
              debug -- toggle logging debug info to log; defaults


fx> readcmi id
=============== FabreX PORT ==================
0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 2 2 2 2 2
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
=============================================
C C C C . . . . . . . . . . . . C C C C . . . .
= PRT = Serial Number ==========
   1:    'APF190930104K2'
   2:    'APF190930104JU'
   3:    'APF190930104FP'
   4:    'APF190930104NF'
   5:    None
   6:    None
   7:    None
   8:    None
   9:    None
  10:    None
  11:    None
  12:    None
  13:    None
  14:    None
  15:    None
  16:    None
  17:    'APF18490028YRL'
  18:    'APF18490028YP6'
  19:    'APF18490028YRF'
```

```
  20:    'APF18490028YP7'
  21:    None
  22:    None
  23:    None
  24:    None


fx> readcmi card
s_prt c_prt    card_guid       server_uuid    mode   vers
 01    00   8ad85e0061280061 0000309c230108f6 H16  00.2a.00
 02    01   8ad85e0061280061 0000309c230108f6 H16  00.2a.00
 03    02   8ad85e0061280061 0000309c230108f6 H16  00.2a.00
 04    03   8ad85e0061280061 0000309c230108f6 H16  00.2a.00

fx> readcmi port 24
pres = 0
NO CABLE

fx> readcmi debug
cmi debugging ON

fx> readcmi debug
cmi debugging OFF
```

Showmgr

The manager switch is the name used in the fmtool command to identify which switch will run the fabric management services.

Each switch is configured to be the manager switch by default. If you have a multiple switch configuration:

1. Each switch must have a unique name,
2. One switch needs to be identified as the manager switch.
3. All other switches are considered worker switches. They must have the manager switch name set as the manager switch

You can show the current configured manager switch using the following command:

```
fx>help showmgr
showmgr: Displays the current fabric manager.
         usage: Current Manager   <switch name>
```

The following example of a multi-switch output shows Switch1 as the manager switch.

```
from Switch1
fx>showmgr
Current Manager:  Switch1

from Switch2
fx>showmgr
Current Manager:  Switch1
```

⚠ If your network does not have DNS, use the manager switch's IP address instead of the switch name. If you are using an IP address, ensure that it is a valid IP address.

Setmgr

If you have only one switch in the topology, it will be the manager switch by default. If you have multiple switches, you will need to decide which is the manager switch and which are worker switches You will then need to set the manager switch name for all the worker switches using the following command.

```
fx>help setmgr <switch hostname>
Set the manager for the fabric in which this switch participates.

        usage: setmgr [-i] <hostname>

        Option:
            -i -- ignore-fabric-state. Bypass the fabric state che
                  start initialization.
```

```
                    WARNING: The fabric state check is performed for
                    An unspecified, unsupported, and/or invalid fabr
                    requiring system shutdown, remediation, and rein
                    the fabric. Changing the manager name on this sw
                    participating in an active fabric may cause unde
                    behavior. However, the ignore option might be ap
                    current manager name is stale data, and this swi
                    in an active fabric controlled by the current ma
```

The following example of a multi-switch output shows Switch1 being set as the manager switch.

```
from Switch1

fx>showmgr
Current Manager:  Switch1

to Switch2

fx>setmgr Switch 2
Current Manager:  Switch2
```

ⓘ Now that you have finished setting up your switch, note the switch <hostname>. You will need the hostname to initialize the fabric with fmtool.

Topos

The topos command lists, deletes, or purges any custom topologies installed on the system. If no custom topologies are installed the user receives a message indicating this. The following example shows the topos command options available.

```
topos: List and remove Custom Topologies installed on this switch.
```

```
            usage: topos [list|delete|purge|force]

            Sub-commands:
              list -- list installed custom topologies by index.
              delete <ix> -- delete the topology at index <ix>.
              purge -- delete all custom topologies from this switc
              force [ON|OFF] -- Allow/prevent installation of next

fx>topos list
Custom Topologies
=================
0: 2S-4x4-1

fx>topos delete 0

fx>topos list
No custom topologies are installed.

fx>
```

Uplgio

The `uplgio` command enables users to install a new .gio software package from an installed USB flash drive to switch storage. The command prompts the USB flash drive for an index. The index retrieves the file list for the user. If the user enters a valid reference number, the selected operation begins using the file specified by the index. Perform the following steps to run `uplgio` and upload a .gio package:

1. Insert a flash drive in a USB port.
2. Run `fx>uplgio` from a console window.
3. Enter the number associated with the file to be uploaded.
   The associated file uploads from the USB to switch storage, as the following example shows.

```
fx>help uplgio
List .gio files found on usb devices, and prompt for the index of

           usage uplgio

fx> uplgio

PACKAGES:
  0: FabreX_v1.1.0_Ubuntu
  1: FabreX_v1.2.0_Ubuntu
  2: FabreX_v2.0.0_Ubuntu
  Select File #[0-2]:

fx> 2
```

Inspkg

The inspkg command enables users to install a .gio software package residing on the FabreX Switch. If you plan on updating the switch running an existing fabric, perform the following steps to run `inspkg` and install a .gio package:

1. Upgrade all the host servers' software package by following the 1822589314.
2. Power off the host servers.
3. Power off all IO resources boxes attached to the switch.
4. Run `fx>inspkg` from a console window.
5. Enter the number associated with the file to be uploaded.
   The associated file is installed on the FabreX Switch, as the following example shows.

   ```
   fx>help inspkg
   List available packages on the switch, and prompt for the ind

   usage inspkg [-f] [-i]

              Option:
   ```

```
            -f -- force. Ignore stale locks or ongoing instal
                  and attempt to install anyway. This option
                  in general, but may be necessary to recover
                  installation process cannot proceed because

            -i -- ignore-fabric-state. Bypass the fabric stat
                  start initialization.

                  "WARNING: The fabric state check is perform
                  An unspecified, unsupported, and/or invalid
                  requiring system shutdown, remediation, and
                  the fabric. Normal package installation sho
                  the fabric is NOT CONFIGURED."

fx> inspkg
PACKAGES:
  0: FabreX_v1.1.0_Ubuntu
  1: FabreX_v1.2.0_Ubuntu
  2: FabreX_v2.0.0_Ubuntu
  Select File #[0-2]:

fx> 2
```

6. Once the status tells you the install procedure is complete, use the following steps for fabric bring-up after updating the host servers and switch.
    a. Upgrade the FabreX Manager package on your admin server.
    b. Power on the IO resource boxes.

        > This step applies only to topologies with I/O resources.

        Validate that the DSP (connected to the I/O resource boxes) LTSSM are at L0.
    c. Reboot the switch.
    d. After the switch is ready, configure the fabric using the same host_list.json file, but you will need to use the new ltr.yml file.

        > The switch firmware may have changed, so it's good practice to use

the latest ltr.yml file.

e. Power up the host servers. Validate that the USP (connected to the host servers) LTSSM are at L0.
f. Ensure that the host servers are part of the fabric and are linked to their peers.

Rmpkg

The rmpkg command can remove (delete) a .gio package residing on the FabreX Switch. Perform the following steps to run `rmpkg` and remove a .gio package:

1. Run `fx>rmpkg` from a console window.
2. Enter the number associated with the file to be uploaded.
   The associated file is deleted from the FabreX Switch, as the following example shows.

```
fx>help rmpkg
List available packages on the switch, and prompt for the index of
This removes the package file from switch storage. It does not rem

          usage rmpkg

fx>rmpkg
PACKAGES:
  0: FabreX_v2.4.0
  1: FabreX_v2.5.0
  2: FabreX_v2.6.0
  Select File #[0-2]:

fx> 0
```

Checkpoints

Checkpointing is a tracking mechanism to maintain the FabreX Manager state through

reboots of the FabreX switch. By default, checkpoints should always be 'enabled.' Disabling a checkpoint should only be used for certain FabreX swtich and FabreX Manager error recovery operations.The checkpoints command allows users to check the status of a checkpoint, as well as enable, disable, or clear FabreX Manager checkpoints.

```
fx>help checkpoints
checkpoints: Check status or enable/disable/clear FM checkpointing

 usage: checkpoints [status|enable|disable|clear]

           Sub-commands:
             status  -- check current FM checkpointing status (def
             enable  -- enable/resume FM checkpointing
             disable -- disable/stop FM checkpointing
             clear   -- deletes FM checkpoint file. IMPORTANT: For
                        have the desired effect, the fxmanager ser
                        must be stopped first, and then restarted
                        doing a "checkpoints clear". In other word
                        sequence would be:
                            fx> service stop fxmanager
                            fx> checkpoints clear
                            fx> service start fxmanager

fx> checkpoints status
FM checkpoints ENABLED
```

History

The fabshell supports navigating through the shell command history list using the ↑ (up) ↓ (down) arrow keys. Entering <ctr-r> allows users to do a reverse search through the command history list. You can delete the command history list using the delhst command:

```
fx>help delhst
```

```
delhst: Delete history

fx>delhst
```

Autoflash, Flashfw, and Fbank

The switch supports auto recovery mode. The switch in this mode uses a default firmware if the switch finds a firmware failure during POST. You can enable this recovery to occur automatically by setting `autoflash` to ON. Note that the default setting is OFF. If `autoflash` is enabled on the switch, the current configuration and topology will be overwritten.

```
fx>autoflash
AUTOFLASH mode OFF
fx>autoflash ON
```

If you had `autoflash` set to OFF, and the switch detects a firmware failure, you can force the switch to use the default firmware by using the `flashfw` command. This command will overwrite the current configuration and topology.

```
fx>flashfw
```

The switch has a second flash bank that can be used if the first flash bank becomes corrupted or has failed. To switch to this second flash bank, use the `fbank` command.

```
fbank: Display the current flash bank name.

           usage: fbank [[-t] or [-T]]

            Option:
              -t: Toggle the current flash bank to the alternate f
              -T: FORCE the toggle operation without confirmation.

fx>fbank -t
```

Reboot the FabreX Switch

> ⓘ If rebooting the switch, servers should be powered down; otherwise, they may experience a kernel panic due to PCIe sub-tree disappearing while the switch is rebooting.

You can reboot the FXS running on the switch without affecting the PCIe switch chip by using the `reboot` command. This differs from `swreset` command which resets the whole switch.

```
fx>reboot
```

Displaying Port Status

Running the `swinfo` command in fabshell displays the current Port status. The following example only shows this portion of the swinfo display.

```
fx> swinfo

================================================================
                         Port Status
------PORT:
 -                     1          5          9         13         17
 - Binding info          usp 0.0    usp 1.0    dsp 1.1    dsp 1.2    d
 - LTSSM State           L0         DETECT     DETECT     DETECT     L
 - Config Width          x16        x16        x16        x16        x
 - Negotiate Width       x16        x0         x0         x0         x
 - Initial Speed         16G        16G        16G        16G        1
 - Config Speed          16G        16G        16G        16G        1
 - Force Link Speed      Inv        Inv        Inv        Inv        I
 - Negotiate Speed       16G        2.5G       2.5G       2.5G       8
 - BW Timestamp          17:58:14   17:58:14   17:58:14   17:58:14   1
 - BW Interval (uS)      5000258    5000258    5000258    5000258    5
```

```
  - Computed BandWidth    000.000 B 000.000 B 000.000 B 000.000 B 0
  - Total BandWidth       000.000 B 000.000 B 000.000 B 000.000 B 0

==========================================================
```

The following table defines the Port Status conditions shown in the example.

Fields	Description
Port number	The physical port number (0, 8, 16, 24, 32, and 40) that are used internally by the switch.
Binding info	The PCIe physical link status of the port: UP or DOWN • USP=upstream port • DSP=downstream port. After the port link status, the partition is listed. Each USP has its own partition, such as 0.0 or 1.0. DSPs are bound to a partition by the number after the period, such as 0.1 or 1.3. A partition must have 1 upstream and 0 or more downstream partitions.
LTSSM State	PCIe Link Training and Status State Machine Each state consists of sub-states that, taken together, comprise the state. The L0 state is the normal working state where data is transferred on the link. The second state entered during link training is the POLLING state. During the POLLING state, the bit symbol lock and lane polarity are established.
Config Width	The maximum bifurcation allocated for this port entry. This field is the configured PCIe lane width and x16 indicates a 16-lane connection.
Negotiate Width	Negotiated PCIe lane width; indicates the actual lane width currently in use. This may be less than or equal to Config Width. So while the configuration width may be x16, the negotiated width might be x4 or x8.
Config Speed	The PCIe rate for the port: Gen 1=2.5 GT/s, Gen 2=5.0

	GT/s, Gen 3=8.0 GT/s, Gen 4=16.0 GT/s. For example, 8G indicates 8.0 GT/s. The configuration speed is always present, whether or not there is bad or improperly connected cable.
Negotiate Speed	The negotiated link speed. If there is no link, there will be no negotiated speed. The negotiated link speed may be less than or equal to Config Speed. So, while the configuration speed may be 8G, the negotiated speed might be 2.5G or 5G.
Ing <P/C/N> TLP BandWidth	Ingress <P=Posted, N=Non-Posted, C=Completion> Transaction Layer Packet (TLP) bandwidth is calculated by adding up the bytes (Bs or kilobytes (KB)) entering the switch.
Egs <P/C/N> TLP BandWidth	Egress <P=Posted, N=Non-Posted, C=Completion> TLP bandwidth is calculated by adding up the bytes (Bs or KB) leaving the switch.

Port Status Display Fields

Displaying LED Status

Running the `swinfo` command in fabshell displays the current LED status. The following example gives below only shows this portion of the `swinfo` display.

```
fx> swinfo

============================================================
                      LED Status
 - Portn[00] U P Up 00  00  8/ 8 gen4 N0 B blue       blue
 - Portn[01]   P    ff                N1 B blue       blue
 - Portn[02] U P    01  00  0/ 8 gen1 N0   red      B blue
 - Portn[03]   P    ff                N0   red      B blue
 - Portn[04] U      02  00  0/ 8 gen1 N0   black    B blue
 - Portn[05]        ff                N0   black    B blue
 - Portn[06] U      03  00  0/ 8 gen1 N0   black    B blue
```

```
 - Portn[07]         ff                N0   black    B blue
 - Portn[08] D P Up 00 R15  4/16 gen3 N0   red      B yellow
 - Portn[09]   P    ff                N0   red      B yellow
 - Portn[10]   P    ff                N0   red      B yellow
 - Portn[11]   P    ff                N3 B green      yellow
 - Portn[12] D P Up 01  00 16/16 gen3 N0 B green      yellow
 - Portn[13]   P    ff                N1 B green      yellow
 - Portn[14]   P    ff                N2 B green      yellow
 - Portn[15]   P    ff                N3 B green      yellow
 - Portn[16] U P Up 04  00 16/16 gen4 N0 B blue       green
 - Portn[17]   P    ff                N1 B blue       green
 - Portn[18]   P    ff                N2 B blue       green
 - Portn[19]   P    ff                N3 B blue       green
 - Portn[20] U P Up 05  00 16/16 gen4 N0 B blue       green
 - Portn[21]   P    ff                N1 B blue       green
 - Portn[22]   P    ff                N2 B blue       green
 - Portn[23]   P    ff                N3 B blue       green
=============================================================
```

The following table defines the LED status conditions shown in the example.

LED Status Display Fields

Notation	Description
Port[n]	Switch port number from 0 to 23
U or D	Upstream or Downstream port. On a x8 or x16 port, only the lowest port is given.
Blank or P	Cable presence indicated (may be useful in debugging). Output available in ledctrl.log.
Blank or Up	Link is up or not up. Shows 'Up' when the link is up and blank when the link is not up.
00..ff	The value indicates the partition number; ff indicates invalid.
R15, 00, ..	If indicates 'R' then the lanes are reversed. The number is the

	first active lane.
8/16	Actual width / configured width; for example, 16/16 PCIe Gen 3.
gen#	PCIe generation representing the data transfer rate; where # indicates Gen 4 (16.0 GT/s), Gen 3 (8.0 GT/s), Gen 2 (5.0 GT/s), or Gen 1 (2.5 GT/s).
N0..N3	Neighbor value. If 0, the blink/activity is taken from this connector. If 1, 2 or 3, the information is taken from the 1st, 2nd or 3rd LED to the left.
B	B indicates a left blink next to the left color and a right blink when next to the right color.
color	The color of the left LED color indicates the current link rate and status, while the color or the right LED displays the lane width (x4, x8, or x16), direction (downstream or upstream), and port configuration status. See the following table listing right and left status conditions for each LED color.

The FabreX Switch has 24 x4 PCI Express ports organized in six quad port blocks. Each port provides 32 Gbits/sec at half duplex and 64 Gbits/sec at full duplex. Depending on the switch software configuration each x4 port can be combined with its neighboring ports to form either a x8 or x16 port within the same quad port block. Above each FabreX Switch connector the port has a left and right tricolor (red/green/blue) LED. The left LED color indicates the current link rate and status while the right LED color indicates the lane width (x4, x8, or x16), direction (downstream or upstream), and port configuration status.

The following table summarizes the LED colors and their associated right/left status condition.

ⓘ In the following table, the LED color should be assumed solid (non-blinking), unless otherwise indicated.

Port LED Indicator Status

Color	Left LED	Right LED
Black (not lit)	• if "right" LED is lit • Port inactive, or • Cable not detected (debug: may be the cable is not fully seated) • if "right" LED is unlit • see right LED column	Port is not in the current switch configuration
Various	Fast blink at link rate color when link is active. PCIe Gen rate colors: Unlit (BLACK), Gen 1 (CYAN), Gen 2 (YELLOW), Gen 3 (GREEN), or Gen 4 (BLUE)	Slowly blinks at the downstream (DS) or upstream (US) link rate color when not at the configured width: DS colors: Unlit (BLACK), x4 (WHITE), x8 (CYAN), or x16 (YELLOW) US colors: Unlit (BLACK), x4 (MAGENTA), x8 (BLUE), x16 (GREEN)
White	NA	DS x4
Cyan	Gen 1 (2.5 GT/s)	DS x8
Yellow	Gen 2 (5 GT/s)	DS x16
Magenta	NA	US x4
Blue	Gen 4 (16 GT/s)	US x8
Green	Gen 3 (8 GT/s)	US x16
Red	• Cable present but link	If a link is up but has only 1 or 2

	down, or • Link trained to low width for each inactive connector	active lanes.

NA – Not Applicable

Gathering FabreX Switch Logs

The fabshell logs command gathers switch logs, then compresses and encrypts the log files. The logs command may take several minutes to execute, depending on the size of the log files. Once the log files are prepared, they may be moved to a FTP directory, where the encrypted log file(s) may be downloaded and stored.

```
fx> help logs
logs: Fetch switch logs into the public ftp area for download.

fx>logs
FETCHING ... patience required.
ARCHIVING
.........................................
.........................................
...
.........................................
.....................................
COMPLETED!
   Logs available via public ftp i.e. ftp:// <switch_name>/pub

fx>
```

To move the logs to the location where they can be downloaded and stored, do the following

1. Point a browser, like Chrome, to: ftp://<switch ip address>.
2. Enter the username, ftp and what else?

3. Navigate to the <switch_name>/pub directory to see a list of available files to download over FTP.
   The switchd log (if there is one) and the post.log should be available as raw text.
   The remaining logs should be present in a .gio file and may be very large in size.

Exiting the FabreX Shell

The FabreX Shell is terminated via the 'quit' command. If an application terminates as a result of an error, the fabshell interface may terminate as well. Termination of fabshell logs the 'admin' user out. The fabshell does not escape to a command line shell on the FabreX System. If a command is running, a user may be able to terminate it with a <ctrl-c>. The following example shows the quit command.

```
fx>quit
shell exit
Connection to manager_switch closed.
```

Password Recovery

If the FabreX switch administer password is lost or forgotten, it may be reset through the password recovery feature. This feature restores the 'admin' password's value to the default of “password1”. To use it, you must have physical access to the switch: attach a keyboard and monitor. Login to the switch as: 'pwreset'. The password for this login is: “pwrestore1”

You should see a login screen that looks similar to this:

```
*********** Password Reset ***********
This feature will restore the 'admin'
password to the factory default.
************************************
```

```
Reset Password? [Y/N]:
```

Choose 'Y' to reset the 'admin' password, or 'N' to abort the request.

Password Recovery can be disabled from the fabshell via the 'pwrs' command. This command will query or set the password restoration mode to 'ON' or 'OFF'.

> **Important!** If you have multiple switches in your fabric, the username and password must be kept *exactly* the same for all switches.

FabreX Switch Web User Interface

The FabreX Switch includes a Web Interface that provides brief status information about the current running state of the switch and allows a privileged user to update the current package and firmware.

Menu Items

The following sections cover the Web UI menu items listed below:

Web UI Menu Items

Item	Description
Home	Home Page
User	User-Profile, when logged in; otherwise, displays the Login Page
Update	Switch Update Page, when logged in; otherwise, displays the Login Page.

Home Page

To access the home page, navigate with a browser to the switch using https://, this is followed by either its IP address or its host and domain name — provided a name server will perform resolution within the context of your network.

Switch Detail Example

Web UI Switch Detail Status Descriptions

Column	Description
PAR	The number of the switch partition being described. Configurations divide PCIe switch activity into one or more partitions.
FBRXPORT	The FabreX front panel switch port number. • On x16 connections, four ports are used and are described as a range; for example, 1..4 describes a x16 connection using four ports, 1, 2, 3, and 4. • On x8 connections, two ports are used and are described as a range; for example, 1..2 describes a x8 connection, consisting of two ports, 1 and 2.
MAX	The maximum bifurcation allocated for this port entry. This is the configured PCIe lane width. A x16 indicates a 16-lane connection.
STAT	Port Status indication of UP or DOWN.
NEG	Negotiated PCIe lane width, which indicates the actual lane width currently in use. This may be less than or equal to MAX.
DIR	Port Direction. USP = Upstream Port, DSP = Downstream Port. A partition must have 1 Upstream Port and may have 0 or more Downstream Ports.
LID	Logical Bridge ID. This number indicates the order in which the switch allocates its P2P bridges. The USP always has an LID of 0. DSPs may occupy any index. This number relates to the lspci report of port number.
PHY	Physical Port number used internally by the switch.

BDF	Bus Device Function Number. If a port has been configured by its host, this data should be meaningful and match that of a host's lspci output.
LTSSM	PCIe Link Training and Status State Machine
RATE	PCIe Rate.
IN-OUT	Ingress and Egress Byte Counts for the individual port.

Login Page

The User or Update Menu will present the login page when a user has not logged in.

Web UI Login Page

A special updater account is provided with which the user may initially log into the switch administrator's account. An email address is used as the login name.

Email: `admin@gigaio.com`

Default password: **`password1`**

Once logged in, the user may change the email and password on the User Profile page.

> **Important!** If you have multiple switches in your fabric, the username and password must be kept *exactly* the same for all switches.

User Profile Page

Web UI User Profile Page

User Profile information is displayed along with a LOGOUT button.

The email address, which functions as the login name for the user, can be changed.

Also, the password for the user may be changed with this page as well.

> **Important!** If you lose your password, see the section, "**Password Recovery**,1822589314for instructions.

Update Page

When a new release is available, the user must obtain a .gio package file, by visiting Software Releases on the GigaIO Service Desk (login required). After downloading, the .gio file may be uploaded to the switch via the **Upload Configuration Package** function shown in the following screen.

> A switch upgrade from 2.6.0 => 2.7.0 or downgrade 2.7.0 => 2.6.0 requires a reboot as the final step.

Web UI Switch Update Page

A file chooser is provided to allow the selection of the .gio file from the user's system, after which the user should submit the upload request, using the **Submit** button.

Once a .gio file has been properly uploaded, the page displays an Installation Option above the **Execute** button, as shown in the following screen.

Web UI Switch Update Selected

> Note that in your web UI screen the version should reflect the version you are installing. A switch upgrade from 2.6.0 => 2.7.0 or downgrade 2.7.0 => 2.6.0 requires a reboot as a final step.

The Available Packages section provides a list of available package files on the switch. Package Operation allows for a package file to be "Installed" or "Removed" using these options. Removal of a package file simply deletes the package file. Removal does not alter the current running software on a switch. The Install operation actually updates the running software on the switch.

A user must select the desired operation and submit the request via the **Execute** button. **Reboot** the switch to complete the upgrade process.

Update the FabreX Switch using the Web UI

The following table lists the preliminary, Web UI, and fabric bring-up tasks required to update an existing fabric on the switch.

Basic Steps to Update the FabreX Switch via the Web UI

If you plan on updating the switch running an existing fabric, perform the following procedure first:

Step	Preliminary tasks
1	Upgrade all the host servers' software package. Go to the section, "**Installation Instructions**."
2	Power off the host servers.
3	Power off all the IO resources attached to the switch.

The following table lists the basic tasks involved in updating the FabreX Switch using the Web UI screens.

Step	Web UI tasks
1	Using a web browser, access the GigaIO FabreX Switch Web UI. For details, go to the section "**Home Page**235667567
2	Access the USER tab to login as the updater user with credentials as follows for email and password, respectively.

	u: admin@gigaio.com p: password1 Login is successful and the webpage UI updates. For details, go to the section "**Login Page**."
3	Browse to the UPDATE tab of the FabreX Switch Web UI. Select CHOOSE FILE button. Using the Web UI file chooser, select the .gio file from the local user's system. For details, go to the section, "**Update Page**."
4	Verify The "Upload Configuration Package" section is updated with the .gio file name. Select the SUBMIT button to upload the GIO package to the switch. For details, go to the section, "**Update Page**."
5	Verify Screen updates with upload successful message, and the FabreX upgrade package is available in the drop-down menu of "Available Packages" section. For details, go to the section, "**Update Page**."
6	Choose the install package from the drop-down menu of "Available Packages" section and select the EXECUTE button. **NOTE**: Installation of GIO package should take around 10 minutes. For details, go to the section, "**Update Page**."
7	Reboot the switch. ⓘ When power cycling a switch, allow the switch power supplies to fully power down (approximately 30 seconds).

Use the following procedure for fabric bring-up after updating the host servers and switch.

Step	Fabric bring-up tasks
1	Upgrade the FabreX Manager package on your admin server.
2	Power on switch. Wait for the switch to fully power on.
3	After the switch is ready, configure the fabric using the same host_list.json file. Be sure to use the new <ltr_file|topo_id>file. (The switch firmware may have changed; best practice is to always use the latest ltr.yml file.)
4	Power on any IO resources and wait until they are fully online before going to the next step. **Note**: This step applies only to topologies with IO resources. Validate that the DSP (connected to the IO resource boxes) LTSSM are at L0.
5	Power up the host servers. Validate that the USP (connected to the host servers) LTSSM are at L0.
6	Ensure that the host servers are part of the fabric and are linked to their peers.

FabreX Software Features and Configuration

Libfabric

Libfabric LPP Functionality Matrix

Name	fi_lpp - The LPP Fabric Provider
Overview	The LPP provider runs on FabreX PCIe networks. FabreX provides high performance RDMA capabilities forming the foundation of the LPP provider. Higher level primitives are implemented at the libfabric and LPP kernel module (KLPP) layers.

Supported features	The LPP provider supports a subset of the libfabric API. Key features include: • **Endpoint types** LPP supports the **`FI_EP_RDM`** endpoint type with resource management. • **Endpoint capabilities** LPP supports **`FI_MSG`**, **`FI_RMA`**, and **`FI_TAGGED`** message types. • **Additional capabilities** LPP supports the additional features: **`FI_DIRECTED_RECV`**, **`FI_MULTI_RECV,`** **`FI_INJECT`**, and **`FI_DELIVERY_COMPLETE`** • **Progress** LPP currently supports only **`FI_PROGRESS_MANUAL`**; therefore, user applications are required to poll for progress.
Limitations	The following features are not supported: • connection management, • event queue, • scalable endpoint, • passive endpoint, • shared receive context, and • atomics.
Runtime parameters	The LPP provider checks for the following environment variables: • **`FI_LPP_DISABLE_OSBYPASS`** A bool which disables direct userspace writes when set. This can be used as a debugging aid, however it will degrade performance. • **`FI_LPP_MAX_WR_OSBYPASS_SIZE`** Sets the maximum size for PIO when performing write RDMA transfers. Transfers at or below this size will use CPU copy, while transfers above it will use a DMA engine. • **`FI_LPP_MAX_RD_OSBYPASS_SIZE`** Sets the maximum size for PIO when performing read RDMA transfers. Transfers at or below this size will use CPU copy,

	while transfers above it will use a DMA engine. • **FI_LPP_CQ_OVERCOMMIT** A bool which allows operations to start that may overrun the CQ. Normally, when resource management is enabled, the LPP provider will attempt to throttle operations that might overrun the CQ. This parameter disables that behavior while leaving the remainder of the resource management features enabled. • **FI_LPP_DOMAIN_CLEANUP** A bool which controls whether closing a domain with active resources will cause those resources to be automatically closed. If true (the default), the LPP provider will automatically close the resources.

MPICH Tuning Guide

MPICH Tuning Guidelines

Action	Tuning Guidelines
Disable dynamic debugging	If you have dynamic debug statements enabled, it can be detrimental to statements are usually enabled by a module param in /etc/modprobe.d/r For example: options ntb_dm dyndbg=+p The modules klpp, ntb_dm, ntb_transport should have dynamic debuggi testing. To disable at runtime: `echo 'module ntb_transport -pmfl'> /sys/kernel/debug/d` `echo 'module ntb_dm -pmfl'> /sys/kernel/debug/dynamic_` `echo 'module klpp -pmfl'> /sys/kernel/debug/dynamic_de`
Run on correct NUMA node	If the host is configured in NUMA mode, applications should generally ru as the ntb resources. To determine the NUMA node of the PCI resources on a 2 NUMA node s

	`` root@host# lspci | grep "Bridge: PMC" 04:00.1 Bridge: PMC-Sierra Inc. Device 8536 ``` If the bus number (04 in the above example) is greater than 80, it is NUM NUMA node 0. To run an application on a specific NUMA node: ``` numactl -N0 your_program ``` See the MPI documentation for instructions on binding MPI applications
Tune parameters	While the defaults will often be optimal, there are tuning parameters ava performance for some workloads: • Libfabric parameters • KLPP module parameters and sysfs tunables • ntb_dm module parameters and tunables
	• **Libfabric parameters** Environment variables ⓘ Prior to FXS v1.1, the FI_LPP_MAX_OBYPASS_SIZE variable contro max OSBYPASS size. • **FI_LPP_MAX_WR_OSBYPASS_SIZE** Sets maximum size (default 256) for direct PIO write (FI_WRITE) ope Above this size, a DMA engine is used. The DMA engine is faster for l overhead that dominates latency in small size transfers. • **FI_LPP_MAX_RD_OSBYPASS_SIZE** Sets maximum size (default 128) for direct PIO read (FI_READ) opera Above this size, a DMA engine is used. The DMA engine is faster for l overhead which dominates latency in small size transfers. • **FI_LPP_CQ_OVERCOMMIT** (default 0) — Setting this to 1 allows opera

	overrun the CQ. Normally, such overruns are prevented by throttling This variable provides a way to disable that throttling.
	• **KLPP module parameters and sysfs tunables** The KLPP module implements kernel support for the LPP libfabric provic Module parameters **`use_dma_read`** — For use_dma_read- if "Y", FI_READ DMA operations v transactions. Otherwise, reads will be internally reversed into write trans generally perform better when this option is "Y". However, not all config the DMA engine and switch configuration may impact availability of this for a non-compliant configuration, large size FI_READ operations will fail
	• **sysfs tunables**: Sysfs tunables can be found in /sys/class/klpp/klpp0 **`generated_mr_max_bytes`** — It takes time to pin and IO map buffers fo KLPP will not immediately release memory after the transfer is complete cached in case it is needed again soon. However, if the amount of cache tunable, KLPP will begin purging the cache. This does not apply to buffe fi_mr_reg(), which are pinned and IO mapped at all times. **`generated_mr_ttl_ms`** — A companion to the above tunable. KLPP wil DMA transfer involving a cached IO buffer before purging that buffer fro **`rmr_region_per_node_regions`** — This defines how many MRs can b from a remote node. If the user attempts to perform FI_READ or FI_WRIT for a given node than this value, they will continue to function, however, userspace, OS bypass writes are not possible, and performance may the value to accommodate all the MRs would allow OS bypass FI_READ and occur. **`max_small_msg_size`** — This defines the maximum size for a "small" F messages are transferred in a single step, however, doing this requires b receive side. Large messages, in contrast, are transferred directly from t receive buffer but requires more overhead to set up the transfer.
	• **ntb_dm module parameters and tunables** The defaults are generally correct for optimal performance. However, ex parameters could improve performance under certain conditions. Additi

	seriously and adversely affect performance if they are inadvertently set **ntb_dm tunables** **`dm_max_msg_size`** — This defines the largest command size that may b "`max_small_msg_size`" is derived from this ntb_dm parameter, therefo adjust both when experimenting. **`dm_max_msg_count`** — This defines the size of the command queue. If t libfabric operations may fail with EAGAIN completions. Increasing the siz may reduce instances of EAGAIN failures. **`dma_use_intr`** — This should generally be "Y". Using the default interna dramatically increase latency. **`xxx_stryx`** — This setting should be "Y" for i7 systems, and "N" for any possible to run with "N" on an i7 system, however performance of large degraded.

NVMe Over Fabrics

There are two different server configurations that support NVMe Over Fabrics (NVMe-oF). The first is the target server configuration that has NVMe storage, which is directly connected.The second is the host server configuration that connects to the target server over the FabreX network. These two configuration types are covered in the following sections.

Target Server Configuration

The target server configuration uses the ConfigFS kernel interface and file operations in `/sys/kernel/config/nvmet/` tree. To facilitate the configuration process, the nvmetcli tool was developed. This open-source tool can build a configuration using JSON files.

1. Install the **nvmetcli** tool for CentOS/Rocky Red Hat Enterprise Linux 8 (RHEL) by entering the following command:

```
$ yum install nvmetcli
```

Important! NVMe-OF works only on topologies with NT endpoints. Perform steps 2 and 3 any time you add or subtract NVMe devices, or when you need to bind or unbind other IO resources. Adding or subtracting IO resources may change the PCIe Bus Function Device numbers listed for the NVMe and Bridge devices.

2. Disable the ACS between NVMe devices and the NT device by finding the BDFs of all devices of interest. The following example shows NVMe drives and their BDFs.

```
# lspci | egrep "[0-9] Bridge: PMC-Sierra|NVMe"
69:00.1 Bridge: PMC-Sierra Inc. Device 4000
6e:00.0 Non-Volatile memory controller: Samsung Electronics C
70:00.0 Non-Volatile memory controller: Samsung Electronics C
72:00.0 Non-Volatile memory controller: Samsung Electronics C
74:00.0 Non-Volatile memory controller: Samsung Electronics C
76:00.0 Non-Volatile memory controller: Samsung Electronics C
78:00.0 Non-Volatile memory controller: Samsung Electronics C
7a:00.0 Non-Volatile memory controller: Samsung Electronics C
7c:00.0 Non-Volatile memory controller: Samsung Electronics C
7e:00.0 Non-Volatile memory controller: Samsung Electronics C
80:00.0 Non-Volatile memory controller: Samsung Electronics C
82:00.0 Non-Volatile memory controller: Samsung Electronics C
84:00.0 Non-Volatile memory controller: Samsung Electronics C
```

Use the following tool, fabrex-p2p-devs.sh, which automatically determines the appropriate devices and displays the associated BDFs, as shown in the following example:

```
# fabrex-p2p-devs.sh
69:00.1 6e:00.0 70:00.0 72:00.0 74:00.0 76:00.0 78:00.0 7a:00
```

3. Use `fabrex-acs-ctl` with the option flag `-s` to disable ACS between all the

NVMe drives and the NT. Disabling the ACS allows the NVMe drives to communicate directly with the NT as the following example shows.

```
# fabrex-acs-ctl -s 69:00.1 6e:00.0 70:00.0 72:00.0 74:00.0 7
grubby -- args='pci=disable_acs_redir=68:10.0\;6d:12.0\;6a:01
```

4. Create a json file that has the NVMe storage configuration for the target server. To set up a configuration from a json file, run the following command:

```
$ sudo nvmetcli restore <json_file_name>
```

The following sample shows a json file with two subsystems on the target server batray. Use the target server's hostname in place of batray. Also, find the manufacturer and serial number in the /dev/disk/by-id/ directory for each drive. This is so that every time you reboot the target it assigns the same drive to the same nvme-node#.

> ⓘ In the sample note that the path defines the physically connected device you want to logically map. The NVM Express Specification v1.3d, paragraph 7.9, defines the NQN, subsystem NQN, nguid, and uuid formats. Be sure to use unique values when selecting a value. Take care not to use the same value in multiple places.

```
{
  "hosts": [
    {
      "nqn": "nqn.2014-08.org.nvmexpress:NVMf:uuid:0ffc73f6-7
    }
  ],
  "ports": [
    {
      "addr": {
        "adrfam": "",
        "traddr": "fabrex0.lpp0.batray",
        "treq": "not specified",
```

```
        "trtype": "ntb"
      },
      "portid": 1,
      "referrals": [],
      "subsystems": [
        "nqn.batray.gigaio.com:nvme:nvm-node0",
        "nqn.batray.gigaio.com:nvme:nvm-node1"
      ]
    }
  ],
  "subsystems": [
    {
      "allowed_hosts": [],
      "attr": {
        "allow_any_host": "1"
      },
      "namespaces": [
        {
          "device": {
            "nguid": "1f6901ed-a060-4ca6-a0f5-c7d3563f4a7e",
            "path": "/dev/disk/by-id/nvme-<manufacturer-and-s
            "uuid": "c54f74c0-7c16-49c1-b56c-e4ff8492a739"
          },
          "enable": 1,
          "nsid": 1
        }
      ],
      "nqn": "nqn.batray.gigaio.com:nvme:nvm-node0"
    },
    {
      "allowed_hosts": [],
      "attr": {
        "allow_any_host": "1"
      },
      "namespaces": [
        {
          "device": {
            "nguid": "1f6901ed-a062-4ca6-a0f5-c7d3563f4a7e",
```

```
            "path": "/dev/disk/by-id/nvme-<manufacturer-and-s
            "uuid": "c54f74c0-7c18-49c1-b56c-e4ff8492a739"
          },
          "enable": 1,
          "nsid": 1
        }
      ],
      "nqn": "nqn.batray.gigaio.com:nvme:nvm-node1"
    }
  ]
}
```

5. List the devices you just configured for NVMe-oF usage.

```
$ sudo nvmetcli ls
```

Configuring and Connecting Host Servers to Target Subsystems

Host servers connect to target subsystems by writing connection parameters to `/dev/nvme-fabrics`. The nvme-cli tool facilitates this process.

> The nvme-cli tool install is an optional host driver package and only used for NVMe over Fabrics (NVMe-oF).

1. Install the `gigaio-nvme-cli*.rpm` file. Be sure to change the directory to where the GigaIO host driver package was unzipped, and then run the following command:

```
$ sudo yum localinstall -y gigaio-nvme-cli-*.rpm
```

2. To connect to a target subsystem, run the following command:

```
$ sudo /opt/gigaio-nvme-cli/nvme connect --transport ntb \
            --host-traddr fabrex0.lpp0.<host-server> \
            --traddr fabrex0.lpp0.<target-server> \
```

```
                --nqn nqn.<target-server>.gigaio.com:nvme:nvm-no
                --hostnqn="nqn.<host-server>.org.nvmexpress:NVMf
                --queue-size=64 --nr-io-queues=8 --reconnect-del
```

Where:

host-traddr — the address of the host port initiating the connection, should be obtained from FM

hostnqn — the Host NVMe Qualified name of the initiator server that wants to connect to the target subsystem

traddr — the address of the target port to connect, should be obtained from FM

nqn — the name of the target subsystem to connect, can be obtained from target configuration JSON

queue-size, ***nr-io-queues*** — the number and size of IO queues

reconnect-delay, ***ctrl-loss-tmo*** — parameters used for connection failure handling; delay for reconnect attempt, wait time before giving up reconnect attempts.

3. Check that the NVMe drive is now attached by running `lsblk` to see whether it is present as storage device **nvme0n1** (assuming there are no NVMe devices in the host server before NVMe-oF was set up).

```
$ lsblk
NAME                  MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
nvme0n1               259:0    0 745.2G  0 disk
sda                     8:0    0 447.1G  0 disk
├─sda2                  8:2    0 446.1G  0 part
│ ├─centos_host-swap  253:1    0     4G  0 lvm  [SWAP]
│ ├─centos_host-home  253:2    0 392.1G  0 lvm  /home
│ └─centos_host-root  253:0    0    50G  0 lvm  /
└─sda1                  8:1    0     1G  0 part /boot
```

GPUDirect RDMA (GDR)

GPU Direct RDMA (GDR) enables a GPU application to be run over multiple servers with GPUs attached to them. The GPUs would be able to use NCCL to pass messages between them across the FabreX network.

Prerequisites

- NVIDIA GPUs are installed in an external GigaIO Acclerator Pooling Appliance, not in a PCI slot on the motherboard. Motherboard PCI slots are typically on a different PCI root port than the FabreX Network Adapter Card. Passing TLPs between root ports tends to provide poor performance and stability.
- NVIDIA drivers and CUDA are installed. See the section, Qualified Software Versions, for qualified and recommended NVIDIA drivers, CUDA tools, and NCCL versions.
- If planning to run NCCL, install it according to the instructions in the README. See https://github.com/NVIDIA/nccl.

  > ⓘ There is support for NCCL version 2.4.x.

Qualified GDR Software Matrix

The following table lists recommended GDR software and versions that GigaIO has qualified for each FabreX release.

> ⓘ While newer or older versions of the software listed should work, the table lists only the versions that have been verified.

Qualified GDR Software Matrix

Software	FabreX 2.7
NVIDIA driver	465.19.01
CUDA compilation tools	11.3, v11.3.109

NCCL Version	2.9.6

Install klpp-peer-dkms rpm

1. Install the gigaio-nv-klpp-peer-dkms rpm. Be sure to change the directory to where the GigaIO host driver package was unzipped, and then run the following command:

```
$ sudo yum localinstall -y gigaio-nv-klpp-peer-dkms*.rpm
```

2. Install FabreX NCCL Plugin using the following command. The NCCL plugin can be found with the other host server drivers in the centos/package.

```
$ sudo yum localinstall -y gigaio-fabrex-nccl-*.rpm
```

3. Verify that both the Nvidia and nv_klpp_peer drivers are installed using the following command:

```
$ dkms status
nvidia, 440.44, 5.3.0.release.47.6909b14f842e, x86_64: instal
nv_klpp_peer, 2.0.0.release.2.63b2b3a, 5.3.0.release.47.6909b
```

Load a GDR-capable switch config

The following example lists the topology options currently available (subject to change):

> ⓘ For available GDR-capable topologies, see the Topologies site on the GigaIO Service Desk. Note that GigaIO's Service Desk requires a valid support account to sign in. Designations are PCIe Gen 3 is .../sj1/ and PCIe Gen 4 is .../sj5/.

```
$ find /opt/gigaio-fabrexfm-tool/topologies/release/sj* -name *-G*
```

```
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/1S-2x4-1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/1S-2x4-X1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/1S-4x2-1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/1S-4x2-X1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/1S-6x0-1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/1S-8x2-X1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/2S-4x8-X1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/2S-8x4-1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/2S-8x4-X1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/2S-8x4-X2
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/GC2P.5S-12x16-X1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/GC3P.7S-18x24-X1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/GDK.1S-3x4-X1
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/GP.2S-6x8-X1
```

Disable ACS between NT Endpoint and GPU devices

1. Find the BDFs of all the devices of interest. For example, the first column of the following output shows the BDFs of the installed NVIDIA K80s:

   ```
   # lspci | egrep "[0-9] Bridge: PMC-Sierra|NVID"
   5b:00.1 Bridge: PMC-Sierra Inc. Device 4000
   64:00.0 3D controller: NVIDIA Corporation GK210GL [Tesla K80]
   65:00.0 3D controller: NVIDIA Corporation GK210GL [Tesla K80]
   68:00.0 3D controller: NVIDIA Corporation GK210GL [Tesla K80]
   69:00.0 3D controller: NVIDIA Corporation GK210GL [Tesla K80]
   ```

   Use the following tool, fabrex-p2p-devs.sh, which automatically determines the appropriate devices and displays the associated BDFs, as shown in the following example:

   ```
   # fabrex-p2p-devs.sh
   5b:00.1 64:00.0 65:00.0 68:00.0 69:00.0
   ```

2. Use fabrex-acs-ctl with the option flag –s to disable ACS between the NT Endpoint and all the GPUs. Disabling ACS allows GPUs to communicate directly

with each other (peer to peer (P2P)), and also between remote hosts and the local GPUs, as the following example shows.

```
# fabrex-acs-ctl -s 5b:00.1 64:00.0 65:00.0 68:00.0 69:00.0
grubby --args='pci=disable_acs_redir=5c:01.0\;63:10.0\;63:08.
```

Configure p2p_dma_ops

1. To verify the module has been properly initialized, look in dmesg for messages similar to the following. There should be one message for each GPU.

   ```
   # dmesg | grep p2p_dma [ 12.466707] p2p_dma_ops: set dma_ops
   ```

2. Set environmental variables.

   ```
   export FBX_NCCL_PTR_CUDA=1
   NCCL_NET_GDR_READ=0
   ```

   The first variable enables GDR in the FabreX NCCL plugin. The second variable disables eager messages in libfabric, which does not work with GDR.

Testing GDR with nccl-tests

1. Verify that the GDR is working using tests included with nccl-tests package.

   ⚠ All servers participating should already have Nvidia drivers, CUDA, and NCCL installed. Also, a keyless ssh needs to be set up.

2. Install nccl-tests according to the instructions in the README.
   See https://github.com/NVIDIA/nccl-tests.
3. Follow the instructions for running the various nccl-tests.

NTB (Non-Transparent Bridge) IP Network Interface

NTB Interface Functionality Matrix

Overview	TCP/IP communication services over FabreX are provided by the NTB IP Network Interface (`ntb_netdev`) kernel module. This module registers as a standard Linux netdev device; it plugs into the Linux TCP/IP networking framework and can be managed like any other standard network device.
Supported features	Unique properties of the `ntb_netdev` in FabreX are: 1. Only supports point-to-point connectivity with other hosts in the FabreX fabric. As such, each ntb_netdev network interface represents a specific peer and is required to have a unique IP address assigned. No multicast. 2. The network interface name is dependent on the hostname (utsname.nodename) of the respective peer to which the local host is connected, and has the format: ntb<NT-EP-instance><peer-nodename>. For example, when defining a fabric made up of three hosts: Fabric: HostA + HostB + HostCNetwork interface (DEVICE) names would be: HostA: ntb0HostB, ntb0HostC HostB: ntb0HostA, ntb0HostC HostC: ntb0HostA, ntb0HostB
Runtime parameters	The "`ntb0`" prefix represents the NT Endpoint instance through which the connection is made. In this example, each host only has one NT Endpoint connecting it to the FabreX fabric. Determining the MAC or Hardware Address for a ntb_netdev interface, the associated MAC address for a ntb_netdev interface can be retrieved via: `/sys/class/net/{IFDEVNAME}/address` where, {IFDEVNAME} is the ntb_netdev network interface

	name, such as the ntb0HostA, ntb0HostB, listed above. To set up a `ifcfg` network script for a ntb_netdev interface: **TYPE** — Select TYPE as "**Ethernet**". **UUID** — Generate via "**uuidgen {HOST**}". **IPADDR** — Should be consistent with your lab set-up.
Tunable properties	• **ntb_netdev tunables** The following kernel module parameters affect ntb_netdev behavior/performance.These parameters can be set via **`/etc/modprobe.d .conf`** file settings for the ntb_netdev module. Note that because the ntb_netdev module are loaded very early during the boot process, it is necessary to update the respective **`initramfs`** of the booted kernel once the **`.conf`** file has been updated. See **`dracut -f`**. The copy_bytes is the buffer size at which a DMA channel is used to transfer the data. Buffers below this size are transferred directly by a CPU via `memcpy` function**.** • IF (buffer_size < copy_bytes) THEN use CPU `memcpy`. • IF (buffer_size >= copy_bytes) THEN use DMA. (Default = 1024, units = bytes)**`transport_mtu Maximum`** — Transmission Unit (MTU) size for the respective network device. (Default = 65536, units = bytes)**`transport_msg_count`** Queue size — Represents the size of the ring buffers used for sending/receiving packets. Must be consistent between peers on the same point-to-point connection. (Default = 512, units = packet count)**`transport_skb_prealloc`** — Number of skbuffs to pre-allocate and populate in the RX side for incoming packets. A skbuff is a virtually contiguous block of data used in the Linux network stack.

	Pre-allocating skbuffs is a form of priming the pump for incoming data. (Default = 100, units = packet count)**`use_poll (1)`** — Indicates that communication should utilize polling for recognizing that packets have arrived and not to rely on interrupts. Note that polling will consume a lot of CPU time. (Default = 0, units = binary)**`use_rxdirect (1)`** — Indicates communication by the TX side should copy the transmission data directly into the target skbuff on the RX side. When not enabled (0), TX data uses a bounce buffer on the RX side before the data is copied into a target skbuff. (Default = 1, units = binary)

FabreX Switch Redfish API

DMTF's Redfish® is a standard API designed to deliver simple and secure management for converged, hybrid IT, and Software-Defined Data Center (SDDC).

As an open-industry standard specification and schema, Redfish specifies a RESTful interface and uses defined JSON payloads — usable by existing client applications and browser-based GUI.

GigaIO's Redfish API implementation can be used for the following:

- Selecting a topology to initialize and clearing a topology to tear down
- Viewing Non-Transparent (NT) PCIe Fabric status of host-to-host communication links
- Viewing Transparent PCIe Fabric composition status and PCIe endpoint device catalog
- Viewing Fabric-related properties, involving components such as endpoints and switches, port information on those switches, and more
- Viewing PCIeDevice and PCIeFunction information for discovered PCIe endpoints, such as GPUs, FPGAs, storage devices and so on
- Listing CMI Cable IDs that are connected to a port on the switch, configured and negotiated port width, and configured and negotiated PCIe protocol generation

- Viewing Composed Systems that belong to a particular fabric instance
- Viewing other miscellaneous information, such as metadata about the switch and IO Chassis information
- Reassigning DSPs (and corresponding IO) between USPs (hosts/partitions) *after* the fabric has been initialized
- Creating or viewing subscriptions to events and telemetry

For the complete information about the FabreX Fabric, including setup and teardown, resource examples, and a list of supported APIs, see the *GigaIO Redfish API User's Manual* link in the 1352007746 section.

Troubleshooting FabreX Software

Ubuntu Software Installation Errors

Ubuntu may display an error during the GigaIO Software Installation when the GDR package is installed. The GigaIO GDR package includes modules for NCCL and DKMS.

> The GigaIO GDR package should only be installed only after setting up the GPUs and relevant drivers on the host system. Otherwise, the installation may fail and report errors.

The following screen examples are displayed on a GUI Desktop and show an Ubuntu internal error and associated crash report when installing DKMS:

The next screen shows an Ubuntu error when the installation is performed from a terminal shell. The installation log captures these errors for the user.

```
"Building module:
cleaning build area...
make -j32 KERNELRELEASE=5.13.0-40-generic LEXMVP_DIR=/usr/src/kern
ERROR: Cannot create report: [Errno 17] File exists: '/var/crash/g
Error! Bad return status for module build on kernel: 5.13.0-40-gen
Consult /var/lib/dkms/nv_klpp_peer/2.7.0r1.release.31.cb9bdfd/buil
```

Web User Interface Error Messages

Web User Interface Error Messages

Error Message	Description and/or Resolution
The Web User Interface shows "an internal error has occurred with the FabreX Switch.	Perform a power cycle. Power off the Host(s), FabreX Switch, and IO resources. Power on the devices in following order: 1. FabreX Switch 2. IO resources ⓘ When power cycling the switch, allow 30 seconds for its power supplies to fully power down before applying power again. 3. Host(s)

FM Tool Error Messages

FM Tool Error Messages

Error Message
If you do not have your authorization information correct or in the default location, you

```
user@manager_switch:~$ cat ~/.fabrex_auth_bad
Zm1hZ2VudEBnaWdhaW8uY29tOkhhbmcxMFRheWxvck1hZA==
$ fmtool -a /home/user/.fabrex_auth_bad -s manager_switch
Requesting status from manager_switch...
```

```
Error 401, invalid authorization
```

If you do not have your authorization information correct you will also see the following

```
Apr 10 21:21:20 manager_switch switchd:  [INFO    ] Request: 127.
Apr 10 21:21:20 manager_switch switchd:  [ERROR   ] AUTH: authori
Apr 10 21:21:20 manager_switch switchd:  [ERROR   ] /status: inva
Apr 10 21:21:20 manager_switch switchd:  [INFO    ] Response: 0x7
```

Hosts not communicating with the GigaIO Switch

> For details about host tables (htables), see the FabreX Shell htable command option.

The GigaIO Switch and Hosts require Hostname or IP Address entries to communicate with each other. This is a requirement for the GigaIO Fabric.

If your Network environment is not set up for DHCP and DNS, you will need to verify your Hosts have hostname entries for all the servers and the GigaIO Switch. The following example shows an edited etc/hosts file from a Host Server:

The GigaIO Switch must also be able to communicate with the Hosts. Configuration is performed using the command, `htable`, available from the fabshell interface loaded on the FabreX switch.

The example following shows adding one GigaIO Switch and one Host, using htable:

htable add "192.168.99.200 GigaIO-Switch1.gigaio.com GigaIO-Switch1"
htable add "192.168.99.142 host1.gigaio.com host1"

This next example shows a listing of all entries that have been manually created in the htable:

Verify the GigaIO Switch and Hosts can all ping each other. Once confirmed, the GigaIO Switch and Hosts should be rebooted (or power cycled) for the change to take effect.

Other Software Issues

System Issues and Errors

issue	Solution
Host server doesn't join the fabric.	Ensure that the fabrexfm-host.service is active. Ensure that the server file /etc/fabrex.conf exists and contains the switch (<switch name or switch IP address>). Ensure that the fabric has been initialized with the <server name> in the Host List file used by the fmtool host_list.json LTR.yml <switch name> command.
Worker switch doesn't join the fabric.	Ensure that the showmgr fabshell command shows the correct manager switch name.
If you wish to prevent a host server from joining the fabric or remove it once it has joined.	Stop the fxworker on the host server using the following command: `systemctl stop fxworker.service`
If you wish the host server to rejoin the fabric.	Restart the fxworker on the host server using the following command: `systemctl restart fxworker.service`

GPU gets stuck during reboot or power-down process with soft lock GPU stuck - nouveau message.	Disable the nouveau driver. For details, see the section, "**Disabling the nouveau driver**."
Host server does not see IO devices assigned to it.	Power down the host server and power cycle the IO devices.
If you run, GET https://<switchname>/redfish/v1/CompositionService, you see the following response:	This response indicates that the `host_list.json` file may be missing Redfish-supported information, or the network has not been initialized. Use the latest fmtool package and/or initialize the network.
Topology will not initialize because the fabric is already configured: `fmtool host_list.json ltr.yml <switch_name>` `fmtool: 2021-08-20 at 14:55:13` `Configuring fabric...` `Sending host_list.json to <switch_name>4...` `Request:` `PUT https://<switch_name>/fabrex/v2/hostlist` `Response:` `HTTP/1.1 409 CONFLICT` `{` `"msg": " Host list reload not permitted, Fabric Configured!"` `}` `cannot process request` `Process terminated... exiting...`	Uninitialize the fabric with `fmtool –u` and try again.

... "ServiceEnabled": true, "Status": { "Health": "Warning", "HealthRollup": "Warning", "State": "StandbyOffline" } ... If you run GET https://<switchname>/redfish/v1/CompositionService to see the following: ... "ServiceEnabled": false, "Status": { "Health": "Critical", "HealthRollup": "Critial", "State": "Disabled" } ...	This response indicates that the network has been un-initialized. Initialize the network using `fmtool` or REST API.
If *fabrex-status.sh* shows the following: – `fxworker.service STOPPED` See 1822589314, for what *systemctl status fxworker.service* shows.	Check that the BIOS settings are correct. Set MMIO High Size to **1024GB**, MMIO Base to **56TB**, and the 64-bit PCIe addressing by setting above 4G Decoding to **Enabled.**
If *fabrex-status.sh* shows the following: – `fxworker.service STOPPED` See 1822589314, for what *systemctl status fxworker.service* shows.	Ensure that you have the file `/etc/fabrex.conf` present with the following contents: `Switch: myswitch`
Connection not coming up with correct lane width.	Check that all cables in the link connection are correctly plugged into the connectors. If you need to swap cables or replace a cable, follow the

	proper bring up and reboot procedure afterwards.
Redfish is unable to bind resources with a new admin password.	If you have multiple switches in your fabric, the username and password must be kept exactly the same for all switches.
Fabric Management issues HTTP 401 unauthorized responses	Check that you have a valid fabrex_auth file in the proper location. A switch reboot may be required. If a reboot still does not solve the issue, then re-initialize the entire fabric after the switch reboot.

Example 1: systemctl status fxworker.service

```
$ sudo systemctl status fxworker.service
● fxworker.service - FabreX Host Worker Service
   Loaded: loaded (/etc/systemd/system/fxworker.service; enabled;
   Active: active (running) since Mon 2021-08-23 11:26:00 PDT; 4 d
 Main PID: 1741 (fxwd)
   CGroup: /system.slice/fxworker.service
           ├─1741 /opt/gigaio-fabrexfm-host-worker/fxwd
           └─1742 /opt/gigaio-fabrexfm-host-worker/fxwd

Aug 24 17:17:37 seaturtle fxwd[1742]:   nfp_peer_ccs_msi_off: 0x0
Aug 24 17:17:37 seaturtle fxwd[1742]:   nfp_ccs_msi_data: 0x0
Aug 24 17:17:37 seaturtle fxwd[1742]:   nfp_xlate_addr: 0x0
Aug 24 17:17:37 seaturtle fxwd[1742]:   nfp_xlate_size: 0x0
Aug 24 17:17:37 seaturtle fxwd[1742]:   nfp_tgt_partition: 0x0
Aug 24 17:17:37 seaturtle fxwd[1742]: *** *** *** ***
Aug 24 17:17:37 seaturtle fxwd[1742]: ms:<32> INFO       [md: fm_h
Aug 24 17:17:37 seaturtle fxwd[1742]: ms:<32> INFO       [md: fm_h
```

```
Aug 24 17:17:37 seaturtle fxwd[1742]: clear_nic(): - CALLING ioctl
Aug 24 17:17:37 seaturtle fxwd[1742]: clear_nic(): status: 0, errn
Hint: Some lines were ellipsized, use -l to show in full.
```

Example 2: systemctl status fxworker.service

```
$ sudo systemctl status fxworker.service
● fxworker.service - FabreX Fabric Host Worker Service
   Loaded: loaded (/etc/systemd/system/fxworker.service; enabled;
   Active: activating (auto-restart) (Result: exit-code) since Thu
  Process: 36213 ExecStart=/opt/gigaio-fabrexfm-host-worker/fxwd (
 Main PID: 36213 (code=exited, status=11)

Apr 02 14:01:38 server1 systemd[1]: Unit fxworker.service entered
```

Deprecated - FabreX Manager v2.6 and Earlier Tool Features

FabreX Manager Tool Setup Guide

FabreX Manager Tool is an optional command line interface. Its usage is not required when Redfish or other orchestration software tools are available.

> When installing the FabreX Manager Tool it is necessary to use a system that is outside the fabric and run it from a CentOS 7.x (or later) or Ubuntu 16.04.x (or later) LTS-based management workstation.

The following packages are available for FabreX Manager Tool:

- Centos: gigaio-fabrexfm-tool-*.x86_64.rpm
- Ubuntu: gigaio-fabrexfm-tool_*_amd64.deb

These files are downloadable from the Service Desk. Visit Software Releases to download the latest version.

The topology directories will install in the `/opt/gigaio-fabrexfm-tool/topologies/release/<topology>` directory and contain the ltr.yml and host_list.json files needed to initialize the fabric.

CentOS Installation

1. Download the latest **CentOS 7.x** software release.
2. Install **gigaio-fabrexfm-tool-*.x86_64.rpm** package.

```
sudo yum localinstall -y gigaio-fabrexfm-tool-*.x86_64.rpm
```

Ubuntu Installation

1. Download the latest **Ubuntu 16.04.x** software release.
2. Install **gigaio-fabrexfm-tool_*_amd64.deb** package.

```
sudo apt install ./gigaio-fabrexfm-tool_*_amd64.deb
```

FabreX Manager Tool Files

Once the FabreX Manager Tool (fmtool) package has been installed on a management workstation, the fabric is ready to be configured. To create a configuration, perform the following steps:

It is recommended that updates to fmtool be kept in-sync with the FabreX software running on the switch.

1. Create an authorization file.
2. Select a FabreX Topology Record (LTR) from the list of supported topologies (for

details visit Topologies on the Service Desk). See the section, "**FabreX Topology Record**" for information about LTRs.

3. Edit the accompanying host_list.json file that identifies the site-specific component names in the topology.
4. Configure the topology using the LTR and host_list.json file.

The following sections describe the various files used by the FabreX Manager Tool.

Authorization File

You must provide HTTP Basic authorization credentials to fmtool. The credentials should be stored in a file, the default is $HOME/.fabrex_auth. To create the credentials file, run the following command:

```
$ echo -n admin@gigaio.com:password1 | base64 > ~/.fabrex_auth
$ chmod 600 ~/.fabrex_auth
$ cat ~/.fabrex_auth
YWRtaW5AZ2lnYWlvLmNvbTpwYXNzd29yZDE=
```

Important! If you have multiple switches in your fabric, keep the username and password *exactly* the same for all switches. If you have changed the admin user password (on the manager switch) via the fabshell interface, you should change the password on all other worker switches to match. You will also need change the password used to create ~/.fabrex_auth file to match.

If you would like to provide an alternate location for the authorization file, you can direct fmtool to use it by providing the arguments -a <path_to_auth_file> to fmtool. With the default authorization file in place, you should now be able to run the fmtool command.

```
$ fmtool -s <switch_name>

fmtool: 2020-06-21 at 14:16:13
Requesting status from <switch_name>...
```

```
Response:
{
    "status_code": 200,
    "body": {
        "fabric": {
            "Fabric State": "NOTCONFIGURED",
            "hosts": [],
            "switches": [],
            "Host List State": "NOTCONFIGURED"
        }
    }
}
Success: Status from <switch_name>
```

If you do not have your authorization information correct or in the default location, you will see the following error returned by fmtool:

```
$ cat ~/.fabrex_auth_bad
Zm1hZ2VudEBnaWdhaW8uY29tOkhhbmcxMFRheWxvck1hZA==
$ fmtool -a ~/.fabrex_auth_bad -s <switch_name>

fmtool: 2020-02-25 at 13:25:30
Requesting status from <switch_name>...
Response:
{
    "status_code": 401,
    "body": {
        "msg": "The server could not verify that you are authorize
    }
}
Error: Cannot process request
use --verbose for more information

$ fmtool -s <switch_name>

fmtool: 2020-02-25 at 13:22:32
Requesting status from <switch_name>...
```

```
Error: No such file or directory - /home/user/.fabrex_auth
```

FabreX Topology Records (LTR)

FabreX Topology Records (LTRs) describe a specific topology consisting of one or more FabreX switches and one or more servers. The list of supported LTR files is delivered in the FabreX FM Tool package and can be found on the management server. Each of these directories also has a host_list.json file that needs to be modified for components within the topology installed onto the fabric.

```
/opt/gigaio-fabrexfm-tool/topologies/release/sj5/1S-2x4-1/
-rw-rw-r--  1 root root  291 Aug 23 10:56 host_list.json
-rw-rw-r--  1 root root 2114 Aug 23 10:56 ltr.yml
-rw-rw-r--  1 root root 1513 Aug 23 10:56 README.md
```

Each LTR contains symbolic tags identifying components in the topology.

> Users should never edit or modify the `ltr.yml` file.

The host_list.json file

An example host_list.json file is found in the topology directory and is used to identify required, optional, and restricted components:

- The unique names of FabreX switch(es).
- The unique aliases of any IO devices connected to the "dsp" ports on the switch(es).
- The resource "zone" identifier name(s). A resource zone is a collection of IO devices connect to "dsp" ports (ResourceBlocks) which can be composed together. For a single hybrid switch configuration, the entire switch is one zone. For multiple switch configurations, you may have multiple zones, and each should have a unique name. Redfish uses these zones to keep track of which IO resources can be connected to which "host" (USP) ports.

- The unique host names of servers connected to the "host" ports of the switch(es).

The initialization PATCH command to `redfish/v1/Fabrics/Oem/GigaIO/Topologies/<id>` should follow these rules:

Rules for Entities

REQUIRED	switch-tags, zone-tags, dsp tags	Switch values must equal the DNS hostname of the switch
OPTIONAL	host-tags	Whether or not supplied by the Redfish template
RESTRICTED	Everything else	

> ⓘ All switches and server host names must resolve to an IP address.

If the connection is to be blank, you can use any unique name to identify that the connection will not be used. You can change this unique name at a later date when you add another server. All default tag-values MUST be replaced. For any device that is not connected, a dummy name is still required for proper Redfish functionality.

> ⓘ Host-tags are no longer (since FabreX 2.3) part of the host_list.json content. Any host may join the fabric if connected to an appropriate USP host port. These port connections are in every topology map (tmap.json) and diagram. The Dynamic Host Mode (DHM or dynahost) conditional flag available in the FabreX Shell can be set to include host-tags or exclude them when fetching topology templates via Redfish. The FabreX Manager (FM) behaves *dynamically* with host addition or removal, so long as the host-tags were excluded from the initialization PATCH for the topology. The FM behaves *statically* to host addition and removal if host-tags were included during topology initialization. For more information on 'dynahost' usage, see the "**FabreX Shell**" section.

The following example shows the content of the host_list.json file from the 1S-2x4-1 topology.

> ✓ Note the absence of a comma for the last entry in the example. You leave out the

comma In the last entry, which is the required syntax. Be sure to use the exact syntax shown when configuring your host JSON file. Only change the parameters on the right side of the colon between the quotes.

```
$ cat host_list.json
{
    "s1p9dsp": "<SWITCH1_PORT9_NAME>",
    "s1p13dsp": "<SWITCH1_PORT13_NAME>",
    "s1p17dsp": "<SWITCH1_PORT17_NAME>",
    "s1p21dsp": "<SWITCH1_PORT21_NAME>",
    "switch1": "<SWITCH1_NAME>",
    "zone1": "<ZONE1_NAME>"
}
```

If you need to change the server after you have a fabric up and running, see the section, "**Changing a Server Hostname in an Existing Fabric**."

Each data line in the template is a TYPE, VALUE pair. Elements on the left (before each colon) are TYPE elements that give the generic identifier for the topological element. Elements to the right of each colon are the VALUE elements, which must be substituted with user provided data.

For example, the tag, switch1, is the interface for switch1's DNS name. If your switch is named, frodo as shown in the following example, you would replace the angle-bracket-portion with "frodo" in a PATCH command to the redfish/v1/Fabrics/Oem/GigaIO/Topologies/1S-2x4-1 resource.

In addition, other non-switch-oriented tags in the template must be substituted as well, and the substituted values become <id> portions of other Redfish resources.

```
$ cat host_list.json
{
    "s1p9dsp": "JBOG_PORT_9",
```

```
    "s1p13dsp": "UNUSED_PORT_13",
    "s1p17dsp": "JBOF_PORT_17",
    "s1p21dsp": "UNUSED_PORT_21",
    "switch1": "frodo",
    "zone1": "zone1"
}
```

Quite simply, the DSP port tags (and zone tags) will need some kind of alias, which is used in referencing Redfish information queried later on. So, choose an alias that makes sense and use Linux-like hostname rules for those aliases.

FabreX Manager Tool Usage

The FabreX Manager Tool (fmtool), along with the files in the 1822589314 section, are used to configure a topology. The fmtool executable installs in the /usr/bin directory.

Its usage is given below:

```
usage: fmtool [-h] [--help]
       fmtool [-V] [--version]
       fmtool [-a <auth_file>] [-v] [-f] [-i] <host_file> <ltr_fil
       fmtool [-a <auth_file>] [-v] -b <blacklist_file> <fabric_ma
       fmtool [-a <auth_file>] [-v] -B switch:<switch_name>,part_i
       fmtool [-a <auth_file>] [-v] -l <loglevel> <fabric_manager>
       fmtool [-a <auth_file>] [-v] -L <worker1>:<loglevel1>,...,<
       fmtool [-a <auth_file>] [-v] -s <fabric_manager>
       fmtool [-a <auth_file>] [-v] -u <fabric_manager>
       fmtool [-a <auth_file>] [-v] -U switch:<switch_name>,port_i
...
```

⚠ The fabric_manager is the same as the manager_switch.

General options

FM Tool General Options

Option	Usage
man fmtool	Brings up the fmtool manual. Similar output as fmtool -h
-a --auth	Path to base64 encoded auth file, such as 'echo -n email:password | base64 > ~/.fabrex_auth' (default: /home/user/.fabrex_auth)
-B --bind	Allows a user to bind downstream port(s) to a partition on a switch. switch:<SWITCH_NAME>,part_id:<PARTITION_ID>,port_id:<PORT_ID>, --bind switch:<SWITCH_NAME>,part_id:<PARTITION_ID>,port_id:<PORT_ID>
-f --force	Forces a reset of switch(es) and iniitializes the fabric. Meaningful only when loading a topology with deprecated form
<fabric_manager>	This is the name of the switch identified as the <manager_switch> for the topology. It must be one of the switch names identified in the host_list.json file and must be used exclusively for subsequent fmtool commands. See the section, "**Initializing a Topology on the Fabric**." All other switches in the topology will need to have their "showmgr" setting be set to the manager switch name using the "setmgr" command. By default, each switch has sw-rs4024 or sw-rs3024 as the manager switch. In multi-switch topologies, no two switches in topology can have the exact same name.
host_list.json	Provides tags for components within the topology being

	installed onto the fabric. See the section, "**Initializing a Topology on the Fabric**."
-l , --loglevel	Specifies the level of logging for the fabric_manager
-L --worker-loglevel	Specifies the level of logging for worker(s)
ltr.yml	Defines the topology and initial configuration for installing the topology onto the fabric
-s --status	Status of fabric command
-u --uninitialize	Remove the existing fabric configuration command
-U --unbind	Unbind DSP (downstream port) command
-v --verbose	Verbose mode
-V --version	Version

Displaying the fmtool man page

The following example shows the usage to display the FabreX Manager Tool 'fmtool' Manual.

```
user@manager_switch:~$ man fmtool
FMTOOL(1)

NAME
       fmtool - fmtool 2.2.0

SYNOPSYS
       fmtool [-h] [--help]
       fmtool [-V] [--version]
       fmtool [-a AUTH_FILE] [-v] [-f] HOST_LIST_FILE LTR_FILE FAB
       fmtool [-a AUTH_FILE] [-v] -B SWITCH:<SWITCH_NAME>,PART_ID:
       fmtool [-a AUTH_FILE] [-v] -l <LOGLEVEL> FABRIC_MANAGER
       fmtool [-a AUTH_FILE] [-v] -L <WORKER1>:<LOGLEVEL1>,...,<WO
```

```
       fmtool [-a AUTH_FILE] [-v] -s FABRIC_MANAGER
       fmtool [-a AUTH_FILE] [-v] -u FABRIC_MANAGER
       fmtool [-a AUTH_FILE] [-v] -U SWITCH:<SWITCH_NAME>,PORT_ID:

DESCRIPTION
       Client for FabreX Fabric Management; allows a user to initi

       Supported Version(s): FabreX 2.2.0

OPTIONS
       Following argument list describes all supported arguments,

   POSITIONAL ARGUMENTS
       HOST_LIST_FILE
              Path  to  host_list.json  file.  Usually delivered a
              nected to the port(s) on the switch(es) (c) The alia
              string such as 'blank1', 'blank2' etc.

       LTR_FILE
              Path to ltr.yml file. FabreX Topology Records (LTRs)

       FABRIC_MANAGER
              This is the hostname of the switch identified as the

  ...
```

Uninitializing a Topology on the Fabric

To remove the currently established topology, use the uninitialize command. The `fmtool` usage for uninitializing a fabric is given by:

```
fmtool -u <switch_name>
```

The expected output is shown in the following example:

```
fmtool: 2021-08-20 at 13:42:54
Sending uninit fabric to <switch_name>...
Request:
PUT https://&lt;SWITCH>/fabrex/v2/uninit
Response:
HTTP/1.1 200 OK
SUCCESS: uninitialize fabric complete on <switch_name>
```

Initializing a Topology on the Fabric

> Make sure to unintialize a fabric before initializing it.

The `fmtool` usage for configuring a fabric is given by:

```
fmtool [-v] [-l <loglevel>] [-f] [host_list.json ltr.yml] <manager
```

You must include the `host_list.json` file you modified from the same directory as the `ltr.yml` file. In the following example, the single switch topology (LTR) is used; the `<manager_switch>` is the switch name. The command is run from the `/opt/gigaio-fabrexfm-tool/topologies/release/sj5/1S-2x4-1/` directory where the two files are located. If there is more than one switch in the topology, then you must select one of the switches to be the <manager_switch>. All subsequent fmtool commands must use the same <manager_switch> name.

```
[user@fmtool-server 1S-2x4-1] $ fmtool -f host_list.json ltr.yml <
fmtool: 2020-02-21 at 09:34:12
Configuring fabric...
Sending host_list.json to <switch_name>...
Success: Sent host_list.json to <switch_name>
Sending ltr.yml to <switch_name>...
Success: Sent ltr.yml to <switch_name>
[user@fmtool-server 1S-2x4-1] $
```

Checking the Status of the Fabric

The `fmtool` usage to obtain the status of a fabric is given by:

```
fmtool [-v] -s <manager_switch>
```

The status information is returned in JSON form and is quite long. It consists of the following structure:

- Fabric
- Hosts
- Switches

Fabric Status

The following example shows obtaining status for the single switch topology 1S-2x4-1. Here the **Fabric** status is `CONFIGURED`, meaning a LTR and Host List file have been used to configure the topology:

```
[user@fmtool-server 1S-2x4-1] $ fmtool -s <switch_name>

fmtool: 2020-02-21 at 13:59:06
Requesting status from <switch_name>...
Response:
...
            "Fabric State": "CONFIGURED"
        }
    },
    "status_code": 200
}
Success: Status from <switch_name>
```

Also note that the final `status_code` for the `fmtool -s` command was 200, an indication that the command is complete.

Here is example output for `fmtool -s` for a switch (or topology) that is `NOT CONFIGURED`:

```
[user@fmtool-server 1S-2x4-1] $ fmtool -s <switch_name>

fmtool: 2020-02-21 at 14:16:13
Requesting status from <switch_name>...
Response:
{
    "status_code": 200,
    "body": {
        "fabric": {
            "Fabric State": "NOTCONFIGURED",
            "hosts": [],
            "switches": [],
            "Host List State": "NOTCONFIGURED"
        }
    }
}
Success: Status from <switch_name>
```

Host Server State and Status

The following table lists host states.

Host Server States

Field	Host state indicates if the host is...
DISCONNECTED	Disconnected from fabric management communicating with the host server. This is usually because the host server is powered off.
NOTCONFIGURED	In the initial state of a host server before it connects to or

	configures with fabric management. If the host server is powered, but still in this state, the host driver may not be installed.
CONNECTED	Up and communicating with fabric management, with the host server connected and ready to accept requests from the manager.
CONFIGURED	Configured by fabric manager; the host server is now ready to communicate with other peer servers that are also CONFIGURED.

The following sample output shows two hosts that are `CONNECTED` and `CONFIGURED,` while the third host is `DISCONNECTED.`

The peers server status indicates if the host server's peers are:

- NOTCONFIGURED - Fabric manager has not configured the peer host for communication with the host server.
- CONFIGURED - Fabric manager has configured the peer server for communication with the host server.

```
...

            "Host List State": "CONFIGURED",
            "hosts": [
                {
                    "pc_peer_id": "HOST1",
                    "peers": [
                        {
                            "pc_peer_id": "HOST2",
                            "state": "CONFIGURED"
                        },
                        {
                            "pc_peer_id": "HOST3",
```

```
                        "state": "NOTCONFIGURED"
                    }
                ],
                "state": "CONFIGURED"
            },
            {
                "pc_peer_id": "HOST2",
                "peers": [
                    {
                        "pc_peer_id": "HOST1",
                        "state": "CONFIGURED"
                    },
                    {
                        "pc_peer_id": "HOST3",
                        "state": "NOTCONFIGURED"
                    }
                ],
                "state": "CONFIGURED"
            }
            {
                "pc_peer_id": "HOST3",
                "peers": [
                    {
                        "pc_peer_id": "HOST1",
                        "state": "NOTCONFIGURED"
                    },
                    {
                        "pc_peer_id": "HOST2",
                        "state": "NOTCONFIGURED"
                    }
                ],
                "state": "DISCONNECTED"
            }
        ]
...
```

Switch State and Status

The following output shows switch_state details for a worker switch state in the topology. The switch state returns with the following fields:

Worker Switch States

Field	Fabric Management state is...
DISCONNECTED	Disconnected from communicating with the worker switch (in a multi-switch topology). Usually this occurs because the worker switch is powered off.
NOTCONFIGURED	In the initial state of a worker switch before it connects to or configures with fabric management. If the worker switch is in this state it may be due to: • setmgr having the wrong manager_switch set • installed version has a mismatch
CONNECTED	Up and communicating, with the worker switch connected and ready to accept requests from the manager switch.
CONFIGURED	Configured for the worker switch with the correct configuration file.

The following output shows status details for a manager or worker switch status in the topology. The switch status returns with the following fields:

Switch State Status

Field	Status
switch_state	See switch states in the preceding table
binding	Shows the binding status for all DSPs
GUID	Switch GUID
switch_id	Name of the manager or worker switch
config_state	Lists the configuration state of the topology the fmtool

	command loaded
ports	Lists the port details
config_file	Lists the file that is configured for the manager or worker switches

```
...
            "Switches": [
                {
                    "switch_state": "CONFIGURED",
                    "binding": {
...
                    },
                    "GUID": "0000010105190003",
                    "switch_id": "<switch_name>",
                    "config_state": "RUNNING",
                    "ports": [
...
                    "config_file": "sj1-r-c043.yls.2B32G.I32G-1.3-
                }
            ]
...
```

Switch Configuration State and File

A topology consists of one or more switches, where each switch is running a particular configuration. The `config_state` field indicates one of the following states:

Switch Configuration States

State	Configuration file in a switch stanza...
UNAVAILABLE	Is invalid; no action is taken

LOADABLE	Is valid; can be loaded, activated, and run after a switch reset
ACTIVATED	Is loaded and configured/activated; a switch reset will run this file
RUNNING	Is active and running

The `config_file` field indicates which configuration has been loaded on the switch and is running.

```
...
                "config_state": "RUNNING",
...
                "config_file": "sj1-r-c043.yls.2B32G.I32G-1.3--08c
...
```

Switch Partitions

Switch ports are grouped into partitions. Each partition contains one upstream port (USP) and zero or more downstream ports (DSPs).

Port Numbering

Use the lowest numbered Switch Port Number to designate which port connections to bind or unbind. All x4 and x8 or x16 grouped ports are referenced by the lowest numbered Switch Port Number.

Port Binding Status

Switch ports are `bound` to a partition. DSPs can be unbound from a partition and bound to another partition. The `binding` field indicates the current port binding configuration for the given switch.

Port Binding Status Port Fields

Port Field	Usage
`can_unbind`	True if there are any DSPs in the bound list

`bind_dest`	Valid partition numbers
`unbound`	List of DSPs currently not bound to any partition
`bound`	List of DSPs currently bound to a partition
`open_slots`	True if there is a DSP that can be bound/unbound from a bind_destination
`can_bind`	True if there are any DSPs in the unbound list

The following example shows how the port fields are used to bind or unbind a partition.

```
...
                    "binding": {
                        "can_unbind": true,
                        "bind_dest": [
                            0,
                            1
                        ],
                        "unbound": [],
                        "bound": [
                            9,
                            13,
                            17,
                            21
                        ],
                        "open_slots": true,
                        "can_bind": false
                    },
...
```

Port Status

A switch port represents a PCIe port on the switch. The **ports** field indicates the current PCIe port configuration for the given switch.

Port Configuration Fields

Port Field	Usage
`Dir`	Indicates whether the port is upstream (USP) or downstream (DSP).
`LexPort`	The physical port number(s) corresponding to this PCIe port, with the range of switch ports comprising an x16, x8, or x4 link. In fmtools, it references the lowest port number in the range shown for the port grouping.
`Par`	The partition of the port is currently bound to on the switch
`Max`	The maximum lane count possible for this port or group of ports: x16, x8, or x4
`Neg`	The negotiated lane count of this port or group of ports: x16, x8, or x4
`Status`	The PCIe physical link status of the port: UP or DOWN
`Rate`	The negotiated PCIe link rate of the port, with the following possible values: • 16G: 16 Gbits/sec per lane, Gen 4 PCIe • 8G: 8 Gbits/sec per lane, Gen 3 PCIe • 5G: 5 Gbits/sec per lane, Gen 2 PCIe • 2.5G: 2.5 Gbits/sec per lane, Gen 1 PCIe

The following example shows how the ports field is used to indicate the current PCIe port configuration for the given switch. The ports are grouped by the partition they are bound to.

```
...
                    "ports": [
                        {
                            "Neg": "x16",
```

```
                        "Rate": "8G",
                        "Status": "UP",
                        "LexPort": "1..4",
                        "Dir": "USP",
                        "Max": "x16",
                        "Par": "0"
                    },
                    {
                        "Neg": "x0",
                        "Rate": "2.5G",
                        "Status": "DOWN",
                        "LexPort": "9..12",
                        "Dir": "DSP",
                        "Max": "x16",
                        "Par": "0"
                    },
...
```


Stop the Fabric

The key reasons to run this command:

- To add a new or different host server/switch
- To change the physical connection to any switch in the fabric
- To reboot or power cycle a worker switch (non-manager)

⚠ If rebooting the switch, the host servers should be powered down; otherwise, they may experience a kernel panic due to PCIe sub-tree disappearing while the switch is rebooting.

When power cycling a switch, allow the switch power supplies to fully power down (approximately 30 seconds).

The `fmtool usage to` remove the current topology of a fabric is given by:

```
fmtool [-v] -u <manager_switch>
```

The following example de-configures the single switch topology configured above, where the switch host name is ***<manager_switch>***:

```
$ fmtool -u <manager_switch>
Sending uninit: <manager_switch>...Done
```

Unbind and Bind IO Ports in a Running Topology

> Power down the host server prior to running a bind or unbind operation on IO ports.

The `fmtool` usage for binding and unbinding IO in a fabric is given by:

```
fmtool [-v] -U switch:switch_name,port_id:<port ID> <manager_switc
fmtool [-v] -B switch:switch_name,part_id:<partition ID>,port_id:<
```

The following example unbinds the DSP, identified by `port_id` 13, from its current partition on switch `switch_name`. Note that if you only have one switch, `switch_name` is equal to `<manager_switch>`.

```
$ fmtool -U switch:<switch_name>,port_id:13 <switch_name>
Sending unbind: <switch_name>...Done
```

If you have multiple switches, use `switch_name` to indicate the switch from which to unbind/bind the DSP. The following example shows how to bind `switch_name port 13` to partition *1* on `<manager_switch>:`

```
$ fmtool -B switch:<switch_name>,part_id:1,port_id:13 <manager_swi
Sending bind: <manager_switch>...Done
```

Bind Error Report

In bind and unbind operations, the correct partition and port numbers must be specified. Errors include attempts to:

- bind a bound port
- unbind an unbound port
- unbind a USP
- specify an invalid partition
- specify an invalid partition/port combination

The error output will list valid partitions and ports. The following example shows an attempt to bind switch port 13, which fails because switch port 13 is not in the unbound ports list.

```
$ fmtool -B switch:switch name,part_id:1,port_id:13 <manager_switc
Sending bind: <manager_switch>...Error 500, bind failed
valid paritions: [0,1]
bound ports: ["9","13","17","21"]
unbound ports: []
```

Set Switch log level

Each switch produces a log output to a file. The `fmtool` usage to set the log level for a switch is given by:

```
fmtool [-v] [-l <loglevel>] <manager_switch>
```

Valid values for `loglevel` are:

Value	loglevel
0	DEBUG
1	INFO
2	WARNING
3	ERROR
4	CRITICAL

The default loglevel is 1 (INFO). Note that loglevel 0 includes 1-4, loglevel 1 includes 2-4, and so on.

The following example sets the `loglevel` for switch `<manager_switch>` to 0.

```
$ fmtool -l 0 <manager_switch>
Sending loglevel: <manager_switch>...Done
```

Set fxwd log level

Each switch produces output to the fxworker logging file. Using fmtool command you can set the logging level independently on each host.

```
fmtool [-v] -L server1_hostname:<loglevel1>,...,serverN_hostname:<
```

The following example sets the `loglevel` for hosts `server1_hostname` and `server2_hostname` to 0 (DEBUG):

```
$ fmtool -L server1_hostname:0,uppers:0 <manager_switch>
Sending host loglevel: <manager_switch>...Done
```

The host log output is captured in /var/log/fabrex/fwworker.log on `server2_hostname`. In looking at the log after running the command above, notice the additional *DEBUG* output in the log,

```
Feb  3 21:59:18 host1 fxwd: [INFO    ] SET LOGLEVEL: 0
Feb  3 21:59:29 host1 fxwd: [INFO    ] REMOVE PEER: "server1_hostn
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_peer_id = server1_h
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_peer_flid = 1
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_host_index = 0
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_ntb_guid = 0x500e00
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_dtf_bar: 2
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_dtf_sz: 0x800000000
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_dtf_off: 0
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_mtf_bar: 2
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_mtf_sz: 0x100000
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_mtf_off: 0xfee00000
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_peer_ccs_off = 0
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_peer_ccs_msi_off 0
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_peer_ccs_msi_data 0
Feb  3 21:59:29 host1 fxwd: [DEBUG   ]     nfp_stat = 0x2
```

This `fxwd loglevel` persists until changed by `fmtool`, host is rebooted, or the `fxworker` service is restarted on the host.

Changing a Server in an Existing Fabric

It is highly recommended not to change the server hostname since it requires the running fabric be stopped to include the new server. If you must change the server, use the following procedure:

1. Stop all host applications on the host you are trying to replaced that uses the fabric for communication.
2. Power down the server.

3. Connect to the new server and power on the server.