

NVIDIA UFM Enterprise User Manual v6.23.11 LTS (2025 LTS U1)

Table of contents

Release Notes 7

Changes and New Features 8

Installation Notes 14

Bug Fixes in This Release 22

Known Issues in This Release 23

UFM Overview 24

UFM Benefits 25

Main Functionality Modules 26

Getting Familiar with UFM's Data Model 28

UFM Communication Requirements 29

UFM Software Architecture 34

UFM Web UI Overview 37

Access the WebUI 37

WebUI Layout 38

Set User Preferences 40

UFM Installation and Configuration 46

UFM Installation Steps 46

Downloading UFM Software and License File 47

Installing UFM Server Software 50

Installing UFM Server on Bare Metal Server 57

Installing UFM on Bare Metal Server - High Availability Mode 57

Installing UFM on Bare Metal Server- Standalone Mode 65

Installing UFM Docker Container Mode 66

Installing UFM on Docker Container - High Availability Mode 73

Installing UFM on Docker Container - Standalone Mode 80

Replacing the Standby Node 82
Installing UFM Podman Rootless Container Mode 83
UFM Infra Installation 94
Installing UFM Infra Using Docker Root 94
Installing UFM Infra Using Rootless with Podman 99
Activating Software License 114
UFM Configuration 118
Initial Configuration 118
Optional Configurations 119
Running UFM Server Software 181
Upgrading UFM Software 196
Upgrading UFM on Bare Metal Server 197
Upgrading UFM on Docker Container 201
Upgrading UFM Podman Rootless Container 205
Uninstalling UFM 207
UFM Configuration Backup and Restore 209
UFM Factory Reset 214
UFM Rollback 218
Manage Users 220
Authentication Methods 222
UFM Server Health Monitoring 247
Events and Alarms 255
Threshold-Crossing Events Reference 256
Reports 278
Telemetry 279

High-Frequency (Primary) Telemetry Fields 288
Low-Frequency (Secondary) Telemetry Fields 290
UFM Web UI 297
Access the WebUI 37
WebUI Layout 38
Set User Preferences 40
Multi-Subnet UFM 306
UFM Plugins 322
Plugins Release Notes 324
UFM Plugins Management 325
Plugins Bundle 330
REST-RDMA Plugin 333
NDT Plugin 347
UFM Telemetry Fluentd Streaming (TFS) Plugin 367
UFM Events Fluent Streaming (EFS) Plugin 368
UFM Bright Cluster Integration Plugin 370
IB Link Resiliency Plugin 374
ClusterMinder Plugin 396
Sysinfo Plugin 417
SNMP Plugin 419
Packet Level Monitoring Collector (PMC) Plugin 424
GNMI-Telemetry Plugin 428
UFM Telemetry Manager (UTM) Plugin 467
UFM Consumer Plugin 484

Fast-API Plugin 487
UFM Light Plugin 493
Key Performance Indexes (KPI) Plugin 499
UFM Events Grafana Dashboard Plugin 520
Log Streamer Plugin 523
GNMI NVOS Events Plugin 526
NVLink Plugin 533
Unmanaged Switch Dump (USD) Plugin 552
UFM Infra 559
Installing UFM Infra Using Docker Root 94
Installing UFM Infra Using Rootless with Podman 99
Troubleshooting 586
Appendixes 609
SM Configurations 609
Appendix – SM Default Files 610
Appendix – UFM Subnet Manager Default Properties 610
Appendix – Partitioning 637
Appendix – Enhanced Quality of Service 646
Appendix – Routing Chains 648
Appendix – Adaptive Routing 660
Appendix – Security Features 661
Appendix – SM Activity Report 678
OpenSM Configuration 680
Appendix – UFM AI Mode 681
Appendix – Diagnostic Utilities 682

Appendix - Supported Port Counters and Events 716
Appendix – Used Ports 719
Appendix – Configuration Files Auditing 720
Appendix – IB Router 721
Appendix – NVIDIA SHARP Integration 729
Appendix - UFM SLURM Integration 733
Appendix - Switch Grouping 739
Appendix – Device Management Feature Support 747
Appendix - MAD Limiter 751
Appendix - UFM Versions Manager 753
Installation and Uninstallation 758
Configurations Snapshot 760
System Backup and Restore 770
Migration 777
Upgrade 782
Documentation History 789
Document Revision History 789
Release Notes History 789
Changes and New Features History 790
Bug Fixes History 790
Known Issues History 791
EULA, Legal Notices and 3rd Party Licenses 792

About This Document

NVIDIA ® UFM ® Enterprise is a powerful platform for managing InfiniBand scale-out computing environments. UFM enables data center operators to efficiently monitor and operate the entire fabric, boost application performance and maximize fabric resource utilization.

This user guide provides documentation for network administrators responsible for deploying, configuring, monitoring, and troubleshooting the network in their data center.

For a list of the new features, bug fixes and known issues in this release, see Release Notes.

Software Download

To download the UFM software, please visit NVIDIA's Licensing Portal.

If you do not have a valid license, please fill out the NVIDIA Enterprise Account Registration form to get a UFM evaluation license.

Revision History

For the list of changes made to this document, refer to Documentation History.

Release Notes

NVIDIA® UFM® is a powerful platform for managing InfiniBand scale-out computing environments. UFM enables data center operators to efficiently monitor and operate the entire fabric, boost application performance and maximize fabric resource utilization.

Key Features

UFM provides a central management console, including the following main features:

- Fabric dashboard including congestion detection and analysis
- Advanced real-time health and performance monitoring
- Fabric health reports
- Threshold-based alerts
- Fabric segmentation/isolation
- Quality of Service (QoS)
- Routing optimizations
- Central device management
- Task automation
- Logging
- High availability
- Daily report: Statistical information of the fabric during the last 24 hours
- Event management

> **⚠ Warning**
>
> Prior to installation, please verify that all prerequisites are met. Please refer to System Requirements.

Changes and New Features

UFM Enterprise Changes and New Features

> **ⓘ Note**
>
> Notes:
>
> - For an archive of changes and features from previous releases, please refer to Changes and New Features History.
> - The items listed in the table below apply to all UFM license types.
> - For bare metal installation of UFM, it is required to install DOCA_HOST before the UFM installation. Please make sure to use the UFM installation package that is compatible with your setup, as detailed in Bare Metal Deployment Requirements.

Feature	Description
UFM Versions Manager	Added the UFM Versions Manager command-line tool, which manages UFM system backups, configurations, and migrations. It provides comprehensive backup and restore capabilities for UFM systems running on root-less Podman containers, with full support for both Standalone (SA) and High Availability (HA) environments. For more information, refer to Appendix - UFM Versions Manager.
Upgraded Python Version	Upgraded runtime UFM Python version from 3.9 to 3.12.

Feature	Description
Fabric Health as Unified Tool	Added the ability to use UFM fabric health report as a centralized fabric-wide health monitoring tool. For more information, refer to Fabric Health Tab and the Reports REST API.
Supported Operating Systems	Added support for RedHat 10. For more information, refer to Installation Notes.
System Dump Script	Reduced the UFM system dump collected by default via the `ufm_sysdump.sh` script. For more information, refer to Collecting System Dump.
SHARP Reservations	Added support for automatic synchronization between the PKey database and SHARP reservations in multi-tenant clusters - upon PKey modifications the corresponding reservations are automatically updated . For more information, refer to UFM SHARP REST API .
SHARP	Enabled SHARP by default in UFM. For more information, refer to Enabling SHARP Aggregation Manager.
Cloud Profile Mode	Added support for Secured Cloud Profile mode for CSPs (Cloud Service Providers) which leverages enhanced security settings when enabled. For more information, refer to Cloud Profile Mode.
Clustered Telemetry	Added the Clustered Telemetry feature which enables multiple telemetry data collection instances across multiple network adapters (HCAs) in the InfiniBand fabric. This feature provides improved performance and scalability for large-scale deployments through workload distribution. For more information, refer to Telemetry.
Telemetry Enhancements	Optimized the UFM telemetry restart mechanism only upon newly added ports.
Visibility into UFM Services	Added the ability to monitor and track the initialization and progress of all UFM services in real-time during UFM startup. For more information, refer to UFM Startup Logging.
Syslog API	Updated the Syslog API validation to support hostnames in addition to IPv4 and IPv6 addresses for the destination parameter.
UFM Configuration Parameters	Aligned the UFM large-scale subnet configuration parameters. For more information, refer to Adjusting UFM Configuration Files Based on Fabric Size.
Azure Authentication	Added the ability to increase Azure AD username length. For more information, refer to Azure AD Authentication.

Tools Changes and New Features

Tool	Version	Changes and New Features	
SHARP	3.13.0	Automatic Synchronization of SHARP Reservations and PKeys	Added support for automatic synchronization between the PKey database and SHARP reservations (create, modify, and delete) in multi-tenant clusters. When this mode is enabled, there is no need to invoke the SHARP reservation REST API manually.
		Stochastic Rounding Support (**Beta**)	Added beta-level support for Stochastic Rounding during SHARP mathematical calculations.
		Default Client-Server Communication	New installations of SHARP_am now use UCX as the default communication method with libsharp instead of sockets. During upgrades, SHARP_am retains the previously configured communication method. If the system was previously using sockets, customers are advised to switch to UCX for improved reliability.
		Reduce-Scatter and All-Gather	Added Reduce-Scatter and All-Gather operations support on Quantum-3 systems.
		Default Configuration in UFM	`sharp_enable` is now set to `true` by default, eliminating the need to update the settings in the configuration file.
		For more information, please refer NVIDIA Scalable Hierarchical Aggregation and Reduction Protocol (SHARP) Rev 3.13.0.	

Tool	Version	Changes and New Features	
OpenSM	5.25.1	Routing	Improved routing calculation time on planarized networks.
			Reduced routing table updates on planarized networks.
		General	Added support for SHARP multicast tress.
			Added support for writing port partition table to SMDB file.
			Added support for specifying BW limitation to devices connected to XDR switches.
			Added support for configuring SL2VL Mapping from device configuration file.
			Added information on whether a parameter can be changed during runtime to generated configuration file.
		Logging	Added support for additional APort validity checks.
		For more information, please refer MLNXSM (InfiniBand Subnet Manager) Utility Release Notes v5.25.1.	

Tool	Version	Changes and New Features
IBUtils2 Utility	2.24.0	• Added support for CPO. • Added support for Multi-Cast Private LFT. • Added support for GPU devices. • Updated routing analysis in XDR fabrics. • Added support for PPLM register. • Added support for ARHC and ARHR HCA registers. • Added support for new PEMI layouts. • Updated PEMI section numbering according to PRM. • Disabled by default PEMI SNR and PEMI OEDP. • Disabled by default SLRG, SLRP, SLRIP, and SLSIR registers. • Disabled Diagnostic Data Pages; using Access Registers only. • Changed PDDR cable register timeout to long timeout. • Updated APort file. • Updated fetch mechanism for all MADs. • Reduced the number of recurring MAD fetch errors. • Added delta for Performance Histogram Port Data. • Added delta for Performance Histogram Buffer Data. • Added Template GUID to Physical Hierarchy section.
		For more information, please refer IBUtils2 Utility Documentation v2.24.0.

Plugins Changes and New Features

For plugin changes, new features, bug fixes or known issues, refer to plugin documentation under UFM Plugins.

Unsupported Functionalities/Features

The following distributions are no longer supported in UFM:

- RH7.0-RH7.7 / CentOS7.0-CentOS7.7
- SLES12 / SLES 15
- EulerOS2.2 / EulerOS2.3
- Ubuntu18.04

Deprecated Features:

- Mellanox Care (MCare) Integration
- UFM on VM (UFM with remote fabric collector)
- Logical server auditing
- The UFM high availability script - **/etc/init.d/ufmha** - is no longer supported
- The **UFM Multi-site portal** feature is no longer supported. The Multi-Subnet feature can be used instead
- As of UFM Enterprise v6.19.0, the Autonomous Link Maintenance **(ALM)** and **PDR Deterministic** plugins are no longer supported.
- The **GRPC-Streamer plugin** is deprecated.
- As of UFM Enterprise v6.18.0, **UFM Agent discovery** will be disabled by default, and managed switches will be discovered in-band
- As of UFM Enterprise v6.18.0, the `ibdiagpath` diagnostic utility is deprecated
- As of UFM Enterprise version 6.14.0, **UFM Monitoring Mode** is deprecated and is no longer supported
- As of UFM Enterprise v6.12.0, the **Logical Elements tab** is removed
- Removed the following fabric validation tests: **CheckPortCounters & CheckEffectiveBER**

Note

In order to continue working with **/etc/init.d/ufmha** options, use the same options using the **/etc/init.d/ufmd** script.

For example:

Instead of using **/etc/init.d/ufmha model_restart,** please use **/etc/init.d/ufmd model_restart** (on the primary UFM server)

Instead of using **/etc/init.d/ufmha sharp_restart,** please use **/etc/init.d/ufmd sharp_restart** (on the primary UFM server)

The same goes for any other option that was supported on the **/etc/init.d/ufmha** script

Installation Notes

Supported Devices

Supported NVIDIA Externally Managed Switches

Type	Model	Latest Tested Firmware Version
NDR switches	• MQM9790	31.2016.2054
HDR switches	• MQM8790	27.2014.2170
EDR switches	• SB7790 • SB7890	15.2010.4402

Supported NVIDIA Internally Managed Switches

Type	Model	Latest Tested OS Version
XDR switches	• Q3200_RA • Q3400_RA	NVOS 25.02.5030
NDR switches	• MQM9700	MLNX-OS 3.12.6000 NVOS 25.01.4000
HDR switches	• MQ8700 • MCS8500 • TQ8100-HS2F • TQ8200-HS2F	MLNX-OS 3.12.1002
EDR switches	• SB7700 • SB7780 • SB7800 • CS7500 • CS7510 • CS7520	MLNX-OS 3.10.4400

System Requirements

Bare Metal Deployment Requirements

Platform	Type and Version
OS (Relevant for Standalone and High-Availability deployments)	64-bit OS: • RedHat 8 • RedHat 9 • RedHat 10 • Ubuntu 20.04 • Ubuntu 22.04 • Ubuntu 24.04
CPU[(a)]	x86_64

Platform	Type and Version
HCAs	• NVIDIA ConnectX®-5 with Firmware 16.35.4506 and above • NVIDIA ConnectX®-6 with Firmware 20.44.0982 and above • NVIDIA ConnectX®-7 with Firmware 28.47.0416 and above • NVIDIA ConnectX®-8 with Firmware 40.47.0366 and above • NVIDIA Mezzanine Board with Four ConnectX-7 ASICs for Multi-GPU Connectivity (CEDAR) with Firmware 28.36.0394 and above • NVIDIA BlueField with Firmware 24.33.900 and above • NVIDIA BlueField-2 with Firmware 24.33.900 and above • NVIDIA BlueField-3 with Firmware 32.45.0350 and above
DOCA Host[(b)]	• MLNX_OFED23.x • MLNX_OFED24.x • MLNX_OFED25.x

Note

[(a)] CPU requirements refer to resources consumed by UFM. You can also dedicate a subset of cores on a multicore server. For example, 4 cores for UFM on a 16-core server.

[(b)] For supported HCAs in each DOCA Host version, please refer to DOCA Host Release Notes.

[(c)]UFM v6.15.0 is the last version to support NVIDIA ConnectX-4 adapter cards

Note

For running SHARP Aggregation Manager within UFM, it is recommended to use MLNX_OFED-5.4.X version or newer.

Note

Installation of UFM on minimal OS distribution is not supported.

Note

UFM does not support systems in which NetworkManager service is enabled.

Before installing UFM on RedHat OS, make sure to disable the service.

Docker Installation Requirements

UFM Docker Container is supported on the standard docker environment (engine).

The following operating systems were tested with Docker Container (as standalone container):

Component	Type and Version
Supported OS	• RHEL8 • RHEL9 • RHEL10 • Ubuntu20.04 • Ubuntu22.04 • Ubuntu24.04 • Debian 10 • Oracle Enterprise Linux 8.10

Note

For UFM Docker Container installation in HA mode, please refer to Bare Metal Deployment Requirements for the list of operating systems and kernels which support HA.

Note

On some Ubuntu OSs, Docker is installed via SNAP, which might lead to errors when trying to use UFM Plugins.

To solve this issue, perform the following:

1. Remove Docker installed via SNAP, run:

```
snap remove --purge docker
```

2. Update the local package index, run :

```
apt update
```

3. Install native Docker, run:

```
apt install-y docker.io
```

Podman Installation

Note

Podman is tested on Oracle Linux 8.10 distribution.

Installation Instructions

Since Docker and Podman are mutually exclusive, ensure that any existing Docker packages are removed before installing Podman. You can do this with the following command:

```
yum remove $(rpm -qa | grep docker)
```

Once Docker is removed, proceed with the installation of Podman and the Podman Docker-compatible CLI using the following commands:

```
dnf install podman podman-docker
```

Finally, enable and start the Podman socket to allow for socket-based communication:

```
systemctl enable --now podman.socket
```

After installing Podman, the deployment of UFM Docker Containers and UFM Plugins should function in the same way as the corresponding Docker deployments.

UFM Server Resource Requirements per Cluster Size

Fabric Size	CPU Requirements	Memory Requirements	Disk Space Requirements	
			Minimum	Recommended
Up to 1000 nodes	16-core server	16 GB	100 GB	500 GB
1000-5000 nodes	32-core server	64 GB	250 GB	1000 GB (1 TB)
5000-10000 nodes	64-core server	64 GB	1000 GB	2000 GB (2 TB)
Above 10000 nodes	64-core server	64 GB	2000 GB	4000 GB (4 TB)

UFM GUI Client Requirements

The platform and GUI requirements are detailed in the following tables:

Platform	Details
Browser	Edge, Internet Explorer, Firefox, Chrome, Opera, Safari
Memory	• Minimum: 8 GB • Recommended: 16 GB

MFT Package Version

Platform	Details
MFT	Integrated with MFT version 4.34.1-10

UFM SM Version

Platform	Type and Version
SM	UFM package includes SM version 5.25.1

Note

Assuming the SM is connected to the production cluster, it can handle any events (IB traps) coming from the fabric that is being built; such events should not affect the routing on the production cluster. If events occurred in the production cluster, the routing could be changed.

However, NVIDIA recommends isolating fabric sections to allow faster bring-ups, **faster troubleshooting and misconfiguration avoidance** that can cause routing errors. Isolation provides clearer SM and CollectX logs, avoiding warnings/errors from masking real production issues.

UFM NVIDIA SHARP Software Version

Platform	Type and Version
NVIDIA® Scalable Hierarchical Aggregation and Reduction Protocol (SHARP)™	UFM package includes NVIDIA SHARP software version 3.13.11

Used Ports by UFM Server

For a list of ports used by the UFM Server for internal and external communication, refer to Appendix – Used Ports

Software Update from Prior Versions

The installer detects versions previously installed on the machine and prompts you to run a clean install of the new version or to upgrade while keeping user data and configuration unchanged.

The upgrade from previous versions maintains the existing database and configuration, allowing a seamless upgrade process.

Info

Upgrading UFM Enterprise software version is supported up to two previous GA software versions (GA -1 or -2).

For example, if you wish to upgrade to UFM Enterprise v6.17.0, it is possible to do so only from UFM Enterprise v6.16.0 or v6.15.0.

Note

Due to a possible conflict, SM and SHARP installed by the MLNX_OFED must be uninstalled. The installation procedure will detect and print all MLNX_OFED packages that must be removed.

Bug Fixes in This Release

UFM Enterprise Bug Fixes

N/A

UFM High-Availability Bug Fixes

For UFM High-Availability related bug fixes, refer to UFM High-Availability v6.1.1 Bug Fixes.

UFM Plugins Bug Fixes

For more information, refer to plugin documentation under Plugins Release Notes

Tools Bug Fixes

Tool	Description
Open SM v5.25.1	• Fixed an issue with line breaks in Virtualization DB dump file • Fixed an issue with reconfiguring PFRN on routers when changing configuration during runtime • Fixed an issue with redundant HCA configuration on topology changes • Fixed trailing newlines from syslog messages • Fixed hang due to race when handling CC/VS/N2N KeyViolation MADs • Fixed an issue with failing DFP routing engine after failing to assign LID to CA • Fixed an issue with failing to configure routing tables on all switches • Fixed an issue with reporting APort attributes misalignment • Fixed an issue with multicast routing of SendOnly members in planarized networks • Fixed an issue with multicast routing of Aggregation Nodes
SHARP v3.13.0	• Fixed an issue where the metrics for active jobs and active SAT jobs were decreased on every job clean attempt, even when the cleanup failed, resulting in incorrect metric values. • Fixed an issue where a failure to clean a job caused the metric for jobs ended due to client failure to continuously increase, resulting in incorrect values being reported in telemetry.

Known Issues in This Release

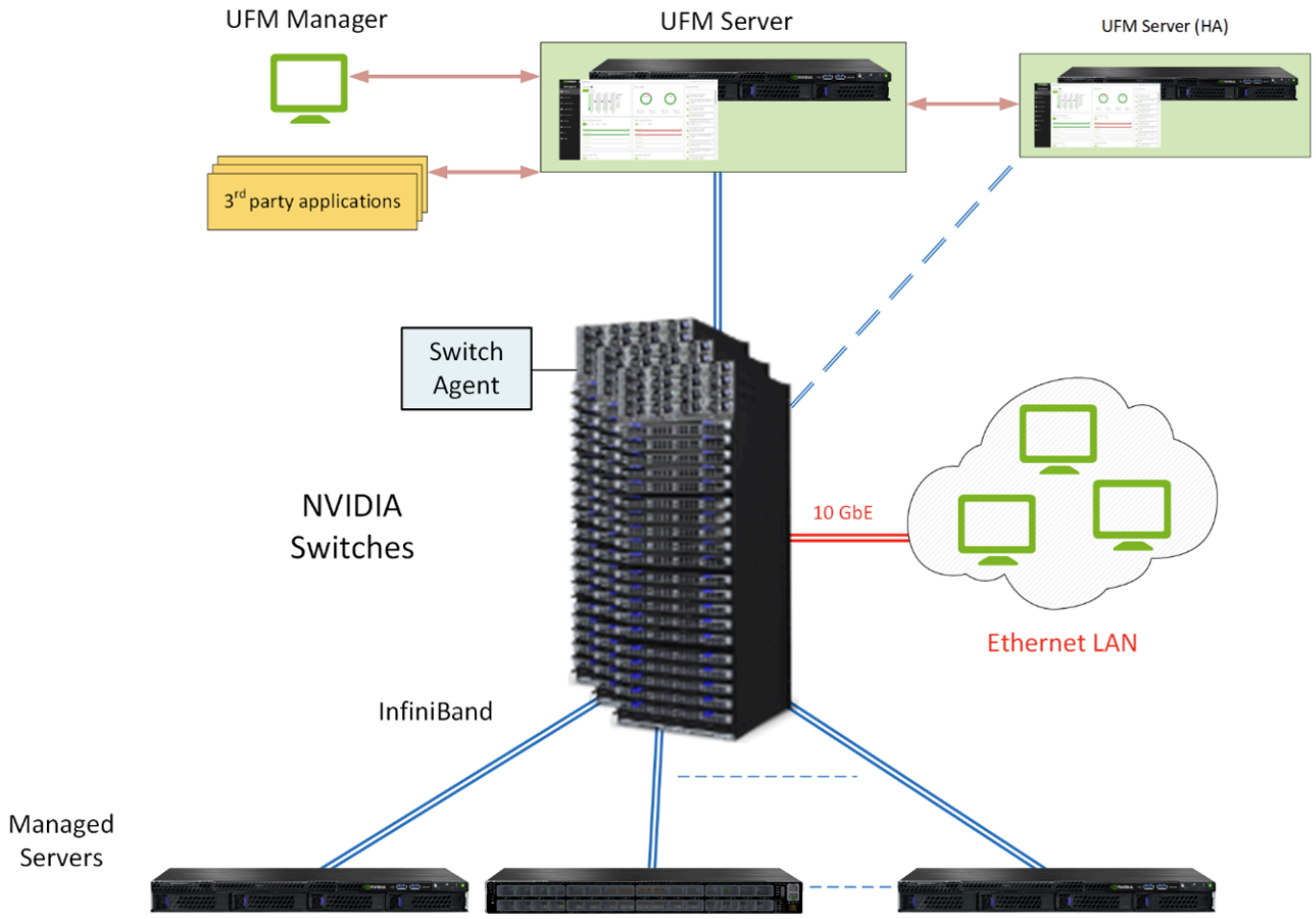
N/A

UFM Overview

Scale-Out Your Fabric with Unified Fabric Manager

NVIDIA®UFM is a host-based solution that provides all the management functionalities required for managing fabrics.

Fabric Topology with UFM



UFM Server is a server on which UFM is installed and has complete visibility over the fabric to manage routing on all devices.

UFM HA Server is a UFM installed server on a secondary server for High Availability deployment.

Managed Switching Devices are fabric switches, gateways, and routers managed by UFM.

Managed Servers are the compute nodes in the fabric on which the various applications are running, and UFM manages all servers connected to the fabric.

UFM Host Agent is an optional component that can be installed on the Managed Servers. UFM Host Agent provides local host data and host device management functionality.

The UFM Host Agent provides the following functionality:

- Discovery of IP address, CPU, and memory parameters on host
- Collection of CPU/Memory/Disk performance statistics on host
- Upgrading HCA Firmware and OFED remotely
- Creating an IP interface on top of the InfiniBand partition

UFM Switch Agent is an embedded component in NVIDIA switches that allows IP address discovery on the switch and allows UFM to communicate with the switch.

UFM Benefits

Benefit	Description
Central Console for Fabric Management	UFM provides all fabric management functions in one central console. The ability to monitor, troubleshoot, configure and optimize all fabric aspects is available via one interface. UFM's central dashboard provides a one-view fabric-wide status view.
In-Depth Fabric Visibility and Control	UFM includes an advanced granular monitoring engine that provides real-time access to switch and host data, enabling cluster-wide monitoring of fabric health and performance, real-time identification of fabric-related errors and failures, quick problem resolution via granular threshold-based alerts and a fabric utilization dashboard.
Advanced Traffic Analysis	Fabric congestion is difficult to detect when using traditional management tools, resulting in unnoticed congestion and fabric under-utilization. UFM's unique traffic map quickly identifies traffic trends, traffic bottlenecks, and congestion events spreading over the fabric, which enables the administrator to identify and resolve problems promptly and accurately.

Benefit	Description
Enables Multiple Isolated Application Environments on a Shared Fabric	Consolidating multiple clusters into a single environment with multi-tenant data centers and heterogeneous application landscapes requires specific policies for the different parts of the fabric. UFM enables segmentation of the fabric into isolated partitions, increasing traffic security and application performance.
Service-Oriented Automatic Resource Provisioning	UFM uses a logical fabric model to manage the fabric as a set of business-related entities, such as time critical applications or services. The logical fabric model enables fabric monitoring and performance optimization on the application level rather than just at the individual port or device level. Managing the fabric using the logical fabric model provides improved visibility into fabric performance and potential bottlenecks, improved performance due to application-centric optimizations, quicker troubleshooting and higher fabric utilization.
Quick Resolution of Fabric Problems	UFM provides comprehensive information from switches and hosts, showing errors and traffic issues such as congestion. The information is presented in a concise manner over a unified dashboard and configurable monitoring sessions. The monitored data can be correlated per job and customer, and threshold-based alarms can be set.
Seamless Failover Handling	Failovers are handled seamlessly and are transparent to both the user and the applications running on the fabric, significantly lowering downtime. The seamless failover makes UFM in conjunction with other Mellanox products, a robust, production-ready solution for the most demanding data center environments.
Open Architecture	UFM provides an advanced Web Service interface and CLI that integrate with external management tools. The combination enables data center administrators to consolidate management dashboards while flawlessly sharing information among the various management applications, synchronizing overall resource scheduling, and simplifying provisioning and administration.

Main Functionality Modules

Module	Description
Fabric Dashboard	UFM's central dashboard provides a one-view fabric-wide status view. The dashboard shows fabric utilization status, performance metrics, fabric-wide events, and fabric health alerts. The dashboard enables you to efficiently monitor the fabric from a single screen and serves as a starting point for event or metric exploration.
Fabric Segmentation (PKey Management)	In the PKey Management view you can define and configure the segmentation of the fabric by associating ports to specific defined PKeys. You can add, remove, or update the association of ports to the related PKeys and update the qos_parameters for PKey (mtu, rate, service_level).
Fabric Discovery and Physical View	UFM discovers the devices on the fabric and populates the views with the discovered entities. In the physical view of the fabric, you can view the physical fabric topology, model the data center floor, and manage all the physical-oriented events.
Central Device Management	UFM provides the ability to centrally access switches and hosts, and perform maintenance tasks such as firmware and software upgrade, shutdown and restart.
Monitoring	UFM includes an advanced granular monitoring engine that provides real-time access to switch and server data. Fabric and device health, traffic information and fabric utilization are collected, aggregated and turned into meaningful information.
Configuration	In-depth fabric configuration can be performed from the Settings view, such as routing algorithm selection and access credentials. The Event Policy Table, one of the major components of the Configuration view, enables you to define threshold-based alerts on a variety of counters and fabric events. The fabric administrator or recipient of the alerts can quickly identify potential errors and failures, and actively act to solve them.
Fabric Health	The fabric health tab contains valuable functions for fabric bring-up and on-going fabric operations. It includes one-click fabric health status reporting, UFM Server reporting, database and logs' snapshots and more.
Logging	The Logging view enables you to view detailed logs and alarms that are filtered and sorted by category, providing visibility into traffic and device events as well as into UFM server activity history.

Module	Description
High Availability	In the event of a failover, when the primary (active) UFM server goes down or is disconnected from the fabric, UFM's High Availability (HA) capability allows for a secondary (standby) UFM server to immediately and seamlessly take over fabric management tasks. Failovers are handled seamlessly and are transparent to both the user and the applications running in the fabric. UFM's High Availability capability, when combined with NVIDIA's High Availability switching solutions allows for non-disruptive operation of complex and demanding data center environments.

Please refer to the following sections for UFM's main functionalities:

- Events and Alarms
- Reports
- Telemetry

Getting Familiar with UFM's Data Model

Overview of Data Model

UFM enables the fabric administrator to manage the fabric based on discovery data collected from the fabric. This data is mapped into model elements (objects) available to the end user via UFM REST API and UFM Web UI.

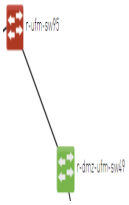
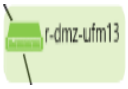
UFM Model Basics

The fabric managed by UFM consists of a set of physical and logical objects, including their connections. The Object Model has a hierarchical object-oriented tree structure with objects as the tree elements. Each object defines an abstraction for physical or logical fabric elements.

Physical Model

The Physical Model represents the physical resources and connectivity topology of the Network. UFM enables discovery, monitoring and configuration of the managed physical objects.

Physical Objects

Icon	Name	Description
N/A	Port Object	Represents the external physical port on switch or on Host Channel Adapter (HCA). A port is identified by its number. UFM provides InfiniBand standard management and monitoring capabilities on the port level.
N/A	Module Object	Represents the Field Removable Unit, Line card, and Network card on switch or HCA on host. For NVIDIA Switches, Line and Network Cards are modeled as modules.
	Link Object	Represents the physical connection between two active ports.
N/A	Cable Object	Represents the physical cable or the transceiver connected to one of the link edges.
	Computer Object	Represents the computer (host) connected to the Fabric. The UFM Agent installed on the host provides extended monitoring and management capabilities. Hosts without agents are limited to InfiniBand standard management and monitoring capabilities.
	Switch Object	Represents the switch chassis in the Fabric. A Switch object is created for every NVIDIA Switch. Switches of other vendors are represented as InfiniBand Switches and limited by InfiniBand standard management and monitoring capabilities.
	Rack Object	Represents the arbitrary group of switches or computers. When linked devices are shown as a group, the link is shown between the group and the peer object.

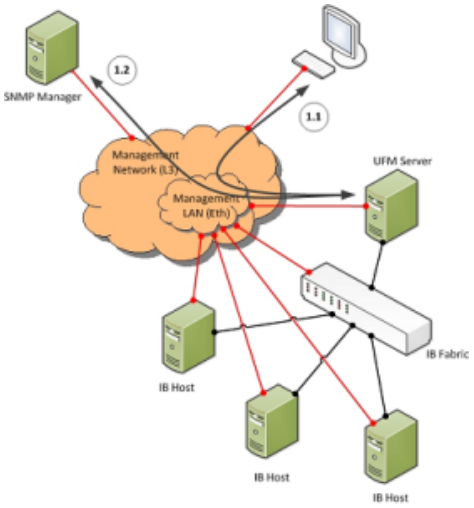
UFM Communication Requirements

This chapter describes how the UFM server communicates with InfiniBand fabric components.

UFM Server Communication with Clients

The UFM Server communicates with clients over IP. The UFM Server can belong to a separate IP network, which can also be behind the firewall.

UFM Server Communication with Clients



UFM Server Communication with UFM Web UI Client

Communication between the UFM Server and the UFM web UI client is HTTP(s) based. The only requirement is that TCP port 80 (443) must not be blocked.

UFM Server Communication with SNMP Trap Managers

The UFM Server can send SNMP traps to configured SNMP Trap Manager(s). By default, the traps are sent to the standard UDP port 162. However, the user can configure the destination port. If the specified port is blocked, UFM Server traps will not reach their destination.

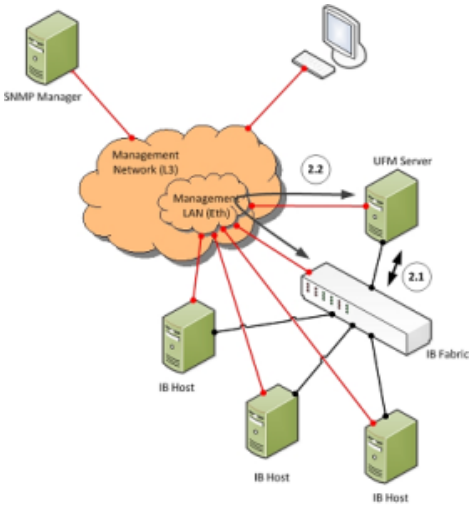
Summary of UFM Server Communication with Clients

Affected Service	Network	Address / Service / Port	Direction
Web UI Client	Out-of-band management*	HTTP / 80 HTTPS / 443	Bi-directional
SNMP Trap Notification	Out-of-band management*	UDP / 162 (configurable)	UFM Server to SNMP Manager

*If the client machine is connected to the IB fabric, IPoIB can also be used.

UFM Server Communication with InfiniBand Switches

UFM Server Communication with InfiniBand Switches



UFM Server InfiniBand Communication with Switch

The UFM Server must be connected directly to the InfiniBand fabric (via an InfiniBand switch). The UFM Server sends the standard InfiniBand Management Datagrams (MAD) to the switch and receives InfiniBand traps in response.

UFM Server Communication with Switch Management Software (Optional)

The UFM Server auto-negotiates with the switch management software on Mellanox Grid Director switches. The communication is bound to the switch Ethernet management port.

The UFM Server sends a multicast notification to MCast address 224.0.23.172, port 6306 (configurable). The switch management replies to UFM (via port 6306) with a unicast message that contains the switch GUID and IP address. After auto-negotiation, the UFM server uses Switch JSON API (HTTPS based) to retrieve inventory data and to apply switch actions (software upgrade and reboot) on the managed switch.

The following Device Management tasks are dependent on successful communication as described above:

- Switch IP discovery
- FRU Discovery (PSU, FAN, status, temperature)
- Software and firmware upgrades

The UFM Server manages IB Switch Devices over **HTTPS** (default port **443** – configurable) and / or SSH (default port 22 – configurable).

UFM Server Communication with Externally Managed Switches (Optional)

UFM server uses Ibdiagnet tool to discover chassis information (PSU, FAN, status, temperature) of the externally managed switches.

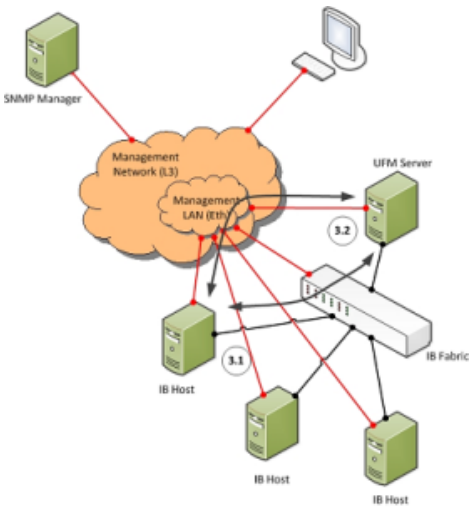
By monitoring chassis information data, UFM can trigger selected events when module failure occurs or a specific sensor value is above threshold.

Summary of UFM Server Communication with InfiniBand Switches

Affected Service	Network	Address / Service / Port	Direction
InfiniBand Management / Monitoring	InfiniBand	Management Datagrams	Bi-directional
Switch IP Address Discovery (auto-negotiation with switch management software)	Out-of-band management	Multicast 224.0.23.172, TCP / 6306 (configurable)	Multicast: UFM Server to switch TCP: Bi-directional
Switch Chassis Management / Monitoring	Out-of-band management	TCP / UDP / 6306 (configurable) SNMP / 161 (configurable) SSH / 22 (configurable)	Bi-directional

UFM Server Communication with InfiniBand Hosts

UFM Server Communication with InfiniBand Hosts



UFM Server InfiniBand Communication with HCAs

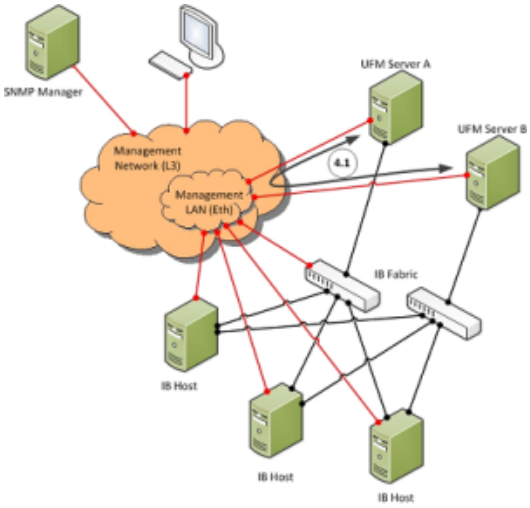
The UFM Server must be connected directly to the InfiniBand fabric. The UFM Server sends the standard InfiniBand Management Datagrams (MADs) to the Host Card Adapters (HCAs) and receives InfiniBand traps.

UFM Server Communication with InfiniBand Hosts

Affected Service	Network	Address / Service / Port	Direction
InfiniBand Management / Monitoring	InfiniBand	Management Datagrams	Bi-directional

UFM Server High Availability (HA) Active—Standby Communication

UFM Server HA Active—Standby Communication



UFM Server HA Active—Standby Communication

UFM Active — Standby communication enables two services: heartbeat and DRBD.

- *heartbeat* is used for auto-negotiation and keep-alive messaging between active and standby servers. *heartbeat* uses port 694 (udp).
- DRBD is used for low-level data (disk) synchronization between active and standby servers. DRBD uses port 8888 (tcp).

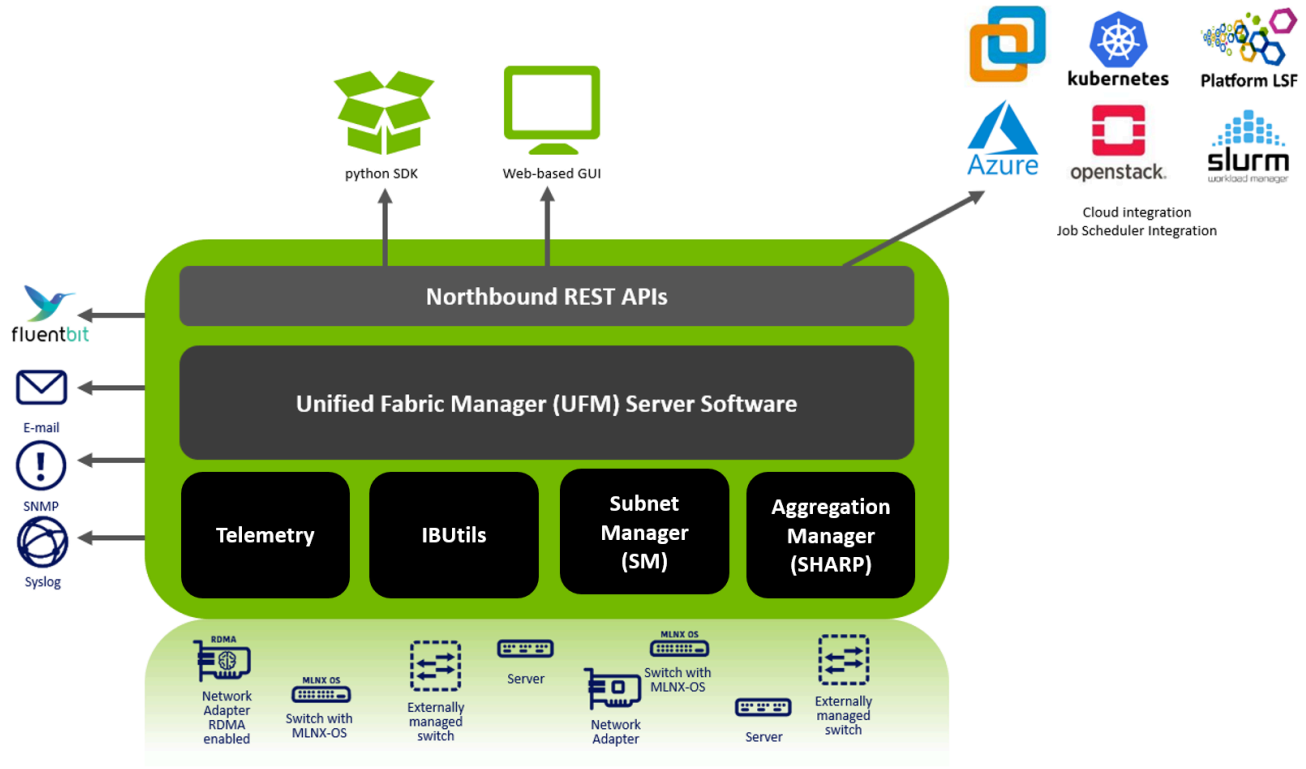
Affected Service	Network	Address / Service / Port	Direction
UFM HA heartbeat	Out-of-band management*	UDP / 694	Bi-directional
UFM HA DRBD	Out-of-band management*	TCP / 8888	Bi-directional

*An IPoIB network can be used for HA, but this is not recommended, since any InfiniBand failure might cause split brain and lack of synchronization between the active and standby servers.

UFM Software Architecture

The following figure shows the UFM high-level software architecture with the main software components and protocols. Only the main logical functional blocks are displayed and do not necessarily correspond to system processes and threads.

UFM High-Level Software Architecture



Graphical User Interface

UFM User Interface is a web application based on JavaScript and Angular JS, which is supported by any Web Browser. The Web application uses a standard REST API provided by the UFM server.

Client Tier API

Third-party clients are managed by the REST API.

Client Tier SDK Tools

Support for UFM's API and a set of tools that enhance UFM functionality and interoperability with third-party applications are provided as part of UFM.

UFM Server

UFM server is a central data repository and management server that manages all physical and logical data. UFM-SDN Appliance receives all data from the Device and Network tiers and invokes Device and Network tier components for management and configuration

tasks. UFM-SDN Appliance uses a database for data persistency. The UFM-SDN Appliance is built on the Python twisted framework.

Subnet Manager

Subnet Manager (SM) is the InfiniBand “Routing Engine”, a key component used for fabric bring-up and routing management. UFM uses the Open Fabric community OpenSM Subnet Manager. UFM uses a plug-in API for runtime management and fabric data export.

NVIDIA Scalable Hierarchical Aggregation and Reduction Protocol (SHARP)™ Aggregation Manager

NVIDIA Scalable Hierarchical Aggregation and Reduction Protocol (SHARP) is a technology that improves the performance of mathematical and machine learning applications by offloading collective operations from the CPU to the switch network.

Aggregation Manager (AM) is a key component of NVIDIA SHARP software, used for NVIDIA SHARP resources management.

For further information about NVIDIA SHARP AM, refer to Appendix – NVIDIA SHARP Integration.

Performance Manager

The UFM Performance Manager component collects performance data from the managed fabric devices and sends the data to the UFM-SDN Appliance for fabric-wide analysis and display of the data.

Device Manager

The Device Manager implements the set of common device management tasks on various devices with varying management interfaces. The Device Manager uses SSH protocol and operates native device CLI (command-line interface) commands.

UFM Switch Agent

UFM Switch Agent is an integrated part of NVIDIA switch software. The agent supports system parameter discovery and device management functionality on switches.

Communication Protocols

UFM uses the following communication protocols:

- Web UI communicates with the UFM server utilizing **Web Services** carried on **REST API**.
- The UFM server communicates with the switch Agent located on managed switches by proprietary **TCP/UDP**-based discovery and monitoring protocol and **SSH**.
- Monitoring data is sent by the switch Agent to UFM server Listener by a proprietary **TCP**-based protocol.

UFM Web UI Overview

The UFM Web User Interface (GUI) lets you access UFM through a web browser, where you can visualize your network and interact with the display using a keyboard and mouse.

The WebUI is designed to be viewed on a display with a minimum resolution of 1920 × 1080 pixels.

- Access the WebUI
- WebUI Layout
- Set User Preferences

Access the WebUI

UFM Web UI Supported Browsers

UFM Web UI is supported on all the following web browsers: Internet Explorer, Firefox, Chrome and Opera.

For optimal UFM Web UI performance, make sure you are using the latest version available of Google Chrome.

Launching UFM Web UI Session

Before accessing the UFM Web UI:

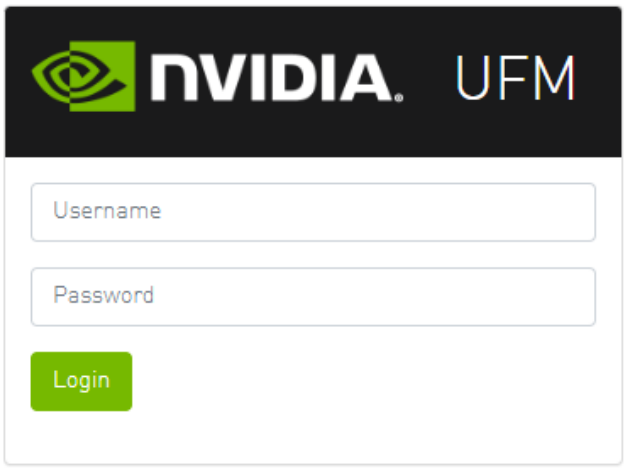
- If required, you can change the configuration of the connection (port and protocol) between the UFM server and the APACHE server in the file *gv.cfg*:
    - *ws_protocol* = http or https
        - Setting the parameter *ws_protocol* to *http* allows unsecured access
        - Setting the parameter *ws_protocol* to *https* denies unsecured access.
    - *ws_port* = port number

To launch a UFM Web UI session, do the following:

1. Launch the Web UI by entering the following URL in your browser:

   http://<UFM_server_IP>/ufm

   https://<UFM_server_IP>/ufm



2. In the Login page, enter your **User Name** and your predefined user **Password** and click **Login**.

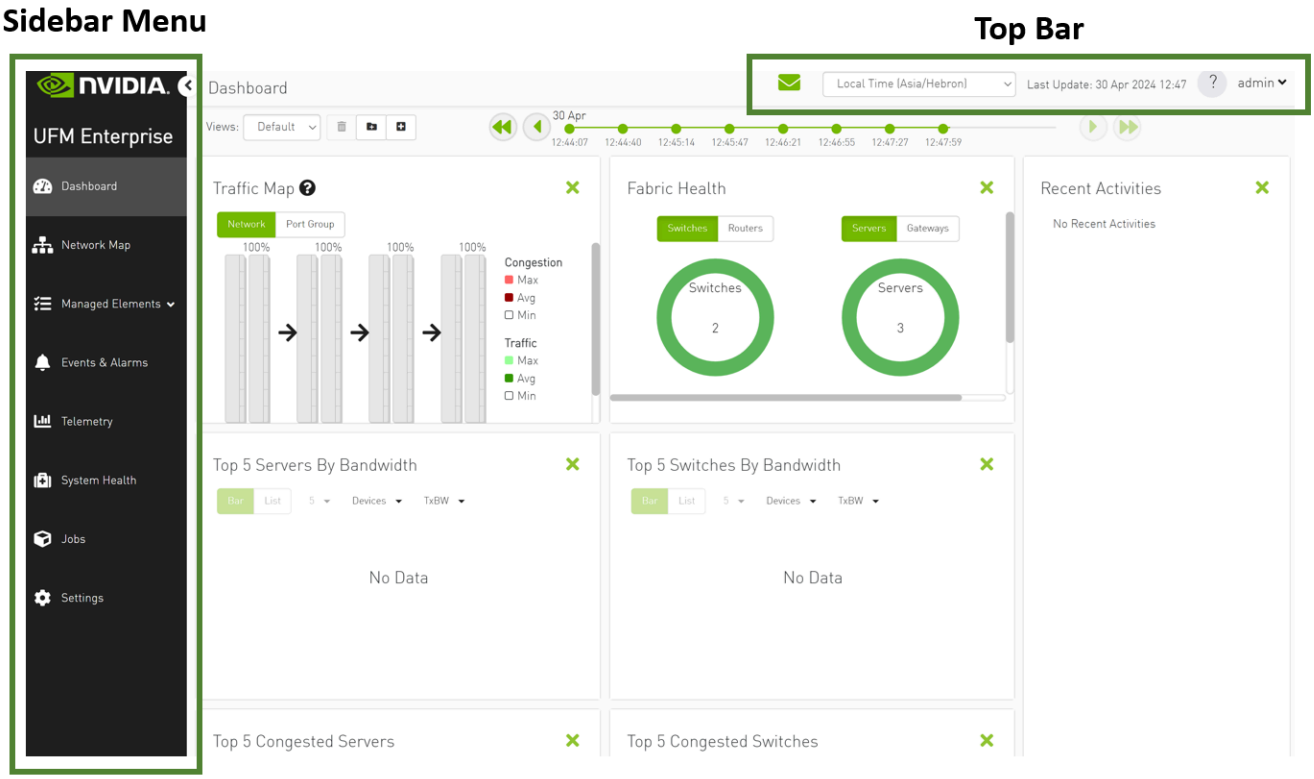
Once you have entered your user name and password, the main window shows the UFM Dashboard. For more information, see the Fabric Dashboard.

WebUI Layout

The UFM WebUI contains two main areas:

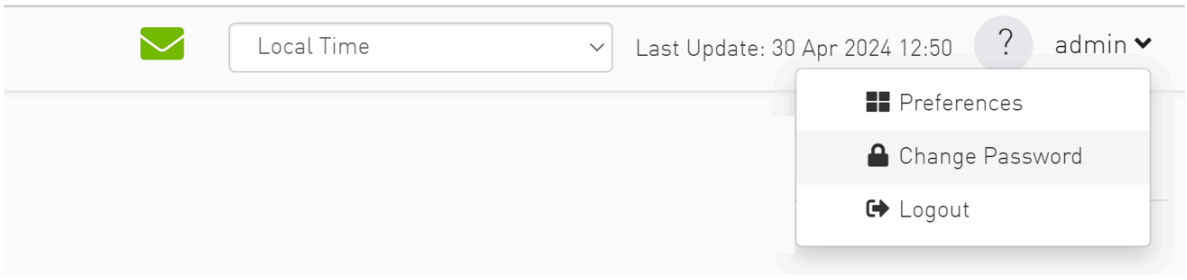
1. Top Bar - Contains local time zone and user information on the top right side of the screen.

2. Sidebar Menu - Contains a taskbar accessible from a sidebar menu on the left side of the screen. For more information on each tab, refer to UFM Web UI.

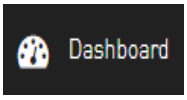
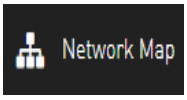
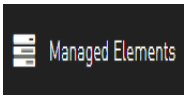
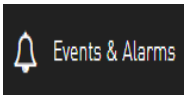
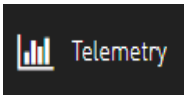
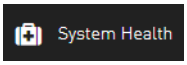
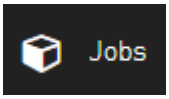
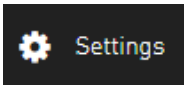


Top Bar

Each user can customize the UFM display, time zone, and date format, change their account password, and manage their preferences. For details, refer to Set User Preferences.



Sidebar Menu

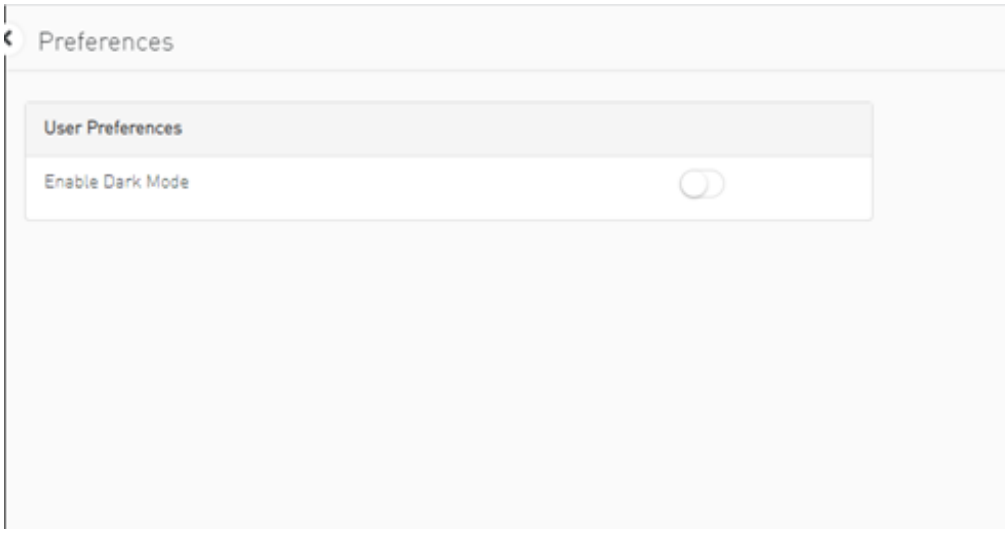
Tab Icon	Description
	Provides a summary view of the fabric status.
	Provides a hierarchical topology view of the fabric.
	Provides information on all fabric devices. This information is presented in a table format.
	Provides information on all logical servers. This information is presented in a table format.
	Provides information on the events & alarms generated by the system.
	Enables establishing monitoring sessions on devices or ports.
	Enables running and viewing fabric reports, UFM reports, and system logs. You can also back up UFM configuration files.
	Provides information on all jobs created, as a result of UFM actions.
	Enables configuring UFM server and UFM fabric settings, including events policy, device access, network management, subnet manager, and user management

Set User Preferences

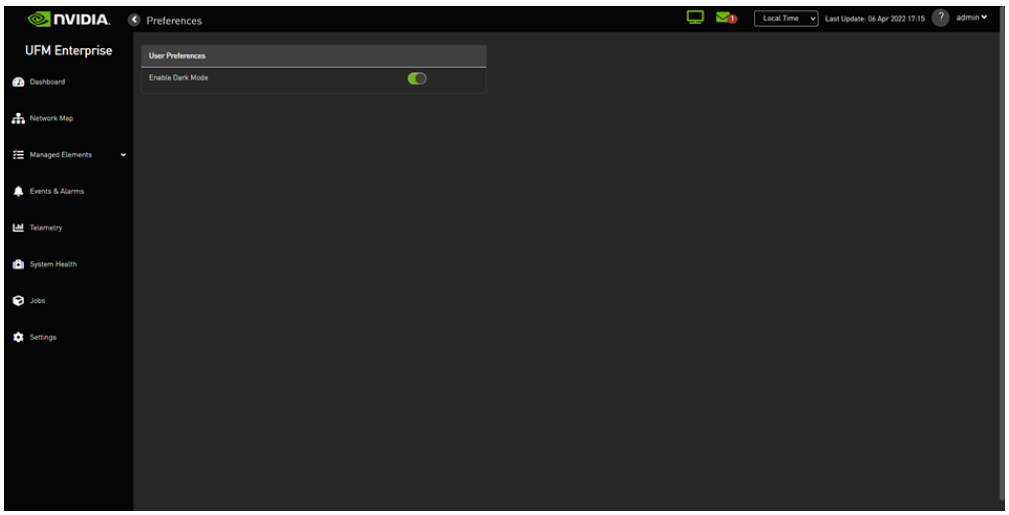
This section describes how to customize your UFM display settings and change your password,

Dark/Light Theme

1. Select **Preferences.**



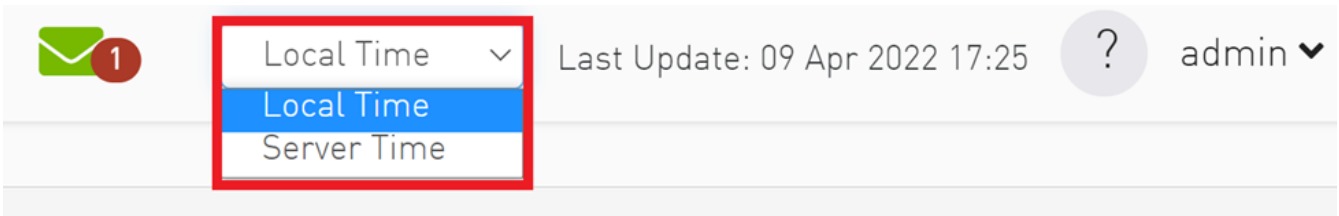
2. In **User Preferences**, enable dark mode for UFM presentation in a dark theme. The following figure shows the dark theme:

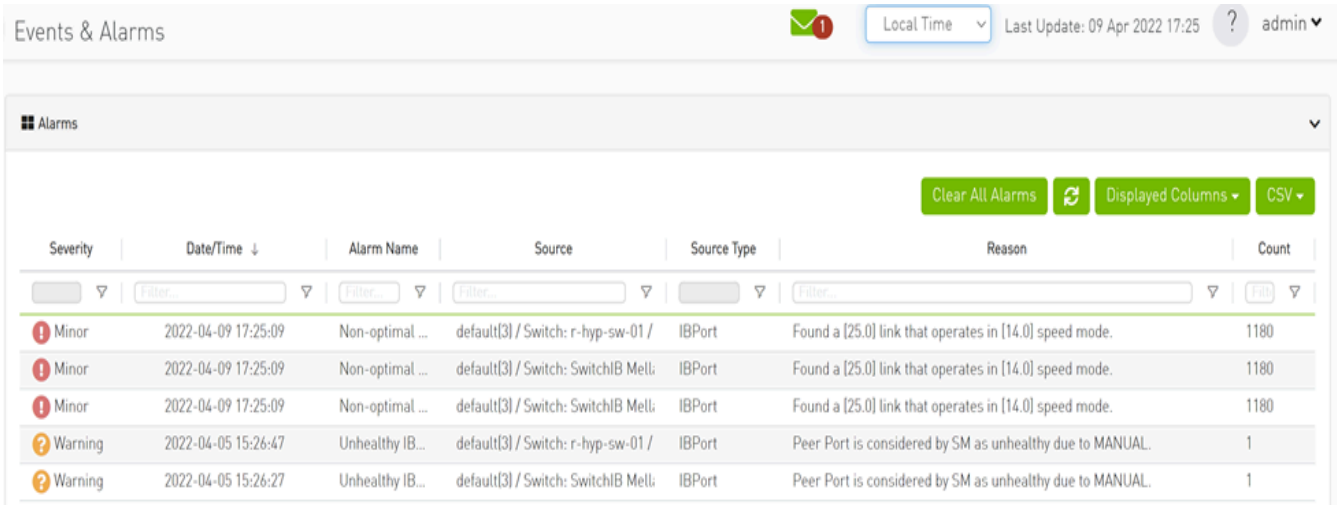


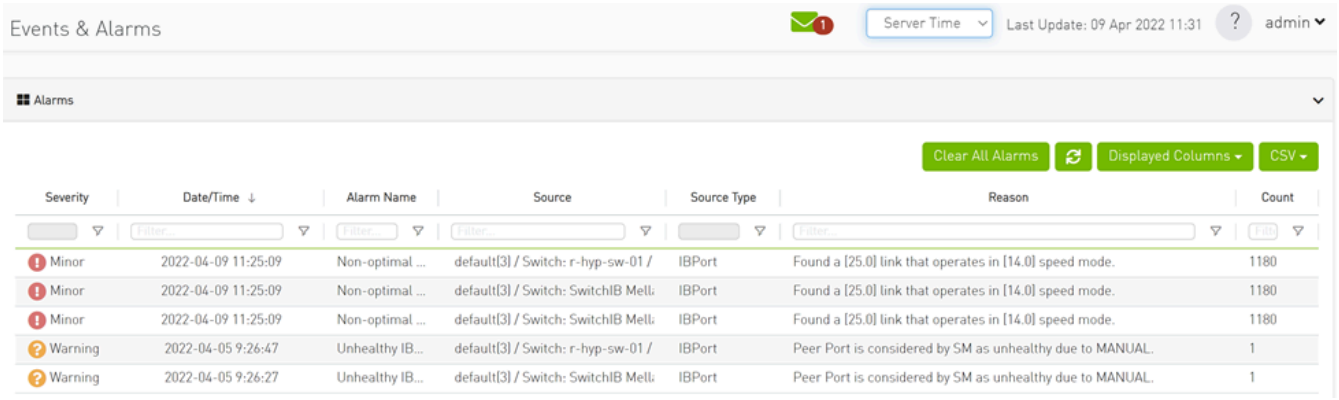
Time Zone Converter

Allows you to unify all times in UFM like events and alarms, ibdiagnet, telemetry and logs. You can switch between local and machine time.

In the status bar drop-down menu, switch between local and server/machine time.





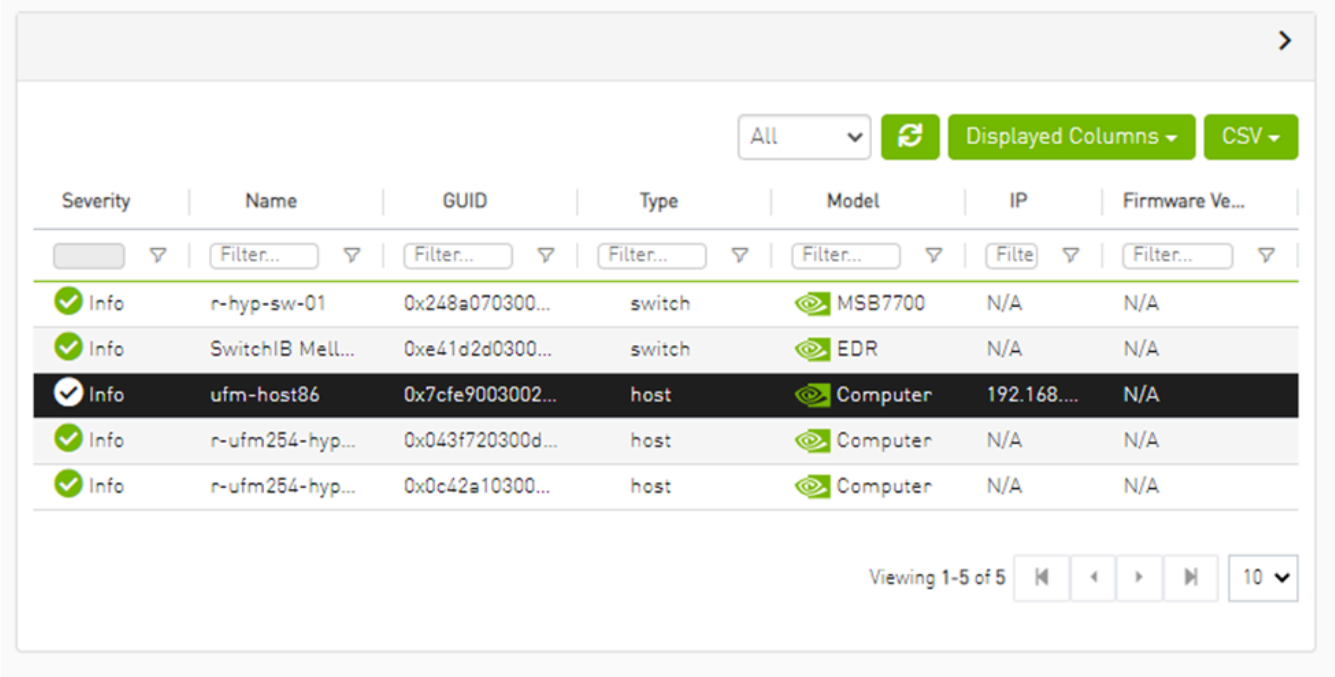


Note

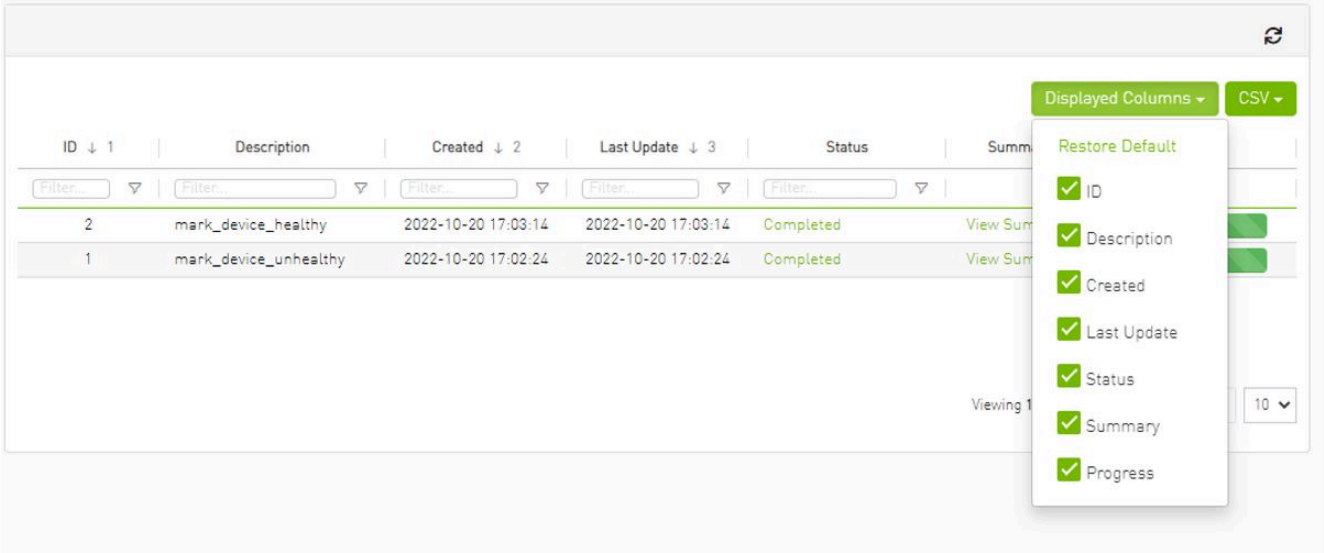
In the screenshots, the difference between Server Time and Local Time is 6 hours.

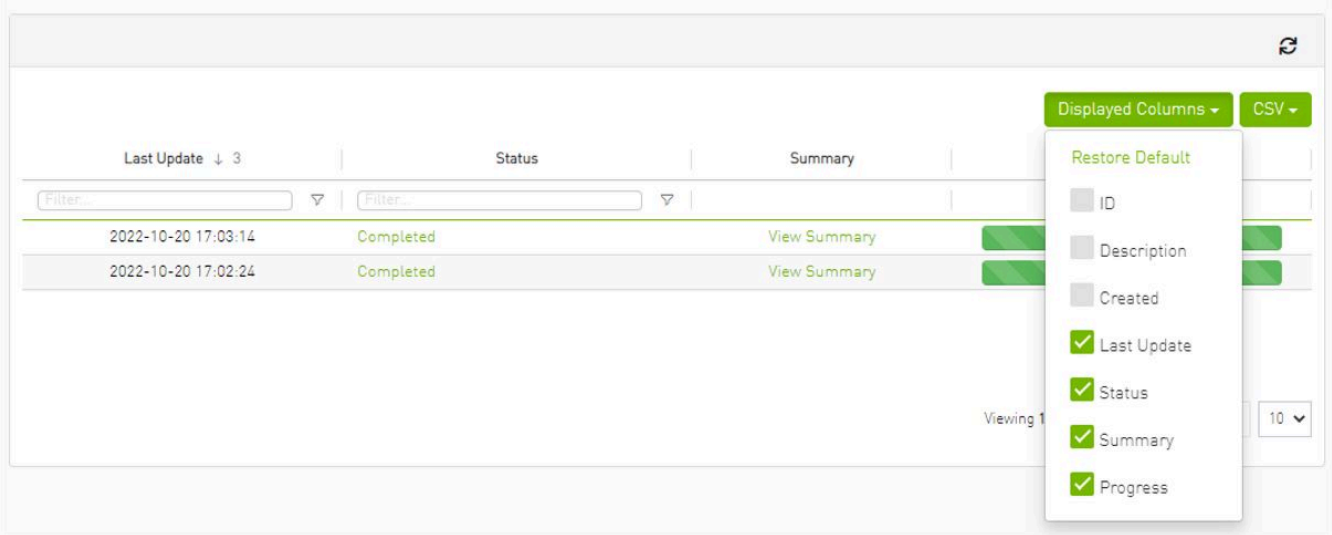
Table Enhancements

Look and Feel Improvements



Displayed Columns





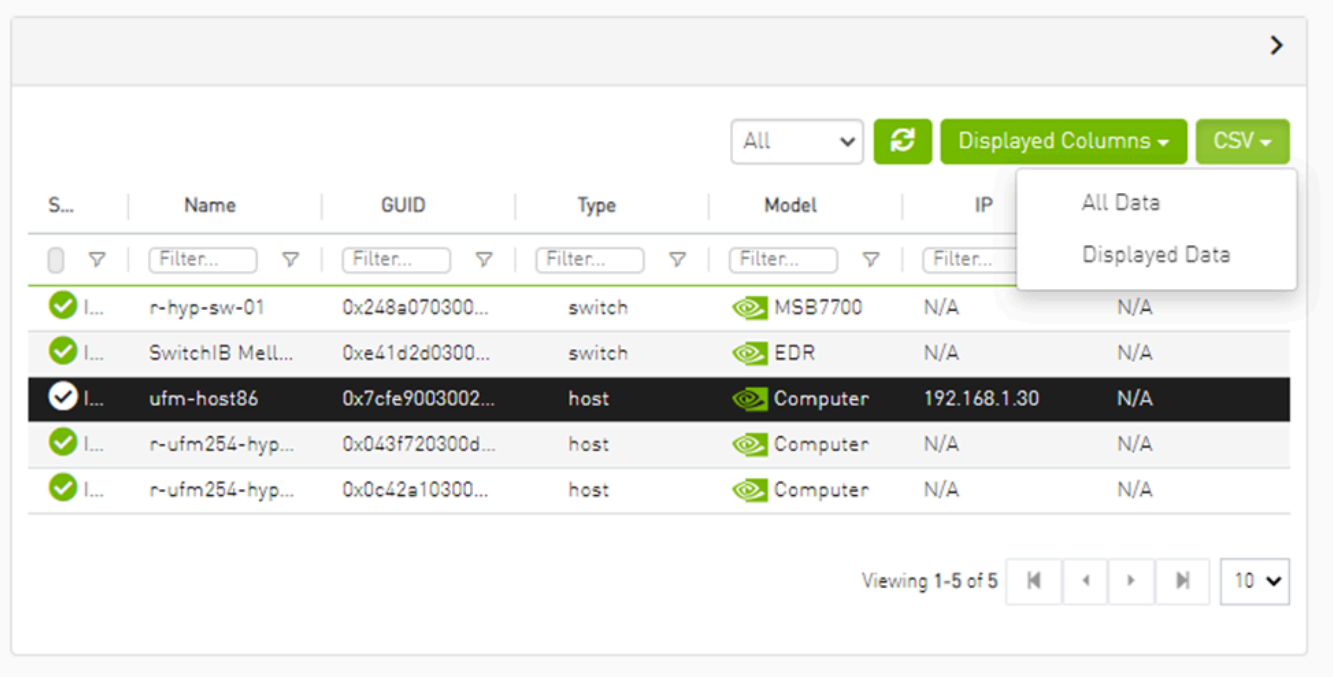
Note

Displayed columns of all tables are persistent per user, with the option to restore defaults.

Export All Data as CSV

There are two options for exporting as CSV

- All Data: all data returned from server.
- Displayed Data: only displayed rows.


All
Displayed Columns
CSV
All Data
Displayed Data
S...
Name
GUID
Type
Model
IP
Filter...
I...
r-hyp-sw-01
0x248a070300...
switch
MSB7700
N/A
N/A
I...
SwitchIB Mell...
0xe41d2d0300...
switch
EDR
N/A
N/A
I...
ufm-host86
0x7cfe9003002...
host
Computer
192.168.1.30
N/A
I...
r-ufm254-hyp...
0x043f720300d...
host
Computer
N/A
N/A
I...
r-ufm254-hyp...
0x0c42a10300...
host
Computer
N/A
N/A
Viewing 1-5 of 5
10

UFM Installation and Configuration

UFM® software includes Server and Agent components. UFM Server software should be installed on a central management node. For optimal performance, and to minimize interference with other applications, it is recommended to use a dedicated server for UFM. The UFM Agent is an optional component and should be installed on fabric nodes. The UFM Agent should not be installed on the Management server.

The following sections provide step-by-step instructions for installing and activating the license file, installing the UFM server software, and installing the UFM Agent.

Prerequisites for UFM Server Software Installation

Please refer to Installation Notes for information on system prerequisites.

UFM Installation Steps

To install the UFM software:

1. Download UFM Software and License File
2. Install the UFM Server Software and Activate the Software License
3. Perform initial configuration
4. Run the UFM server software

UFM Installation Steps

- Downloading UFM Software and License File
- Installing UFM Server Software
- UFM Infra Installation

Downloading UFM Software and License File

Before you obtain a license for the UFM® software, prepare a list of servers with the MAC address of each server on which you plan to install the UFM software. These MAC addresses are requested during the licensing procedure.

Obtaining License

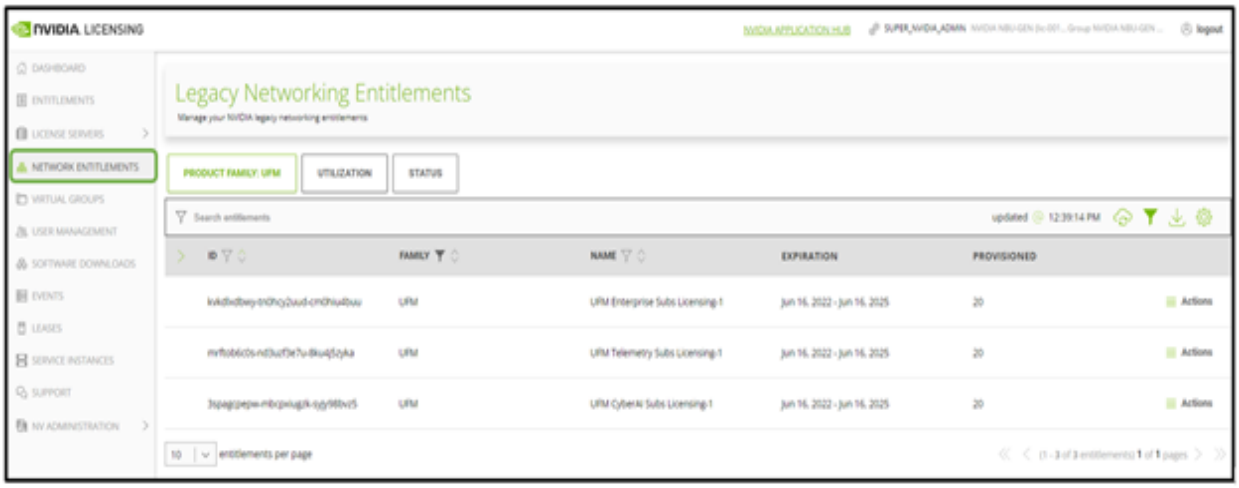
UFM is licensed per managed device according to the UFM license agreement.

When you purchase UFM, you will receive an email with instructions on obtaining your product license. A valid UFM license is a prerequisite for the installation and operation of UFM.

to UFM licenses are per managed HCA and are aggregative. If you install an additional license, the system adds the previous HCA number and the new HCA number and manages the sum of the HCAs. For example, if you install a license for 10 managed HCAs and an additional license for 15 HCAs, UFM will be licensed for up to 25 managed HCAs.

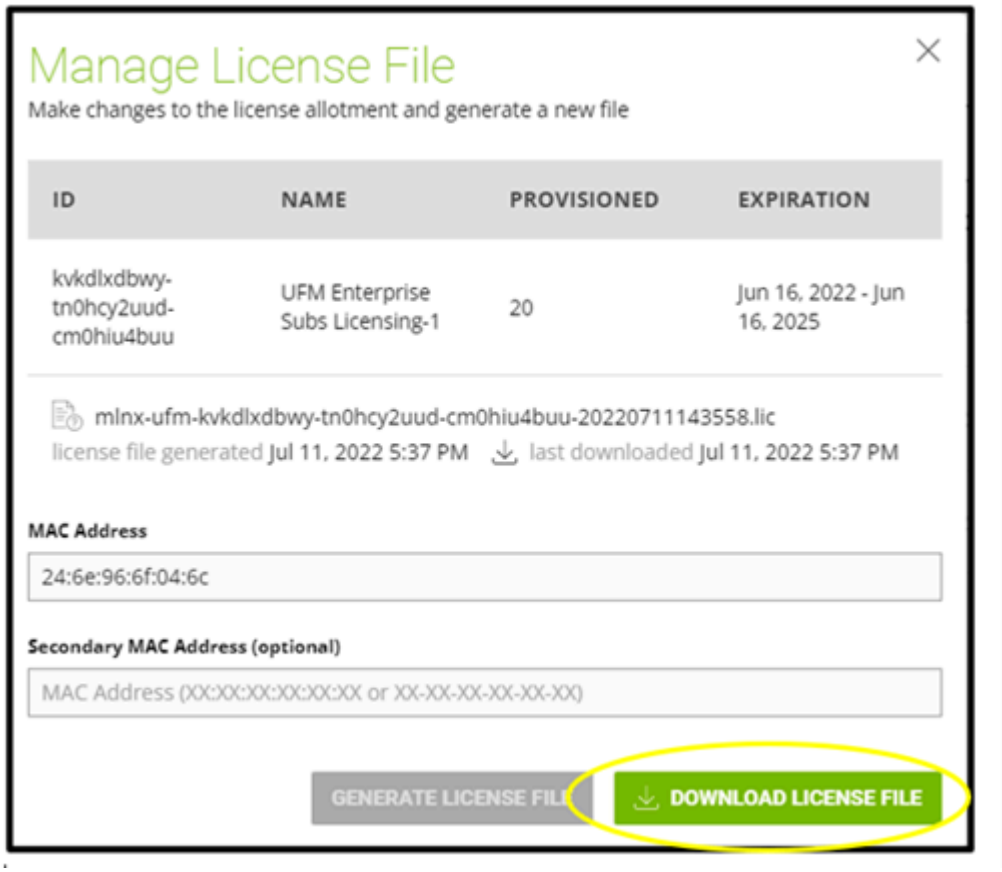
To obtain the license:

1. Go to NVIDIA's Licensing and Download Portal and log in as specified in the licensing email you received.
    - If you did not receive your NVIDIA Licensing and Download Portal login information, contact your product reseller.
2. If you purchased UFM directly from NVIDIA and you did not receive the login information, contact **enterprisesupport@nvidia.com.** Click on the Network Entitlements tab. You'll see a list with the serial licenses of all your software products and software product license information and status.



3. Select the license you want to activate and click on the “Actions” button.

4. In the MAC Address field, enter the MAC address of the delegated license-registered host. If applicable, in the HA MAC Address field, enter your High Availability (HA) server MAC address. If you have more than one NIC installed on a UFM Server, use any of the MAC addresses.



5. Click on Generate License File to create the license key file for the software.

6. Click on Download License File and save it on your local computer.

If you replace your NIC or UFM server, repeat the process of generating the license to set new MAC addresses. You can only regenerate a license two times. To regenerate the license after that, contact NVIDIA Sales Administration at **enterprisesupport@nvidia.com**.

Downloading UFM Software

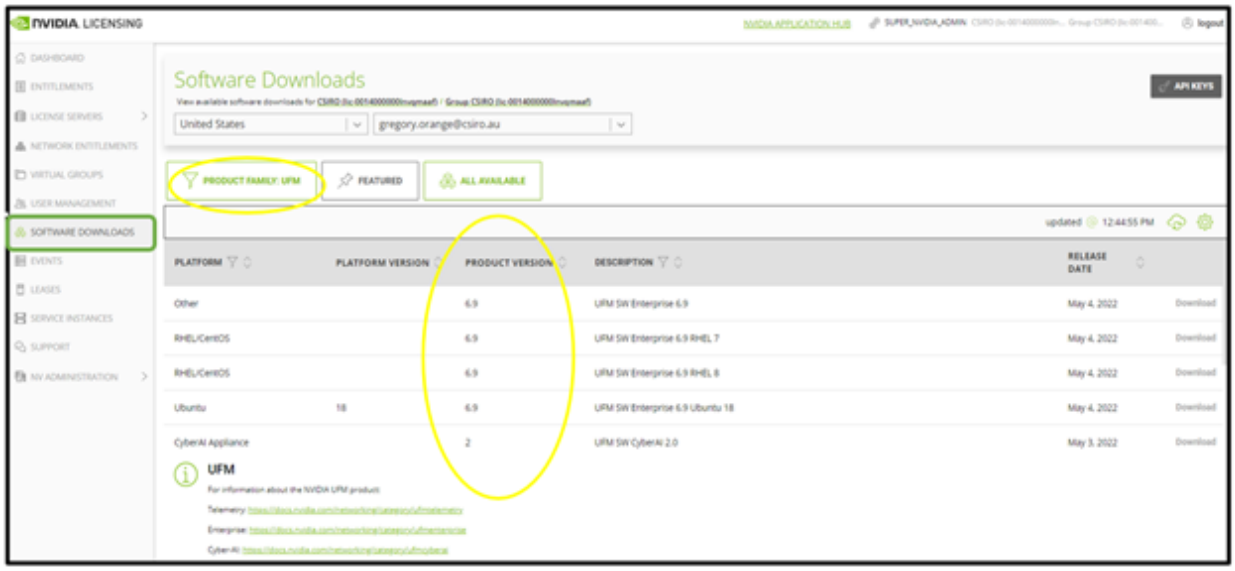
> **Note**
>
> Due to internal packaging incompatibility, this release has two different packages for each of the supported distributions:
>
> - One for UFM deployments over MLNX_OFED 5.X (or newer)
>
> Please make sure to use the UFM installation package compatible to your setup.

This software download process applies to software updates and first-time installation.

If you own the UFM Media Kit and this is your first-time installation, skip this section.

To download the UFM software:

1. Click on Software Downloads, filter the product family to UFM, and select the relevant version of the software. Click on Download.



2. Save the file on your local drive.

3. Click Close.

Installing UFM Server Software

The default UFM® installation directory is */opt/ufm*.

For instructions on installing the UFM server software, please refer to following instructions per desired installation mode.

- Installing UFM Server on Bare Metal Server
    - Installing UFM on Bare Metal Server- Standalone Mode
    - Installing UFM on Bare Metal Server - High Availability Mode
- Installing UFM Docker Container Mode
    - Installing UFM on Docker Container - Standalone Mode
    - Installing UFM on Docker Container - High Availability Mode

The following processes might be interrupted during the installation process:

- httpd (Apache2 in Ubuntu)

- dhcpd

> **Note**
>
> To install UFM over static IPv4 configuration (instead of DHCP) please refer to Configuring UFM Over Static IPv4 Address before installation.

After installation:

1. Activate the software license
2. Perform initial configuration

> **Note**
>
> Before you run UFM, ensure that all ports used by the UFM server for internal and external communication are open and available. For the list of ports, see Appendix – Used Ports.

Prerequisites for UFM Server Software Bare Metal Installation

Verify that a supported version of Linux is installed on your machine. For details, see UFM System Requirements.

The following table lists the packages that must be installed on your machine (according to the system OS) before you install the UFM server software.

Prerequisites for UFM Server Software Installation

Verify that a supported version of Linux is installed on your machine. For details, see UFM System Requirements.

The following table lists the packages that must be installed on your machine (according to the system OS) before you install the UFM server software.

RedHat 8	RedHat 9	Ubuntu 20.04	Ubuntu 22.04
acl	acl	acl	acl
apr-util-openssl	apr-util-openssl	apache2	apache2
bc	bc	bc	bc
dos2unix	dos2unix	chrpath	chrpath
gnutls	gnutls	cron	cron
httpd	httpd	dos2unix	dos2unix
iptables	iptables-nft	gawk	gawk
jansson	jansson	lftp	lftp
lftp	lftp	libcurl4	libcurl4
libnsl	libnsl	logrotate	logrotate
libxml2	libxml2	python3	python3
libxslt	libxslt	qperf	qperf
mod_session	mod_session	rsync	rsync
mod_ssl	mod_ssl	snmpd	snmpd
net-snmp	net-snmp	sqlite3	sqlite3
net-snmp-libs	net-snmp-libs	sshpass	sshpass
net-snmp-utils	net-snmp-utils	ssl-cert	ssl-cert
net-tools	net-tools	sudo	sudo
php	php	telnet	telnet
psmisc	psmisc	zip	zip
python36	python3		
qperf	qperf		
rsync	rsync		
sqlite	sqlite		
sshpass	sshpass		
sudo	sudo		

RedHat 8	RedHat 9	Ubuntu 20.04	Ubuntu 22.04
telnet	telnet		
zip	zip		

Note

On some Ubuntu OSs, Docker is installed via SNAP, which might lead to errors when trying to use UFM Plugins.

To solve this issue, perform the following:

1. Remove Docker installed via SNAP, run:

```
snap remove --purge docker
```

2. Update the local package index, run :

```
apt update
```

3. Install native Docker, run:

```
apt install-y docker.io
```

In addition, ensure the following before you begin installation:

- The computer hostname is not defined as 127.0.0.1 and localhost is defined as 127.0.0.1.
- The hostname must NOT appear on the loopback address line. An example of the loopback address is: 127.0.0.1 localhost.localdomain localhost.

- Disable the firewall service (`/etc/init.d/iptables stop`), or ensure that the required ports are open (see the prerequisite script, refer to Used Ports).
- Disable SELinux.
- If more than one fabric is managed by different UFM instances, set up different management network spaces for each fabric (not the same LAN).
- Uninstall any previously installed Subnet Manager from the UFM server machine.
- NVIDIA Host Infiniband Networking packages (MLNX_OFED 5.x version or NVIDIA DOCA 2.9.0 or greater) is installed prior to installing UFM.
- As of UFM v.6.12.0, it is **NOT mandatory** to configure the IPoIB fabric interface with an IP address.

  In cases where the IP is configured, it is **mandatory** that the IP is permanently configured and that it starts automatically upon server reboot (the IPoIB fabric interface should be active even if the network is down).

  > **Note**
  >
  > The user can set a persistent IP address using Netplan (mainly for Ubuntu systems) or modifying the interface network script (RedHat systems).

- The default NVIDIA Host InfiniBand Networking package provided by MLNX_OFED installation includes OpenSM. Remove MLNX_OFED OpenSM before UFM installation like the following examples:

  RedHat:

  ```
  rpm -e opensm-3.3.9.MLNX_20111006_e52d5fc-0.1
  ```

  Ubuntu:

```
apt purge opensm
```

- If using NVIDIA Host InfiniBand Networking packages provided by DOCA package, install the drivers and UFM dedicated package only.

  RedHat:

```
yum install doca-ufm doca-kernel
```

  Ubuntu:

```
apt update
apt install doca-ufm doca-kernel
```

  By default, ib0 and eth0 are configured as primary access points for the UFM management. If different management and/or InfiniBand interfaces (including bond interfaces) are used as the primary access points, you should modify the configuration file by running the script */opt/ufm/scripts/change_fabric_config.sh* as described in the section Configuring General Settings in gv.cfg.

  Change the UFM Agent interface to the Ethernet and/or IPoIB interfaces used for communication with UFM Agent:

```
ufma_interfaces = ib0,eth0
```

Additional Prerequisites for UFM High Availability Installation

- Reliable and high-capacity out-of-band IP connectivity between the UFM Primary and Secondary servers (1 Gb Ethernet is recommended). This connectivity is used for DRBD synchronization.

- Format two identical servers with dedicated disk partitions for UFM replication. Since the UFM configuration file is replicated to the standby server, both master and standby servers must have the same interfaces.

- Allocate exactly the same size partition on both servers (master and slave) for the replicated data. See UFM Server Requirements for the recommended partition size.

  Partitions should not be mounted and must be zeroed (the file system should not be installed on the partitions). For disk partitioning, see the Linux user manual (man fdisk).

- We recommend establishing a passwordless SSH (via /root/.ssh/authorized_keys file) between the two servers before the installation.

- In fabrics consisting of multiple tiers of switches, it is recommended that the management ports (ib0) of the primary and secondary UFM server be connected to different fabric switches on the same tier (the outermost edge in CLOS 5 designs).

  This is because by default, UFM manages the IB fabric via ib0, port 1 of the HCA. Failure or disconnect of ib0, the IB management port, causes a failure condition in UFM resulting in HA failover.

  When the management ports (ib0) of the primary and secondary UFM server are connected to the same switch, a failure of this switch will result in a disconnect of both UFMs from the fabric, and therefore UFM will not be able to manage the fabric.

Note

Subnet Manager is running over the native InfiniBand layer, therefore bonding the IpoIB interfaces will not provide high availability. For additional information, please refer to section UFM Failover to Another Port.

The UFM installation includes the InfiniBand Performance Management module (IBPM). This module is responsible for reporting performance information back to UFM and upper layer applications. When available, this process is offloaded to the non-management port (default ib1) of the UFM server. Failure or disconnect of the non-management port (ib1) on the primary UFM server will not cause UFM to failover. By default, the UFM Health Monitoring process is configured to try to restart the IBPM. For more information, see UFM Health Configuration in the UFM User Manual.

Installing UFM Server on Bare Metal Server

Installing UFM server on Bare Metal server can be done with the following modes:

- Installing UFM on Bare Metal Server - High Availability Mode
- Installing UFM on Bare Metal Server- Standalone Mode

Installing UFM on Bare Metal Server - High Availability Mode

Before installing UFM server software in High-Availability mode, ensure that the Additional Prerequisites for UFM High Availability Installation are met.

The UFM High-Availability configuration requires dual-link connectivity based on two separate interfaces between the two UFM HA nodes. This configuration comprises of a primary link that is exclusively reserved for DRBD operations and a secondary link

designated for backup purposes. Crucially, it is imperative that communication between the servers is established in a bidirectional manner across both interfaces and validated through user-initiated testing, such as a 'ping' command or other suitable alternatives before HA configuration can be implemented. In cases where only one link is available among the two UFM HA nodes/servers, manually configure UFM with a single link. Refer to Configure HA without SSH Trust (Single Link Configuration).

> **Note**
>
> UFM HA package requires a dedicated partition with the same name for DRBD on both servers. This guide uses `/dev/sda5` as an example.

1. On both servers, Install UFM Enterprise in Stand Alone (SA) mode.

   > **Note**
   >
   > Do not start UFM service.

2. Install the latest pcs and drbd-utils and resource-agents-extra drivers on both servers.

   For Ubuntu:

   ```
   apt install pcs pacemaker drbd-utils resource-agents-extra
   ```

   For CentOS/Red Hat:

```
yum install pcs pacemaker drbd84-utils kmod-drbd84 resource-agents-extra
```

OR

```
yum install pcs pacemaker drbd90-utils kmod-drbd90 resource-agents-extra
```

3. Download UFM-HA latest package from using this command:

```
wget http://www.mellanox.com/downloads/UFM/ufm_ha_6.1.1-2.tgz
```

> **Note**
>
> For more information on the UFM-HA package and all installation and configuration options, please refer to UFM High-Availability User Guide.

4. Extract the downloaded UFM-HA package on both servers under `/tmp/.`

5. Go to the directory you extracted /tmp/ufm_ha_XXX and run the installation script. For example, if your DRBD partition is `/dev/sda5` run:

```
./install.sh -l /opt/ufm/files/ -d /dev/sda5 -p enterprise
```

6. Configure the HA cluster. There are the three methods:

- Configure HA with SSH Trust (Dual Link Configuration) - Requires passwordless SSH connection between the servers.
- Configure HA without SSH Trust (Dual Link Configuration) - Does not require passwordless SSH connection between the servers, but asks you to run configuration commands on both servers.
- Configure HA without SSH Trust (Single Link Configuration) - Can be used in cases where only one link is available among the two UFM HA nodes/servers.

Configure HA with SSH Trust (Dual Link Configuration)

1.

    1. On the **master server only**, configure the HA nodes. To do so, from /tmp, run the `configure_ha_nodes.sh` command as shown in the below example

```
configure_ha_nodes.sh \
--cluster-password 12345678 \
--master-primary-ip 10.10.10.1 \
--standby-primary-ip 10.10.10.2 \
--master-secondary-ip 192.168.10.1 \
--standby-secondary -ip 192.168.10.2 \
--no-vip
```

> **Note**
>
> The script `configure_ha_nodes.sh` is is located under /usr/local/bin/, therefore, by default, you do not need to use the full path to run it.

Note

The `--cluster-password` must be at least 8 characters long.

Note

To set up a Virtual IP for UFM and gain access to UFM through this IP, regardless of which server is running UFM, you may employ the `--no-vip` OR `--virtual-ip` command and provide an IP address as an argument. This can be achieved by navigating to https://<Virtual-IP>/ufm on your web browser.

Note

When using back-to-back ports with local IP addresses for HA sync interfaces, ensure that you add your IP addresses and hostnames to the `/etc/hosts` file. This is needed to allow the HA configuration to resolve hostnames correctly based on the IP addresses you are using.

Note

`configure_ha_nodes.sh` requires SSH connection to the standby server. If SSH trust is not configured, then you are prompted to enter the SSH password of the standby server during configuration runtime

2. Depending on the size of your partition, wait for the configuration process to complete and DRBD sync to finish.

Configure HA without SSH Trust (Dual Link Configuration)

If you cannot establish an SSH trust between your HA servers, you can use **ufm_ha_cluster directly to configure HA. To configure HA, follow the below instructions:**

> **Note**
>
> Please change the variables in the commands below based on your setup.

1.

    1. [On Standby Server] Run the following command to configure Standby Server:

```
ufm_ha_cluster config -r standby \
--local-primary-ip 10.10.50.1 \
--peer-primary-ip 10.10.50.2 \
--local-secondary-ip 192.168.10.1 \
--peer-secondary-ip 192.168.10.2 \
--hacluster-pwd 123456789 \
--no-vip
```

    2. [On Master Server] Run the following command to configure Master Server:

```
ufm_ha_cluster config -r master --local-primary-ip
10.10.50.1 \
--peer-primary-ip 10.10.50.2 \
--local-secondary-ip 192.168.10.1 \
--peer-secondary-ip 192.168.10.2 \
--hacluster-pwd 123456789 \
--no-vip
```

You must wait until after configuration for DRBD sync to finish, depending on the size of your partition. To check the DRBD sync status, run:

```
ufm_ha_cluster status
```

Configure HA without SSH Trust (Single Link Configuration)

> ⚠ **Warning**
>
> This is not the recommended configuration and, in case of network failure, it might cause HA cluster split brain.

If you cannot establish an SSH trust between your HA servers, you can use **ufm_ha_cluster directly to configure HA. To configure HA, follow the below instructions:**

> ⓘ **Note**
>
> Please change the variables in the commands below based on your setup.

1.

    1. [On Standby Server] Run the following command to configure Standby Server:

```
ufm_ha_cluster config \
-r standby \
-e 10.212.145.5 \
-l 10.212.145.6 \
--enable-single-link
```

    2. [On Master Server] Run the following command to configure Master Server:

```
ufm_ha_cluster config -r master \
-e 10.212.145.6 \
-l 10.212.145.5 \
-i 10.212.145.50 \
--enable-single-link
```

    You must wait until after configuration for DRBD sync to finish, depending on the size of your partition. To check the DRBD sync status, run:

```
ufm_ha_cluster status
```

Starting HA Cluster

- To start UFM HA cluster:

```
ufm_ha_cluster start
```

- To check UFM HA cluster status:

```
ufm_ha_cluster status
```

Stopping UFM HA cluster:

```
ufm_ha_cluster stop
```

> **Note**
>
> For complete details on high availability, refer to NVIDIA UFM High-Availability User Guide.

Installing UFM on Bare Metal Server-Standalone Mode

To install the UFM server software as a standalone for InfiniBand:

1. Create a temporary directory (for example */tmp/ufm*).
2. Open the UFM software zip file that you downloaded. The zip file contains the following installation files:
   - RedHat 9/CentOS 9/OEL 9: ufm-X.XX.X-X.el9.x86_64.tgz
   - RedHat 8/Centos 8: ufm-X.X-XXX.el8.x86_64.tgz
   - Ubuntu 18.04: ufm-X.X-XXX.Ubuntu18.x86_64.tgz
   - Ubuntu 20.04: ufm-X.X-XXX.Ubuntu20.x86_64.tgz
   - Ubuntu 22.04: ufm-X.X-XXX.Ubuntu22.x86_64.tgz

3. Extract the installation file for your system's OS to the temporary directory that you created.

4. From within the temporary directory, run the following command as root:

```
./install.sh
```

> **Note**
>
> Running with the option "-o ib" is no longer required. For automatic installation, use the -q flag.
>
> For “quiet” installation -q flag can be added (automatically answer yes for each question the installer asks).

> **Note**
>
> Export MULTISUBNET_CONSUMER=1 environment variable before running the installation script to install the UFM server in Multisubnet Consumer mode.

The UFM software is installed. You can now remove the temporary directory.

Installing UFM Docker Container Mode

General Prerequisites

- UFM dedicated NVIDIA Host InfiniBand Networking packages provided by DOCA-Host must be installed on the server that will run UFM Docker.

  RedHat:

```
yum install doca-ufm doca-kernel
```

Ubuntu:

```
apt install doca-ufm doca-kernel
```

- For UFM to work, you must have an InfiniBand port configured with an IP address and in "up" state.

> **Note**
>
> For InfiniBand support, please refer to DOCA-Host documentation.

- If the hosting server is equipped with XDR HCAs (ConnectX-8/ConnectX-9 SuperNICs), SMI devices for the InfiniBand interfaces intended to be used as fabric interfaces in UFM (for OpenSM) must be created if they do not already exist.

  Fabric interfaces are defined in the **Server** section of the `gv.cfg` file.

```
[Server]
# SM Management interface
fabric_interface = ib0
# disabled (default) | enabled (configure OpenSM with
multiple GUIDs) | ha_enabled (configure multiport SM with
high availability).
multi_port_sm = disabled
# When enabling multi_port_sm, specify here the additional
fabric interfaces for OpenSM conf.
# Example: ib1,ib2,ib5 (OpenSM will support the first 8 GUIDs
where first GUID will
# be extracted from the fabric_interface, and remaining GUIDs
from additional_fabric_interfaces.
additional_fabric_interfaces =
```

Prerequisites for Creating SMI Devices

1.

    1. Install the `mlnx-iproute2` package (if it is not already installed):

        ```
        apt install mlnx-iproute2
        ```

    2. Retrieve the list of active InfiniBand (IB) devices based on the defined fabric interfaces.

        Use the following bash command to obtain the device names:

        `ibdev2netdev | grep ib0 | cut -d" " -f1`

        For the interface name `ib0`, the corresponding device name will be `mlx5_0`.

    3. Create SMI devices for the active IB interfaces using the following command:

```
/opt/mellanox/iproute2/sbin/rdma dev add
[interface_name] type SMI parent [device name]

for device mlx5_0 smi device name will be smi1 (should
not be smi0. Just add one to device name extension
(mlx5_0)
Example:
/opt/mellanox/iproute2/sbin/rdma dev add smi1 type SMI
parent mlx5_0
```

- Make sure to stop the following services before running UFM Docker container, as it utilizes the same default ports that they do: Pacemaker, httpd, OpenSM, and Carbon.
- If firewall is running on the host, please make sure to add an allow rule for UFM used ports (listed below):

> **Note**
>
> If the default ports used by UFM are changed in UFM configuration files, make sure to open the modified ports on the host firewall.

  - 80 (TCP) and 443 (TPC) are used by WS clients (Apache Web Server)
  - 8000 (UDP) is used by the UFM server to listen for REST API requests (redirected by Apache web server)
  - 6306 (UDP) is used for multicast request communication with the latest UFM Agents
  - 8005 (UDP) is used as a UFM monitoring listening port
  - 8888 (TCP) is used by DRBD to communicate between the UFM Primary and Standby servers

- 2022 (TCP) is used for SSH

Prerequisites for Upgrading UFM Docker Container

- Supported versions for upgrade are UFM v.6.10.0 and above.
- UFM files directory from previous container version mounted on the host.

Step 1: Loading UFM Docker Image

To load the UFM docker image, pull the latest image from docker hub:

```
docker pull mellanox/ufm-enterprise:latest
```

Note

You can see full usage screen for ufm-installation by running the container with `-h` or `-help` flag:

```
docker run --rm mellanox/ufm-enterprise-
installer:latest -h
```

If an Internet connection is not available, perform the following:

- Copy the UFM image to your machine.
- Load the image from the file using this command:

```
docker image load -i <image-path>
```

Step 2: Installing UFM Docker

Installation Command Usage

```
docker run -it --name=ufm_installer --rm \
-v /var/run/docker.sock:/var/run/docker.sock \
-v /etc/systemd/system/:/etc/systemd_files/ \
-v /opt/ufm/files/:/installation/ufm_files/ \
-v [LICENSE_DIRECTORY]:/installation/ufm_licenses/ \
mellanox/ufm-enterprise:latest \
--install [OPTIONS]
```

Modify the variables in the installation command as follows:

- `[UFM_LICENSES_DIR]`: UFM license file or files location.

> **Note**
>
> Example: If your license file or files are located under `/downloads/ufm_license_files/` then you must set this volume to be `-v /downloads/ufm_license_files/:/installation/ufm_licenses/`

- `[OPTIONS]`: UFM installation options. For more details see the table below.

Command Options

Flag	Description	Default Value
`--install`	Flag to indicate that UFM should perform the installation	N/A

Flag	Description	Default Value
`-f | --fabric-interface`	IB fabric interface name.	ib0
`-g | --mgmt-interface`	Management interface name.	eth0
`-m | --multisubnet-consumer`	UFM Multisubnet Consumer mode	N/A
`--local-certs-dir <directory>`	A directory on the host that contains the SSL certificate and key (`server.crt` and `server.key`) and the ca-intermediate certificate (ca-intermediate.crt), all in PEM format.	N/A
`-h | --help`	Show help	N/A

Options applicable only on Podman Unprivileged User Installation

Flag	Description	Default Value
`--rootless`	Flag to indicate that the installation should be rootless.	N/A
`--plugin-path`	Path to the plugin data created in the previous step.	N/A
`--ufm-user`	The user that will run UFM on the host (e.g., `ufmadm`).	N/A
`--ufm-group`	The user group that will run UFM on the host (e.g., `ufmadm`).	N/A

Installation Modes

UFM Enterprise installer supports several deployment modes:

- Installing UFM on Docker Container - High Availability Mode

- Installing UFM on Docker Container - Standalone Mode

Installing UFM on Docker Container - High Availability Mode

Pre-Deployments Requirements

- Install **pacemaker**, **pcs**, and **drbd-utils** on both servers

  For Ubuntu:

  ```
  apt install pcs pacemaker drbd-utils
  ```

  For CentOS/Red Hat:

  ```
  yum install pcs pacemaker drbd84-utils kmod-drbd84
  ```

  OR

  ```
  yum install pcs pacemaker drbd90-utils kmod-drbd90
  ```

- A partition for DRBD on each server (**with the same name** on both servers) such as `/dev/sdd1`. Recommended partition size is 10-20 GB, otherwise DRBD sync will take a long time to complete.

- CLI command `hostname -i` must return the IP address of the management interface used for pacemaker sync correctly (update `/etc/hosts/` file with machine IP)

- Create the directory on each server under `/opt/ufm/files/` with read/write permissions on each server. This directory will be used by UFM to mount UFM files, and it will be synced by DRBD.

- Disable the firewall service (`/etc/init.d/iptables` stop), or ensure that the required ports are open (see the prerequisite script).
- Disable SELinux.

Installing UFM Containers

On the main server, install UFM Enterprise container with the command below:

```
docker run -it --name=ufm_installer --rm \
-v /var/run/docker.sock:/var/run/docker.sock \
-v /etc/systemd/system/:/etc/systemd_files/ \
-v /opt/ufm/files/:/installation/ufm_files/ \
-v /tmp/license_file/:/installation/ufm_licenses/ \
mellanox/ufm-enterprise:latest \
--install
```

On the standby (secondary) server, install the UFM Enterprise container like the following example with the command below:

```
docker run -it --name=ufm_installer --rm \
-v /var/run/docker.sock:/var/run/docker.sock \
-v /etc/systemd/system/:/etc/systemd_files/ \
-v /opt/ufm/files/:/installation/ufm_files/ \
mellanox/ufm-enterprise:latest \
--install
```

Downloading UFM HA Package

Download the UFM-HA package on both servers using the following command:

```
wget http://www.mellanox.com/downloads/UFM/ufm_ha_6.1.1-2.tgz
```

Installing UFM HA Package

For more information on the UFM-HA package and all installation and configuration options, please refer to UFM High Availability User Guide.

1. [On Both Servers] Extract the downloaded UFM-HA package under `/tmp/`
2. [On Both Servers] Go to the extracted directory `/tmp/ufm_ha_XXX` and run the installation script. For example, if your DRBD partition is `/dev/sda5` run the following command:

```
./install.sh -l /opt/ufm/files/ -d /dev/sda5 -p enterprise
```

Configuring UFM HA

There are the three methods to configure the HA cluster:

- Configure HA with SSH Trust (Dual Link Configuration) - Requires passwordless SSH connection between the servers.
- Configure HA without SSH Trust (Dual Link Configuration) - Does not require passwordless SSH connection between the servers, but asks you to run configuration commands on both servers.
- Configure HA without SSH Trust (Single Link Configuration) - Can be used in cases where only one link is available among the two UFM HA nodes/servers.

Configure HA with SSH Trust (Dual Link Configuration)

1. On the **master server only**, configure the HA nodes. To do so, from /tmp, run the `configure_ha_nodes.sh` command as shown in the below example

```
configure_ha_nodes.sh \
--cluster-password 12345678 \
--master-primary-ip 10.10.50.1 \
--standby-primary-ip 10.10.50.2 \
--master-secondary-ip 192.168.10.1 \
--standby-secondary-ip 192.168.10.2 \
--no-vip
```

Note

The script `configure_ha_nodes.sh` is is located under /usr/local/bin/, therefore, by default, you do not need to use the full path to run it.

Note

The `--cluster-password` must be at least 8 characters long.

Note

When using back-to-back ports with local IP addresses for HA sync interfaces, ensure that you add your IP addresses and hostnames to the `/etc/hosts` file. This is needed to allow the HA configuration to resolve hostnames correctly based on the IP addresses you are using.

> **Note**
>
> `configure_ha_nodes.sh` requires SSH connection to the standby server. If SSH trust is not configured, then you are prompted to enter the SSH password of the standby server during configuration runtime

2. Depending on the size of your partition, wait for the configuration process to complete and DRBD sync to finish. To check the DRBD sync status, run:

```
ufm_ha_cluster status
```

Configure HA without SSH Trust (Dual Link Configuration)

If you cannot establish an SSH trust between your HA servers, you can use `ufm_ha_cluster` directly to configure HA. You can see all the options for configuring HA in the Help menu:

```
ufm_ha_cluster config -h
```

To configure HA, follow the below instructions:

> **Note**
>
> Please change the variables in the commands below based on your setup.

1. [On **Standby Server**] Run the following command to configure **Standby Server**:

```
ufm_ha_cluster config -r standby -e <peer ip address> -l
<local ip address> -p <cluster_password>
```

2. [On **Master Server**] Run the following command to configure **Master Server**:

```
ufm_ha_cluster config -r master -e <peer ip address> -l
<local ip address> -p <cluster_password> -i <virtual ip
address>
```

Configure HA without SSH Trust (Single Link Configuration)

> ⚠ **Warning**
>
> This is not the recommended configuration and, in case of network failure, it might cause HA cluster split brain.

If you cannot establish an SSH trust between your HA servers, you can use **ufm_ha_cluster** directly to configure HA. To configure HA, follow the below instructions:

> ⓘ **Note**
>
> Please change the variables in the commands below based on your setup.

1.

    1. [On **Standby Server**] Run the following command to configure **Standby Server**:

```
ufm_ha_cluster config \
-r standby \
-e 10.212.145.5 \
-l 10.212.145.6 \
--enable-single-link
```

2. [On **Master Server**] Run the following command to configure **Master Server**:

```
ufm_ha_cluster config -r master \
-e 10.212.145.6 \
-l 10.212.145.5 \
-i 10.212.145.50 \
--enable-single-link
```

You must wait until after configuration for DRBD sync to finish, depending on the size of your partition. To check the DRBD sync status, run:

```
ufm_ha_cluster status
```

IPv6 Example:

```
ufm_ha_cluster config -r standby -l
fcfc:fcfc:209:224:20c:29ff:fee7:d5f2 -e
fcfc:fcfc:209:224:20c:29ff:fecb:4962 --enable-single-link -
p some_secret
```

Starting HA Cluster

- To start UFM HA cluster:

```
ufm_ha_cluster start
```

- To check UFM HA cluster status:

```
ufm_ha_cluster status
```

- To stop UFM HA cluster:

```
ufm_ha_cluster stop
```

- To uninstall UFM HA, first stop the cluster and then run the uninstallation command as follows:

```
/opt/ufm/ufm_ha/uninstall_ha.sh
```

Installing UFM on Docker Container - Standalone Mode

1. Copy only your UFM license file(s) to a temporary directory which we're going to use in the installation command. For example: `/tmp/license_file/`

2. Run the UFM installation command according to the following example which will also configure UFM fabric interface to be ib1:

```
docker run -it --name=ufm_installer --rm \
                            -v
/var/run/docker.sock:/var/run/docker.sock \
                            -v
/etc/systemd/system/:/etc/systemd_files/ \
                            -v
/opt/ufm/files/:/installation/ufm_files/ \
                            -v
/tmp/license_file/:/installation/ufm_licenses/ \
                           mellanox/ufm-enterprise:latest \
                           --install \
                           --fabric-interface ib1
```

3. Reload systemd:

```
systemctl daemon-reload
```

4. To Start UFM Enterprise service run:

```
systemctl start ufm-enterprise
```

Installing UFM on Podman Using Unprivileged User While Setting Client Certificate Authentication

1. Copy only your UFM license file(s) to a temporary directory which we're going to use in the installation command. For example: `/tmp/license_file/`

2. Run the UFM installation command according to the following example which will also configure UFM fabric interface to be ib0:

```
sudo -u ufmadm podman run  -it --rm --name=ufm_installer \
                           -v /run/podman-ufm/podman-
ufm.sock:/var/run/docker.sock \
                           -v
/opt/ufm/:/installation/ufm_files/ \
                           -v
/opt/ufm/systemd:/etc/systemd_files/ \
                           mellanox/ufm-enterprise:latest \
                           --install \
                           --fabric-interface ib0 \
                           --mgmt-interface enp1s0 \
                           --rootless \
                           --plugin-path
/opt/ufm/ufm_plugins_data \
                           --ufm-user ufmadm \
                           --ufm-group ufmadm \
                           --local-certs-dir /ufm_certs
```

3. Reload systemd on the host:

```
systemctl daemon-reload
```

4. To start UFM Enterprise service on the host, run:

```
systemctl start ufm-enterprise
```

Replacing the Standby Node

- Install the HA package for the new node (standby).

- Disconnect the standby node (the old standby) and run the following command on the master node:

```
ufm_ha_cluster detach
```

- Configure the new standby node; please refer to the relevant section depending on the installation
- Connect the new standby to the cluster by running the command on the master node:

```
ufm_ha_cluster attach -l <local primary ip address> -e <peer
primary ip address> -E <peer secondary ip address> -p <clust
```

Installing UFM Podman Rootless Container Mode

Step 1: Check Firewall Status

```
systemctl status firewalld
```

Step 2: Configure Firewall (if active)

```
# check if firewalld is running
   systemctl status firewalld
   # Permanently add port 8443 to firewalld
   firewall-cmd --permanent --add-port=8443/tcp
   # reload firewalld config
   firewall-cmd --reload
```

Step 3: Create UFM Directory

```
mkdir -p /opt/ufm
```

Step 4: Create UFM Group

```
groupadd ufmadm -g 733
```

Step 5: Create UFM User

```
useradd -d /opt/ufm -m -u 733 -g ufmadm ufmadm
```

Step 6: Set Directory Ownership

```
chown -R ufmadm:ufmadm /opt/ufm
```

Step 7: Add SubUID/SubHID Configuration

```
echo "ufmadm:100000:65536" >> /etc/subuid
echo "ufmadm:100000:65536" >> /etc/subgid
```

Step 8: Enable Login Linger for UFM User

```
loginctl enable-linger ufmadm
```

Step 9: Configure Rootless Podman Storage

```
sudo -u ufmadm mkdir -p /opt/ufm/.config/containers
cat <<EOF | sudo -u ufmadm tee
/opt/ufm/.config/containers/storage.conf > /dev/null
[storage]
driver = "overlay"
runroot = "/run/user/733"
EOF
```

10. Create Podman UFM Socket

```
cat <<EOF > /usr/lib/systemd/system/podman-ufm.socket
[Unit]
Description=Podman API Socket For Nvidia UFM

[Socket]
SocketUser=ufmadm
SocketGroup=ufmadm
ListenStream=%t/podman-ufm/podman-ufm.sock
SocketMode=0660

[Install]
WantedBy=sockets.target
EOF
```

11. Create Podman UFM Service

```
cat <<EOF > /usr/lib/systemd/system/podman-ufm.service
[Unit]
Description=Podman API Service for Nvidia UFM
Requires=podman-ufm.socket
After=podman-ufm.socket
StartLimitIntervalSec=0

[Service]
Delegate=true
Type=exec
User=ufmadm
Group=ufmadm
KillMode=process
Environment=LOGGING="--log-level=info"
ExecStart=/usr/bin/podman \$LOGGING system service
LimitMEMLOCK=infinity

[Install]
WantedBy=default.target
EOF
```

12. Create Podman Cleanup Service

```
cat <<EOF > /usr/lib/systemd/system/podman-ufm-cleanup.service
[Unit]
Description=podman-ufm-cleanup - clean stuck rootless containers
at boot
After=podman-ufm.service
Before=ufm-enterprise.service

[Service]
Type=oneshot
User=ufmadm
Group=ufmadm
ExecStart=/usr/bin/podman system migrate

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

Step 13: Enable and Start Podman Services

```
systemctl daemon-reload
systemctl enable --now podman-ufm.socket
systemctl enable --now podman-ufm.service
systemctl enable --now podman-ufm-cleanup.service
```

Step 14: Create Udev Rules for InfiniBand Devices

```
cat <<EOF > /etc/udev/rules.d/70-umad.rules
KERNEL=="umad*", SUBSYSTEM=="infiniband_mad", MODE="0600", OWNER="ufmadm",
GROUP="ufmadm"
KERNEL=="issm*", SUBSYSTEM=="infiniband_mad", MODE="0600", OWNER="ufmadm",
GROUP="ufmadm"
EOF

udevadm control --reload-rules
udevadm trigger
```

Step 15: Clear and Create UFM Directories

```
rm -rf /opt/ufm/systemd
sudo -u ufmadm mkdir -p /opt/ufm/ufm_plugins_data
sudo -u ufmadm mkdir -p /opt/ufm/systemd
sudo -u ufmadm mkdir -p /opt/ufm/etc/apache2
```

Step 16: Load UFM Image

```
sudo -u ufmadm podman load -i <path to ufm*-docker.img.gz
```

Step 17: Run UFM Installer

```
sudo -u ufmadm podman run -it --rm --name=ufm_installer \
                          -v /run/podman-ufm/podman-
ufm.sock:/var/run/docker.sock \
                          -v /opt/ufm/:/installation/ufm_files/ \
                          -v /opt/ufm/systemd:/etc/systemd_files/
\
                          mellanox/ufm-enterprise:latest \
                          --install \
                          --fabric-interface ib0 \
                          --rootless \
                          --plugin-path /opt/ufm/ufm_plugins_data
\
                          --ufm-user ufmadm \
                          --ufm-group ufmadm
```

Note

Note: Replace `ib0` with your actual InfiniBand interface name, if it is not the default ib0. **Note**: - All other UFM install flags are supported and can be added to the command.

Step 18: Install UFM Enterprise Service Files

```
mv /opt/ufm/systemd/ufm-enterprise.service
/etc/systemd/system/ufm-enterprise.service
systemctl daemon-reload
```

Step 19: Enable And Start UFM Service

```
systemctl start ufm-enterprise.service
```

Optional: Client Certificate Authentication Setup

This section provides steps to configure UFM with client certificate authentication using locally refreshed certificates. This is an optional security enhancement that can be configured after the basic UFM installation is complete.

Prerequisites for Client Certificate Setup

- Valid client certificates available
- Local certificates directory mapped to the UFM container

Client Certificate Configuration Steps

Step 1: Install UFM with Local Certificates Directory (Alternative to Step 18)

If you want to use client certificates, first prepare your certificate directory with the required files, then modify step 18 to include the `--local-certs-dir` flag.

> **Note**
>
> **Important**: The `--local-certs-dir` flag can only be used during initial installation. If UFM is already installed, you must reinstall to use this feature.

The local certificates directory must contain the following files in PEM format:

- `server.crt` - SSL certificate
- `server.key` - SSL key

- `ca-intermediate.crt` - CA intermediate certificate

```
sudo -u ufmadm podman run -it --rm --name=ufm_installer \
                          -v /run/podman-ufm/podman-
ufm.sock:/var/run/docker.sock \
                          -v /opt/ufm/:/installation/ufm_files/ \
                          -v /opt/ufm/systemd:/etc/systemd_files/
\
                          mellanox/ufm-enterprise:latest \
                          --install \
                          --fabric-interface ib0 \
                          --mgmt-interface enp1s0 \
                          --rootless \
                          --plugin-path /opt/ufm/ufm_plugins_data
\
                          --ufm-user ufmadm \
                          --ufm-group ufmadm \
                          --local-certs-dir /path/to/local/certs
```

Replace `/path/to/local/certs` with the actual path to your directory containing the certificate files on the host.

Step 2: Enter the UFM Container

Once the UFM service is running, enter the container:

```
sudo -u ufmadm podman exec -it ufm /bin/bash
```

Step 3: Enable Client Certificate Authentication

```
/opt/ufm/scripts/manage_client_authentication.sh enable-client-cert-authen
```

Step 4: Set Client Certificate Subject Identifier

```
/opt/ufm/scripts/manage_client_authentication.sh set-subject-identifier --identifier CN
```

> **Note**
>
> Use CN or SAN.

Step 5: Associate Client Certificate with UFM User

```
/opt/ufm/scripts/manage_client_authentication.sh associate-user --cn <CN> --username <UFM_USER>
```

Replace:

- `<CN>`: The Common Name from your client certificate
- `<UFM_USER>`: The UFM username to associate with the certificate

Step 6: Set SSL Server Certificate Hostname

```
/opt/ufm/scripts/manage_client_authentication.sh set-ssl-cert-
hostname --hostname <hostname>
```

> **Note**
>
> Replace `<hostname>` with your server's hostname.

Step 7: Restart Apache Web Server

```
/opt/ufm/scripts/manage_client_authentication.sh
restart_ufm_websrv
```

Client Certificate Authentication Notes

- Steps B-G should be performed after the UFM container is running
- This command can be embedded in automation scripts and invoked after certificates are refreshed
- Users will need valid client certificates matching the configured CN associations to access UFM
- Verify certificates are properly formatted and the hostname matches your server configuration

UFM Infra Installation

The **UFM Infra** feature introduces a structured architecture where services are divided into two categories, each deployed differently based on functionality:

- UFM Infra: A set of persistent infrastructure services that run on all nodes. These services support system-level operations and ensure distributed availability.
- UFM Enterprise: Services that run exclusively on the master node, responsible for management, orchestration, and user-facing functionality.

For more information on the UFM Infra architecture, refer to UFM Infra

- Installing UFM Infra Using Docker Root
- Installing UFM Infra Using Rootless with Podman

Installing UFM Infra Using Docker Root

UFM Docker Root Installation Steps with Infra Mode and NFS/Shared Storage

This section provides installation instructions for UFM Enterprise running in Docker root mode with UFM infra support and NFS or shared storage. The configuration uses versioned directories and soft links to enable seamless upgrades.

Prerequisites

- Docker installed and running
- NFS or shared storage must be mounted at `/opt/ufm/shared_files`
- UFM and FAST-API tar files available
- Root or sudo privileges
- rsync
- Redis - if using an external Redis

Architecture Overview

- **NFS Mount Point**: `/opt/ufm/shared_files` - Main shared storage location
- **Version Directory**: `/opt/ufm/shared_files/ufm-<version>` - Version-specific directory
- **Soft Link**: `/opt/ufm/files` → `/opt/ufm/shared_files/ufm-<version>` - Points to current version
- **UFM Mode**: Docker root + UFM infra (creates two services: ufm-enterprise.service and ufm-infra.service)

Installation Steps

Step 1: Create UFM Base Directory

```
mkdir -p /opt/ufm
mkdir -p /opt/ufm/ufm_plugins_data
```

Step 2: Load UFM Image and Extract Version

```
# Define the UFM image file path
UFM_IMAGE_FILE="<path to ufm*-docker.img.gz>"

# Extract UFM version from filename (e.g., ufm_6.22.0-7.ubuntu24.x86_64-docker.img.gz -> 6_22_0_7)
UFM_VERSION=$(basename "$UFM_IMAGE_FILE" | sed 's/ufm_\([0-9][^.]*\.[^.]*\.[^.]*-[^.]*\)\.ubuntu.*/\1/' | tr '.-' '_')
echo "UFM Version: $UFM_VERSION"

# Load the UFM image
docker load -i "$UFM_IMAGE_FILE"
```

Step 3: Create Version-Specific Directory and Soft Link

```
# Create version-specific directory in shared storage
mkdir -p /opt/ufm/shared_files/ufm-${UFM_VERSION}

# Remove existing files link if it exists
rm -f /opt/ufm/files

# Create soft link to version-specific directory
ln -s /opt/ufm/shared_files/ufm-${UFM_VERSION} /opt/ufm/files

# Verify the soft link
ls -la /opt/ufm/files
```

Step 4: Run UFM Installer

```
docker run --name=ufm_installer --rm \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v /etc/systemd/system/:/etc/systemd_files/ \
    -v /opt/ufm/files/:/installation/ufm_files/ \
    -v [LICENSE_DIRECTORY]:/installation/ufm_licenses/ \
    mellanox/ufm-enterprise:latest \
    --install \
    --fabric-interface ib0 \
    --ufm-infra
```

Note

- Replace `ib0` with your actual InfiniBand interface name, if it is not the default ib0.
- All other UFM install flags are supported and can be added to the command.

Step 5: Load Redis Image

Load the given Redis image (in case you are not using external Redis) docker load -i ""

Step 6: Load Fast API Plugin Image

```
# Define the Fast API plugin image file path
FAST_API_IMAGE="<PATH TO FAST API.img.gz file>"

# Load the Fast API plugin image
docker load -i "$FAST_API_IMAGE"

# Extract Fast API version from Docker image tag (exclude latest tag)
FAST_API_VERSION=$(docker images --format "{{.Repository}}:{{.Tag}}" | grep "mellanox/ufm-plugin-fast_api" | grep -v ":latest" | head -1 | cut -d':' -f2)
echo "Fast API Version: $FAST_API_VERSION"
```

Step 7: Install Fast API Plugin

```
docker run --hostname $HOSTNAME --rm --name=ufm_plugin_mgmt --entrypoint="" \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v /opt/ufm/files:/opt/ufm/shared_config_files \
    -v /dev/log:/dev/log \
    -v /sys/fs/cgroup:/sys/fs/cgroup:ro \
    -v /lib/modules:/lib/modules:ro \
    -v /opt/ufm/ufm_plugins_data:/opt/ufm/ufm_plugins_data \
    -e UFM_CONTEXT=ufm-infra \
    mellanox/ufm-enterprise:latest \
    /opt/ufm/scripts/manage_ufm_plugins.sh add -p fast_api -t ${FAST_API_VERSION} -c ufm-infra
```

Step 8: Reload systemd and Start Services (if not running HA)

```
# Reload systemd daemon
systemctl daemon-reload

# Start UFM infra service (starts first)
systemctl start ufm-infra
systemctl start ufm-enterprise
```

Installing UFM Infra Using Rootless with Podman

UFM Infra Rootless Deployment with Podman (Restricted to Oracle Linux Only)

Prerequisites

1. Download the UFM and plugins bundle tar file to `/tmp`.
2. Extract the contents using the command:

```
tar -xvf <bundle tar>
```

This archive (tar file) includes the following components:

- Relevant UFM container image
- Relevant FAST-API container image

- Relevant Infra container image (for internal Redis usage). Refer to Redis-Related Configuration for more information.
- Default plugin bundle for UFM
- UFM-HA package

HA Installation Requirements

To enable the UFM Infra feature, UFM HA must be installed in a new mode (`external-storage`), using a new product (`enterprise-multinode`).

Additionally, NFS must be configured as follows:

NFS Setup Prerequisites

- Select a dedicated NFS server to host the shared directories.
- Create a shared directory on the NFS server for UFM configuration and logs.
- Install the NFS client on each UFM node if not already present.

Enable HA ports in Firewall

If you have firewall rules that blocks non-standard ports, we need to open these ports so high availability services could communicate with each other on the HA nodes. To do so, run these commands:

```
firewall-cmd --permanent --add-service=high-availability
# or
firewall-cmd --add-service=high-availability
# and then reload the rules
firewall-cmd --reload
```

Create and Mount the UFM Directory

> **Note**
>
> At this stage, apply point #2 (Mount the UFM directory) only on the master machine.
>
> Other nodes will be visited for mount later.

1. Create the UFM directory:

```
mkdir -p /opt/ufm/files/
```

2. Mount the UFM directory:

   - If using NFS 4.2:

```
 mount -t nfs4 -o context="system_u:object_r:container_file_t:s0"
<server>:/shared_folder /opt/ufm/shared_files
```

   - If using NFS 3:

```
 mount -t nfs -o vers=3,context="system_u:object_r:container_file_t:s0"
<server>:/shared_folder /opt/ufm/shared_files
```

3. Ensure the NFS version and mount options are compatible with the NFS server.

4. Verify that the following HA packages are installed: `pcs`, `pacemaker`, and `corosync`. Install them if they are missing.

5. Follow the HA installation steps in Run the HA Installation.

Run the HA Installation

Follow the HA installation instructions at UFM High-Availability Installation and Configuration.

When running the HA installation script, use the following command:

```
./install.sh -p enterprise-multinode -l /opt/ufm/shared_files
```

- The `-l` flag must always point to the shared directory path: `/opt/ufm/shared_files`
- No need to provide the DRBD disk argument to the installation script.

Installation Instructions

1. Check firewall status:

   ```
   systemctl status firewalld
   ```

2. Configure Firewall (if active):

```
# check if firewalld is running
   systemctl status firewalld
# Permanently add port 8443 to firewalld
   firewall-cmd --permanent --add-port=8443/tcp
# reload firewalld config
   firewall-cmd --reload
```

3. Create UFM directory:

```
mkdir -p /opt/ufm
```

4. Create UFM group:

```
groupadd ufmadm -g 733
```

5. Create a UFM user:

```
useradd -d /opt/ufm -m -u 733 -g ufmadm ufmadm
```

6. Set directory ownership:

```
chown -R ufmadm:ufmadm /opt/ufm
chown -R ufmadm:ufmadm /opt/ufm/shared_files
```

7. Configure SubUID and SubGID:

```
echo "ufmadm:100000:65536" >> /etc/subuid
echo "ufmadm:100000:65536" >> /etc/subgid
```

8. Enable Login Linger for UFM ser:

```
loginctl enable-linger ufmadm
```

9. Configure Rootless Podam storage

```
sudo -u ufmadm mkdir -p /opt/ufm/.config/containers
cat <<EOF | sudo -u ufmadm tee
/opt/ufm/.config/containers/storage.conf > /dev/null
[storage]
driver = "overlay"
runroot = "/run/user/733"
EOF
```

10. Create Podman UFM socket:

```
cat <<EOF > /usr/lib/systemd/system/podman-ufm.socket
[Unit]
Description=Podman API Socket For Nvidia UFM

[Socket]
SocketUser=ufmadm
SocketGroup=ufmadm
ListenStream=%t/podman-ufm/podman-ufm.sock
SocketMode=0660

[Install]
WantedBy=sockets.target
EOF
```

11. Create Podman UFM service

```
cat <<EOF > /usr/lib/systemd/system/podman-ufm.service
[Unit]
Description=Podman API Service for Nvidia UFM
Requires=podman-ufm.socket
After=podman-ufm.socket
StartLimitIntervalSec=0

[Service]
Delegate=true
Type=exec
User=ufmadm
Group=ufmadm
KillMode=process
Environment=LOGGING="--log-level=info"
ExecStart=/usr/bin/podman \$LOGGING system service
LimitMEMLOCK=infinity

[Install]
WantedBy=default.target
EOF
```

12. Create Podman cleanup service:

```
cat <<EOF > /usr/lib/systemd/system/podman-ufm-
cleanup.service
[Unit]
Description=podman-ufm-cleanup - clean stuck rootless
containers at boot
After=podman-ufm.service
Before=ufm-enterprise.service

[Service]
Type=oneshot
User=ufmadm
Group=ufmadm
ExecStart=/usr/bin/podman system migrate

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

13. Enable and start Podman services:

```
systemctl daemon-reload
systemctl enable --now podman-ufm.socket
systemctl enable --now podman-ufm.service
systemctl enable --now podman-ufm-cleanup.service
```

14. Create Udev Rules for InfiniBand Devices

```
cat <<EOF > /etc/udev/rules.d/70-umad.rules
KERNEL=="umad*", SUBSYSTEM=="infiniband_mad", MODE="0600",
OWNER="ufmadm", GROUP="ufmadm"
KERNEL=="issm*", SUBSYSTEM=="infiniband_mad", MODE="0600",
OWNER="ufmadm", GROUP="ufmadm"
EOF

udevadm control --reload-rules
udevadm trigger
```

15. Clean and create UFM directories

```
rm -rf /opt/ufm/systemd
sudo -u ufmadm mkdir -p /opt/ufm/ufm_plugins_data
sudo -u ufmadm mkdir -p /opt/ufm/systemd
sudo -u ufmadm mkdir -p /opt/ufm/etc/apache2
```

16. Load UFM image and extract version:

```
# Extract UFM version from filename (e.g., ufm_6.22.0-
7.ubuntu24.x86_64-docker.img.gz -> 6_22_0_7)
UFM_VERSION=$(basename "$UFM_IMAGE_FILE" | sed 's/ufm_\([0-9][^.]*\.[^.]*\.[^.]*-
[^.]*\)\.ubuntu.*/\1/' | tr '.-' '_')
echo "UFM Version: $UFM_VERSION"

# Load the UFM image
sudo -u ufmadm podman load -i "$UFM_IMAGE_FILE"
```

17. Create version-specific directory and soft link:

```
# Create version-specific directory in shared storage
sudo -u ufmadm mkdir -p
/opt/ufm/shared_files/ufm-${UFM_VERSION}

# Remove existing files link if it exists
rm -f /opt/ufm/files

# Create soft link to version-specific directory
sudo -u ufmadm ln -s /opt/ufm/shared_files/ufm-${UFM_VERSION}
/opt/ufm/files

# Verify the soft link
ls -la /opt/ufm/files
```

18. Run UFM installer:

```
sudo -u ufmadm podman run -it --rm --name=ufm_installer \
                          -v /run/podman-ufm/podman-
ufm.sock:/var/run/docker.sock \
                          -v
/opt/ufm/:/installation/ufm_files/ \
                          -v
/opt/ufm/files:/installation/ufm_files/files \
                          -v
/opt/ufm/systemd:/etc/systemd_files/ \
                          mellanox/ufm-enterprise:latest \
                          --install \
                          --fabric-interface ib0 \
                          --rootless \
                          --plugin-path
/opt/ufm/ufm_plugins_data \
                          --ufm-user ufmadm \
                          --ufm-group ufmadm \
                          --ufm-infra
```

Note

Note: Replace `ib0` with your actual InfiniBand interface name, if it is not the default ib0.

Note: - All other UFM install flags are supported and can be added to the command.

19. Load Redis Image (if not using external Redis):

```
Load the given Redis image (in case you are not using external
Redis)
sudo - u ufmadm load -i "<PATH TO GIVEN REDIS IMAGE>"
```

20. Load Fast API Plugin image:

```
sudo - u ufmadm run --hostname $HOSTNAME --rm --
name=ufm_plugin_mgmt --entrypoint="" \
    -v /run/podman-ufm/podman-ufm.sock:/var/run/docker.sock \
    -v /opt/ufm/files:/opt/ufm/shared_config_files \
    -v /dev/log:/dev/log \
    -v /sys/fs/cgroup:/sys/fs/cgroup:ro \
    -v /lib/modules:/lib/modules:ro \
    -v /opt/ufm/ufm_plugins_data:/opt/ufm/ufm_plugins_data \
    -e UFM_CONTEXT=ufm-infra \
    mellanox/ufm-enterprise:latest \
    /opt/ufm/scripts/manage_ufm_plugins.sh add -p fast_api -t
${FAST_API_VERSION} -c ufm-infra
```

21. Install service files:

```
mv /opt/ufm/systemd/ufm-enterprise.service
/etc/systemd/system/ufm-enterprise.service
mv /opt/ufm/systemd/ufm-infra.service
/etc/systemd/system/ufm-infra.service
systemctl daemon-reload
```

To **start UFM as a standalone instance**, run:

```
systemctl daemon-reload
systemctl start ufm-infra
systemctl start ufm-enterprise
```

Running in HA Mode

> **Note**
>
> Do not manually start any services.

1. Ensure UFM and UFM-HA are installed on all nodes as described in the above sections.
2. Mount /opt/ufm/files on all standby nodes as described point #2 (Mount the UFM directory)
3. On one node, edit the HA configuration file:

   ```
   /etc/ufm_ha/ha_nodes.cfg
   ```

   Fill each node parameters

```
[Node.1]
# valid role options: master/standby
role = master
# Mandatory
primary_ip =
# Mandatory if dual_link = true
secondary_ip =

[Node.2]
role = standby
primary_ip =
secondary_ip =

[Node.3]
role = standby
primary_ip =
secondary_ip =
```

4. Ensure the file sync mode is set to `external-storage`, and that the shared file system is mounted prior to HA configuration.

```
[FileSync]
# valid options are: drbd/external-storage
# in case of external-storage the user MUST mount the files
system PRIOR to ha configuration
mode = external-storage
```

5. Copy the edited file to all nodes at the same path.

6. Configure the cluster, starting from standby nodes and ending with the master node:

```
ufm_ha_cluster config -p <password>
```

> **Note**
>
> Use the same password on all nodes.

7. After finishing the configuration on all nodes, run:

```
ufm_ha_cluster status
```

8. Start the cluster:

```
ufm_ha_cluster start
```

9. Check cluster status again to ensure all services have started successfully.

Activating Software License

1. Before starting the UFM software, copy your license file(s) downloaded from NVIDIA Licensing and Download Portal (*volt-ufm-<serial-number>.lic*) to the master server under the */opt/ufm/files/licenses* directory. We recommend that you back up the license file(s).

   In High Availability mode, the license files are replicated to the standby machine automatically. Your software is now activated.

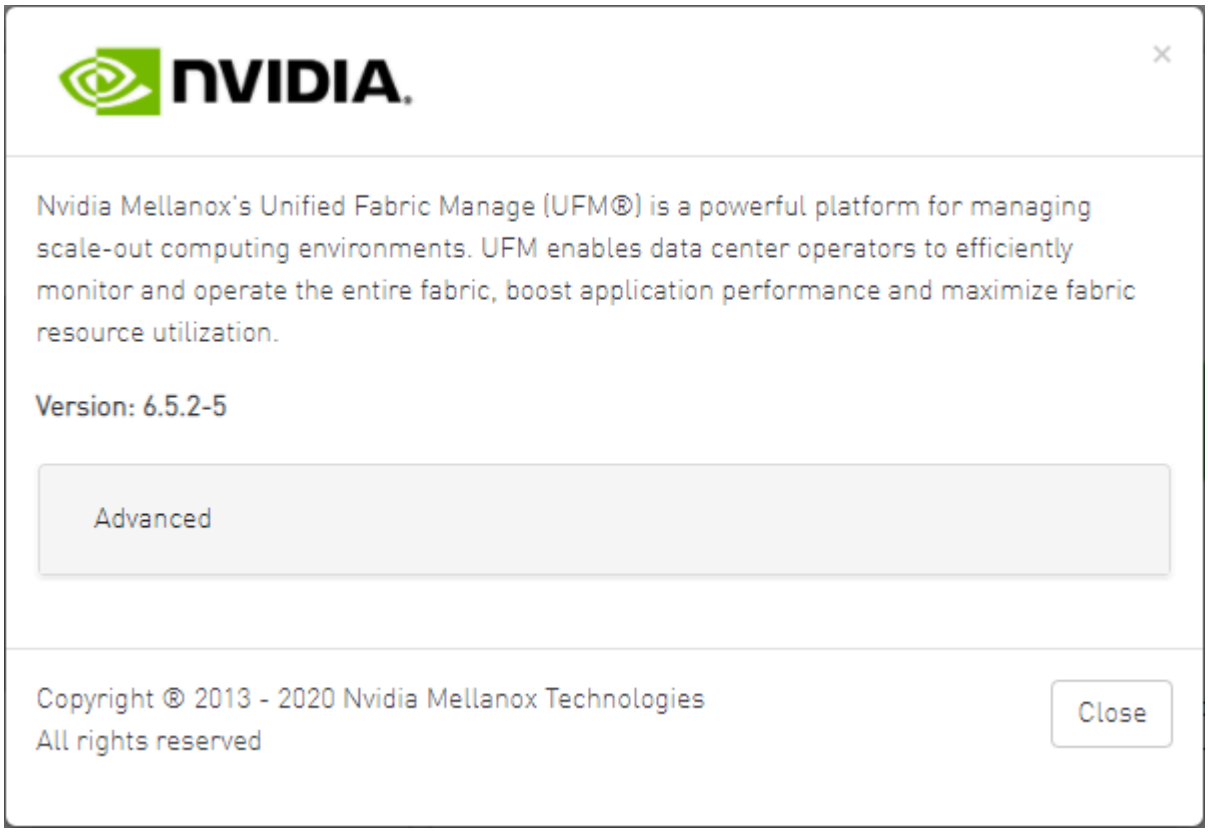
2. Run the UFM software as described in the following sections.

Note

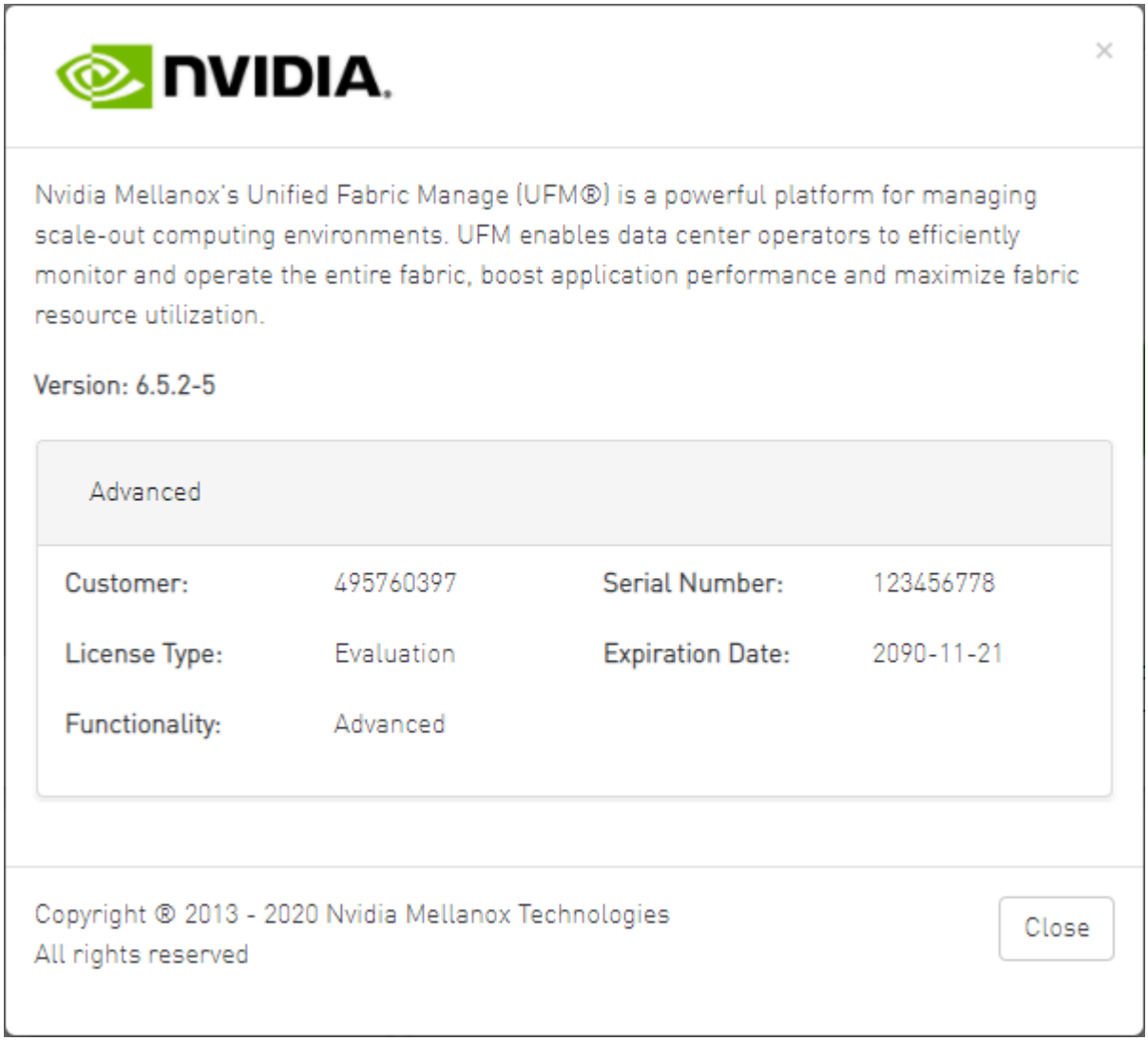
When a UFM license is not provided for activation upon the first UFM installation, the UFM runs on an auto-generated evaluation license which expires after 30 days from the first start-up of the UFM.

Licensing

1. After installing and activating your software, you can view your licenses in the Web UI by clicking the About icon ( ) in the main window.



2. To view the advanced license information, click the Advanced button. The advanced license details will be displayed below.



3. Product Functionality is updated only after startup. If you replace the UFM license, UFM continues to work in the previous mode until the UFM server is restarted.

To view license information from the CLI:

Run CLI Command "**ufmlicense**" to display information about all installed licenses on the UFM server under /opt/ufm/files/licenses. This includes invalid and expired license information.

UFM license could support up to 3 different MAC addresses in case there are more than one standby machines (one master and two standbys).

In below example there are two UFM HA licenses where each license includes 2 different MACs: one for the primary machine and one for the standby machine.

In a given time, for each license, only one MACs is detected to be "Valid" (exists on the local machine) where the other MAC is detected as "Invalid" (exist on the standby machine).

See below ou*tput example when running the CLI command* `ufmlicense` in SA and HA Modes.

HA Mode Output Example:

```
[root@ip-10-224-16-49-dg11 ~]# docker exec -ti ufm bash
root@ ip-10-224-16-49-dg11~# ufmlicense
|----------------------------------------------------------------------------------------------------------------------|
| Customer ID | SN | swName | Type | MAC Address | Exp. Date |Limit| Functionality |
Status ||----------------------------------------------------------------------------------------------------------------------| |NVIDIA
Corp|xxxxxxx-xxxxxxx|UFM Enterprise|Subscription |e4:43:4b:18:3c:e0|2025-08-29 |128 |3 Years |Invalid | |-
----------------------------------------------------------------------------------------------------------------------| |NVIDIA
Corp|xxxxxxx-xxxxxxx|UFM Enterprise|Subscription |e4:43:4b:18:49:a0|2025-08-29 |128 |3 Years |Valid | |---
----------------------------------------------------------------------------------------------------------------------| |NVIDIA
Corp|xxxxxxx-xxxxxxx|UFM Enterprise|Subscription |e4:43:4b:18:3c:e0|2025-07-13 |128 |3 Years |Invalid | |-
----------------------------------------------------------------------------------------------------------------------| |BBK Electronics
Corp Ltd|xxxxxxx-xxxxxxx|UFM Enterprise|Subscription |e4:43:4b:18:49:a0|2025-07-13 |128 |3 Years |Valid |
```

SA Mode Output Example:

```
root@ufm-production:~# ufmlicense
|-------------------------------------------------------------------
----------------------------------------------------------------|
|   Customer ID   |     SN     |   swName      |        Type
|   MAC Address   |   Exp. Date   |Limit| Functionality | Status
|
|-------------------------------------------------------------------
----------------------------------------------------------------|
|495760397         |123456778   |UFM Enterprise |Evaluation
|NA                |2090-11-21      |1024 |Advanced       |Valid   |
|-------------------------------------------------------------------
----------------------------------------------------------------|
```

To remove a license:



Delete the license file from /opt/ufm/files/licenses.

UFM Configuration

- Initial Configuration
- Optional Configurations

Initial Configuration

After installing the UFM® server software and before running UFM, perform the following:

- Mandatory Configuration:
  - Configure General Settings in gv.cfg
- Optional Configurations:
  - General Configuration options
  - Quality of Service
  - Activate and Enable Lossy Configuration Manager (Advanced License Only)
  - Activate and Enable Congestion Control Manager (Advanced License Only)

Configuring Fabric Interface

In most common cases, UFM is run in management mode; the UFM SM manages the InfiniBand fabric. In such cases, the only mandatory configuration is setting the **fabric_interface** parameter.

The fabric interface should be set to one of the InfiniBand IPoIB interfaces, which connect the UFM/SM to the fabric:

```
fabric_interface = ib0
```

Note

- By default, `fabric_interface` is set to `ib0`
- `fabric_interface` must be up and running before UFM startup. Otherwise, UFM will not be able to run.

For additional configuration options, please refer to the Optional Configurations.

Optional Configurations

Note

Unless noted otherwise, a UFM restart is required for configuration changes to take effect.

General Settings in gv.cfg

Configure general settings in the `conf/gv.cfg` file.

Note

When running UFM in HA mode, the gv.cfg file is replicated to the standby server.

Enabling SHARP Aggregation Manager

SHARP Aggregation Manager is enabled by default. To disable it, set:

```
[Sharp]
sharp_enabled = true
```

To disable this feature, set: `sharp_enabled = false`

A UFM restart is required for the change to take effect.

Enabling MAD Limiter

MAD Limiter is a security tool designed to mitigate Denial-of-Service (DoS) and Distributed Denial-of-Service (DDoS) attacks targeting the management node.

MAD Limiter is disabled by default. To enable it, set:

```
[MADLimiter]
mad_limiter_enabled = false
```

Enabling Predefined Groups

By default, pre-defined groups are enabled. In very large-scale fabrics, pre-defined groups can be disabled in order to allow faster startup of UFM.

```
enable_predefined_groups = true
```

Enabling Multi-NIC Host Grouping

Starting with UFM Enterprise v6.4.1, multi-NIC host grouping is enabled by default for fresh installations. However, for users upgrading from earlier versions, this feature remains disabled by default.

Note

It is recommended to set the value of this parameter before running UFM for the first time.

```
multinic_host_enabled = true
```

Defining Node Description Black-List

Note

Node descriptions from the black-list should not be used for Multi-NIC grouping.

During the process of host reboot or initialization/bring up, the majority of HCAs receive a default label rather than an actual, real description. To prevent the formation of incorrect multi-NIC groups based on these default labels, this feature offers the option to establish a blacklist containing possible node descriptions that should be avoided when grouping Multi-NIC HCAs during host startup. Once a legitimate node description is assigned to the host, the HCAs are organized into multi-NIC hosts based on their respective descriptions. It is recommended to configure this parameter before initiating the UFM for the first time.

For instance, nodes initially identified with descriptions listed in the `exclude_multinic_desc` will not be initially included in Multi-NIC host groups until they obtain an updated, genuine node description.

Modify the `exclude_multinic_desc` parameter in the cv.fg file:

```
exclude_multinic_desc = localhost,generic_name_1,generic_name_2
```

Running UFM Over IPv6 Network Protocol

The default multicast address is configured to an IPv4 address. To run over IPv6, this must be changed to the following in section UFMAgent of gv.cfg.

```
[UFMAgent]
...
# if ufmagent works in ipv6 please set this multicast address to
FF05:0:0:0:0:0:0:15F
mcast_addr = FF05:0:0:0:0:0:0:15F
```

Configuring SM Plugins via UFM

UFM allows users to configure Subnet Manager (SM) plugins through the `event_plugin_name` and `event_plugin_options` parameters in the configuration file. When UFM starts the SM, it automatically launches the specified plugins with the provided options.

To define the SM plugins, use the following parameter:

```
# Event plugin name(s)
event_plugin_name osmufmpi lossymgr
```

Add the plug-in options file to the `event_plugin_options` option:

```
# Options string that would be passed to the plugin(s)
event_plugin_options --lossy_mgr -f <lossy-mgr-options-file-name>
```

Note

These plugin settings are applied only when UFM is operating in **Management mode** and is copied to the `opensm.conf` file accordingly.

Multi-port SM

SM can use up to eight-port interfaces for fabric configuration. These interfaces can be provided via `/opt/ufm/conf/gv.cfg`. The users can specify multiple IPoIB interfaces or bond interfaces in /opt/ufm/conf/gv.cfg, subsequently, the UFM translates them to GUIDs and adds them to the SM configuration file (/opt/ufm/conf/opensm/opensm.conf). If users specify more than eight interfaces, the extra interfaces are ignored.

```
[Server]

# disabled (default) | enabled (configure opensm with multiple
GUIDs) | ha_enabled (configure multiport SM with high
availability)
 multi_port_sm = disabled
 # When enabling multi_port_sm, specify here the additional
fabric interfaces for OpenSM conf
 # Example: ib1,ib2,ib5 (OpenSM will support the first 8 GUIDs
where first GUID will
 # be extracted the fabric_interface, and remaining GUIDs from
additional_fabric_interfaces
 additional_fabric_interfaces =
```

Note

UFM treats bonds as a group of IPoIB interfaces. So, for example, if bond0 consists of the interfaces ib4 and ib8, then expect to see GUIDs for ib4 and ib8 in opensm.conf.

Note

Duplicate interface names are ignored (e.g. ib1,ib1,ib1,ib2,ib1 = ib1,ib2).

Configuring UDP Buffer

This section is relevant only in cases where `telemetry_provider=ibpm`. (By default, telemetry_provider=telemetry).

To work with large-scale fabrics, users should set the set_udp_buffer flag under the [IBPM] section to "yes" for the UFM to set the buffer size (default is "no").

```
# By deafult, UFM does not set the UDP buffer size. For large
scale fabrics
# it is recommended to increase the buffer size to 4MB (4194304
bits).
set_udp_buffer = yes
# UDP buffer size
udp_buffer_size = 4194304
```

Virtualization

The virtualization feature allows for supporting virtual ports in UFM.

```
[Virtualization]
# By enabling this flag, UFM will discover all the virtual ports
assigned for all hypervisors in the fabric
enable = false
# Interval for checking whether any virtual ports were changed in
the fabric
interval = 60
```

Static SM LID

Users may configure a specific value for the SM LID so that the UFM SM uses it upon UFM startup.

```
[SubnetManager]
# 1- Zero value (Default): Disable static SM LID functionality
and allow the SM to run with any LID.
#    Example: sm_lid=0
# 2- Non-zero value: Enable static SM LID functionality so SM
will use this LID upon UFM startup.
sm_lid=0
```

Note

To configure an external SM (UFM server running in sm_only mode), users must manually configure the opensm.conf file (`/opt/ufm/conf/opensm/opensm.conf`) and align the value of `master_sm_lid` to the value used for `sm_lid` in gv.cfg on the main UFM server.

Configuring Log Rotation

This section enables setting up the log files rotate policy. By default, log rotation runs once a day by cron scheduler.

```
[logrotate]
#max_files specifies the number of times to rotate a file before
it is deleted (this definition will be applied to
#SM and SHARP Aggregation Manager logs, running in the scope of
UFM).
#A count of 0 (zero) means no copies are retained. A count of 15
means fifteen copies are retained (default is 15)
max_files = 15
#With max_size, the log file is rotated when the specified size
is reached (this definition will be applied to
#SM and SHARP Aggregation Manager logs, running in the scope of
UFM). Size may be specified in bytes (default),
#kilobytes (for example: 100k), or megabytes (for example: 10M).
if not specified logs will be rotated once a day.
max_size = 3
```

Configuring UFM Logging

The [Logging] section in the gv.cfg enables setting the UFM logging configurations.

For more information, refer to UFM Startup Logging.

Field	Default Value	Value Options	Description
`level`	WARNING	CRITICAL, ERROR, WARNING, INFO, DEBUG	The definition components.
`smclient_level`	WARNING	CRITICAL, ERROR, WARNING, INFO, DEBUG	The logging le

Field	Default Value	Value Options	Description
`event_log_level`	INFO	CRITICAL, ERROR, WARNING, INFO, DEBUG	The logging le
`rest_log_level`	INFO	CRITICAL, ERROR, WARNING, INFO, DEBUG	The Logging le
`authentication_service_log_level`	INFO	CRITICAL, ERROR, WARNING, INFO, DEBUG	The logging le
`startup_logging_enabled`	False	True,False	When this par startup progre and status. This creates s `/opt/ufm/f` startup logs), `/opt/ufm/f enterprise` and `/opt/ufm/f modelmain.` (persistent pro The logging pr process with t updates. When disable startup proces
`startup_logging_log_level`	INFO	CRITICAL, ERROR, WARNING, INFO, DEBUG	The logging le

Field	Default Value	Value Options	Description
log_dir	/opt/ufm/files/log	N/A	It is possible t directory. The configure permission fo UFM should b between the U of the default creation and i
max_history_lines	100000	N/A	The maximum UI output for U
stream_all_events	FALSE	TRUE, FALSE	Enabling this p logs and syslo alarmable.

```
[Logging]
# Optional logging levels: CRITICAL, ERROR, WARNING, INFO, DEBUG.
level = WARNING
smclient_level = WARNING
event_log_level = INFO
rest_log_level = INFO
authentication_service_log_level = INFO
# The configured log_dir must have read, write and execute
permission for ufmapp user (ufmapp group).
log_dir = /opt/ufm/files/log
max_history_lines = 100000
stream_all_events = FALSE
```

Configuring UFM Over Static IPv4 Address

Follow this procedure to to run UFM on a static IP configuration instead of DHCP:

1. Modify the defined management Ethernet interface network script to be static. Run:

```
# vi /etc/sysconfig/network-scripts/ifcfg-enp1s0
```

Update the required interface with the static IP configuration (IP address, netmask, broadcast, and gateway):

```
NAME="enp1s0"DEVICE="enp1s0"
ONBOOT="yes"
BOOTPROTO="static"
IPADDR="10.209.37.153"
NETMASK="255.255.252.0"
BROADCAST="10.209.39.255"
GATEWAY="10.209.36.1"
TYPE=Ethernet
DEFROUTE="yes"
```

2. Add host entries to the /etc/hosts file. Run:

```
# vi /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4
localhost4.localdomain4
::1         localhost localhost.localdomain localhost6
localhost6.localdomain6

10.209.37.153 <hostname>
```

3. Check hostname. Run:

```
# vi /etc/hostname
<hostname>
```

4. Set up DNS resolution at /etc/resolv.conf. Run:

```
# vi /etc/resolv.conf
search mtr.labs.mlnx
nameserver 8.8.8.8
```

5. Restart network service. Run:

```
service network restart
```

6. Check Configuration. Run:

```
# hostname
<hostname>
# hostname -i
10.209.37.153
```

Configuring Syslog

This configuration enables the UFM to send log messages to syslog, including remote syslog. The configuration described below is located in the [Logging] section of the gv.cfg file.

Field	Default Value	Value Options	Desc
`syslog`	false	True or False	Enab syslo

Field	Default Value	Value Options	Desc
`syslog_addr`	/dev/log # for remote rsyslog_hostname:514	N/A	UFM (syslo For w syslo /dev/l exter value host:p the d serve As th could serve confi remo The /e shoul secti uncor below syslo recep imud 514# recep imtcp 514 Resta servi `serv rest`
`ufm_syslog`	false	True or False	Sets main mess send.
`smclient_syslog`	false	True or False	Sets Open mess send.

Field	Default Value	Value Options	Desc
`event_syslog`	false	True or False	Sets event False Send
`rest_syslog`	false	True or False	Sets UFM mess send.
`authentication_syslog`	false	True or False	Set s authe mess send.
`syslog_level`	WARNING	CRITICAL, ERROR, WARNING, INFO, DEBUG	Sets mess The s comr comp The s sent highe level level secti
`syslog_facility`	LOG_USER	LOG_KERN, LOG_USER, LOG_MAIL, LOG_DAEMON, LOG_AUTH, LOG_SYSLOG, LOG_LPR, LOG_NEWS, LOG_UUCP ,LOG_CRON,LOG_AUTHPRIV, LOG_FTP, LOG_NTP,LOG_SECURITY, LOG_CONSOLE, LOG_SOLCRON	Inclu syslo value facilit

```
syslog = false
#syslog configuration (syslog_addr)
# For working with local syslog, set value to: /dev/log
# For working with external machine, set value to: host:port
syslog_addr = /dev/log
# The configured log_dir must have read, write and execute
permission for ufmapp user (ufmapp group).
log_dir = /opt/ufm/files/log
# Main ufm log.
ufm_syslog = false
smclient_syslog = false
event_syslog = false
rest_syslog = false
authentication_syslog = false
syslog_level = WARNING
# Syslog facility. By default - LOG_USER
#  possible facility codes:
LOG_KERN,LOG_USER,LOG_MAIL,LOG_DAEMON,LOG_AUTH,LOG_SYSLOG,
#  LOG_LPR,LOG_NEWS,LOG_UUCP,LOG_CRON,LOG_AUTHPRIV,LOG_FTP,
LOG_NTP,LOG_SECURITY,LOG_CONSOLE,LOG_SOLCRON
# for reference https://en.wikipedia.org/wiki/Syslog
syslog_facility = LOG_USER
```

Excluding Unhealthy Ports from Fabric Health Report

In gv.cfg file there is a section named **UnhealthyPorts** and parameters in this section are used for unhealthy ports managing in UFM.

Unhealthy port state could be defined by used using UI or REST API request or reported by OpenSM or ibutilities.

UFM has an ability to check periodically fabric ports healthiness and to report unhealthy ports out or to perform automatically predefined isolation action for unhealthy ports.

In addition, using `exclude_unhealthy_ports` key in **UnhealthyPorts** section unhealthy ports could be excluded from ibdiagnet report.

By default, the value for this parameter is set as *false*. It means that unhealthy ports will appear in ibdiagnet reports, but if need to exclude unhealthy port from ibdiagnet reports

this parameter should be set to true and UFM server should be restarted so this action will take effect.

UFM starting flow will configure indiagnet configuration file with appropriate parameters and unhealthy ports will not appear in UFM health and Fabric health reports.

```
[UnhealthyPorts]
enable_ibdiagnet = true
log_level = INFO
syslog = false
# scheduling_mode possible values: fixed_time/interval.
# If fixed_time - ibdiagnet will run every 24 hours on the
specified time - <fixed_time>.
# If interval - ibdiagnet will run first time after <start_delay>
minutes from UFM startup and every <interval> hours (default
scheduling mode).
scheduling_mode = interval
# First ibdiagnet start delay interval (minutes)
start_delay = 5
# ibdiagnet run interval (hours)
interval = 3
# ibdiagnet run at a fixed time (example: 23:17:35)
fixed_time = 23:00:00
# By enabling this flag all the discovered high ber ports will be
marked as unhealthy automatically by UFM
high_ber_ports_auto_isolation = false
# Auto isolation mode - which type of ports should be isolated.
# Options: switch-switch,switch-host,all (default: switch-switch).
auto_isolation_mode = switch-switch
# Trigger Partial Switch ASIC Failure whenever number of
unhealthy ports exceed the defined percent of the total number of
the switch ports.
switch_asic_fault_threshold = 20
# exclude unhealthy ports from ibdiagnet reports
exclude_unhealthy_ports=false
```

Configuration Examples in gv.cfg

The following show examples of configuration settings in the gv.cfg file:

- Polling interval for Fabric Dashboard information

```
ui_polling_interval = 30
```

- [**Optional**] UFM Server local IP address resolution (by default, the UFM resolves the address by gethostip). UFM Web UI should have access to this address.

```
ws_address = <specific IP address>
```

- HTTP/HTTPS Port Configuration

```
# WebServices Protocol (http/https) and Port
ws_port = 8088
ws_protocol = http
```

- Connection (port and protocol) between the UFM server and the APACHE server

```
ws_protocol = <http or https>
ws_port = <port number>
```

For more information, see Launching a UFM Web UI Session.

- SNMP `get-community` string for switches (fabric wide or per switch)

```
# default snmp access point for all devices
[SNMP]
port = 161
gcommunity = public
```

- Enhanced Event Management (Alarmed Devices Group)

```
[Server]
auto_remove_from_alerted = yes
```

- Log verbosity

```
[Logging]
# optional logging levels
#CRITICAL, ERROR, WARNING, INFO, DEBUG
level = INFO
```

For more information, see UFM Logs.

- Settings for saving port counters to a CSV file

```
[CSV]
write_interval = 60
ext_ports_only = no
```

- Max number of CSV files (UFM Advanced)

```
[CSV]
max_files = 1
```

> **Note**
>
> The access credentials that are defined in the following sections of the conf/gv.cfg file are used only for initialization:
>
> - SSH_Server
> - SSH_Switch
> - TELNET
> - IPMI
> - SNMP
> - MLNX_OS
>
> To modify these access credentials, use the UFM Web UI. For more information, see "Device Access".

- Configuring the UFM communication protocol with MLNX-OS switches. The available protocols are:
  - http
  - https (default protocol for secure communication)

F

➢ or configuring the UFM communication protocol after fresh installation and prior to the first run, set the MLNX-OS protocol as shown below.

Example:

```
[MLNX_OS]
protocol = https
port = 443
```

Once UFM is started, all UFM communication with MLNX-OS switches will take place via the configured protocol.

➢ ***For changing the UFM communication protocol while UFM is running, perform the following:***

1. Set the desired protocol of MLNX-OS in the conf/gv.cfg file (as shown in the example above).

2. Restart UFM.

3. Update the MLNX-OS global access credentials configuration with the relevant protocol port. Refer to Device Access for help.

   For the http protocol - default port is 80.

   For the https protocol - default port is 443.

4. Update the MLNX-OS access credentials with the relevant port in all managed switches that have a valid IP address.

Managing Dynamic Telemetry

The management of dynamic telemetry instances involves the facilitation of user requests for the creation of multiple telemetry instances. As part of this process, the UFM enables users to establish new UFM Telemetry instances according to their preferred counters and configurations. These instances are not initiated by the UFM but rather are monitored for their operational status through the use of the UFM Telemetry bring-up tool

For more information on the supported REST APIs, please refer to UFM Dynamic Telemetry Instances REST API.

The configuration parameters can be found in the gv.cfg configuration file under the `DynamicTelemetry` section.

Name	Description	De val
max_instances	Maximum number of simultaneous running UFM Telemetries.	5

Name	Description	De val
new_instance_delay	Delay time between the start of two UFM Telemetry instances, in minutes.	5
update_discovery_delay	The time to wait before updating the discovery file of each telemetry instance if the fabric has changed, in minutes.	10
endpoint_timeout	Telemetry endpoint timeout, in seconds.	5
bringup_timeout	Telemetry bringup tool timeout, in seconds.	60
initial_exposed_port	Initial port for the available range of ports (range(initial_exposed_port, initial_exposed_port + max_instances)).	90(
instances_sessions_compatibility_interval	Every instances_sessions_compatibility_interval minutes the UFM verifies compliance between instances and sessions to avoid zombie sessions. if 0 is configured this process won't be activate	10

SM Trap Handler Configuration

The SMTrap handler is the SOAP server that handles traps coming from OpenSM.

There are two configuration values related to this service:

- `osm_traps_debounce_interval` – defines the period the service holds incoming traps
- `osm_traps_throttle_val` – once `osm_traps_debounce_interval` elapses, the service transfers `osm_traps_throttle_val` to the Model Main

Note

By default, the SM Trap Handler handles up to 1000 SM traps every 10 seconds.

Setting CPU Affinity on UFM

This feature allows setting the CPU affinity for the major processes of the UFM (such as ModelMain, SM, SHARP, Telemetry).

In order to increase the UFM's efficiency, the number of context-switches is reduced. When each major CPU is isolated, users can decrease the number of context-switches, and the performance is optimized.

The CPU affinity of these major processes is configured in the following two levels:

- Level 1- The major processes initiation.
- Level 2- Preceding initiation of the model's main subprocesses which automatically uses the configuration used in level 1 and designates a CPU for each of the sub-processes.

According to user configuration, each process is assigned with affinity.

By default, this feature is disabled. In order to activate the feature, configure `Is_cpu_affinity_enabled` with true, check how many CPUs you have on the machine, and set the desired affinity for each process.

For example:

```
[CPUAffinity]
Is_cpu_affinity_enabled=true
Model_main_cpu_affinity=1-4
Sm_cpu_affinity=5-13
SHARP_cpu_affinity=14-22
Telemetry_cpu_affinity=22-23
```

The format should be a comma-separated list of CPUs. For example: 0,3,7-11.

The ModelMain should have four cores, and up to five cores. The SM should have as many cores as you can assign. You should isolate between the ModelMain cores and the SM cores.

SHARP can be assigned with the same affinity as the SM. The telemetry should be assigned with three to four CPUs.

Quality of Service (QoS) Support

Infiniband Quality of Service (QoS) is disabled by default in the UFM SM configuration file.

To enable it and benefit from its capabilities, set the qos flag to TRUE in the /opt/ufm/files/conf/opensm/opensm.conf file.

Example:

```
# Enable QoS setup
qos FALSE
```

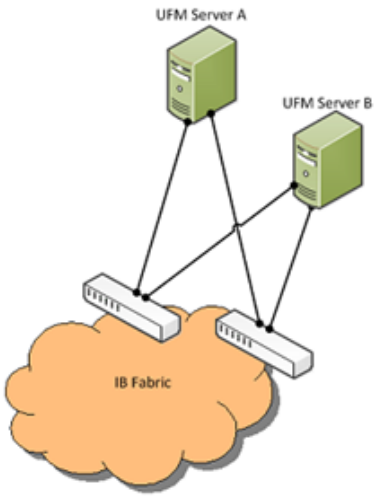
> **Note**
>
> The QoS parameters settings should be carefully reviewed before enablement of the qos flag. Especially, sl2vl and VL arbitration mappings should be correctly defined.

For information on Enhanced QoS, see Appendix – Used Ports.

UFM Failover to Another Port

When the UFM Server is connected by two or more InfiniBand ports to the fabric, you can configure UFM Subnet Manager failover to one of the other ports. When failure is detected on an InfiniBand port or link, failover occurs without stopping the UFM Server or other related UFM services, such as mysql, http, DRDB, and so on. This failover process prevents failure in a standalone setup, and preempts failover in a High Availability setup, thereby saving downtime and recovery.

Network Configuration for Failover to IB Port



To enable UFM failover to another port:

- Configure bonding between the InfiniBand interfaces to be used for SM failover. In an HA setup, the UFM active server and the UFM standby server can be connected differently; but the bond name must be the same on both servers.

- Set the value of fabric_interface to the bond name. using the /opt/ufm/scripts/*change_fabric_config.sh* command as described in Configuring General Settings in gv.cfg. If ufma_interface is configured for IPoIB, set it to the bond name as well. These changes will take effect only after a UFM restart. For example, if bond0 is configured on the ib0 and ib1 interfaces, in gv.cfg, set the parameter fabric_interface to bond0.

- If IPoIB is used for UFM Agent, add bond to the ufma_interfaces list as well.

When failure is detected on an InfiniBand port or link, UFM initiates the give-up operation that is defined in the Health configuration file for OpenSM failure. By default:

- UFM discovers the other ports in the specified bond and fails over to the first interface that is up (SM failover)

- If no interface is up:

    - In an HA setup, UFM initiates UFM failover

    - In a standalone setup, UFM does nothing

If the failed link becomes active again, UFM will select this link for the SM only after SM restart.

Configuring Managed Switches Info Persistency

UFM uses a periodic system information-pulling mechanism to query managed switches inventory data. The inventory information is saved in local JSON files for persistency and tracking of managed switches' status.

Upon UFM start up, UFM loads the saved JSON files to present them to the end user via REST API or UFM WebUI.

After UFM startup is completed, UFM pulls all managed switches data and updates the JSON file and the UFM model periodically (the interval is configurable). In addition, the JSON files are part of UFM system dump.

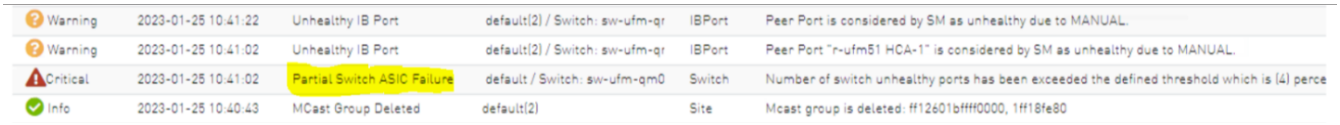
The following parameters allow configuration of the feature via gv.cfg fie:

```
[SrvMgmt]
# how often UFM should send json requests for sysinfo to switches
(in seconds)
systems_poll = 180
# To create UFM model in large setups might take a lot of time.
# This is an initial delay (in minutes) before starting to pull
sysinfo from switches.
systems_poll_init_timeout = 5
# to avoid sysinfo dump overloading and multiple writing to host
# switches sysinfo will be dumped to disc in json format every
set in this variable
# sysinfo request. If set to 0 - will not be dumped, if set to 1 -
will be dumped every sysinfo request
# this case (as example defined below) dump will be created every
fifth sysinfo request, so if system_poll is 180 sec (3 minutes)
sysinfo dump to the file will e performed every 15 minutes.
sysinfo_dump_interval = 5
# location of the sysinfo dump file (it is in /opt/ufm/files/logs
(it will be part of UFM dump)
sysinfo_dump_file_path = /opt/ufm/files/log/sysinfo.dump
```

Configuring Partial Switch ASIC Failure Events

UFM can identify switch ASIC failure by detecting pre-defined portion of the switch ports, reported as unhealthy. By default, this portion threshold is set to 20% of the total switch ports. Thus, the UFM will trigger the partial switch ASIC event in case the number of unhealthy switch ports exceeds 20% of the total switch ports.

You can configure UFM t o control Partial Switch ASIC Failure events. To configure, you may use the gv.cfg file by updating the value of `switch_asic_fault_threshold` parameter under the UnhealthyPorts section. For an example, in case the switch has 32 ports, once 7 ports are detected as unhealthy ports, the UFM will trigger the partial switch ASIC event. Example:



Enabling Network Fast Recovery

> **Note**
>
> To enable the Network Fast Recovery feature, ensure that all switches in the fabric use the following MLNX-OS/firmware versions:
>
> - MLNX-OS version 3.10.6004 and up
> - Quantum firmware versions:
>   - Quantum FW v27.2010.6102 and up
>   - Quantum2 FW v31.2010.6102 and up

Fast recovery is a switch-firmware based facility for isolation and mitigation of link-related issues. This system operates in a distributed manner, where each switch is programmed with a simple set of rule-based triggers (conditions) and corresponding action protocols. These rules permit the switch to promptly react to substandard links within its locality, responding at a very short reaction time - as little as approximately 100 milliseconds. The policy is provided and managed via the UFM and SM channel. Moreover, every autonomous action taken by a switch in the network is reported to the UFM.

The immediate reactions taken by the switch enable SHIELD and pFRN. These mechanisms collaborate to rectify routing within the proximity of the problematic link before it can disrupt transactions at the transport layer. Importantly, this process occurs rapidly, effectively limiting the spreading of congestion to a smaller segment of the network.

To enable the Network Fast Recovery feature, you must activate the appropriate trigger (condition) in the `gv.cfg` file. This allows you to specify which of the four supported triggers UFM should handle.

Additionally, you can configure how UFM responds when a port becomes unhealthy due to a Network Fast Recovery condition:

- `no_reset` – Monitor only, without taking any action.
- `physical_reset` or `logical_reset` – Monitor and respond by performing a reset.

The selected behavior will apply to all configured conditions.

As stated in the gv.cfg file, the feature is disabled by default and the below are the supported fields and options:

```
[NetworkFastRecovery]
# Fast Recovery configuration.
# Supported values:
#    0: Ignore fast recovery related MADs and configuration
(default)
#    1: Disable fast recovery
#    2: Enable fast recovery
fast_recovery_mode = 0

# This will be supported by the Network Fast Recovery.
network_fast_recovery_conditions =
SWITCH_DECISION_CREDIT_WATCHDOG,SWITCH_DECISION_RAW_BER,SWITCH_DEC
# Network Fast Recovery action configuration.
# Supported values:
#    no_reset: Only monitor the conditions (default)
#    physical_reset : Monitor and react by performing physical
reset
#    logical_reset : Monitor and react by performing logical
reset
network_fast_recovery_action = no_reset
```

> **Note**
>
> To enable the Network Fast Recovery feature, the value of `fast_recovery_mode` should be set to 2. For the change to take effect, restart of UFM Enterprise is required.

Parameter	Description
`SWITCH_DECISION_CREDIT_WATCHDOG`	The Switch decided to close the port due to Credit watchdog
`SWITCH_DECISION_RAW_BER`	The Switch decided to close the port due to High raw errors
`SWITCH_DECISION_EFFECTIVE_BER`	The Switch decided to close the port due to High effective errors (after FEC)
`SWITCH_DECISION_SYMBOL_BER`	The Switch decided to close the port due to High symbol errors (after PLR)

By default, the Network Fast Recovery feature operates in monitoring mode. This means the switch does not reset ports, however, it reports issues related to them. To view these port-related issues, you must deploy the UFM PMC (Packet Monitoring Collector) plugin and use its UI to access the relevant network events.

For more details on the PMC plugin, including deployment instructions and how to view Network Fast Recovery events, please refer to Packet Level Monitoring Collector (PMC) Plugin.

Disabling Rest Roles Access Control

By default, the Rest Roles Access Control feature is enabled. It can be disabled by setting the `roles_access_control_enabled` flag to false:

```
[RolesAccessControl]
roles_access_control_enabled = true
```

Enabling/Disabling Authentication

Kerberos Authentication

By default, Kerberos Authentication is disabled. To enable it, set the `kerberos_auth_enabled` flag to true. Additionally, provide the required configurations such as `kerberos_cred_key_path`, `kerberos_use_local_name` and `kerberos_auto_sign_up`.

```
[KerberosAuth]
# This section responsible to manage kerberos authentication
# Set to true to enable the kerberos auth feature, and set to false
to disable it. Default is false.
kerberos_auth_enabled = false
# The path of the keytab file containing credentials for GSSAPI
authentication.
kerberos_cred_key_path = /etc/kadm5.keytab
# Set to true to configure the Apache server to map authenticated
principal names (which represent different clients) to local
usernames,
# and set to false to use the principle names as usernames. Default
is true (this value will be reflected in the 'GssapiLocalName' directive
in Apache).
kerberos_use_local_name = true
# Set to true to enable auto sign up of users who do not exist in
UFM DB. Default is true.
kerberos_auto_sign_up = true
# The default role assigned to create users if they do not exist when
'kerberos_auto_sign_up' is set to true.
kerberos_default_role = System_Admin
```

kerberos_auth_enabled: By default, Kerberos authentication remains disabled. To activate it, the user must set this flag to 'true' and then restart UFM.

kerberos_cred_key_path: This specifies the path to the keytab file containing credentials for GSSAPI authentication.

kerberos_use_local_name: Set to true to configure the Apache server to map authenticated principal names (which represent different clients) to local usernames, and set to false to use the principal names as usernames. Default is true (this value will be reflected in the '`GssapiLocalName`' directive in Apache).

kerberos_auto_signup: For successful authentication via Kerberos, the user must already exist within the UFM database, otherwise, the authentication will be refused by UFM. If this property is set to 'true,' UFM will create the non-existing users in the UFM DB.

kerberos_default_role: The default role is assigned to create users if they do not exist when '`kerberos_auto_sign_up`' is set to true.

Finally, restart the UFM to use Kerberos authentication.

UFM Authentication Server

By default, UFM Authentication Server is enabled. To disable it, you need to set the "`auth_service_enabled`" parameter to '`false`' and then restart the UFM service to initiate the authentication server. Additionally, you can use enable/disable flags for Basic, Session, and Token authentication:

```
[AuthService]
auth_service_enabled = true
auth_service_interface = 127.0.0.1
auth_service_port = 8087 # the serving port for the authentication
server
basic_auth_enabled = true
session_auth_enabled = true
token_auth_enabled = true
```

Azure AD Authentication

By default, Azure AD Authentication is disabled. To enable it, set the `azure_auth_enabled` flag to 'true'. Additionally, provide the required configurations from the Azure AD Application such as TENANT_ID, CLIENT_ID and CLIENT_SECRET which can be found under the "**Overview**" section of the registered application in the Azure portal. Finally, the UFM Authentication Server should be enabled to use the Azure AD Authentication.

```
[AzureAuth]
azure_auth_enabled = false
# TENANT ID of app registration
TENANT_ID =
# Application (client) ID of app registration
CLIENT_ID =
# Application's generated client secret
CLIENT_SECRET =
```

Changing Maximum SSL Request Size when Using Client Certificate

1. Activate Client Certificate Authentication. For more details, refer to the Client Authentication REST API.

2. In the Server section in the `gv.cfg` file, there is a configuration option for controlling the maximum request size when using client certificates:

    - The maximum request size, specified in bytes, is set to a default value of 1,572,864 (1536 KB / 1.5 MB). If not explicitly defined, the system will default to Apache's value of 131,072 bytes (128 KB).

    - `max_ssl_request_size = 1572864`

Note

- This configuration is expressed in bytes.
- The **Setting Up SSL and CA Certificates in UFM** and **Changing Maximum SSL Request Size when Using Client Certificate** features cannot be used simultaneously. Removing, updating, or configuring will delete the certificates.

Adjusting UFM Configuration Files Based on Fabric Size

This function allows users to automate the process of updating the UFM configuration files by parsing a primary configuration file called `large_scale_subnet.cfg` file and applying the values to multiple target files or resetting to default values using the small_scale_subnet.cfg.

The below are instructions on how to use a Python script to parse a configuration file (`large_scale_subnet.cfg`) and update the values of specific parameters in multiple target UFM configuration files (`gv.cfg, reports.cfg, opensm.cfg` , and `sharp_am.cfg`). The script can operate in two modes:

- **Large Scale Subnet Mode**: This mode directly updates the UFM configuration files based on the parsed configuration from the `large_scale_subnet.cfg` file.
- **Small Scale Subnet (Default) Mode**: Sets the UFM configuration files to their default values by parsing the `small_scale_subnet.cfg file` .

Configuration File and Parameters

The primary configuration file contains all the parameters, and their values must be updated over the multiple UFM configuration files.

Primary Configuration Files

- `/opt/ufm/files/conf/large_scale_subnet.cfg`

- `/opt/ufm/files/conf/small_scale_subnet.cfg`

Target UFM Configuration Files

- `/opt/ufm/files/conf/gv.cfg`
- `/opt/ufm/files/conf/reports.cfg`
- `/opt/ufm/files/conf/opensm/opensm.conf`
- `/opt/ufm/files/conf/sharp/sharp_am.cfg`

Example structure of `large_scale_subnet.cfg` and `small_scale_subnet.cfg`:

```
[GV]
[GV.Server]
# disabled (default) | enabled (configure opensm with multiple
GUIDs) | ha_enabled (configure multiport SM with high
availability).
multi_port_sm = ha_enabled
# report_events that will determine which trap to send  to ufm
all/security/none
report_events = security

[GV.FabricAnalysis]
# initial_delay (in minutes) - the initial delay for running
fabric analysis for the first time after UFM was started
initial_delay = 10


[GV.logrotate]
#max_files specifies the number of times to rotate a file before
it is deleted.
#A count of 0 (zero) means no copies are retained. A count of 10
means fifteen copies are retained (default is 10)
max_files = 10

[REPORTS]
[REPORTS.FabricHealth]
# Fabric health report timeout
timeout = 1800

[REPORTS.TopologyCompare]
# Topology compare report timeout
timeout = 1800

[REPORTS.FabricAnalysis]
# Fabric analysis report timeout
timeout = 1800
```

```
[OPENSM]
routing_threads_num = 24

# Max number of threads that are allowed to run
# on the same processor during parallel computing
# If 0, threads assignment per processor is up to
# operating system initial assignment
max_threads_per_core = 1

# 0 - No reports in SA
reports = 0

[SHARP]
lst_file_timeout = 10
lst_file_retries = 30
```

Script Usage Example in the CLI:

- **Large Scale Subnet Mode:**
  `/opt/ufm/scripts/set_ufm_scale_profile.sh --mode large_scale_subnet --force_update`
- **Small Scale Subnet (Default) Mode:**
  `/opt/ufm/scripts/ set_ufm_scale_profile.sh --mode small_scale_subnet`

The `force_update` script parameter adds any parameters found in `large_scale_subnet.cfg` and `small_scale_subnet.cfg` that are not present in the UFM configuration files. For example, if a user adds a new parameter called `test_param` = 500 to the `large_scale_subnet.cfg` file under the [Server] section and this parameter does not exist in the `gv.cfg` file, running the script with the `--force_update` option will add `test_param` = 500 to the [Server] section of the `gv.cfg file.`

Expected Output

1. **Large Scale Subnet Mode:**

- The script reads `large_scale_subnet.cfg`.
- It updates the parameters in the target UFM configuration files based on the parsed data.
- It logs messages for any skipped parameters.

2. Small Scale Subnet - Default Mode:

- The script reads `small_scale_subnet.cfg`.
- It updates the parameters in the target UFM configuration files based on the parsed data.
- It logs messages for any skipped parameters or adds the parameter to the configuration file if the `force_update` was True.

> **Note**
>
> **Note:** In case of the script running failure, the script will reset the UFM configuration files to their default values.

Setting Up SSL and CA Certificates in UFM

This feature allows you to set up SSL and CA certificates, issued and signed by a publicly trusted certificate authority (CA) in UFM .

To utilize this feature, please ensure you have the following prerequisites:

- UFM Docker image
- Certificate files:

- A certificate file named `server.crt`
- A key file named `server.key`
- An optional chain of authority file named `ca-intermediate.crt`

All of these files must be located in the directory `/opt/ufm/files/conf/webclient`.

Once these prerequisites are met, UFM will utilize the provided certificates.

> **Note**
>
> The **Setting Up SSL and CA Certificates in UFM** and **Changing Maximum SSL Request Size when Using Client Certificate** features cannot be used simultaneously. Removing, updating, or configuring will delete the certificates.

Cloud Profile Mode

The configuration model introduces the cloud_profile parameter with the following possible values:

- Mode 0 (Disabled): Default state with no additional security controls.
- Mode 1 (Bare Metal Profile): Perform extra validations on Pkey members, such as ensuring that all HCAs are assigned to the same Pkey.
- Mode 2 (Secured Bare Metal Profile): Enforces comprehensive security validation, auto-configuration, and advanced control mechanisms.

Cloud Profile

By setting the `cloud_profile` to 1, it will configure UFM to assign default management network memberships as *limited*. It also activates additional validations for PKey members, including checks to ensure all HCAs are associated with the same PKey and that there are no duplicate PKey assignments using index 0.

Secured Cloud Profile

By setting `cloud_profile` to `2`, UFM not only applies all the additional validations included in the Bare Metal Profile but also enables rigorous security controls tailored for bare-metal cloud environments. These include:

- Enforcing a restrictive PKEY posture, with the default management PKEY tightly limited.
- Enabling all SM MAD keys with a periodically randomized seed, including:

  `MKEY`, `VSKEY`, `CCKEY`, `Class C key (N2N)`, `SHARP job keys`, `SMKEY`, and `SAKEY`.
- Activating the Service Key mechanism for SHARP Aggregation Manager (AM) and future services.
- Enabling the enhanced SA trust model, introducing registration limits to mitigate security risks.
- Applying SA registration limitations with default values.
- Activating the MAD Limiter using default configurations to improve security.
- Enforcing MFT with M_Key as mandatory.
- Validating allowed SM GUIDs to restrict which ports can operate as Subnet Managers (SMs).
- Restoring original parameter values when the feature is disabled after being enabled.

To configure UFM running in Cloud Profile mode, perform the following:

1. Set the following parameter in the `gv.cfg` file to one of the supported values:

```
# Allow setting of cloud profile. Supported values:
# 0 - Disabled. (default)
# 1 - Bare metal profile (Perform extra validations on Pkey
members, such as ensuring that all HCAs are assigned to the
same Pkey.).
# 2 - Secured bare metal profile (UFM enforces a hardened
InfiniBand (IB) security posture suitable for bare-metal cloud
environments.).
cloud_profile = 2
```

2. Restart UFM.
3. If `cloud_profile` is set to `2`, ensure that the `allowed_sm_guids` parameter is configured in the `opensm.conf` file.

   The `allowed_sm_guids` key is mandatory when the secured bare-metal profile is enabled; otherwise, UFM will fail to start.

Faster Detection of Fan/PSU Removal on Unmanaged Switches

To enable quicker detection of fan and PSU module removals from unmanaged switches (e.g., MQM9790), update the following settings in the `gv.cfg` file:

```
enable_high_freq_fru_check = true
# Time interval between consecutive runs of fabric analysis for
unmanaged switches using minimal MADs to detect FAN and PSU
module removals.
# Setting a value below 1 minute will disable the feature.
high_freq_fru_interval_min = 2
```

After making these changes, restart UFM telemetry or the entire UFM service.

Enabling this feature allows UFM to periodically perform lightweight fabric analysis, resulting in faster indication of fan and PSU removals on unmanaged switches.

Setting UFM Configurations Without Requiring UFM Restart

This section outlines the UFM, OpenSM, and SHARP configuration parameters that can be modified during runtime without requiring a process restart.

UFM Configuration

The following UFM configuration parameters in the `/opt/ufm/files/conf/gv.cfg` file can be updated without restarting the UFM ModelMain process:

Section Name	Parameter Name

Logging	`dump_ibpm_packet`
	`event_log_level` `event_syslog` `rest_log_level` `rest_syslog`
	`smclient_level` `smclient_syslog` `level, ufm_syslog`
	`authentication_service_log_level` `syslog_addr` `syslog`
	`authentication_syslog` `syslog_level` `syslog_facility`
	`max_history_lines`

logrotate	`max_files, max_size`

DailyReport	`daily_report_enabled`
	`mail_send_interval` `attach_fabric_health_report`
	`start_hour` `end_hour` `send_email_method`

Multisubnet	`multisubnet_enabled`
	`multisubnet_chunk_size` `multisubnet_role`
	`multisubnet_site_name`

Section Name	Parameter Name
Server	`events_on_internal_ports` `inband_fw_upgrade_group_size` `dashboard_interval` `polling_errors_alerts_debounce_counter` `ufm_health_model_main_web_request_timeout`

OpenSM Configurations

The following OpenSM parameters in the `/opt/ufm/files/conf/opensm/opensm.conf` file can be changed without requiring a restart of the OpenSM process:

- `m_key, sm_key, sa_key, allowed_sm_guids, m_key_lease_period, m_key_protection_level, m_key_lookup, m_key_per_port`
- `sweep_interval, max_wire_smps, max_wire_smps2, max_smps_timeout, max_sa_reports_queued, max_sa_reports_on_wire`
- `sa_etm_allow_untrusted_proxy_requests, sa_etm_allow_untrusted_guidinfo_rec, sa_etm_allow_guidinfo_rec_by_vf, sa_etm_max_num_mcgs, sa_etm_max_num_srvcs, sa_etm_max_num_event_subs`
- `sa_rate_threshold, sa_check_sgid_spoofing, max_msg_fifo_timeout, sm_priority, qos_config_vl_enabled, max_op_vls, max_op_vls_ca, max_op_vls_sw, max_op_vls_rtr`
- `suppress_mc_pkey_traps, force_link_speed, force_link_speed_ext, force_link_speed_ext2, force_link_width, fdr10, support_mepi_speeds, mepi_enabled_speeds`
- `reassign_lids, ignore_other_sm, disable_multicast, subnet_timeout, packet_life_time, vl_stall_count, leaf_vl_stall_count, head_of_queue_lifetime, leaf_head_of_queue_lifetime`
- `local_phy_errors_threshold, overrun_errors_threshold, use_mfttop, sminfo_polling_timeout, polling_retry_number, force_heavy_sweep, port_profile_switch_nodes, sweep_on_trap`

- routing_engine, enable_queries_during_routing, connect_roots, calculate_missing_routes, max_cas_on_spine, find_roots_color_algorithm, dfp_find_roots_color_algorithm
- log_max_size, log_num_backlogs, log_flags, force_log_flush, accum_log_file, no_partition_enforcement, part_enforce, keep_pkey_indexes, sm_assigned_guid
- qos, suppress_sl2vl_mad_status_errors, override_create_mcg_sl, port_shifting, scatter_ports, updn_lid_tracking_mode, updn_lid_tracking_converge_routes
- updn_lid_tracking_prefer_total_routes, dfp_max_cas_on_spine, max_seq_redisc, aguid_inout_notice, sm_assign_guid_func, mc_primary_root_guid, mc_secondary_root_guid
- max_reverse_hops, routing_threads_num, max_threads_per_core, guid_routing_order_no_scatter, use_scatter_for_switch_lid, offsweep_balancing_enabled, offsweep_balancing_window
- sa_db_dump, sm_db_dump, torus_config, do_mesh_analysis, exit_on_fatal, honor_guid2lid_file, sm_inactive, babbling_port_policy, drop_subscr_on_report_fail
- drop_event_subscriptions, drop_unreachable_event_subscriptions, ipoib_mcgroup_creation_validation, mcgroup_join_validation, use_original_extended_sa_rates_only
- max_rate_enum, reports, use_optimized_slvl, use_optimized_port_mask_slvl, fsync_high_avail_files, default_mcg_mtu, default_mcg_rate
- qos_max_vls, qos_high_limit, qos_vlarb_high, qos_vlarb_low, qos_sl2vl, qos_ca_max_vls, qos_ca_high_limit, qos_ca_vlarb_high, qos_ca_vlarb_low, qos_ca_sl2vl
- qos_sw0_max_vls, qos_sw0_high_limit, qos_sw0_vlarb_high, qos_sw0_vlarb_low, qos_sw0_sl2vl, qos_swe_max_vls, qos_swe_high_limit, qos_swe_vlarb_high, qos_swe_vlarb_low, qos_swe_sl2vl

- `qos_sw2sw_max_vls, qos_sw2sw_high_limit, qos_sw2sw_vlarb_high, qos_sw2sw_vlarb_low, qos_sw2sw_sl2vl, qos_rtr_max_vls, qos_rtr_high_limit, qos_rtr_vlarb_high, qos_rtr_vlarb_low, qos_rtr_sl2vl`
- `mlnx_congestion_control, cc_key_enable, cc_key_lease_period, cc_key_protect_bit, vs_key_enable, vs_key_lease_period, vs_key_ci_protect_bits`
- `n2n_key_enable, n2n_key_lease_period, n2n_key_protect_bit, key_mgr_seed, enable_quirks, no_clients_rereg, client_rereg_mode, consolidate_ipv6_snm_req`
- `consolidate_ipv4_mask, lash_start_vl, sm_sl, log_prefix, max_msg_fifo_len, max_alt_dr_path_retries, sa_pr_full_world_queries_allowed`
- `hm_ports_health_policy_file, force_heavy_sweep_window, validate_smps, enable_inc_mc_routing, allow_sm_port_reset, port_speed_change_action, port_ext_speed_change_action`
- `port_ext_speed2_change_action, port_mepi_speed_change_action, port_mtu_change_action, port_vl_change_action, port_ame_bit_change_action`
- `support_mlnx_enhanced_link, mlnx_enhanced_link_enable, adaptive_timeout_sl_mask, mepi_cache_enabled, virt_max_ports_in_process, virt_default_hop_limit`
- `enable_virt_rec_ext, improved_lmc_path_distribution, syslog_log_flags, sweep_every_hup_signal, osm_stats_interval, osm_stats_dump_limit, osm_perflog_dump_limit`
- `enable_performance_logging, quasi_ftree_indexing, sharp_enabled, rtr_aguid_enable, rtr_pr_flow_label, rtr_pr_tclass, rtr_pr_sl, rtr_pr_mtu, rtr_pr_rate`
- `aguid_default_hop_limit, verbose_bypass_policy_file, enable_subnet_lst, dor_hyper_cube_mode, additional_gi_supporting_devices, additional_mepi_force_devices`

- `activity_report_subjects, enhanced_qos_vport0_unlimit_default_rl, adv_routing_engine, ar_mode, shield_mode, ar_sl_mask, enable_ar_by_device_cap, enable_ar_group_copy`
- `ar_transport_mask, ar_tree_asymmetric_flow, ar_tree_asymmetric_flow_threshold, ar_tree_asymmetric_flow_threshold_limit, routing_flags, dump_ar`
- `cache_ar_group_id, dfp_down_up_turns_mode, enable_vl_packing, topo_config_enabled, rtr_selection_function, rtr_selection_seed, rtr_selection_algo_parameters`
- `get_mft_tables, hbf_sl_mask, hbf_hash_type, hbf_seed_type, hbf_seed, hbf_hash_fields, hbf_weights, pfrn_sl, pfrn_mask_clear_timeout, pfrn_mask_force_clear_timeout`
- `pfrn_over_router_enabled, tenants_policy_enabled, reply_lid_smps_in_dr, respond_unknown_lid_traps, fabric_mode_profile, issu_mode, issu_timeout, issu_pre_upgrade_time`

SHARP Configuration

For parameters found in the `/opt/ufm/files/conf/sharp/sharp_am.cfg` file, no configuration changes can be applied during runtime. Any updates to this file require restarting the `sharp_am` process for changes to take effect.

Setting up Telemetry in UFM

Setting up telemetry deploys UFM Telemetry as bare metal on the same machine. Historical data is sent to SQLite database on the server and live data becomes available via UFM UI or REST API.

Enabling UFM Telemetry

The UFM Telemetry feature is enabled by default and the provider is the UFM Telemetry. The user may change the provider via flag in `conf/gv.cfg`

The user may also disable the History Telemetry feature in the same section.

```
[Telemetry]
history_enabled=True
```

Enabling UFM Telemetry Manager Plugin

UFM Telemetry can be managed via the UFM Telemetry Manager (UTM) Plugin. To enable UTM mode, deploy the plugin, set one or all of the following flags in `/opt/ufm/files/conf/gv.cfg` under the [Telemetry] section to `false` and restart UFM:

```
[Telemetry]
primary_telemetry_legacy_mode = true
secondary_telemetry_legacy_mode = true
dynamic_telemetry_legacy_mode = true
```

To edit UFM Telemetry configuration, edit `/opt/ufm/files/conf/telemetry_defaults/primary_env.cfg` for primary, `/opt/ufm/files/conf/secondary_telemetry_defaults/secondary_env.cfg` for secondary, `/opt/ufm/files/conf/dynamic_telemetry_defaults/dynamic_env.cfg` for dynamic.

Here is a list of available UTM Plugin APIs:

```
Command                    Method    Description
-------------------------  --------  -----------------------------
-------------------------------------------------------------------
-------------------------------------
/status                    GET       Return status of managed
telemetry setup
/status.html               GET       Return status in HTML format
/switches                  GET       Return all switches
/hcas                      GET       Return all HCAs
/guids                     GET       Return all GUIDS PORTS pairs
/add_server                GET       Add a new telemetry instance
at 'url' added to 'group'.
                                     Query parameters:
                                         url=[target ip,
required],
                                         group='default'
/remove_server             GET       Remove paused or running
telemetry instance at URL (required query parameter url=URL)
/pause_server              GET       Pause an existing telemetry
instance at URL (required query parameter url=URL)
/start_server              GET       Start a paused telemetry
instance at URL (required query parameter url=URL)
/restart_server            GET       Restart paused or running
telemetry instance at URL (required query parameter url=URL)
                                                 Apply new
configuration if query parameters are set:
                                                     xcset_name=
regenerate new config ini file for IB Telemetry
                                                     http_port=
replace http port
                                                     sample_rate=
update sample rate
/get_server_xcsets         GET       Get list of available xcsets
available on server at URL  (required query parameter url=URL)
```

```
/deploy_switch_agents     GET       Deploy agent image to
switches to run telemetry within it later
                                          ip_list=[required]    csv
list of target switches IPs or 'all' to deploy to all switches

                                          Examples:
                                              1. deploy to all the
managed switches:
                                                  curl -X GET
127.0.0.1:8888/deploy_switch_agents?ip_list=all
                                              2. deploy to switches
with IPs 127.0.1.1 and 127.0.1.2:
                                                  curl -X GET
127.0.0.1:8888/deploy_switch_agents?ip_list=127.0.1.1,127.0.1.2
/remove_switch_agents     GET       Remove deployed switch agents
and switch telemetry
                                          ip_list=[required]    csv
list of target switches IPs or 'all' to remove from all deployed
switches
/switch_mon_list          GET       Set IPs of managed switches
that will be periodically monitored.
                                          ip_list=[required]    csv
list of target switches IPs or 'all' to deploy to all switches

                                          Examples:
                                              1. deploy to all the
managed switches:
                                                  curl -X GET
127.0.0.1:8888/switch_mon_list?ip_list=all
                                              2. deploy to switches
with IPs 127.0.1.1 and 127.0.1.2:
                                                  curl -X GET
127.0.0.1:8888/switch_mon_list?ip_list=127.0.1.1,127.0.1.2
/managed_switches_status  GET       Get info in JSON-format about
managed switches in Distributed Telemetry.
                                          Query parameters:
```

```
                                                monitored_only=
[1|0]   show only monitored switches (the switches that were set
via /switch_mon_list endpoint).  Default 0.
/start_switch_telemetry   GET       (Re)configure and (re)start
switch telemetry. Query parameters:
                                        ip=[target ip],     set
switch IP to (re)start a single switch telemetry,
                                                            otherw
all the managed switches with installed
                                                            switch
agents will (re)start telemetry
                                        sample_rate,
                                        restart_every,
                                        counter_set,
                                        http_port,
                                        dt_udp_handshake,
                                        dt_udp_data_ack,
                                        gnmi
/stop_switch_telemetry    GET       Stop all running switch
telemetry instance
                                        ip=[target ip],     set
switch IP to stop a single switch telemetry,
                                                            otherw
all the running switch telemetries will be stopped
/set_switch_creds         POST      Set switch credentials.
                                    Required query parameters:
                                            ip=
                                            user=
                                            pass=
/xcset/all                GET       Get cached name-hash xcset
pairs
/xcset                    GET       Get xcset content by name
/xcset                    DELETE    Delete xcset from cache
/xcset                    POST      Add/overwrite cached xcset
                                    Example:
```

```
                                        curl -X POST
0.0.0.0:8888/xcset?name=test --data-binary @path/to/file.xcset
/filter                    POST      Post [x|f]set name to the
host telemetry instance. This data will be accessible via
/[xc,c,f]set/name info endpoint.
                                    Required args:
                                        name=[required]
set name
                                        ext=[required]
set extension (fset, cset, xcset)
                                        session_id=[required]
session ID of the host telemetry instance
                                    Examples:
                                        curl -X POST
0.0.0.0:8888/filter?name=test&ext=fset&session_id --data-binary
@path/to/file.fset
                                        curl -X POST
0.0.0.0:8888/filter?name=test&ext=cset&session_id --data-binary
@path/to/file.cset
                                        curl -X POST
0.0.0.0:8888/filter?name=test&ext=xcset&session_id --data-binary
@path/to/file.xcset
/host/create_telemetry     POST      Deploy and start internal
host telemetry.
                                    Note: works only when
telemetry bringup is installed to UTM image

                                    Required query parameters:
                                        hca=
                                        http_port=
                                        sample_rate=

                                    Optional query parameters:
                                        xcset_name=
regenerate new config ini file for IB Telemetry
```

```
                                        env_params=     csv line
of env parameters
                                        ibdiag_opts=    csv line
of ibdiag options
                                        group=          set
telemetry group name

                                    Optional POST parameters:
                                        env_file        path to
env file. Format is key=value per line.
                                                        Usage:
                                                              curl
... --data-binary  @path/to/file
/host/remove_telemetry    GET       Stop and remove host
telemetry.
                                    Note: works only when
telemetry bringup is installed to UTM image

                                    One of the following query
params is required:
                                        session_id=      to stop
by session ID
                                        group=           to stop
all instances from the group
/host/get_sessions        GET       Get session id to telemetry
instance mapping
/help.html                GET       Print help information in
HTML format
/help                     GET       Print help information
```

For example, to update an existing `cset` in UTM mode, get the `session_id` using `/host/get/sessions` API and update the cset with a post API:

```
curl -k -u admin:123456 -X POST "https://localhost/ufmRest/plugin/utm/filter?name=minimal&ext=cset&session_id=92" --data-binary @/opt/ufm/conf/telemetry/prometheus_configs/cset/minimal.cset
```

Changing Telemetry Bind Address

By default, UFM Telemetry will bind to 127.0.0.1. Users can modify this setting for both primary and secondary telemetry instances using the appropriate flags in `conf/gv.cfg`.

```
[Telemetry]
primary_ip_bind_addr=0.0.0.0
secondary_ip_bind_addr=0.0.0.0
```

Changing Telemetry Endpoint Protocol

The default protocol for the Telemetry endpoint is HTTP. However, this can be switched to HTTPS by meeting two conditions:

1. Ensure that the certificate files, named `server.key` and `server.crt`, are present in the following directories:
    - For bare-metal UFM users: `/var/opt/ufm/webclient`
    - For Docker/Appliance UFM users: `/opt/ufm/files/conf/webclient`
2. Modify the protocol setting in the configuration file `conf/gv.cfg` under the `[Telemetry]` section:

    ```
    [Telemetry]
    prometheus_protocol = https
    ```

Changing UFM Telemetry Default Configuration

There is an option to configure parameters on a telemetry configuration file which takes effect after restarting the UFM or failover in HA mode.

The `launch_ibdiagnet_config.ini` default file is located under `/opt/ufm/conf/telemetry_defaults` and is copied to the telemetry configuration location ((`/opt/ufm/conf/telemetry`) upon startup UFM.

All values taken from the default file take effect at the deployed configuration file except for the following:

Note that normally the user does not have to do anything and they get two pre-configured instances – one for low frequency and one for higher-frequency sampling of the network.

Value	Description
`hca`	-
`scope_file`	-
`plugin_env_PROMETHEUS_ENDPOINT`	The port on which HTTP endpoint is configured
`plugin_env_PROMETHEUS_INDEXES`	Configures how data is indexed and stored in memory
`config_watch_enabled=true`	Configures network watcher to inform ibdiagnet that network topology has changed (as ibdiagnet lacks the ability to re-discover network changes)
`plugin_env_PROMETHEUS_CSET_DIR`	Specifies where the counterset files, which define the data to be retrieved and the corresponding counter names.
`num_iterations`	The number of iterations to run before 'restarting', i.e. rediscovering fabric.
`plugin_env_CLX_RESTART_FILE`	A file that is 'touched' to indicate that an ibdiagnet restart is necessary

The following attributes are configurable via the gv.cfg:

- sample_rate (gv.cfg → dashboard_interval) – only if manual_config is set to false

- prometheus_port

Supporting Generic Counters Parsing and Display

As of UFM v6.11.0, UFM can support any numeric counters from the HTTP endpoint. The list of supported counters are fetched upon starting the UFM from all the endpoints that are configured.

Some of the implemented changes are as follows:

1. Counter naming – all counters naming convention is extracted from the HTTP endpoint. The default `cset` file is configured as follows:

   “`Infiniband_LinkIntegrityErrors=^LocalLinkIntegrityErrorsExtended$`” to get this name to the UFM.

   Counters received as floats should contain an "_f" suffix such as: `Infiniband_CBW_f=^infiniband_CBW$`

2. Attribute units – To see units of a specific counter on the UI graphs, configure the `cset` file to have the counter returned as “`counter_name_u_unit`”.

3. Telemetry History:

   The SQLite history table `(/opt/ufm/files/sqlite/ufm_telemetry.db – telemetry_calculated)`, contains the new naming convention of the telemetry counters.

   In the case of an upgrade, all previous columns that were configured are renamed following the new naming convention, and then, the data is saved.if a new counter that is not in the table needs to be supported, the table is altered upon UFM start.

4. New counter/ `cset` to fetch – if there is a new `cset` /counter that needs to be supported AFTER the UFM already started, preform system restart.

5. Created New API/UfmRestV2/telemetry/counters for the UI visualization. This API returns a dictionary containing the counters that the UFM supports, based on the fetched URLs and their units (if known).

Supporting Multiple Telemetry Instances Fetch

This functionality allows users to establish distinct Telemetry endpoints that are defined to their preferences.

Users have the flexibility to set the following aspects:

- Specify a list of counters they wish to pull. This can be achieved by selecting from an existing, predefined counters set (`cset file`) or by defining a new one.
- Set the interval at which the data should be pulled.

Upon initiating the Telemetry endpoint, users can access the designated URL to fetch the desired counter data.

To enable this feature, under the [Telemetry] section in `gv.cfg,` the flag named “ `additional_cset_ur` l” holds the list of additional URLs to be fetched.

the URLs should be separated by “ “ (with a space) and should follow the following format: `http://<IP>:<PORT>/csv/<CSET_NAME>`. For example `http://10.10.10.10:9001/csv/minimal http://10.10.10.10:9002/csv/test`.

> **Note**
>
> Only csv extensions are supported.

Each UFM Telemetry instance run by UFM can support multiple cset (counters set) in parallel.If the user would like to have a second cset file fetched by UFM and exposed by the same UFM Telemetry instance, the new cset file should be placed under `/opt/ufm/files/conf/telemetry/prometheus_configs/cset/` and configured in gv.cfg to fetch its data as described above.

Low-Frequency (Secondary) Telemetry

As a default configuration, a second UFM Telemetry instance runs, granting access to an extended set of counters that are not available in the default telemetry session. The

default telemetry session is used for the UFM Web UI dashboard and user-defined telemetry views. These additional counters can be accessed via the following API endpoint: **http://<UFM_IP>:9002/csv/xcset/low_freq_debug**.

It is important to note that these exposed counters are not accessible through UFM's REST APIs.All the configurations for the second telemetry can be found under `/opt/ufm/files/conf/secondary_telemetry/`, where the defaults are located under `/opt/ufm/files/conf/secondary_telemetry_defaults/`. The second telemetry instance also allows telemetry data to be exposed on disabled ports, although this feature can be disabled if desired.

The relevant flags in the gv.cfg file are as follows:

- `secondary_telemetry` = true (To enable or disable the entire feature)
- `secondary_endpoint_port` = 9002 (The endpoint's exposed port)
- `secondary_disabled_ports` = true (If set to true, secondary telemetry will expose data on disabled ports)
- `secondary_slvl_support` = false (if set to true, low-frequency (secondary) Telemetry will collect counters per slvl, the corresponding supported xcset can be found under `/opt/ufm/files/conf/secondary_telemetry/prometheus_configs/cset/l` t)

The counters that are supported by default, collected, and exposed can be located in the directory `/opt/ufm/files/conf/secondary_telemetry/prometheus_configs/cset/low_f` .

For the list of low-frequency (secondary) telemetry fields and available counters, please refer to Low-Frequency (Secondary) Telemetry Fields.

Low-Frequency (Secondary) Telemetry - Exposing IPv6 Counters

To allow the low-frequency (secondary) telemetry instance to expose counters on its IPv6 interfaces, perform the following:

1. Change the following flag in the gv.cfg:

```
secondary_ip_bind_addr =0:0:0:0:0:0:0:0
```

2. Restart UFM telemetry or restart UFM.

Stopping Telemetry Endpoint Using CLI Command

To stop low-frequency (secondary) telemetry endpoint only using the CLI you may run the following command:

```
/etc/init.d/ufmd ufm_telemetry_secondary_stop
```

Exposing Switch Aggregation Nodes Telemetry

To expose switches SHARP aggregation nodes telemetry, follow the below steps:

- Configure the low-frequency (secondary) telemetry instance. Run:

```
vi
/opt/ufm/files/conf/secondary_telemetry_defaults/launch_ibdiag
```

  - Set the following:
    - `arg_16=--sharp --sharp_opt dsc`
    - `plugin_env_CLX_EXPORT_API_SKIP_SHARP_PM_COUNTERS=0`
  - Add the wanted attributes to the default `xcset` or to a new one:
    - New `xcset` –

```
vi
/opt/ufm/files/conf/secondary_telemetry/prometheus_con
for your choise>.xcset
```

- After restarting, query curl `http://<UFM_IP>:9002/csv/xcset/<chosen_name>`

- Existing `xcset` –

```
vi
/opt/ufm/files/conf/secondary_telemetry/prometheus_con
```

-
  -
    - Add the following attributes:
      - `packet_sent`
      - `ack_packet_sent`
      - `retry_packet_sent`
      - `rnr_event`
      - `timeout_event`
      - `oos_nack_rcv`
      - `rnr_nack_rcv`
      - `packet_discard_transport`
      - `packet_discard_sharp`

- `aeth_syndrome_ack_packet`
- `hba_sharp_lookup`
- `hba_received_pkts`
- `hba_received_bytes`
- `hba_sent_ack_packets`
- `rcds_sent_packets`
- `hba_sent_ack_bytes`
- `rcds_send_bytes`
- `hba_multi_packet_message_dropped_pkts`
- `hba_multi_packet_message_dropped_bytes`

- Restart telemetry:

```
/etc/init.d/ufmd ufm_telemetry_stop
/etc/init.d/ufmd ufm_telemetry_start
```

Exposing Performance Histogram Counters for Egress Queue Depth Indications (Secondary) Telemetry

To enable the secondary telemetry instance to expose performance histogram counters for all VLs, perform the following:

1. Change the following flag in the `gv.cfg` file:

```
queue_depth_indications_all_vls = true
```

If this flag remains set to `false`, the secondary telemetry instance will only collect counters for VLs 0 and 1.

2. Restart UFM telemetry or restart UFM.

After the secondary telemetry instance restarts, you can find the collected counters at:

```
/opt/ufm/conf/secondary_telemetry/prometheus_configs/cset/low_freq.
```

Validating UFM Configuration Files

This functionality allows users to validate the supported configuration files against the schema of the current UFM version. This validation runs automatically over all supported UFM configuration files upon UFM start.

The validation is enabled by default. The relevant flag in the gv.cfg file is `config_validation_mode`. The supported configuration files are `gv.cfg` and `opensm.conf`.

The expected values for the parameter:

- 0: Disabled (no validation)
- 1: Enabled with warnings (errors are logged, UFM continues).
- 2: Enabled with exit (validation errors cause UFM to terminate startup).
- The default value is set to 2

If UFM fails to start due to a validation error, the error details will not appear in the UFM logs. Instead, check the `systemctl` logs using:

```
journalctl -u ufm-enterprise
```

Manual Execution Examples:

- To validate a specific configuration file:

```
/opt/ufm/scripts/validate_ufm_config_files.sh
     --config-file /opt/ufm/files/conf/opensm/opensm.conf
     --schema-file
/opt/ufm/files/conf/schemas/opensm_schema.json
```

- To validate all configuration files:

```
/opt/ufm/scripts/validate_ufm_config_files.sh
```

Running UFM Server Software

Before running UFM:

- Perform Initial Configuration
- Ensure that all ports used by the UFM server for internal and external communication are open and available. For the list of ports, see Appendix – Used Ports.

  You can run the UFM server software in the following modes:

-
  - Running UFM Server Software in Management Mode
  - Running UFM Software in High Availability Mode
  - Running UFM in High Availability with failover to an external SM

Note

In Management or High Availability mode, ensure that all Subnet Managers in the fabric are disabled *before* running UFM. Any remaining active Subnet Managers will prevent UFM from running.

Running UFM Server Software in Management Mode

After installing, run the UFM Server by invoking:

```
systemctl start ufm-enterprise.service
```

Note

/etc/init.d/ufmd - Available for backward compatibility.

Log files are located under */opt/ufm/files/log* (the links to log files are in */opt/ufm/log*).

Running UFM Software in High Availability Mode

On the Master server, run the UFM Server by invoking:

```
ufm_ha_cluster start
```

You can specify additional command options for the `ufmha` service.

ufm_ha_cluster Command Options

Command	Description
start	Starts UFM HA cluster.
stop	Stops UFM HA cluster.
failover	Initiates failover (change mastership from local server to remote server).
takeover	Initiates takeover (change mastership from remote server to local server).
status	Shows current HA cluster status.
cleanup	Cleans the HA configurations on this node.
help	Displays help text.

HTTP/HTTPS Configuration

By default, UFM is configured to work with the secured HTTPS protocol.

After installation, the user can change the the Web Server configuration to communicate in secure (HTTPS) or non-secure (HTTP) protocol.

For changing the communication protocol, use the following parameter under the [Server] section in the `gv.cfg` file:

- ws_protocol = https

Changes will take effect after restarting UFM.

UFM Internal Web Server Configuration

UFM uses Apache as the main Web Server for client external access. The UFM uses an internal web server process to where the Apache forwards the incoming requests.

By default, the internal web server listens to the local host interface (127.0.0.1) on port 8000.

For changing the listening local interface or port, use the following parameters under the [Server] section in the `gv.cfg` file:

- rest_interface = 127.0.0.1
- rest_port = 8000

Changes will take effect after restarting UFM.

User Authentication

UFM User Authentication is based on standard Apache User Authentication. Each Web Service client application must authenticate against the UFM server to gain access to the system.

The UFM software comes with one predefined user:

- Username: admin
- Password: 123456

You can add, delete, or update users via User Management Tab.

UFM Authentication Server

The UFM Authentication Server, a centralized HTTP server, is responsible for managing various authentication methods supported by UFM.

Configurations of the UFM Authentication Server

The UFM Authentication Server is designed to be configurable and is initially turned off by default. This means that existing authentication methods are managed either by the native Apache functionality (such as Basic, Session, and Client Certificate authentication) or at the UFM level (including Token-Based authentication and Proxy Authentication).

Enabling the UFM Authentication Server provides a centralized service that oversees all supported authentication methods within a single service, consolidating them under a unified authentication API.

Apache utilizes the authentication server's APIs to determine a user's authentication status.

To enable the UFM Authentication Server, refer to Enabling UFM Authentication Server.

All activities of the UFM Authentication Server are logged in the `authentication_service.log` file, located at `/opt/ufm/files/log`.

Azure AD Authentication

Microsoft Azure Authentication is a service provided by Microsoft Azure, the cloud computing platform of Microsoft. It is designed to provide secure access control and authentication for applications and services hosted on Azure.

UFM supports Authentication using Azure Active Directory, and to do so, you need to follow the following steps:

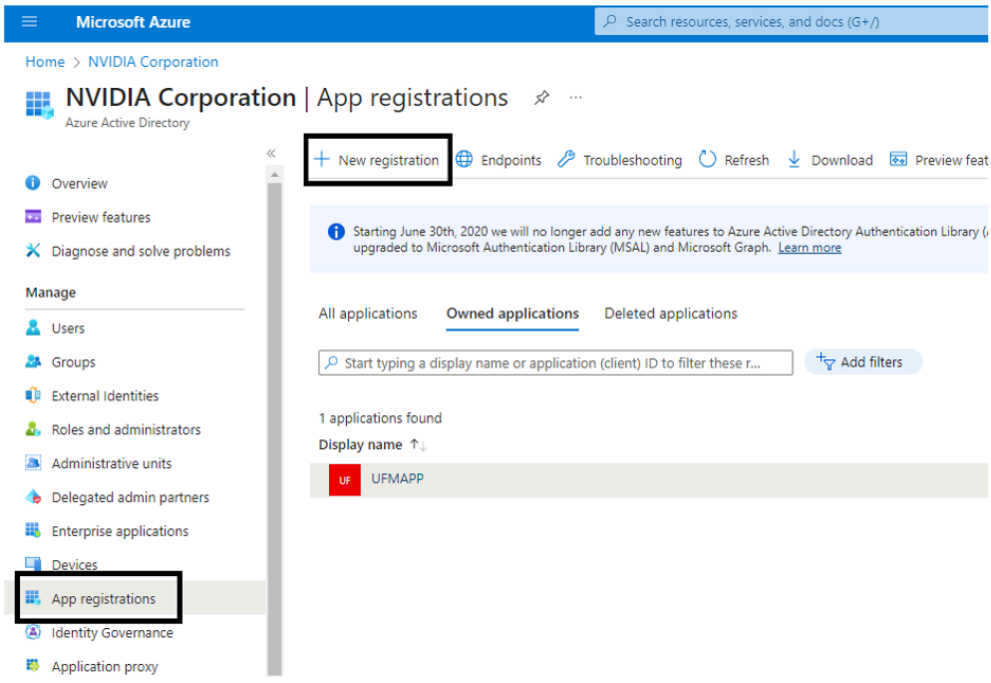
Register UFM in Azure AD Portal

To log in via Azure, UFM must be registered in the Azure portal using the following steps:

1. Log in to Azure Portal, then click "**Azure Active Directory"** in the side menu.

2. If you have access to more than one tenant, select your account in the upper right. Set your session to the Azure AD tenant you wish to use.

3. Under "**Manage"** in the side menu, click App Registrations > New Registration.

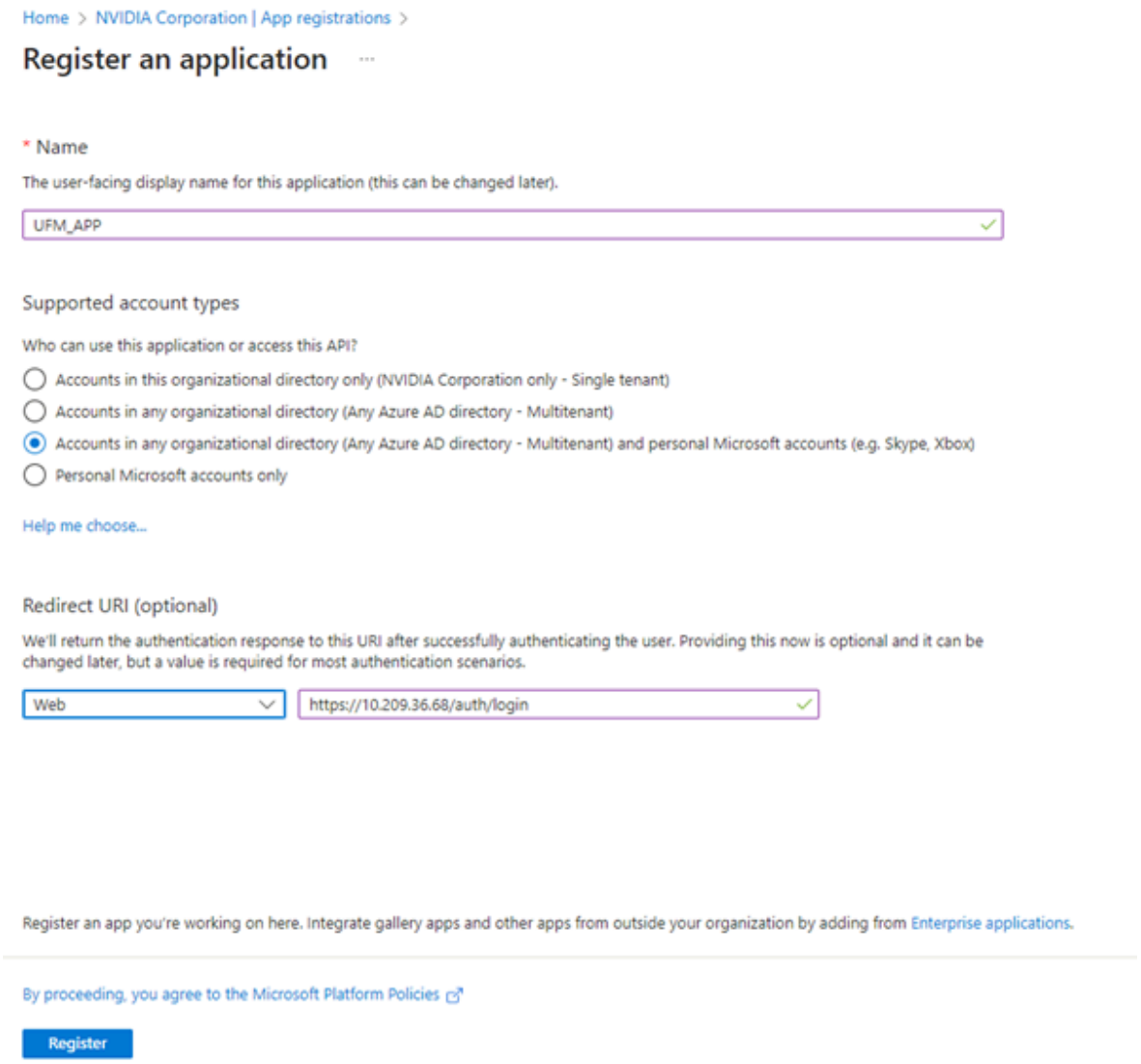


4. Provide the application details:

    1. **Name**: Enter a descriptive name.

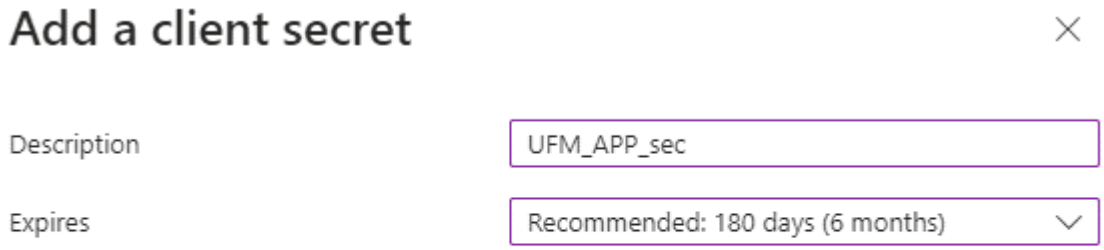
    2. **Supported account types**: Account types that are allowed to login and use the registered application.

3. **Redirect URL**: select the app type **Web**, and Add the following redirect URL https:///auth/login



Then, click **Register**. The app's **Overview** page opens.

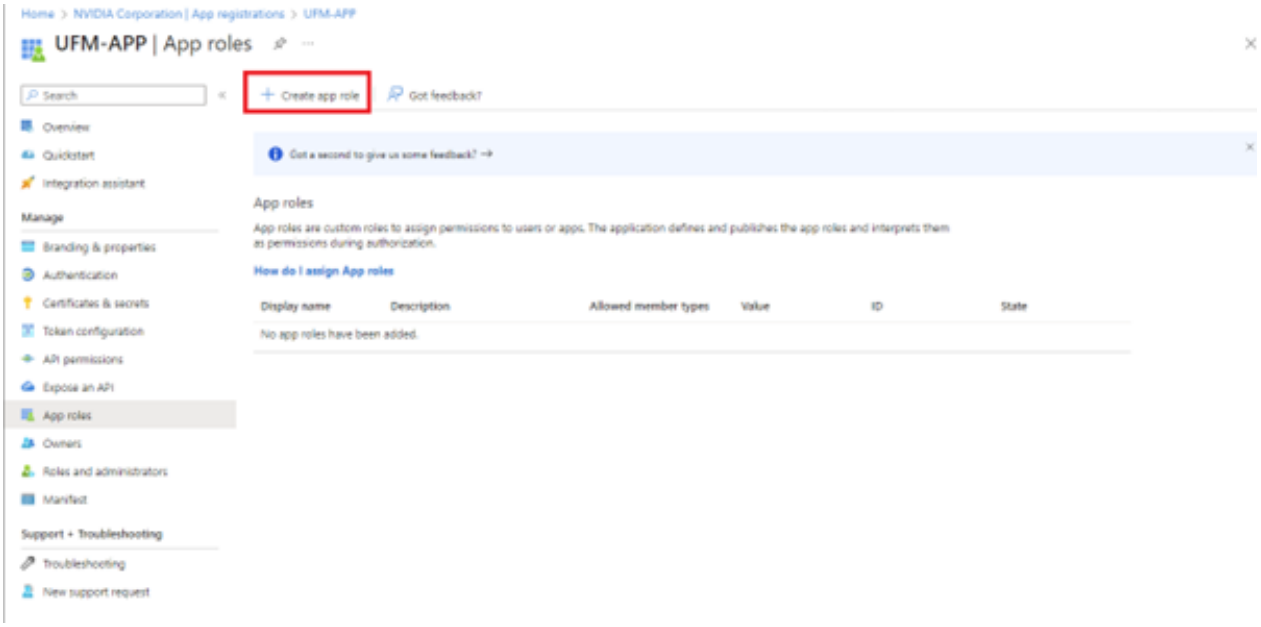
5. Under **Manage** in the side menu, click **Certificates & Secrets** > New client secret.



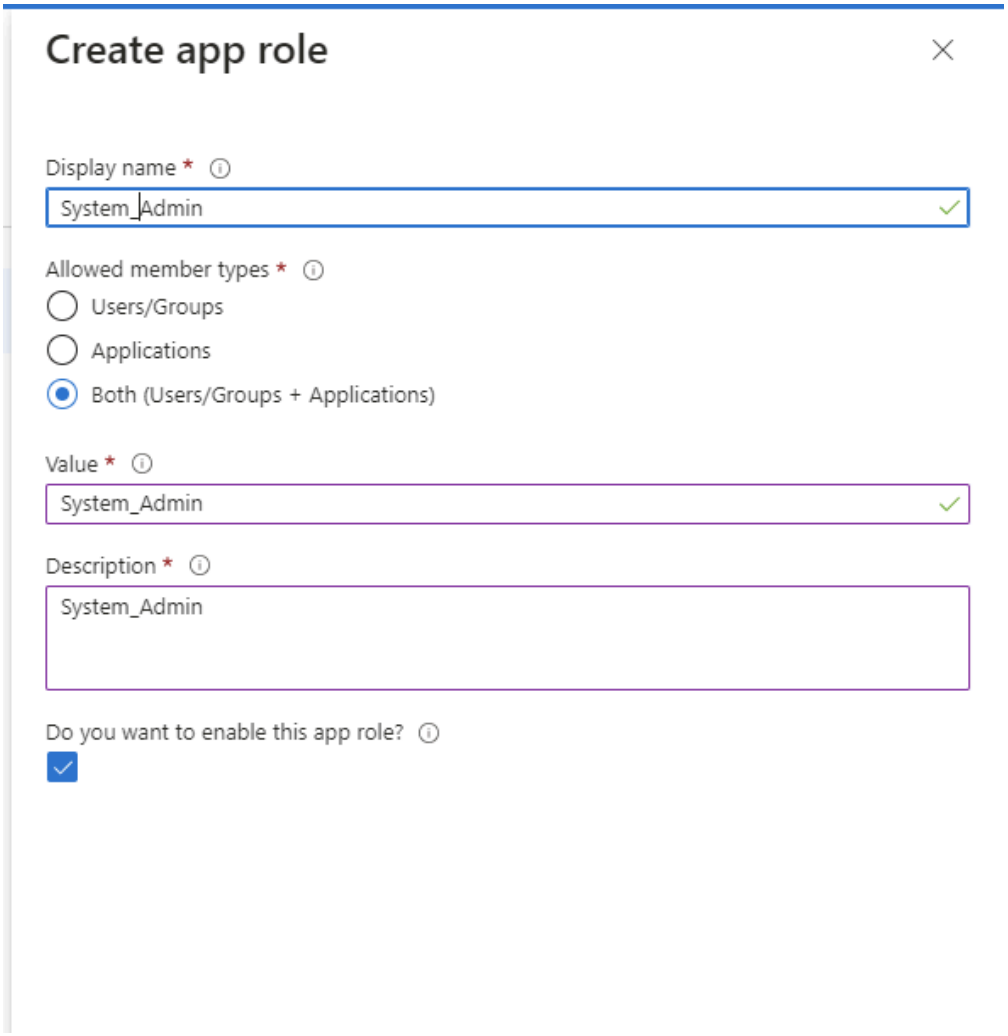
Provide a description for the client secret and set an expiration time, then click "**Add**."

6. Copy the client secret key value which will be needed to configure the UFM with Azure AD (Please note that the value of the generated secret will be hidden and will not be able to be copied/read after you leave the page.

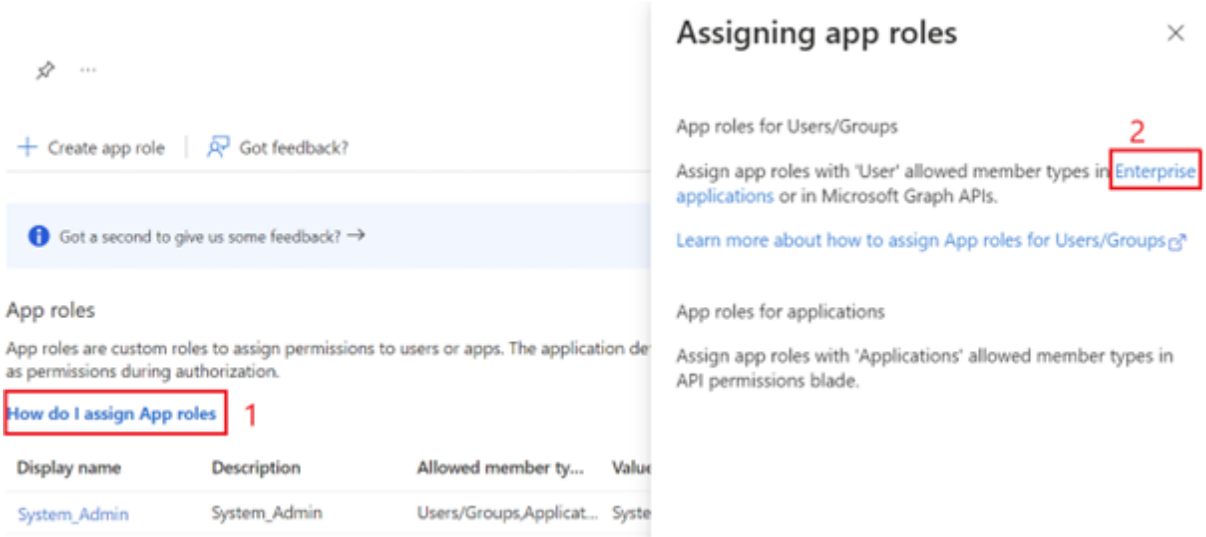
   Under "**Manage"** in the side menu, click App roles > Create app role.

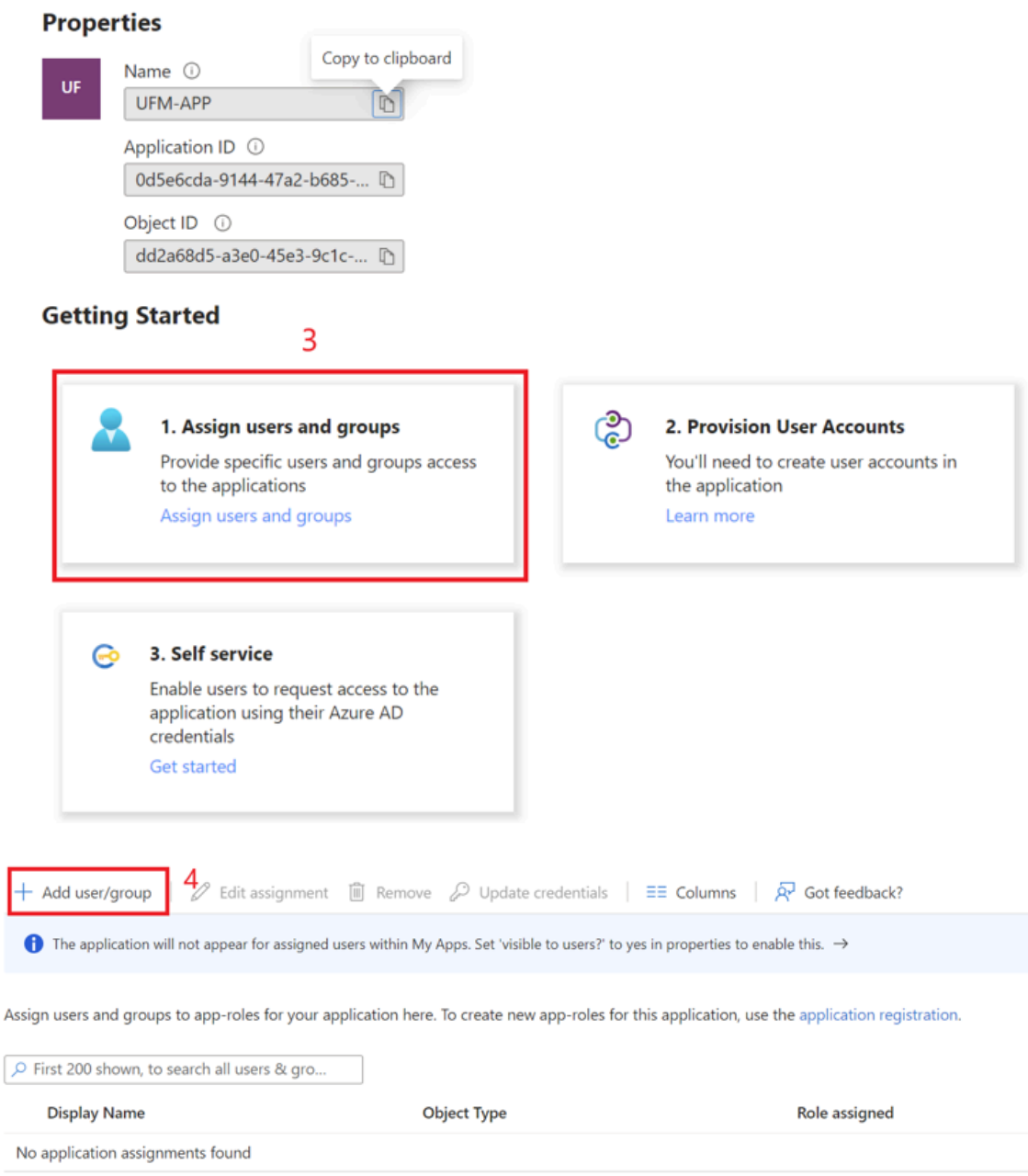


7. Provide the role details. Please note that the role value must be a valid UFM role; otherwise, the login will fail.

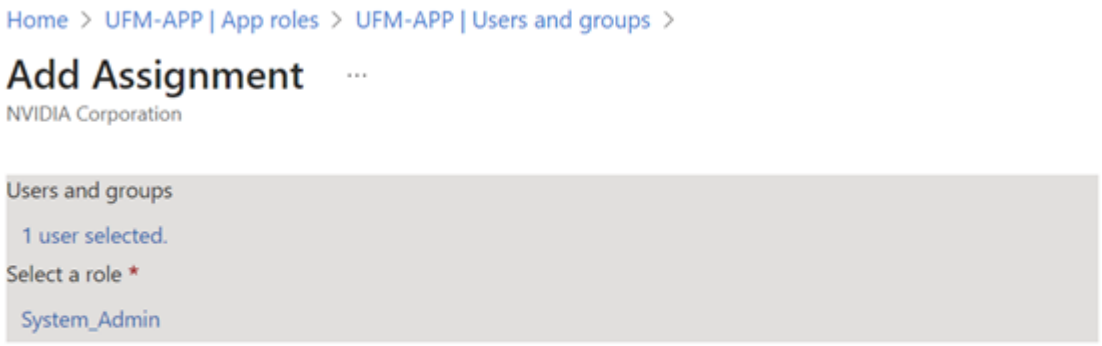


8. Assign the created role to the user. Follow the below steps:




Properties
UF
Name
Copy to clipboard
UFM-APP
Application ID
0d5e6cda-9144-47a2-b685-...
Object ID
dd2a68d5-a3e0-45e3-9c1c-...
Getting Started
3
1. Assign users and groups
Provide specific users and groups access to the applications
Assign users and groups
2. Provision User Accounts
You'll need to create user accounts in the application
Learn more
3. Self service
Enable users to request access to the application using their Azure AD credentials
Get started

Add user/group
4
Edit assignment
Remove
Update credentials
Columns
Got feedback?
The application will not appear for assigned users within My Apps. Set 'visible to users?' to yes in properties to enable this.
Assign users and groups to app-roles for your application here. To create new app-roles for this application, use the application registration.
First 200 shown, to search all users & gro...
Display Name
Object Type
Role assigned
No application assignments found



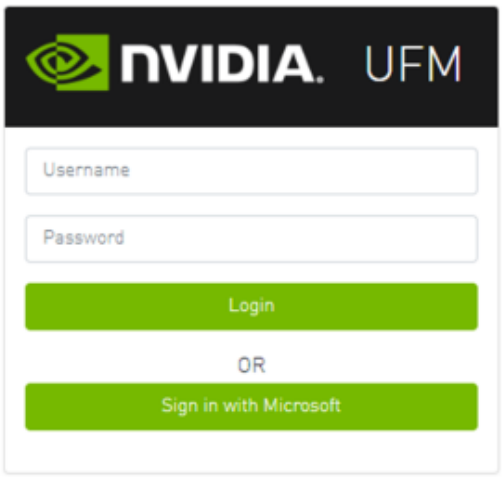
9. Click on "**Overview**" in the side menu to view the application information, such as tenant ID, client ID, and other details.

Enable Azure Authentication From UFM

Azure authentication is disabled by default. To enable it, please refer to Enabling Azure AD Authentication.

Azure Authentication Login Page

After enabling and configuring Azure AD authentication, an additional button will appear on the primary UFM login page labeled 'Sign In with Microsoft,' which will leads to the main Microsoft sign-in page:



Kerberos Authentication

Kerberos is a network authentication protocol designed to provide strong authentication for client-server applications by using secret-key cryptography.

The Kerberos protocol works on the basis of tickets, it helps ensure that communication between various entities in a network is secure. It uses symmetric-key cryptography, which means both the client and servers share secret keys for encrypting and decrypting communication.

To enable Kerberos Authentication, refer to Enabling Kerberos Authentication.

Setting Up Kerberos Server Machine

To set up a system as a Kerberos server, perform the following:

1. Install the required packages:

```
#Redhat
sudo yum install krb5-libs krb5-server
# Ubuntu
sudo apt-get install krb5-kdc krb5-admin-server
```

2. Edit the Kerberos configuration file '`/etc/krb5.conf`' to reflect your realm, domain and other settings:

```
[libdefaults]
    default_realm = YOUR-REALM

[realms]
    YOUR-REALM = {
        kdc = your-kdc-server
        admin_server = your-admin-server
    }

[domain_realm]
    your-domain = YOUR-REALM
    your-domain = YOUR-REALM
```

3. Use the `kdb5_util` command to create the Kerberos database:

```
kdb5_util create -r YOUR-REALM -s
```

4. Add administrative principals:

```
 Kadmin.local addprinc -randkey HTTP/YOUR-HOST-NAME@YOUR-
REALM
```

5. Start KDC and Kadmin services:

```
sudo systemctl start krb5kdc kadmin
sudo systemctl enable krb5kdc kadmin
```

6. Generate a keytab file. The keytab file contains the secret key for a principal and is used to authenticate the service.

```
 kadmin.local ktadd -k /path/to/your-keytab-file HTTP/YOUR-HOST-NAME@YOUR-REALM
```

Replace `/path/to/your-keytab-file` with the actual path where you want to store the keytab file.

Setting Up Kerberos Client Machine

Follow the below steps to set up a system as a Kerberos client.

1. Install the required packages. When installing the UFM, the following packages will be installed as dependencies:

```
#Redhat
krb5-libs krb5-workstation mod_auth_gssapi
# Ubuntu
krb5-config krb5-user libapache2-mod-auth-gssapi
```

2. Configure the /etc/krb5.conf file to reflect your realm, domain, local names map and other settings:

```
[libdefaults]
    default_realm = YOUR-REALM

[realms]
    YOUR-REALM = {
        kdc = your-kdc-server
        admin_server = your-admin-server
        auth_to_local_names = {
            your-principle-name = your-local-user
        }

    }

[domain_realm]
    your-domain = YOUR-REALM
    your-domain = YOUR-REALM
```

3. Copy the keytab file from the Kerberos server to the machine where your service runs (the client). It is important to ensure that it is kept confidential.

   Please ensure that the keytab file exists and that Apache has the necessary read permissions to access the keytab file; otherwise, Kerberos authentication will not function properly.

4. Obtain a Kerberos ticket-granting ticket (TGT):

```
kinit -k -t /path/to/your-keytab-file HTTP/YOUR-HOST-
NAME@YOUR-REALM
```

5. Enable Kerberos Authentication from UFM. Kerberos authentication is disabled by default. To enable it, please refer to Enabling Kerberos Authentication.

6. Test the Kerberos Authentication. You can use curl to test whether the user can authenticate to UFM REST APIs using Kerberos.

```
curl --negotiate -i -u : -k 'https://ufmc-eos01/ufmRestKrb/app/tokens'
```

Licensing

UFM license is subscription-based featuring the following subscription options:

- 1-year subscription
- 3-year subscription
- 5-year subscription
- Evaluation 30-day trial license

> **Note**
>
> UFM will continue to support old license types, but they are no longer available to obtain.

2 months before the expiration of your subscription license, UFM will warn you that your license will expire soon. After the subscription expires, UFM will continue to work with the expired license for two months beyond its expiration.

During this extra two-month period, UFM will generate a critical alarm indicating that the UFM license has expired and that you need to renew your subscription. Failing to do so within that 2-month period activates UFM Limited Mode. Limited mode blocks all REST APIs and access to the UFM web UI.

UFM enables functionality based on the license that was purchased and installed. This license determines the functionality and the maximum allowed number of nodes in the fabric.

To renew your UFM subscription, purchase a new license and install the new license file by downloading the license file to a temp directory on the UFM master server and then

copying the license file to /opt/ufm/files/licenses/ directory.

Note

UFM may not detect new license files if downloaded directly to /opt/ufm/files/licenses. If UFM does not detect the new license file, a UFM restart may be required.

If several licenses are installed on the server (more than one license file exists under /opt/ufm/files/licenses/), UFM uses only the strongest license and takes into consideration the expiration date, and the managed device limits on it, regardless of any other licenses that may exist on the server.

Upgrading UFM Software

After UFM installation, UFM detects existing UFM versions previously installed on the machine and prompts you to run a clean install of the new version or to upgrade. We recommend backing up the UFM configuration before upgrading the UFM as specified in the section UFM Database and Configuration File Backup.

Info

Upgrading the UFM Enterprise software version is supported up to two previous GA software versions (GA -1 or GA -2).

For example, if you wish to upgrade to UFM Enterprise v6.11.0, it is possible to do so only from UFM Enterprise v6.9.0 or v6.10.0.

For the following UFM configuration files: `gv.cfg`, `reports.cfg, opensm.conf, gui.properties, sharp_am.conf`, `UFMInfraHealthConfiguration.xml` and `UFMHealthConfiguration.xml`, parameter values are preserved during the upgrade process. After the upgrade is complete, any values set by the user will remain unchanged.

- Upgrading UFM on Bare Metal Server
- Upgrading UFM on Docker Container
- Upgrading UFM Podman Rootless Container

Upgrading UFM on Bare Metal Server

Upgrading UFM on Bare Metal - Standalone Server Upgrade

You can upgrade the UFM standalone server software for InfiniBand from the previous UFM version.

To upgrade the UFM server software:

1. Create a temporary directory (for example */tmp/ufm*).
2. Open the UFM software zip file that you downloaded. The zip file contains the following installation files for:
    - RedHat 7/CentOS 7/OEL 7: ufm-X.X -XXX.el7.x86_64.tgz
    - RedHat 8/CentOS 8/OEL 8: ufm-X.X -XXX.el8.x86_64.tgz
    - Ubuntu 18.04: ufm-X.X -XXX.ubuntu18.x86_64.tgz
    - Ubuntu 20.04: ufm-X.X -XXX.ubuntu20.x86_64.tgz
    - Ubuntu 22.04: ufm-X.X-XXX.ubuntu22.x86_64.tgz
3. Extract the installation file for your system's OS to the temporary directory that you created.
4. Stop the UFM server. Run:

    ```
    systemctl stop ufm-enterprise
    ```

5. From within the temporary directory, run the following command as root:

```
./upgrade.sh
```

Note

A configuration backup ZIP file will be created in the running directory (e.g. /tmp/ufm). The backup file name is ufm_X.X.X_bkp.zip (X.X.X is the previous version).

1. Upgrade from the previous version: the existing UFM data and configuration are preserved.

2. In case upgrade.sh script stops before completion (e.g. missing prerequisite), the upgrade procedure can be resumed by fixing the issue (e.g. installing missing prerequisite) and rerunning ./upgrade.sh again.

6. Restart the UFM server. Run:

```
systemctl start ufm-enterprise.service
```

Note

`/etc/init.d/ufmd start` - Available for backward compatibility.

7. After the upgrade, remove the temporary directory

Upgrading UFM on Bare Metal - High Availability Upgrade

Note

As of UFM version 6.14.0, UFM upgrade on HA supports in-service upgrade, meaning UFM can continue running during the steps of the upgrade, and there is no need to stop UFM before the upgrade (although this is also supported).

You can upgrade the UFM server HA software for InfiniBand from the previous release. The upgrade is performed on both servers.

To upgrade the UFM server software:

Upgrading the UFM Enterprise Package

1. On the standby server, extract the new UFM Enterprise package to the /tmp folder:

```
tar -xzf ufm-X.X.X-XXXXX.tgz -C /tmp
```

2. On the standby server, enter to the installation folder and upgrade script:

```
standby# cd /tmp/ ufm-X.X.X-X.<OS_NAME>.x86_64.mofed5/
```

3. Run the UFM upgrade script on the standby server:

```
./upgrade.sh
```

4. After the completion of the upgrade script, the UFM code will undergo an upgrade, while the UFM data will remain unchanged. The automatic upgrade of UFM data will take place during the next startup of UFM. To initiate this process, execute a failover from the Master node (or perform a takeover from the Standby node).

```
master# ufm_ha_cluster failover
```

> **Note**
>
> UFM will log the data upgrade to the syslog of the server, in case of issue a backup of the UFM data is saved prior to the upgrade in `/opt/ufm/BACKUP` directory and can be restored.

5. Once UFM is operational on the upgraded node (formerly the standby node), proceed to replicate steps 1 to 3 on the non-upgraded node (previously the master node).

Upgrading the UFM HA Package

1. On **both servers**, download latest UFM-HA package:

```
wget http://www.mellanox.com/downloads/UFM/ufm_ha_6.1.1-2.tgz
```

2. On **both servers**, extract the HA package under `/tmp/` and enter the new directory

3. On the UFM **Standby server**, run the upgrade command from within the extracted HA package located in `/tmp`:

```
./install.sh --upgrade
```

4. On the UFM **Master server,** run the upgrade command from within the extracted HA package located in `/tmp`:

```
./install.sh --upgrade
```

5. Run the following command to verify that the UFM HA cluster is up and running:

```
ufm_ha_cluster status
```

Upgrading UFM on Docker Container

> **Note**
>
> Upgrade the UFM container based on the existing UFM configuration files that are mounted on the server. It is important to use that same directory as a volume for the UFM installation command.
>
> In the below example /opt/ufm_files is used.

Upgrading UFM on Docker Container in Standalone Mode

1. Stop the UFM Enterprise service. Run:

```
systemctl stop ufm-enterprise
```

2. Remove the existing docker image. Run:

```
docker rmi mellanox/ufm-enterprise:latest
```

3. Load the new UFM Enterprise docker image. Run:

```
docker pull mellanox/ufm-enterprise:latest
```

4. Run the docker upgrade command:

```
docker run -it --name=ufm_installer --rm \
-v /var/run/docker.sock:/var/run/docker.sock \
-v /etc/systemd/system/:/etc/systemd_files/ \
-v /opt/ufm/files/:/opt/ufm/shared_config_files/ \
mellanox/ufm-enterprise:latest --upgrade
```

5. Reload system manager configuration:

```
systemctl daemon-reload
```

6. Start UFM Enterprise service:

```
systemctl start ufm-enterprise
```

Upgrading UFM Container in High Availability Mode

Note

As of UFM version 6.14.0, UFM upgrade on HA supports in-service upgrade, meaning UFM can continue running during the steps of the upgrade, and there is no need to stop UFM before the upgrade (although this is also supported).

Upgrading the UFM Enterprise Package

1. Remove the old docker image from the **Standby** server. Run:

```
Stand-by# docker rmi mellanox/ufm-enterprise:latest
```

2. Pull the new UFM Enterprise docker image on the **Standby** server. Run:

```
docker pull mellanox/ufm-enterprise:latest
```

> **Note**
>
> At this stage, the UFM container has been updated with the latest code. The UFM data, however, will be updated during the next UFM run.

3. Run the docker upgrade command:

```
docker run -it --name=ufm_installer --rm \
-v /var/run/docker.sock:/var/run/docker.sock \
-v /etc/systemd/system/:/etc/systemd_files/ \
-v /opt/ufm/files/:/opt/ufm/shared_config_files/ \
mellanox/ufm-enterprise:latest --upgrade
```

4. Reload system manager configuration:

```
systemctl daemon-reload
```

5. Perform a failover to start UFM on the upgraded node. On the **Master** node, run:

```
ufm_ha_cluster failover
```

> **Note**
>
> When UFM starts, it will automatically update the UFM configuration.

6. Repeat steps 1-4 on the un-upgraded node (previous **Master** node).

Upgrading the UFM HA Package

1. On **both servers**, download and extract the latest UFM HA package. Run:

```
wget http://www.mellanox.com/downloads/UFM/ufm_ha_6.1.1-2.tgz
```

2. On both the **Master** and **Standby** servers, execute the upgrade command from within the extracted HA package. Ensure you run it first on the **Standby** server, then on the **Master** server:

```
./install.sh --upgrade
```

3. Run the following command to verify that the UFM HA cluster is up and running:

```
ufm_ha_cluster status
```

Upgrading UFM Podman Rootless Container

This section provides instructions on how to upgrade UFM Enterprise when running as an unprivileged user (rootless mode).

Note

Important Note:

- Do not run any of the below commands from /root directory (user commands will fail based on permissions).

  Change directory into /tmp (e.g.) before starting this procedure.

- Replace the image filename in step 3 with your specific UFM version

Step 1: Pre-checks - Stop UFM Service

Stop the UFM service if it's currently running:

```
systemctl is-active --quiet ufm-enterprise.service && systemctl stop ufm-enterprise.service
```

Step 2: Clean and Prepare Directories

Remove existing system directory and recreate necessary directories:

```
rm -rf /opt/ufm/systemd
sudo -u ufmadm mkdir -p /opt/ufm/ufm_plugins_data
sudo -u ufmadm mkdir -p /opt/ufm/systemd
sudo -u ufmadm mkdir -p /opt/ufm/etc/apache2
```

Step 3: Load New UFM Image

Load the new UFM container image (replace with your specific image file):

```
sudo -u ufmadm podman load -i <UFM docker img file>
```

Step 1: Run UFM Upgrade

Execute the UFM upgrade process with all necessary volume mounts:

```
sudo -u ufmadm podman run -it --rm --name=ufm_installer \
  -v /run/podman-ufm/podman-ufm.sock:/var/run/docker.sock \
  -v /opt/ufm/files:/opt/ufm/shared_config_files \
  -v /opt/ufm:/installation/ufm_files \
  -v /opt/ufm/etc/apache2:/installation/etc/apache2 \
  -v /opt/ufm/systemd:/etc/systemd_files/ \
  mellanox/ufm-enterprise:latest \
  --upgrade --rootless --ufm-user ufmadm --ufm-group ufmadm
```

Step 4: Install Updated Systemd Service File

Move the updated systemd service file to the system location and reload systemd:

```
mv /opt/ufm/systemd/ufm-enterprise.service
/etc/systemd/system/ufm-enterprise.service
systemctl daemon-reload
```

Step 5: Cleanup

Remove the temporary systemd directory:

```
rm -rf /opt/ufm/systemd
```

Uninstalling UFM

The UFM Server can be uninstalled by running an uninstall script in the different server modes:

- Uninstalling UFM in Standalone Mode
- Uninstalling UFM in High Availability
- Uninstalling UFM in Docker Deployment

Uninstalling UFM in Standalone Mode

To uninstall the UFM Server:

1. Go to /opt/ufm.
2. Run ./uninstall.sh.

> **Note**
>
> Child interfaces are not deleted.

3. To delete primary interfaces, restart /etc/init.d/openibd.

Uninstalling UFM in High Availability

To uninstall the UFM Server in high availability mode:

1. Run the following on the master and slave to clean up the UFM HA configuration:

```
ufm_ha_cluser cleanup
```

2. To uninstall the UFM HA configuration, run:

```
/opt/ufm/ufm_ha/uninstall_ha.sh
```

3. To uninstall UFM Enterprise software, run the following on the master and slave:

```
/opt/ufm/uninstall.sh
```

Uninstalling UFM in Docker Deployment

To uninstall the UFM Server in high availability mode:

1. Run the following on the master and slave:

```
ufm_ha_cluser cleanup
```

2. Run:

```
/opt/ufm/ufm_ha/uninstall_ha.sh
```

3. Run the following on the master and slave:

```
/opt/ufm/files/uninstall.sh
```

UFM Configuration Backup and Restore

Overview

UFM migration enables backup and restores UFM configuration files.

Backup UFM Configuration

By default, the following folders (placed in `/opt/ufm/files`) are being backed up:

- `conf`
- `dashboardViews`
- `licenses`
- `networkViews`
- `scripts`
- `sqlite`
- `templates/user-defined`
- `ufmhealth/scripts`
- `userdata`
- `users_preferences`

Note

The user may also backup the UFM historical telemetry data ("-t" argument).

UFM (Bare Metal)

```
/opt/ufm/scripts/ufm_backup.sh --help
usage: ufm_backup.pyc [-h] [-f BACKUP_FILE] [-t]
```

Optional Arguments

-h	--help	show this help message and exit
-f	--backup-file BACKUP_FILE	full path of zip file to be generated
-t	--telemetry	backup UFM historical telemetry

UFM Docker Container

Backup the UFM configuration. Run on the **Master node:**

```
docker exec ufm /opt/ufm/scripts/ufm_backup.sh -f
/opt/ufm/files/ufm-backup_<ufm-version>.zip
```

UFM Appliance

1. Backup UFM configuration. Run:

```
ufm data backup [with-telemetry]
```

2. Upload the backup file to a remote host. Run:

```
ufm data upload <backup file> <upload URL>
```

> **Note**
>
> More details can be found in the log file `/tmp/ufm_backup.log.`

Restore UFM Configuration

> **Note**
>
> All folders which are a part of the UFM backup are restored (filter is done during the backup stage).

UFM Bare Metal

```
/opt/ufm/scripts/ufm_restore.sh --help
usage: ufm_restore.pyc [-h] -f BACKUP_FILE [-u] [-v]
```

Optional Arguments

-h	--help	show this help message and exit
-f BACKUP_FILE	--backup-file BACKUP_FILE	full path of zip file generated by backup script
-u	--upgrade	upgrades the restored UFM files
-v	--verbose	makes the operation more talkative

UFM Docker Container

1. Stop UFM. Run:

```
docker exec ufm /etc/init.d/ufmd stop
```

2. Copy the backup file from the host into UFM docker container. Run:

```
docker cp <backup file> ufm:/tmp/<backup file>
```

3. Restore UFM configuration. Run:

```
docker exec ufm /opt/ufm/scripts/ufm_restore.sh -f
/tmp/<backup file> [--upgrade]
```

4. Start UFM. Run:

```
docker exec ufm /etc/init.d/ufmd start
```

UFM Appliance

1. Stop UFM. Run:

```
no ufm start
```

2. Copy the backup file from a remote host into UFM appliance. Run:

```
ufm data fetch <download URL>
```

3. Restore UFM configuration. Run:

```
ufm data restore <backup file>
```

4. Start UFM. Run:

```
ufm start
```

Note

When restoring the UFM configuration from host to a container, the following parameters in `/opt/ufm/files/conf/gv.cfg` may be reset the following:

- fabric_interface
- ufma_interfaces
- mgmt_interface

Note

UFM configuration upgrade during restore is not supported in UFM Appliance GEN2/GEN2.5

More details can be found in the log files `/tmp/ufm_restore.log` and `/tmp/ufm_restore_upgrade.log`

UFM Factory Reset

This section provides a comprehensive guide on resetting UFM to its original factory settings.

Note

WARNING!!! this operation will remove all user data and configuration and will restore UFM to its factory defaults.

Note

The UFM Factory-Reset will exclusively revert UFM to its original factory settings, leaving HA configurations unaffected. To remove HA, it is essential to execute `ufm_ha_cluster cleanup` before initiating the factory reset.

UFM Docker Container Factory Reset

To reset UFM to its factory defaults when using UFM on a Docker container, follow these steps.

1. Ensure that UFM is not up and running. If UFM is running, stop it.

   For Stand-alone (SA) installations:

   ```
   systemctl stop ufm-enterprise
   # validate that ufm is not running
   systemctl status ufm-enterprise
   ```

   For High-Availability setups (perform the following on the master node only):

```
ufm_ha_cluster stop
# validate that ufm is not running
ufm_ha_cluster status
```

2. Run `mellanox/ufm-enterprise` Docker Container with the following flags:

> **Note**
>
> WARNING: This operation will erase all user data and configurations, resetting UFM to its factory defaults.
>
> CAUTION: This step does not require user confirmation, meaning UFM will be restored to factory defaults immediately once initiated.

```
docker run -it --name=ufm_installer --rm \
                -v /var/run/docker.sock:/var/run/docker.sock
\
                -v /tmp:/tmp \
                -
v /opt/ufm/files/:/opt/ufm/shared_config_files/ \
                mellanox/ufm-enterprise:latest \
                --factory-reset
```

Flag	Type	Description
`--name=ufm_installer`	Mandatory	The container name must be called ufm_installer

Flag	Type	Description
`-v /var/run/docker.sock:/var/run/docker.sock`	Mandatory	The docker socket must be mounted on the docker container.
`-v /tmp:/tmp`	Optional	Logs of the operation can be viewed in `/tmp` on the host in case it is mounted
`-v /opt/ufm/files/:/opt/ufm/shared_config_ufm/`	Mandatory	For the factory reset to persist, it is essential to have the /opt/ufm/files directory mounted from the host.
`mellanox/ufm-enterprise:latest`	Mandatory	The docker image name.
`--factory-reset`	Mandatory	This action will signal the UFM container to initiate the factory reset process.

UFM Enterprise Factory Reset

To restore UFM Enterprise to factory defaults:

1. Ensure that UFM is not up and running. If UFM is running, stop it.

   For Stand-alone (SA) installations:

```
systemctl stop ufm-enterprise
# validate that ufm is not running
systemctl status ufm-enterprise
```

For High-Availability setups (perform the following on the master node only):

```
ufm_ha_cluster stop
# validate that ufm is not running
ufm_ha_cluster status
```

2. Run the `ufm_factory_reset.sh` script:

> **Note**
>
> WARNING: This operation will erase all user data and configurations, resetting UFM to its factory defaults.

```
/opt/ufm/scripts/ufm_factory_reset.sh [-y]
```

Flag:

Flag	Type	Description
`-y`	Optional	Does not require user confirmation.

UFM Rollback

Note

The following instructions apply specifically to UFM on containers only.

It is assumed that UFM Enterprise is already running and functional.

Save a Backup

Refer to UFM Docker Container Backup.

Rollback UFM to a Previous Version

Important: A backup of the UFM files and the UFM container for the same version must be available before performing this procedure. Without a backup, the rollback will not work (though you can still install a fresh copy of the desired version, but no configuration will be restored).

1. **Uninstall UFM**: Please refer to the uninstallation guidelines specific to your current UFM version.

2. **Install the Previous UFM Enterprise Version**: Refer to the installation guidelines for the version you wish to install.

3. **Restore the UFM Backup Files**: Refer to the UFM Configuration Backup and Restore.

Manage Users

Accounts and Roles

Each UFM user can be assigned to one of the following four roles:

- **System Admin** – users can perform all operations including managing other users accounts.
- **Fabric Admin** – users can perform fabric administrator actions such as update SM configuration, update global credentials, manage reports, managing unhealthy ports, and manage PKeys, etc.
- **Fabric Operator** – users can perform fabric operator actions such as device management actions (enable/disable port, add/remove devices to/from groups, reboot device, upgrade software, etc.)
- **Monitoring Only** – users can perform monitoring actions such as view the fabric configuration, open monitoring sessions, define monitoring templates, and export monitoring data to CSV files, etc.

User Account Management

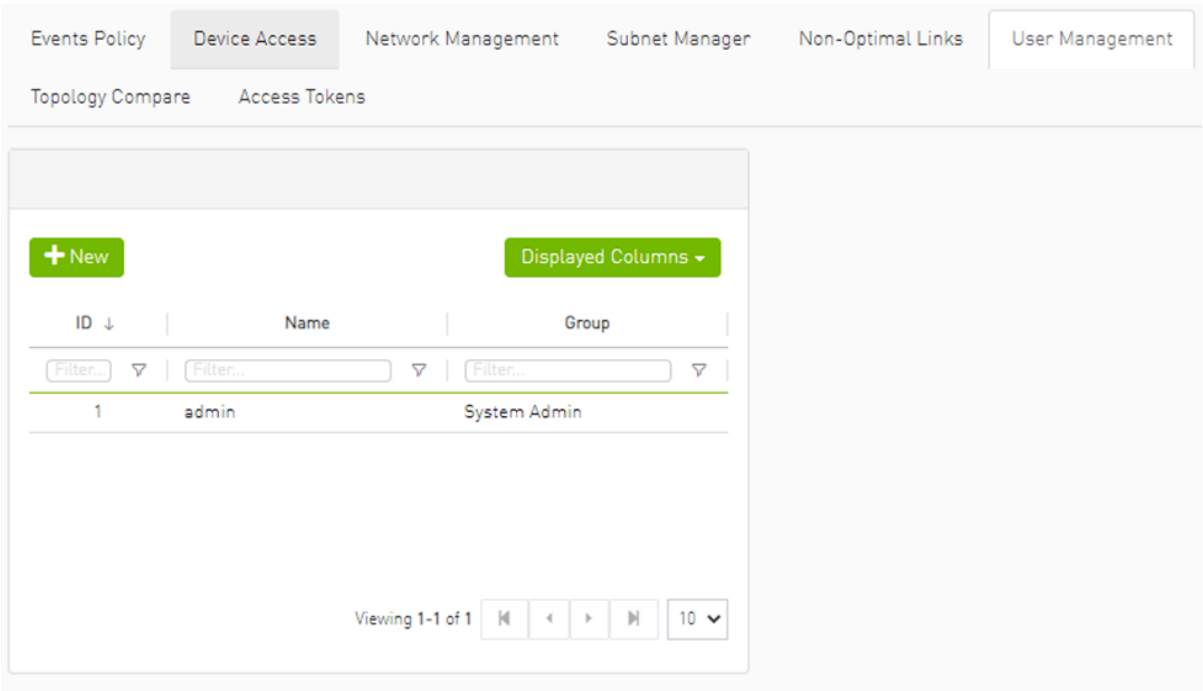
By default and upon every UFM deployment, the **admin** user (System Admin) is generated to allow initial access to the UFM.

A user with system Administration rights can manage other users' accounts, including the creation, deletion, and modification of accounts.

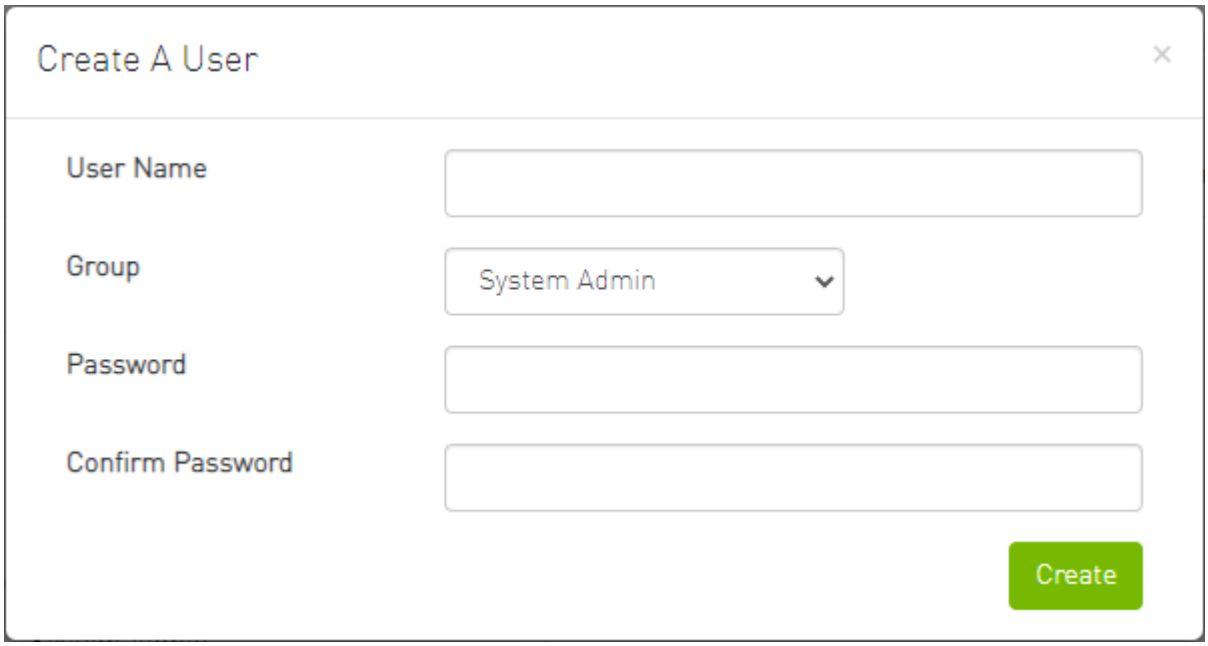
To edit existing user accounts, right-click the account from the list of user accounts and perform the desired action (Change Password/Remove).

Add New User

1. Click the “New” button.

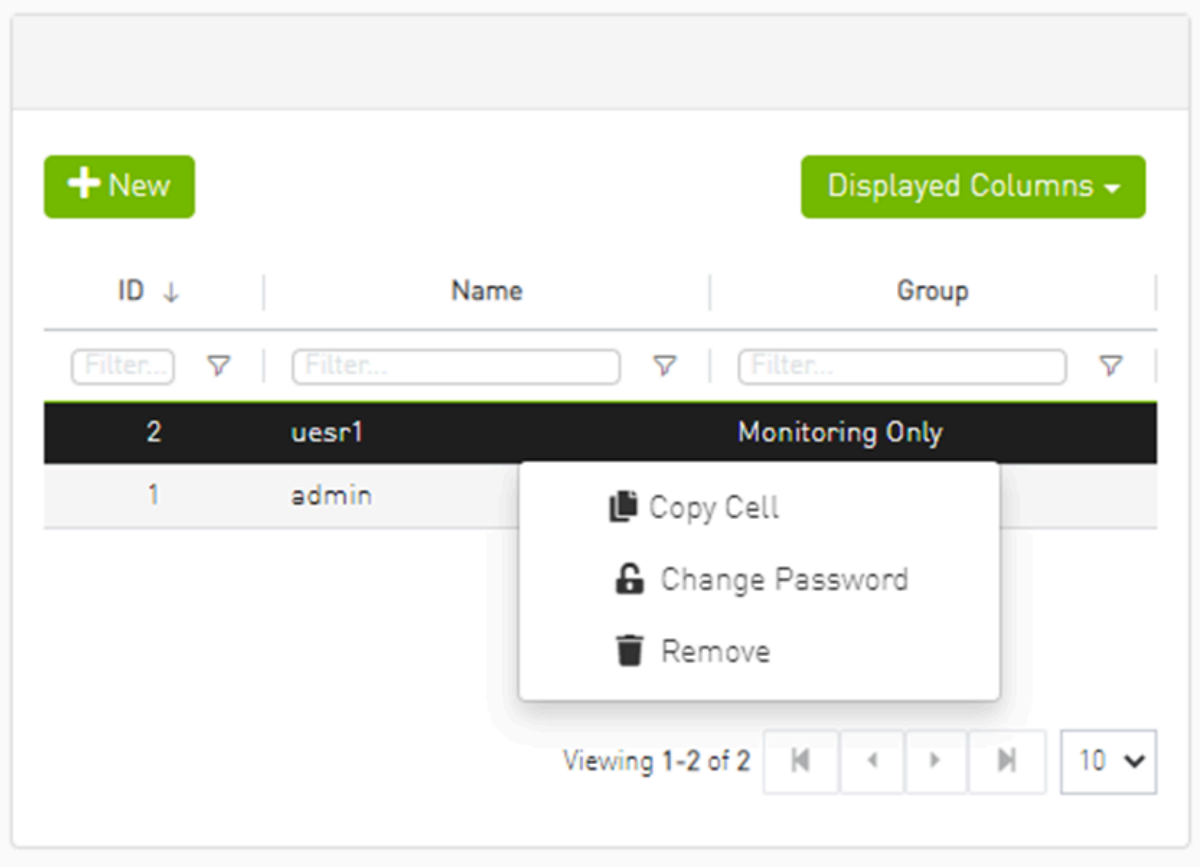


2. Fill in the required fields in the dialog box.



Edit/Remove Existing User

Right-click the account from the list of user accounts and perform the desired action (Change Password/Remove).



Authentication Methods

UFM User Authentication is based on standard Apache User Authentication or Internal UFM Authentication Server.

Each Web Service client application must authenticate against the UFM server to gain access to the system.

The available authentication methods supported by UFM are as follows:

Authentication Method	Description	UFM Related Prefix	REST API Reference
Basic Authentication	Based on user and password, provided by the client. This method is enabled by default.	/ufmRest	Basic Authentication

Authentication Method	Description	UFM Related Prefix	REST API Reference
Session-Based Authentication	A stateful authentication technique where sessions are used to keep track of the authenticated user. This method is enabled by default and is used by the UFM WebUI.	/ufmRestV2	Session-Based Authentication
Client-Based Authentication	Refers to an end user's device proving its own identity by providing a digital certificate that can be verified by a server in order to gain access to UFM resources	/ufmRest	Client-Based Authentication
Token-Based Authentication	Token-based authentication is a protocol which allows users to verify their identity, and in return receive a unique access token. To use UFM, the user should create a token using UFM Web UI or UFM REST API	/ufmRestV3	Token-Based Authentication
Proxy Authentication	Proxy authentication delegates the user authentication to a remote Proxy server.	/ufmRestV2 or /ufmRestV3	N/A
Azure AD Authentication	Microsoft Azure Authentication is a service provided by Microsoft Azure, the cloud computing platform of Microsoft. It is designed to provide secure access control and authentication for applications and services hosted on Azure.	/ufmRestV2	N/A
Kerberos Authentication	Kerberos is a protocol designed to authenticate service requests between trusted hosts over an untrusted network	/ufmRestKrb	N/A
TACACS+ Authentication	TACACS is a network security protocol designed to provide centralize AAA for users accessing UFM REST APIs.	/ufmRest or /ufmRestV2 or /ufmRestV3	N/A

There are three optional services which can provide authentication handling of UFM.

1. Apache Web Server (used by default) - Standard Apache web server and supports the above-mentioned authentication methods.

2. LDAP Authentication - Commonly used for accessing and managing directory information services over a network via a user-configured LDAP server that is responsible for authentication.

3. UFM Authentication Server - A centralized HTTP server, is responsible for managing various authentication methods supported by UFM.

Client Based Authentication

The management port of the UFM is connected to a secure zone within the customer data center. All communications between any user or service and the UFM must remain encrypted, authenticated, and authorized according to user's security standards.

This option allows the user to set/configure required certificates within UFM server for supporting client-based authentication.

Associating Client Certificate Common Name (CN) with UFM User

It is necessary to associate the Common Name (CN) of a client certificate with a specific UFM user. UFM will internally store the CN and link it to the corresponding user.

Once Apache validates the client certificate's chain, UFM will extract the CN from the certificate and compare it against the stored value. If the CN does not match, the request will be denied. If the CN matches, the request will be authorized, and the system will treat it as though the associated user had logged in directly.

Integrating User Identity into UFM

The user component, also referred to as the "Agent," operates locally on both the UFM master and standby nodes. The Agent's role is to transfer valid certificates into a local folder on the host (which is mapped to the UFM container) and to trigger a local UFM script to restart Apache once the certificates have been updated.

Configuration

To configure UFM with client certificate authentication using locally refreshed certificates, follow these steps:

1. **Install the UFM container with the `--local-certs-dir` flag, which maps a local host directory to the UFM container.**

   > **Note**
   >
   > Steps 2-6 should be performed once UFM container is running.

2. **Run the UFM service. Run:**

   ```
   systemctl start ufm-enterprise
   ```

3. **Enter the UFM container**. Run:

   ```
   docker exec -it ufm /bin/bash
   ```

4. **Enable client certificate authentication.** Run:

   ```
   /opt/ufm/scripts/manage_client_authentication.sh enable-
   client-cert-authen
   ```

5. **Set the client certificate subject identifier to CN.** Run:

   ```
   /opt/ufm/scripts/manage_client_authentication.sh set-subject-
   identifier --identifier CN
   ```

6. **Associate a client certificate CN with a UFM user.** Run:

```
/opt/ufm/scripts/manage_client_authentication.sh associate-user --cn <CN> --username <UFM user>
```

7. **Set the SSL server certificate hostname.** Run:

```
/opt/ufm/scripts/manage_client_authentication.sh set-ssl-cert-hostname --hostname <hostname>
```

8. **Restart Apache once the certificates have been refreshed.**

> **Note**
>
> Note: This command should be embedded in the "Agent" and can be invoked after the certificates have been refreshed and UFM container is running. Run:
>
> ```
> docker exec ufm /opt/ufm/scripts/manage_client_authentication.sh restart_ufm_websrv
> ```

LDAP Authentication

LDAP (Lightweight Directory Access Protocol) is commonly used for accessing and managing directory information services over a network.

If an LDAP server is configured, users can authenticate to UFM using LDAP as an external authentication method. To set up LDAP authentication, run the following script:

```
/opt/ufm/scripts/ufm_configure_ldap.py --auth-url='<auth_url>' --bind-dn='<bind_dn>' --–bind-password='<bind_password>'
```

For detailed usage instructions, refer to the script's help documentation.

Important: After executing the script, restart UFM to apply the updated Apache configuration. Once restarted, users can log in to the UFM Web UI and access the UFM REST API using their LDAP credentials. To verify the configuration, attempt to log in to the UFM Web UI with an LDAP username and password.

> **Note**
>
> If UFM is deployed in Docker container mode, run the configuration script inside the container as follows:
>
> ```
> docker exec -it ufm sh -c "/opt/ufm/scripts/ufm_configure_ldap.py --auth-url='<auth_url>' --bind-dn='<bind_dn>' --–bind-password='<bind_password>'"
> ```

TACACS+ Authentication

TACACS (Terminal Access Controller Access-Control System) is a network security protocol designed to provide centralized *Authentication*, *Authorization*, and *Accounting* (AAA) for users accessing UFM REST APIs.

TACACS+ Authentication UFM Configurations

Parameter	Description	Example/Constraints
`tacacs_auth_enabled`	Master switch for TACACS authentication	true/false (default: false)

Parameter	Description	Example/Constraints
`tacacs_address`	IPv4, IPv6, or resolvable hostname of comma separated multiple TACACS servers. *The order determines the fallback order. The first reachable server will be used for subsequent requests.*	192.168.1.100 or tacacs.example.com
`tacacs_port`	TCP port for TACACS communication	49
`tacacs_timeout`	TACACS request timeout in seconds	10 seconds
`tacacs_shared_key`	Pre-shared secret for encrypting TACACS+ packets.	Must match server configuration

The following script provided to help in setting up TACACS+ authentication in UFM:

```
/opt/ufm/scripts/ufm_configure_tacacs_rest_auth.sh \
    --tacacs-address 192.168.1.100,192.168.1.101,192.168.1.102 \
    --tacacs-port 49 \
    --tacacs-shared-key testing123
```

```
/opt/ufm/scripts/ufm_configure_tacacs_rest_auth.sh --enable
```

After setting up the configurations and enabling the TACACS+ authentication, the new settings will be reflected automatically within up to 120 seconds. If you need the changes to take effect immediately, a UFM restart is required.

Note

If UFM is deployed in Docker container mode, run the configuration script inside the container as follows:

```
docker exec -it ufm sh -c
"/opt/ufm/scripts/ufm_configure_tacacs_rest_auth.sh --help"
```

TACACS+ Server Configurations

Below is an example of TACACS users/roles configurations that should be included as part of the external TACACS+ server.

```
# Define where to log accounting data. The default is:
accounting file = /var/log/tac_plus.acct

# This is the key that clients have to use to access TACACS+ - It
should be the same key that provided to UFM
key = testkey123

user = tac_admin {
    default service = permit # allow to access all UFM REST APIs
    login = cleartext testpass
}

group = custom_group {
    # users with custom_group role will be allowed to perform the
below UFM REST APIs
    service = /ufmRest/app/ufm_config {
        method = "GET"
    }
    service = /ufmRest/app/tokens {
        method = "GET"
        method = "POST"
    }
}

user = tac_user {
    member = custom_group
    login = cleartext testpass
}
```

UFM REST API Example with TACACS+ Authentication

Below is an example of accessing UFM REST API using TACACS credentials:

```
curl -k -H “X-Auth-Type: tacacs” -u
tacacs_username:tacacs_password https://<UFM_IP>/ufmRest/app/ufm_config
```

UFM Authentication Server

Configurations of the UFM Authentication Server

The UFM Authentication Server is designed to be configurable and is initially turned on by default.

Enabling the UFM Authentication Server provides a centralized service that oversees all supported authentication methods within a single service, consolidating them under a unified authentication API.

Apache utilizes the authentication server's APIs to determine a user's authentication status.

All activities of the UFM Authentication Server are logged in the `authentication_service.log` file, located at `/opt/ufm/files/log`.

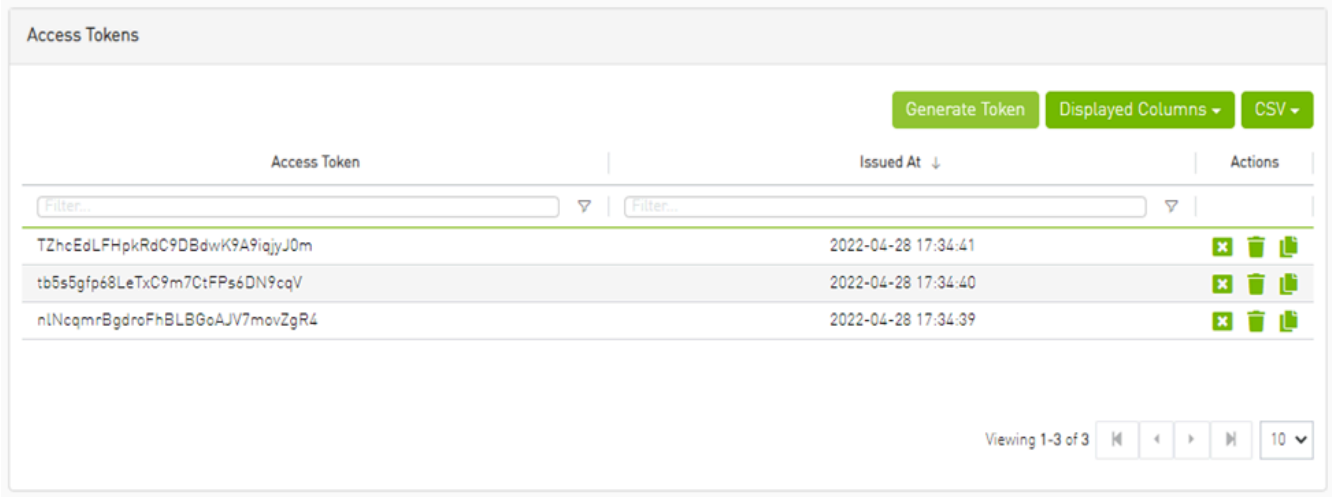
To enable/disable the UFM Authentication Server, refer to .

Token-Based Authentication

Token-based authentication is a protocol which allows users to verify their identity, and in return receive a unique access token. During the life of the token, users then access the UFM APIs that the token has been issued for, rather than having to re-enter credentials each time they need to use any UFM API.

> **Note**
>
> Under the Settings section there is a tab titled called “Access Tokens”.

The functionality of the added tab is to give the user the ability to create new tokens & manage the existing ones (list, copy, revoke, delete):



Actions:

Name	Icon	Description
Revoke		Revoke a specific token. **Note** The revoked token will no longer be valid.
Delete		Delete a specific token.
Copy		Copy specific token into the clipboard.

Note

Each user is able to list and manage only the tokens that have been created by themselves. Only the users with system_admin role will be able to create tokens.

Proxy Authentication

Delegating Authentication to a Proxy

To allow a custom user authentication, you can configure UFM to delegate the user authentication to a remote Proxy server. The remote Proxy server is written by the user, thus, allowing flexibility on deciding how the authentication is performed.

By default, the feature is disabled. To activate the feature, configure `auth_proxy_enabled` with `true`.

In case server authentication is enabled, use /ufmRestV2, otherwise, use /ufmRestV3 to send requests to UFM.

The request header should contain a username and role. The available roles are System_Admin, Fabric_Admin, Fabric_Operator, and Monitoring_Only. If the request header is sent without a username or a role, it is rejected by the UFM.

For example:

```
[AuthProxy]
# Defaults to false, but set to true to enable this feature
auth_proxy_enabled = true
# HTTP Header name that will contain the username
auth_proxy_header_name = X_WEBAUTH_USER
# HTTP Header name that will contain the user roles. The
available roles are as follows: System_Admin, Fabric_Admin,
Fabric_Operator, and Monitoring_Only
auth_proxy_header_role = X_WEBAUTH_ROLE

# Set to `true` to enable auto sign up of users who do not exist in
UFM DB. Defaults to `true`.
auth_proxy_auto_sign_up = true
# Limit where auth proxy requests come from by configuring a list
of IP addresses.
# This can be used to prevent users spoofing the X_WEBAUTH_USER
header.
# This option is required
# Example `whitelist = 192.168.1.1, 192.168.1.0/24, 2001::23, 2001::0/120`
auth_proxy_whitelist =
# Define a list of allowed Common Names (CNs) for mTLS on
authentication proxy requests.
# Only clients with certificates that contain CN values from this
list are allowed to act as an authentication proxy.
# Example: `auth_proxy_cn_whitelist = user1, service1, client-
cert`.
auth_proxy_cn_whitelist =
```

Parameter	Default	Description
auth_proxy_enabled	false	Enables authentication by proxy
auth_proxy_header_name	X_WEBAUTH_USER	The HTTP header that contains the UFM username, to be supplied by the proxy.

Parameter	Default	Description
auth_proxy_header_role	X_WEBAUTH_ROLE	The HTTP header that will include the UFM role. The available predefined roles are: • System_Admin • `Fabric_Admin` • `Fabric_Operator` • `Monitoring_Only`
auth_proxy_auto_sign_up	True	Allows automatic signup of users who are not yet present in UFM.
auth_proxy_whitelist	N/A	Restricts the source of authentication proxy requests by setting up a list of allowed IP addresses. This field is mandatory when the client certificate is disabled and optional when the client certificate is enabled. Example: `whitelist = 192.168.1.1, 192.168.1.0/24, 2001::23, 2001::0/120`
auth_proxy_cn_whitelist	N/A	Specifies a list of permitted Common Names (CNs) for mTLS in authentication proxy requests. Only clients with certificates containing CN values from this list are authorized to act as an authentication proxy. Example: `auth_proxy_cn_whitelist = user1, service1, client-cert` . This field is mandatory when the client certificate is enabled, and the subject identifier of the client certificate is the Common Name (CN). For additional details, see Client-Based Authentication. To use this feature, the Authentication Server must be enabled. For more information, refer to Enabling UFM Authentication Server

Azure AD Authentication

Microsoft Azure Authentication is a service provided by Microsoft Azure, the cloud computing platform of Microsoft. It is designed to provide secure access control and authentication for applications and services hosted on Azure.

UFM supports Authentication using Azure Active Directory, and to do so, you need to follow the following steps:

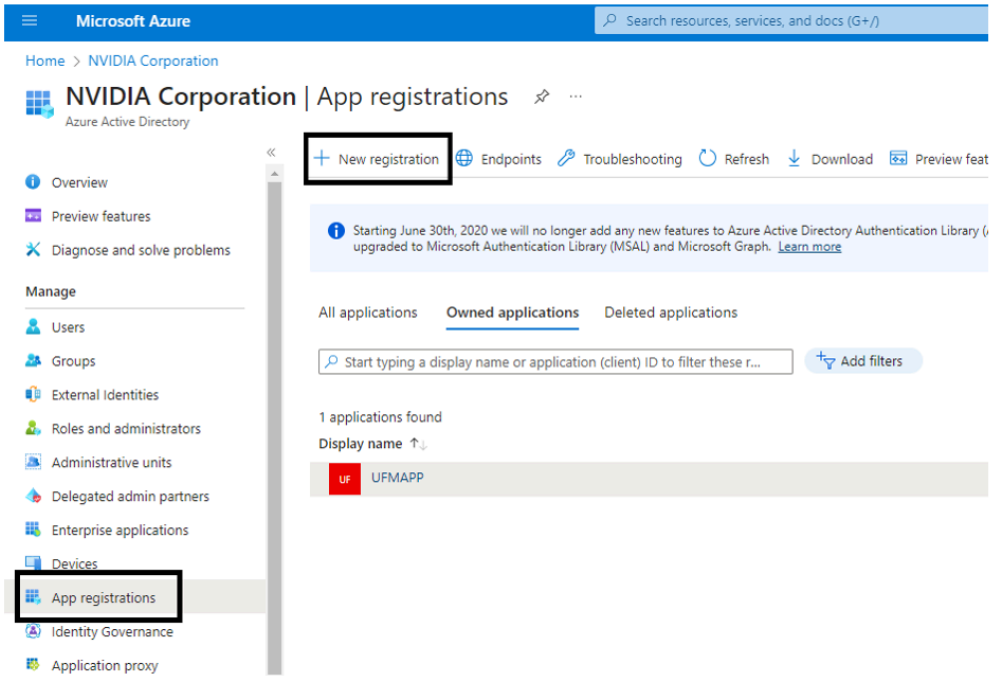
Register UFM in Azure AD Portal

To log in via Azure, UFM must be registered in the Azure portal using the following steps:

1. Log in to Azure Portal, then click "**Azure Active Directory"** in the side menu.

2. If you have access to more than one tenant, select your account in the upper right. Set your session to the Azure AD tenant you wish to use.

3. Under "**Manage"** in the side menu, click App Registrations > New Registration.

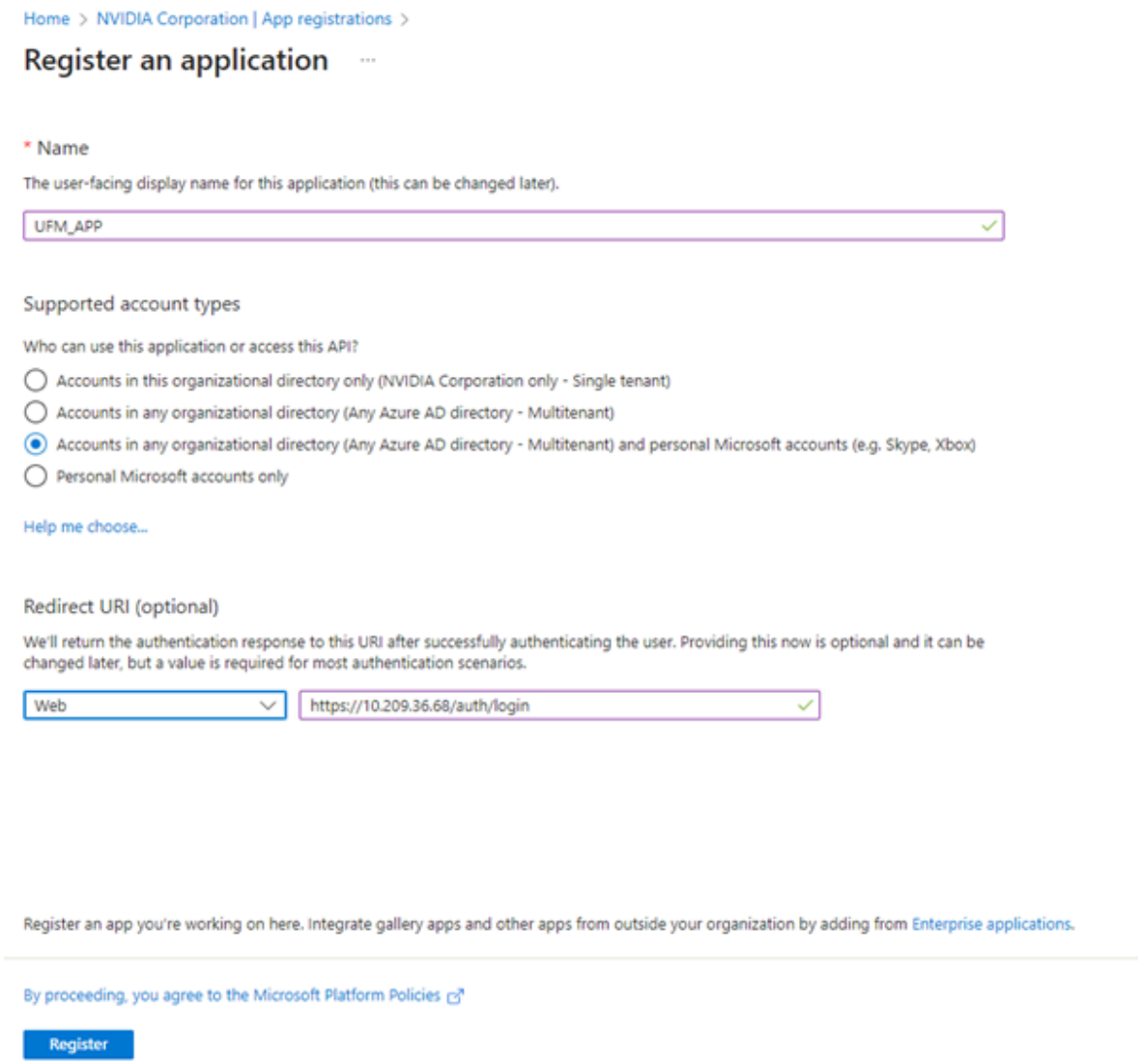


4. Provide the application details:

    1. **Name**: Enter a descriptive name.

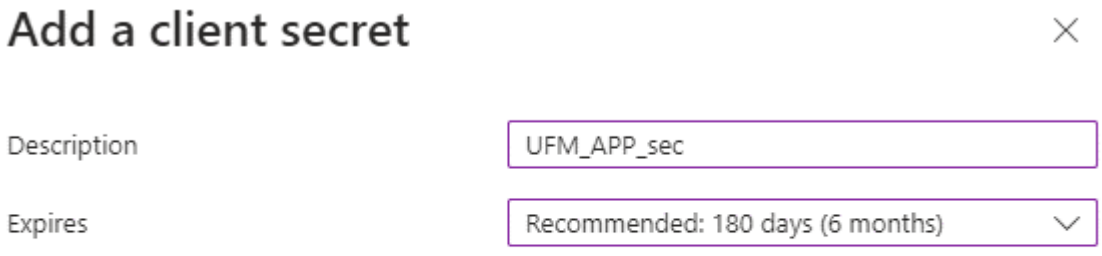
    2. **Supported account types**: Account types that are allowed to login and use the registered application.

    3. **Redirect URL**: select the app type **Web**, and Add the following redirect URL https:///auth/login



Then, click **Register**. The app's **Overview** page opens.

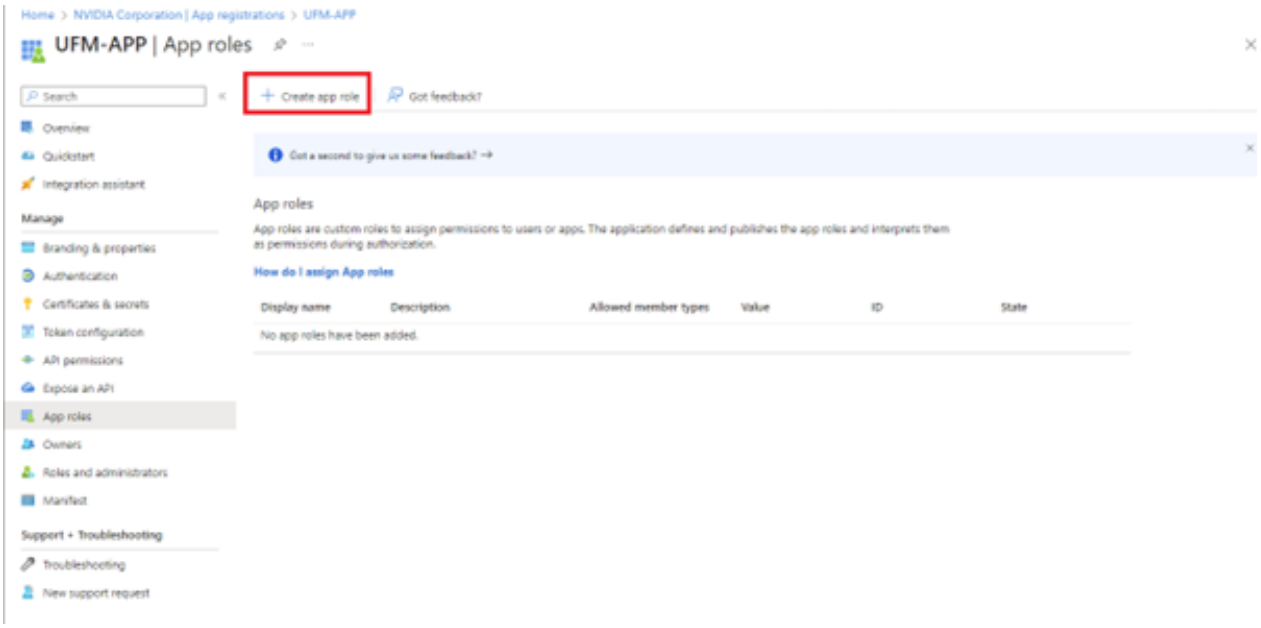
5. Under **Manage** in the side menu, click **Certificates & Secrets** > New client secret.



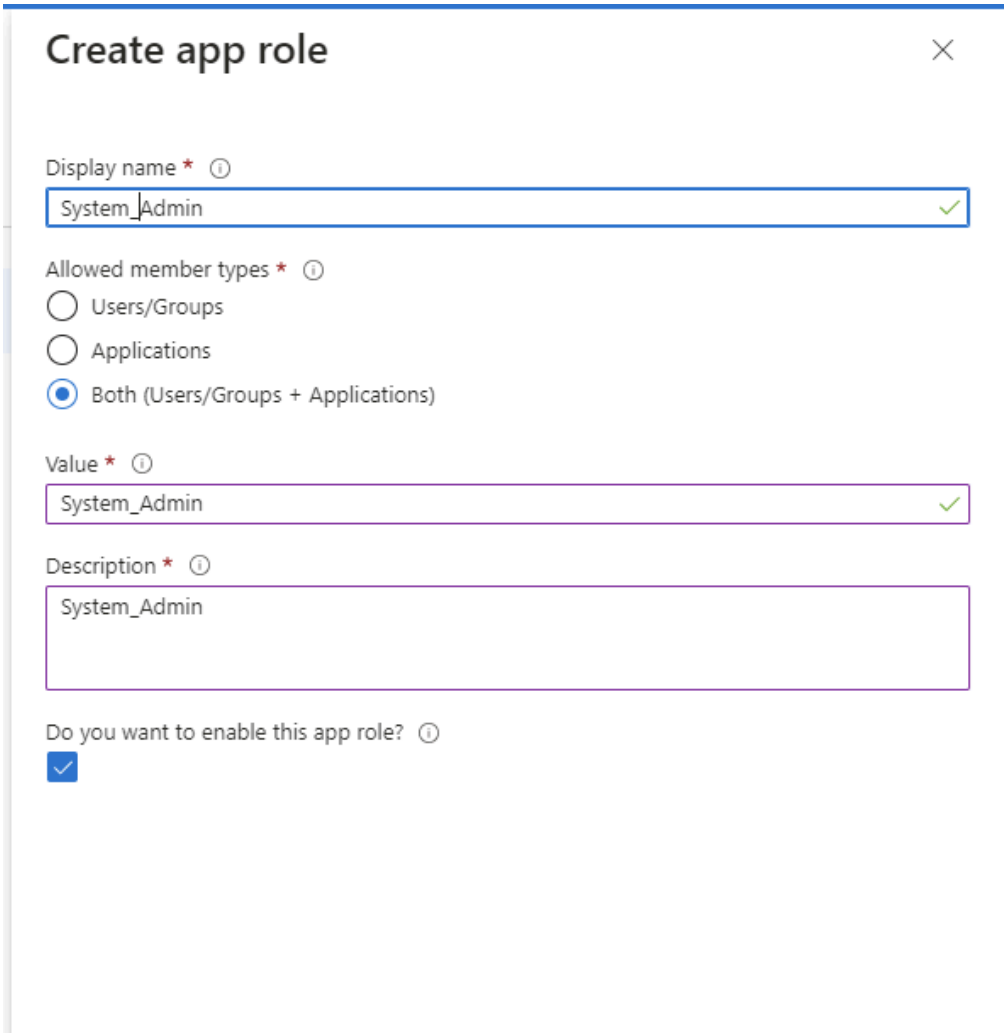
Provide a description for the client secret and set an expiration time, then click "**Add**."

6. Copy the client secret key value which will be needed to configure the UFM with Azure AD (Please note that the value of the generated secret will be hidden and will not be able to be copied/read after you leave the page.

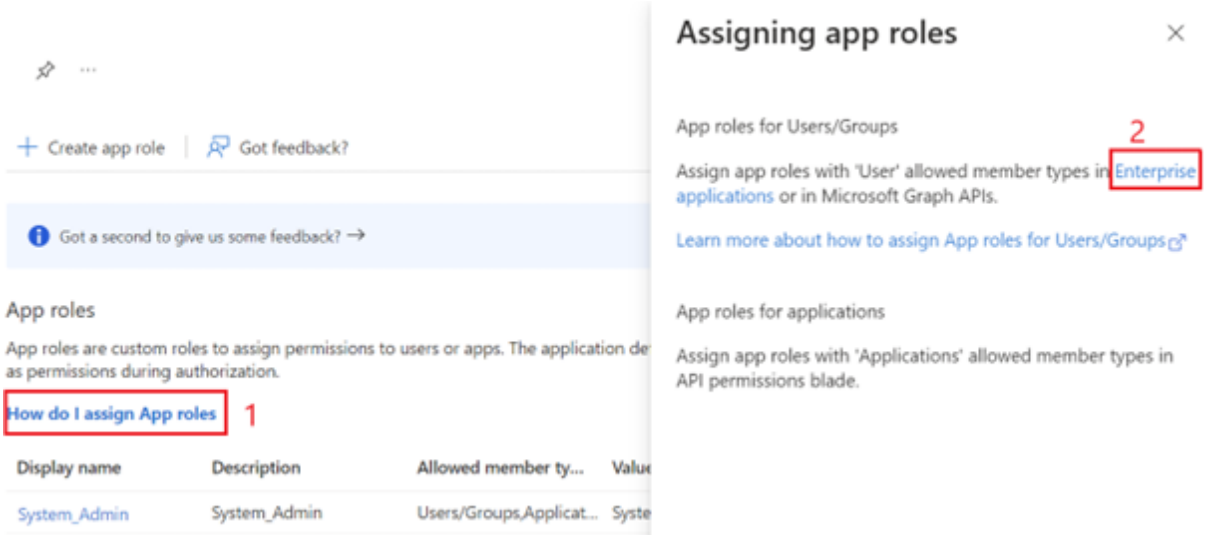
   Under "**Manage"** in the side menu, click App roles > Create app role.

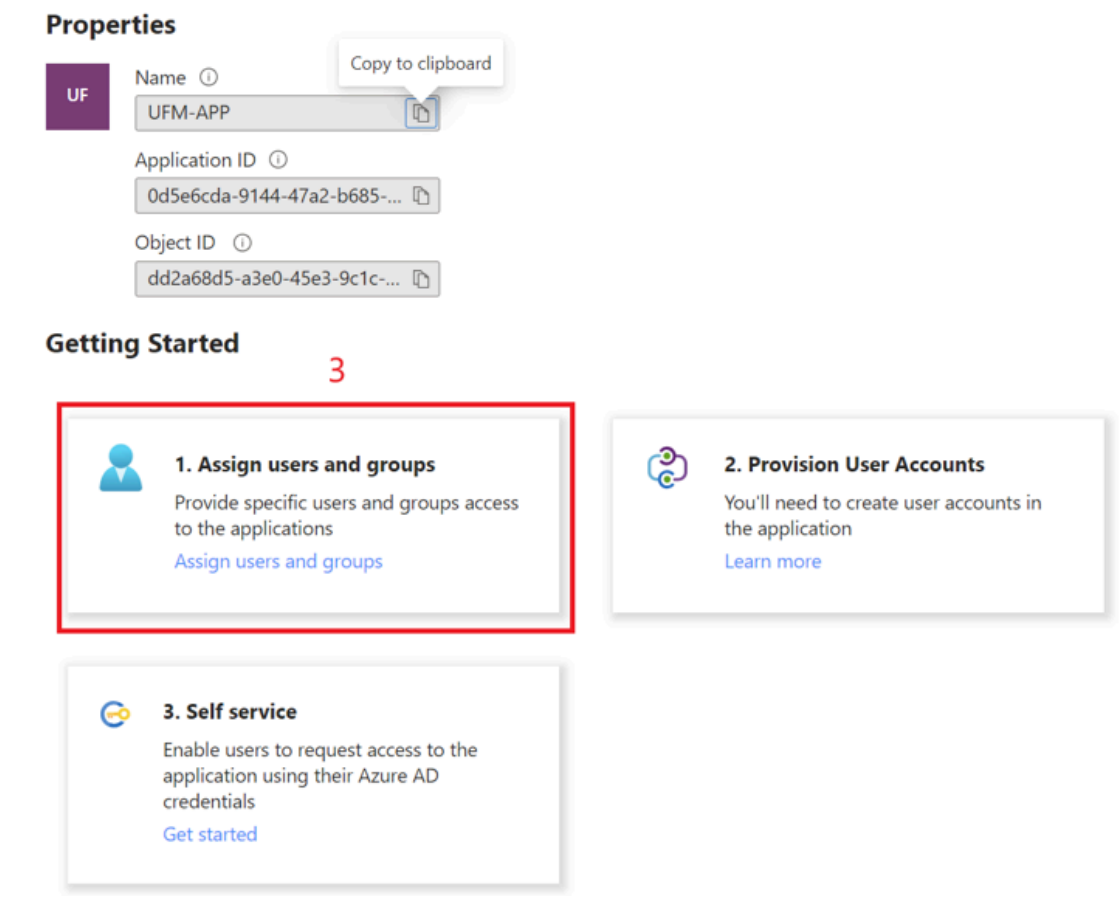


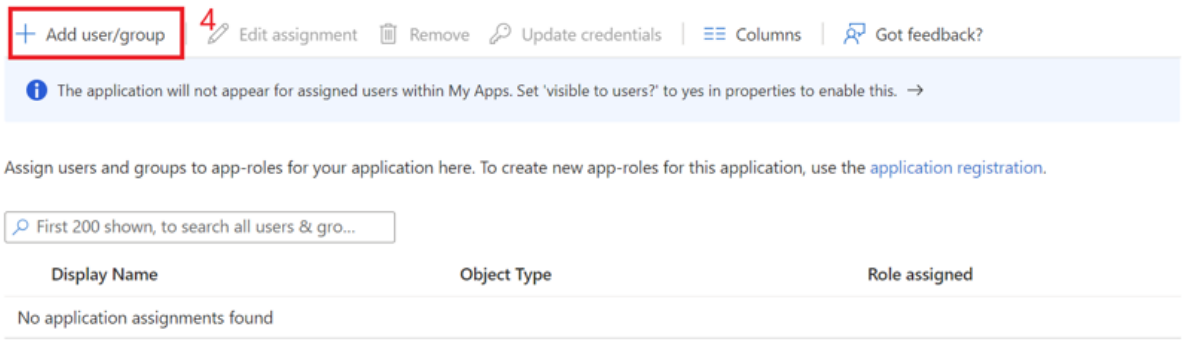
7. Provide the role details. Please note that the role value must be a valid UFM role; otherwise, the login will fail.

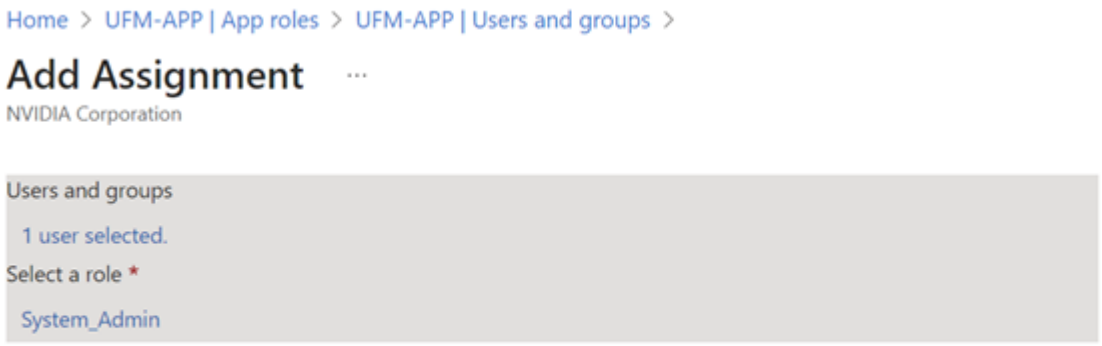


8. Assign the created role to the user. Follow the below steps:




Properties
UF
Name
Copy to clipboard
UFM-APP
Application ID
0d5e6cda-9144-47a2-b685-...
Object ID
dd2a68d5-a3e0-45e3-9c1c-...
Getting Started
3
1. Assign users and groups
Provide specific users and groups access to the applications
Assign users and groups
2. Provision User Accounts
You'll need to create user accounts in the application
Learn more
3. Self service
Enable users to request access to the application using their Azure AD credentials
Get started


Add user/group
4
Edit assignment
Remove
Update credentials
Columns
Got feedback?
The application will not appear for assigned users within My Apps. Set 'visible to users?' to yes in properties to enable this.
Assign users and groups to app-roles for your application here. To create new app-roles for this application, use the application registration.
First 200 shown, to search all users & gro...
Display Name
Object Type
Role assigned
No application assignments found



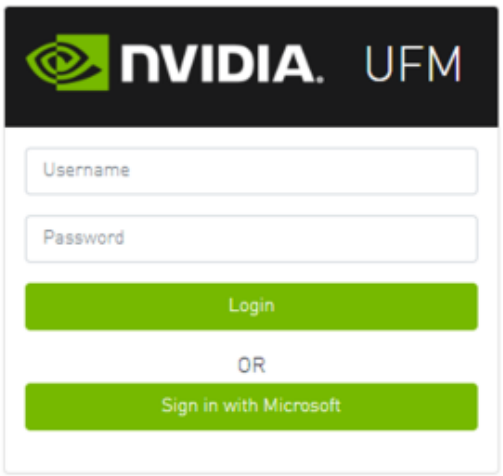
9. Click on "**Overview**" in the side menu to view the application information, such as tenant ID, client ID, and other details.

Enable Azure Authentication from UFM

Azure authentication is disabled by default. To enable it, please refer to Enabling Azure AD Authentication.

Azure Authentication Login Page

After enabling and configuring Azure AD authentication, an additional button will appear on the primary UFM login page labeled 'Sign In with Microsoft,' which will leads to the main Microsoft sign-in page:



Kerberos Authentication

Kerberos is a network authentication protocol designed to provide strong authentication for client-server applications by using secret-key cryptography.

The Kerberos protocol works on the basis of tickets, it helps ensure that communication between various entities in a network is secure. It uses symmetric-key cryptography, which means both the client and servers share secret keys for encrypting and decrypting communication.

To enable Kerberos Authentication, refer to Enabling Kerberos Authentication.

Setting Up Kerberos Server Machine

To set up a system as a Kerberos server, perform the following:

1. Install the required packages:

```
#Redhat
sudo yum install krb5-libs krb5-server
# Ubuntu
sudo apt-get install krb5-kdc krb5-admin-server
```

2. Edit the Kerberos configuration file '`/etc/krb5.conf`' to reflect your realm, domain and other settings:

```
[libdefaults]
    default_realm = YOUR-REALM

[realms]
    YOUR-REALM = {
        kdc = your-kdc-server
        admin_server = your-admin-server
    }

[domain_realm]
    your-domain = YOUR-REALM
    your-domain = YOUR-REALM
```

3. Use the `kdb5_util` command to create the Kerberos database:

```
kdb5_util create -r YOUR-REALM -s
```

4. Add administrative principals:

```
 Kadmin.local addprinc -randkey HTTP/YOUR-HOST-NAME@YOUR-
REALM
```

5. Start KDC and Kadmin services:

```
sudo systemctl start krb5kdc kadmin
sudo systemctl enable krb5kdc kadmin
```

6. Generate a keytab file. The keytab file contains the secret key for a principal and is used to authenticate the service.

```
 kadmin.local ktadd -k /path/to/your-keytab-file HTTP/YOUR-HOST-NAME@YOUR-REALM
```

Replace `/path/to/your-keytab-file` with the actual path where you want to store the keytab file.

Setting Up Kerberos Client Machine

Follow the below steps to set up a system as a Kerberos client.

1. Install the required packages. When installing the UFM, the following packages will be installed as dependencies:

```
#Redhat
krb5-libs krb5-workstation mod_auth_gssapi
# Ubuntu
krb5-config krb5-user libapache2-mod-auth-gssapi
```

2. Configure the /etc/krb5.conf file to reflect your realm, domain, local names map and other settings:

```
kinit -k -t /path/to/your-keytab-file HTTP/YOUR-HOST-NAME@YOUR-REALM
```

```
[libdefaults]
    default_realm = YOUR-REALM

[realms]
    YOUR-REALM = {
        kdc = your-kdc-server
        admin_server = your-admin-server
        auth_to_local_names = {
            your-principle-name = your-local-user
        }

    }

[domain_realm]
    your-domain = YOUR-REALM
    your-domain = YOUR-REALM
```

3. Copy the keytab file from the Kerberos server to the machine where your service runs (the client). It is important to ensure that it is kept confidential.

   Please ensure that the keytab file exists and that Apache has the necessary read permissions to access the keytab file; otherwise, Kerberos authentication will not function properly.

4. Obtain a Kerberos ticket-granting ticket (TGT):

5. Enable Kerberos Authentication from UFM. Kerberos authentication is disabled by default. To enable it, please refer to Enabling Kerberos Authentication.

6. Test the Kerberos Authentication. You can use curl to test whether the user can authenticate to UFM REST APIs using Kerberos.

```
curl --negotiate -i -u : -k 'https://ufmc-eos01/ufmRestKrb/app/tokens'
```

UFM Server Health Monitoring

The UFM Server Health Monitoring module is a standalone module that monitors UFM resources and processes according to the settings in the */opt/ufm/files/conf/UFMHealthConfiguration.xml* file.

For example:

- Each monitored resource or process has its own failure condition (number of retries and/or timeout), which you can configure.
- If a test fails, UFM will perform a *corrective operation*, if defined for the process, for example, to restart the process. You can change the configured corrective operation. If the corrective operation is set to "None", after the defined number of failures, the *give-up* operation is performed.
- If a test reaches the configured threshold for the number of retries, the health monitoring initiates the *give-up* operation defined for the process, for example, UFM failover or stop.
- By default, events and alarms are sent when a process fails, and they are also recorded in the internal log file.

Each process runs according to its own defined schedule, which you can change in the configuration file.

Changes to the configuration file take effect only after a UFM Server restart. (It is possible to kill and run in background the process nohup python /opt/ufm/ufmhealth/UfmHealthRunner.pyo &.)

You can also use the configuration file to improve disk space management by configuring:

- How often to purge MySQL binary log files.
- When to delete compressed UFM log files (according to free disk space).

The settings in the */opt/ufm/files/conf/UFMHealthConfiguration.xml* file are also used to generate the UFM Health Report.

The following section describes the configuration file options for UFM server monitoring.

UFM Health Configuration

The UFM health configuration file contains three sections:

- Supported Operations—This section describes all the operations that can be used in tests, and their parameters.
- Supported Tests—This section describes all the tests. Each test includes:
    - The main test operation.
    - A corrective operation, if the main operation fails.
    - A give-up operation, if the main operation continues to fail after the corrective operation and defined number of retries.

  The number of retries and timeout is also configured for each test operation.
- Test Schedule - This section lists the tests in the order in which they are performed and their configured frequency.

The following table describes the default settings in the */opt/ufm/files/conf/UFMHealthConfiguration.xml* file for each test. The tests are listed in the order in which they are performed in the default configuration file.

You might need to modify the default values depending on the size of your fabric.

For example, in a large fabric, the SM might not be responsive for *sminfo* for a long time; therefore, it is recommended to increase the values for timeout and number of retries for **SMResponseTest**.

Recommended configurations for *SMResponseTest* are:

- For a fabric with 5000 nodes:
    - Number of retries = 12
    - Frequency = 10
- For a fabric with 10000 nodes:
    - Number of retries = 12

- Frequency = 20

Test Name / Description	Test Operation	Corrective Operation (if Test Operation fails)	No. Op
CpuUsageTest Checks total CPU utilization.	CPUTest Tests that overall CPU usage does not exceed 80% (this percentage is configurable).	None If UFM Event Burst Management is enabled, it is automatically initiated when the test operation fails	1 R No
AvailableDiskSpaceTest Checks available disk space.	FreeDiskTest Tests that disk space usage for */opt/ufm* does not exceed 90% (this percentage is configurable).	CleanDisk Delete compressed UFM log files under */opt/ufm*	3 R No
CheckIBFabricInterface Checks state of active fabric interface.	IBInterfaceTest Tests that active fabric interface is up.	BringUpIBFabricInterface Bring up the fabric interface	3 R SM
MemoryTest Checks total memory usage.	MemoryUsageTest Tests that memory usage does not exceed 90% (this percentage is configurable).	None	1 R No
SMProcessTest Checks status of the OpenSM service.	SMRunningTest Tests that the SM process is running.	RestartProcess Restart the SM process	1 R UFI
SMResponseTest Checks responsiveness of SM (when SM process is running).	SMTest Tests SM responsiveness by sending the sminfo query to SM.	None	9 R UFI
IbpmTest Checks status of the IBPM (Performance Manager) service.	ProcessIsRunningTest Tests that the IBPM service is running.	RestartProcess Restart the IBPM service	3 R No
ModelMainTest Checks status of the main UFM service	ProcessIsRunningTest Tests that the UFM service is running.	RestartProcess Restart the UFM service	3 R UFI

Test Name / Description	Test Operation	Corrective Operation (if Test Operation fails)	No. Op
HttpdTest Checks status of the httpd service.	ProcessIsRunningTest Tests that the httpd service is running.	RestartProcess Restart the httpd service	3 R No
PluginIsRunningTest Checks status of the UFM plugin. Checking if UFM plugins are running.	Is[plugin name]PluginRunning Test for deployed plugins if they are up and running and operational.	RestartPlugin Start plugin if not running or restart plugin if not operational.	3 R No
UFMServerVersionTest Checks UFM software version and build.	UfmVersionTest Returns UFM software version information.	None	1 R No
UFMServerLicenseTest Checks UFM License information.	UfmLicenseTest Returns UFM License information.	None	1 R No
UFMMemoryTest Checks available UFM memory.	UfmMemoryUsageTest Tests that UFM memory usage does not exceed 80% (this percentage is configurable).	None	1 R No
UFMCpuUsageTest Checks UFM CPU utilization.	CPUTest Tests that UFM CPU usage does not exceed 60% (this percentage is configurable).	None	1 R No
CheckGvSqliteCorruptionTest Checks corruption in GV SQLite database.	CheckSqliteCorruptionTest Checks if a list of defined tables exists in the SQLite database file and verifies the schema (columns) for all defined tables.	None	No

Note

The Supported Operations section of the configuration file includes additional optional operations that can be used as corrective operations or give-up operations.

UFM Core Files Tracking

To receive a notification every time OpenSM or ibpm creates a core dump, please refer to the list of all current core dumps of OpenSM and ibpm in the UFM health report.

To receive core dump notifications, do the following:

1. Set the core_dumps_directory field in the gv.cfg file to point to the location where all core dumps are created (by default, this location is set to /tmp).

   ```
   core_dumps_directory = /tmp
   ```

2. Set the naming convention for the core dump file. The name must include the directory configured in the step above.

   The convention we recommend is:

   ```
   echo "/tmp/%t.core.%e.%p.%h" > /proc/sys/kernel/core_pattern
   ```

3. Make sure core dumps directory setting is persistent between reboots. Add the kernel.core_pattern parameter with the desired file name format to the /etc/systctl.conf file. Example:

   ```
   kernel.core_pattern=/tmp/%t.core.%e.%p.%h
   ```

4. Configure the core file size to be unlimited.

```
ulimit -c unlimited
```

5. (Only on UFM HA master) Update the UFM configuration file gv.cfg to enable core dump tracking.

```
track_core_dumps = yes
```

Example of Health Configuration

The default configuration for the overall memory test in the *opt/ufm/files/conf/UFMHealthConfiguration.xml* file is:

```
<Test Name="MemoryTest" NumOfRetriesBeforeGiveup="3"
RetryTimeoutInSeconds="10">
    <TestOperation Name="MemoryUsageTest">
        <Parameters>
            <Parameter Name="ThresholdInPercents" Value="90"/>
        </Parameters>
    </TestOperation>
    <CorrectiveOperation Name="None"/>
    <GiveupOperation Name="None"/>
</Test>
```

This configuration tests the available memory. If memory usage exceeds 90%, the test is repeated up to 3 times at 10 second intervals, or until memory usage drops to below 90%. No corrective action is taken and no action is taken after 3 retries.

To test with a usage threshold of 80%, and to initiate UFM failover or stop UFM after three retries, change the configuration to:

```
<Test Name="MemoryTest" NumOfRetriesBeforeGiveup="3"
RetryTimeoutInSeconds="10">
    <TestOperation Name="MemoryUsageTest">
        <Parameters>
            <Parameter Name="ThresholdInPercents" Value="80"/>
        </Parameters>
    </TestOperation>
    <CorrectiveOperation Name="None"/>
    <GiveupOperation Name="UFMFailoverOrStop"/>
</Test>
```

Event Burst Management

UFM event burst management can lower the overall CPU usage following an event burst by suppressing events. Event burst management is configured in the *gv.cfg* configuration file.

When the overall CPU usage exceeds the threshold configured by the CpuUsageTest in the */opt/ufm/files/conf/UFMHealthConfiguration.xml* file, a High CPU Utilization event occurs.

This event initiates the UFM event burst management, which:

- Suppresses events. The default level of suppression enables critical events only.
- If, after a specified period of time (30 seconds, by default), no further High CPU Utilization event occurs, the UFM server enables all events.

To modify Event burst management configuration, change the following parameters in the gv. cfg file:

```
# The events' level in case events are suppressed (the possible
levels are disable_all_events, enable_critical_events, and
enable_all_events)
# The entire feature can be turned off using the level
"enable_all_events"
suppress_events_level = enable_critical_events
# The amount of time in seconds which events are suppressed
suppress_events_timeout = 30
```

Events and Alarms

UFM offers comprehensive diagnostics for your InfiniBand fabric, covering a range of categories:

1. Fabric configurations
2. Fabric topology
3. Hardware issues
4. Communication errors
5. Maintenance
6. Security
7. Switch module status
8. NVIDIA SHARP notifications

Events are notifications generated by UFM, indicating issues within the mentioned categories in the InfiniBand fabric. On the other hand, alerts are urgent notifications derived from events (many events can be configured as alarms based on customer preferences).

These detections are performed both before running applications and during standard operation. They help troubleshoot and notify network administrators of potential network issues before they escalate.

Events can originate from various sources:

- SM traps
- SHARP AM traps
- UFM internal analysis, encompassing:
    - Internal detection of topology changes
    - Internal fabric analysis (based on IBDiagnet)

    - Internal monitoring of managed switches
    - Maintenance activities (device action tracking, licensing, cable integrity)
- Threshold-crossing events determined by telemetry counter readings

	WebUI	REST API
Events	UFM events can be viewed via the Events and Alarms WebUI view. Refer to Events & Alarms	Events REST API
	For device-specific events, refer to the Events	N/A
	Configuration of events is managed within the Events Policy Tab in the Settings window	Events Policy REST API
Alarms	UFM alarms can be viewed via the Events and Alarms WebUI view. Refer to Events & Alarms	Alarms REST API
	Configuration of alarms is managed within the Events Policy in the Settings window	N/A

For showing all the UFM-supported events, refer to Threshold-Crossing Events Reference.

Threshold-Crossing Events Reference

This reference lists the threshold-based events that UFM supports, each including a set of attributes listed below. You can view these messages through Events & Alarms. For details about configuring notifications for these events, refer to Configuring Event Management.

- **Event ID**: Unique event identifier
- **Event Name**: Short description of the event
- **To log**: A flag which defines whether the event will be sent to UFM events log or not (1 or 0, respectively).
- **Alarm**: A flag which defines whether UFM alarm is generated when the specified event is triggered by UFM (1 means an alarm is generated). Alarms are used to allow a notification with significant indication.
- **Default Severity**: Indicates the alarm severity (Info, Warning, Minor, Critical)

- **Default Threshold**: Event dependent threshold (for example, event occurrence, counter threshold or temperature threshold).
- **Default TTL**: TTL (Alarm Time to Live) sets the time during which the alarm on the event is visible on UFM Web UI. TTL is defined in seconds. CAUTION: Setting the TTL to 0 makes the alarm permanent, meaning that the alarm does not disappear from the Web UI until cleared manually.
- **Related Object**: The object (context) to which the UFM event is related to (port, switch, gateway, grid, etc).
- **Category**: Indicates the category to which the event is related to.
- **Source**: Indicates the origin source of the specified event (SM, Telemetry, Licensing, UFM)

For information about defining event policy, see Configuring Event Management.

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
64	GID Address In Service	1	0	Info	1	0	Port	Fabric Notificati
65	GID Address Out of Service	1	1	Warning	1	0	Port	Fabric Notificati
66	New MCast Group Created	1	0	Info	1	0	Port	Fabric Notificati
67	MCast Group Deleted	1	0	Info	1	0	Port	Fabric Notificati
110	Symbol Error	1	1	Warning	200	0	Port	Hardware
111	Link Error Recovery	1	1	Minor	1	0	Port	Hardware
112	Link Downed	1	1	Warning	0	0	Port	Hardware
113	Port Receive Errors	1	1	Warning	5	0	Port	Hardware
114	Port Receive Remote Physical Errors	1	1	Minor	5	0	Port	Hardware

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
115	Port Receive Switch Relay Errors	1	1	Minor	50	0	Port	Fabric Configura
116	Port Xmit Discards	1	1	Minor	200	0	Port	Communi Error
117	Port Xmit Constraint Errors	1	1	Minor	1	0	Port	Communi Error
118	Port Receive Constraint Errors	1	1	Minor	1	0	Port	Communi Error
119	Local Link Integrity Errors	1	1	Minor	5	0	Port	Hardware
120	Excessive Buffer Overrun Errors	1	1	Warning	1	0	Port	Communi Error
121	VL15 Dropped	1	1	Info	500	0	Port	Communi Error
122	Congested Bandwidth (%) Threshold Reached	1	1	Minor	10	0	Port	Communi Error
123	Port Bandwidth (%) Threshold Reached	1	1	Minor	95	0	Port	Communi Error
130	Non-optimal link width	1	1	Minor	1	0	Port	Hardware
134	T4 Port Congested Bandwidth	1	1	Warning	10	0	Port	Communi Error
141	Flow Control Update Watchdog Timer Expired	1	1	Warning	1	0	Port	Hardware

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
144	Capability Mask Modified	1	0	Info	1	0	Port	Fabric Notificati
145	System Image GUID changed	1	1	Info	1	0	Port	Communi Error
156	Link Speed Enforcement Disabled	1	0	Critical	0	0	Site	Fabric Notificati
250	Running in Limited Mode	1	1	Critical	1	0	Grid	Maintena
251	Switching to Limited Mode	1	1	Critical	1	0	Grid	Maintena
252	License Expired	1	1	Warning	1	0	Grid	Maintena
253	Duplicated licenses	1	0	Critical	1	0	Grid	Maintena
254	License Limit Exceeded	1	0	Critical	1	0	Grid	Maintena
255	License is About to Expire	1	0	Warning	1	0	Grid	Maintena
256	Bad M_Key	1	1	Critical	1	0	Port	Security
257	Bad P_Key	1	1	Critical	1	0	Port	Security
258	Bad Q_Key	1	1	Critical	1	0	Port	Security
259	Bad P_Key Switch External Port	1	0	Minor	1	0	Port	Security
271	ISBL LAG Port Up	1	0	Info	1	0	Port	Fabric Notificati
272	ISBL LAG Port Down	0	0	Critical	1	0	Port	Fabric Notificati
273	LAG Port Up	1	0	Info	1	0	Port	Fabric Notificati
274	LAG Port Down	0	0	Warning	1	0	Port	Fabric Notificati

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
275	Port Up	1	0	Info	1	0	Port	Fabric Notificati
276	Port Down	0	0	Warning	1	0	Port	Fabric Notificati
277	Port of LAG Up	1	0	Info	1	0	Port	Fabric Notificati
278	Port of LAG Down	0	0	Warning	1	0	Port	Fabric Notificati
279	Port of ISBL Up	1	0	Info	1	0	Port	Fabric Notificati
280	Port of ISBL Down	0	0	Warning	1	0	Port	Fabric Notificati
328	Link is Up	1	0	Info	1	10	Site	Fabric Top
329	Link is Down	1	0	Warning	1	10	Site	Fabric Top
331	Node is Down	1	0	Warning	1	0	Site	Fabric Top
332	Node is Up	1	0	Info	1	0	Site	Fabric Top
336	Port Action Succeeded	1	0	Info	1	0	Port	Maintena
337	Port Action Failed	0	0	Critical	1	0	Port	Maintena
338	Device Action Succeeded	1	0	Info	1	0	Port	Maintena
339	Device Action Failed	0	0	Critical	1	0	Port	Maintena
344	Partial Switch ASIC Failure	1	1	Critical	1	0	Switch	Maintena
352	Network Added	0	0	Info	1	0	Network	Logical M
353	Network Removed	0	0	Info	1	0	Network	Logical M
380	Switch Upgrade Error	1	1	Critical	1	0	Switch	Maintena

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
381	Switch Upgrade Error	1	1	Critical	1	0	Switch	Maintena
382	Module status NOT PRESENT	1	1	Warning	1	0	Switch	Module S
383	Host Upgrade Failed	1	1	Warning	1	0	Computer	Maintena
384	Switch Module Powered Off	1	1	Info	1	0	Switch	Module S
385	Switch FW Upgrade Started	1	0	Info	1	0	Switch	Maintena
386	Switch SW Upgrade Started	1	0	Info	1	0	Switch	Maintena
387	Switch Upgrade Finished	1	0	Info	1	0	Switch	Maintena
388	Host FW Upgrade Started	1	0	Info	1	0	Computer	Maintena
389	Host OFED Upgrade Started	1	0	Info	1	0	Computer	Maintena
391	Switch Module Removed	1	0	Info	1	0	Switch	Fabric Notificati
392	Module Temperature Threshold Reached (Managed IB switches only)	1	1	Critical	80	0	Module	Hardware
393	Switch Module Added	1	0	Info	1	0	Switch	Fabric Notificati
394	Module status FAULT	1	1	Critical	1	0	Switch	Module S

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
395	Device Action Started	1	0	Info	1	0	Port	Maintena
396	Site Action Started	1	0	Info	1	0	Port	Maintena
397	Site Action Failed	0	0	Minor	1	0	Port	Maintena
398	Switch Chip Added	1	0	Info	1	0	Switch	Fabric Notificati
399	Switch Chip Removed	1	0	Critical	1	0	Switch	Fabric Notificati
400	Switch Module Replaced	1	0	Info	1	0	Switch	Fabric Notificati
403	Device Pending Reboot	1	1	Warning	0	0	Device	Maintena
404	System Information is missing	1	1	Warning	1	0	Switch	Communi Error
405	Switch Identity Validation Failed	1	1	Warning	1	0	Switch	Communi Error
406	Switch System Information is missing	1	1	Warning	1	0	Switch	Communi Error
407	COMEX Ambient Temperature Threshold Reached	1	1	Minor	60	0	Switch	Hardware
408	Switch is Unresponsive	1	1	Critical	1	0	Switch	Communi Error
502	Device Upgrade Finished	1	0	Info	1	0	Device	Maintena
506	Device Upgrade Finished	1	0	Info	1	0	Device	Maintena

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
508	Core Dump Created	1	1	Info	1	0	Grid	Maintena
510	SM Failover	1	1	Critical	1	0	Grid	Fabric Notificati
511	SM State Change	1	1	Info	1	0	Grid	Fabric Notificati
512	SM UP	1	1	Info	1	0	Grid	Fabric Notificati
513	SM System Log Message	1	1	Minor	1	0	Grid	Fabric Notificati
514	SM LID Change	1	1	Warning	1	0	Grid	Fabric Notificati
515	Fabric Health Report Info	1	1	Info	1	0	Grid	Fabric Notificati
516	Fabric Health Report Warning	1	1	Warning	1	0	Grid	Fabric Notificati
517	Fabric Health Report Error	1	1	Critical	1	0	Grid	Fabric Notificati
518	UFM-related process is down	1	1	Critical	1	0	Grid	Maintena
519	Logs purge failure	1	1	Minor	1	0	Grid	Maintena
520	Restart of UFM-related process succeeded	1	1	Info	1	0	Grid	Maintena
521	UFM is being stopped	1	1	Critical	1	0	Grid	Maintena
522	UFM is being restarted	1	1	Critical	1	0	Grid	Maintena
523	UFM failover is being attempted	1	1	Info	1	0	Grid	Maintena

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
524	UFM cannot connect to DB	1	1	Critical	1	0	Grid	Maintena
525	Disk utilization threshold reached	1	1	Critical	1	0	Grid	Maintena
526	Memory utilization threshold reached	1	1	Critical	1	0	Grid	Maintena
527	CPU utilization threshold reached	1	1	Critical	1	0	Grid	Maintena
528	Fabric interface is down	1	1	Critical	1	0	Grid	Maintena
529	UFM standby server problem	1	1	Critical	1	0	Grid	Maintena
530	SM is down	1	1	Critical	1	0	Grid	Maintena
531	DRBD Bad Condition	1	1	Critical	1	0	Grid	Maintena
532	Remote UFM-SM Sync	1	1	Info	1	0	Grid	Maintena
533	Remote UFM-SM problem	1	1	Critical	1	0	Site	Maintena
536	UFM Health Watchdog Info	1	1	Info	1	0	Grid	Maintena
537	UFM Health Watchdog Critical	1	1	Critical	1	0	Grid	Maintena
538	Time Diff Between HA Servers	1	1	Warning	1	0	Grid	Maintena
539	DRBD TCP Connection Performance	1	1	Warning	1	0	Grid	Maintena

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
540	Daily Report Completed successfully	1	0	Info	1	0	Grid	Maintena
541	Daily Report Completed with Error	1	0	Minor	1	0	Grid	Maintena
542	Daily Report Failed	1	0	Critical	1	0	Grid	Maintena
543	Daily Report Mail Sent successfully	1	0	Info	1	0	Grid	Maintena
544	Daily Report Mail Sent Failed	1	0	Minor	1	0	Grid	Maintena
545	SM is not responding	1	1	Critical	1	0	Grid	Maintena
546	Management interface is down	1	1	Critical	1	0	Grid	Maintena
547	UFM stopped polling SM for updates	1	1	Critical	1	0	Grid	Maintena
548	Failed to run generic script test	1	1	Critical	1	0	Grid	Maintena
551	General External Event Notification	1	0	Info	0	0	Grid	Maintena
552	General External Event Notice	1	0	Minor	0	0	Grid	Maintena
553	General External Event Alert	1	0	Warning	0	0	Grid	Maintena

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
554	General External Event Error	1	0	Critical	0	0	Grid	Maintena
560	User Connected	1	0	Info	1	0	Grid	Security
561	User Disconnected	1	0	Info	1	0	Grid	Security
602	UFM Server Failover	1	1	Critical	1	0	Site	Fabric Notificati
603	Events Suppression	1	0	Critical	0	0	Site	Maintena
604	Report Succeeded	1	1	Info	1	0	Grid	Maintena
605	Report Failed	1	1	Critical	1	0	Grid	Maintena
606	Correction Attempts Paused	1	0	Warning	1	0	Site	Fabric Notificati
610	Monitoring History Enabled	1	1	Info	1	0	Grid	Maintena
611	Monitoring History Disabled	1	1	Info	1	0	Grid	Maintena
612	Monitoring History Bad Connection	1	1	Critical	1	0	Grid	Maintena
613	Monitoring History Connection Established	1	1	Info	1	0	Grid	Maintena
614	Monitoring History Inconsistent Version	1	1	Critical	1	0	Grid	Maintena

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
615	Monitoring History Failed save metadata	1	1	Critical	1	0	Grid	Maintena
616	Monitoring History Failed update metadata	1	1	Critical	1	0	Grid	Maintena
617	Monitoring History Failed save data	1	1	Critical	1	0	Grid	Maintena
618	Monitoring History Failed get metadata	1	1	Critical	1	0	Grid	Maintena
619	Monitoring History Failed get data	1	1	Critical	1	0	Grid	Maintena
620	Monitoring History Failed remove file	1	1	Critical	1	0	Grid	Maintena
621	Monitoring History version not matches UFM version	1	1	Critical	1	0	Grid	Maintena
622	Monitoring History Purge DB Occurred	1	1	Warning	1	0	Grid	Fabric Notificati
623	Monitoring History Migration is not completed	1	1	Warning	1	0	Grid	Maintena
624	Monitoring History partition utilization threshold reached	1	1	Critical	1	0	Grid	Maintena

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
625	Monitoring History local report files are about to be cleaned	1	1	Info	1	0	Grid	Maintena
626	Monitoring History oldest DB table is about to be removed	1	1	Info	1	0	Grid	Maintena
627	Monitoring History partition is not mounted	1	1	Critical	1	0	Grid	Maintena
701	Non-optimal Link Speed	1	1	Minor	1	0	Port	Hardware
702	Unhealthy IB Port	1	1	Warning	1	0	Port	Hardware
703	Fabric Collector Connected	1	0	Info	1	0	Grid	Maintena
704	Fabric Collector Disconnected	1	1	Critical	1	0	Grid	Maintena
750	High data retransmission count on port	1	1	Warning	500	0	Port	Hardware
901	Fabric Configuration Started	1	1	Info	1	0	Grid	Fabric Notificati
902	Fabric Configuration Completed	1	1	Info	1	0	Grid	Fabric Notificati
903	Fabric Configuration Failed	1	1	Critical	1	0	Grid	Fabric Notificati

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
904	Device Configuration Failure	1	1	Critical	1	0	Device	Fabric Notificati
905	Device Configuration Timeout	1	1	Critical	1	0	Device	Fabric Notificati
906	Provisioning Validation Failure	1	1	Critical	1	0	Grid	Fabric Notificati
907	Switch is Down	1	1	Critical	1	120	Site	Fabric Top
908	Switch is Up	1	1	Info	1	120	Site	Fabric Top
909	Director Switch is Down	1	1	Critical	1	300	Site	Fabric Top
910	Director Switch is Up	1	1	Info	1	300	Site	Fabric Top
911	Module Temperature Low Threshold Reached (Unmanaged IB switches only)	1	1	Warning	0	0	Module	Hardware
912	Module Temperature High Threshold Reached (Unmanaged IB switches only)	1	1	Critical	0	0	Module	Hardware
913	Module High Voltage	1	1	Warning	15	0	Switch	Module S
914	Module High Current	1	1	Warning	10	0	Switch	Module S
915	Critical BER reported	1	1	Critical	10	0	PORT	Hardware
916	High BER reported	1	1	Warning	10	0	PORT	Hardware

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
917	Critical Symbol BER reported	1	1	Critical	10	0	PORT	Hardware
918	High Symbol BER reported	1	1	Warning	10	0	PORT	Hardware
919	Cable Temperature High	1	1	Critical	0	0	Port	Hardware
920	Cable Temperature Low	1	1	Critical	0	0	Port	Hardware
921	Fan Speed Below Minimum Threshold	1	1	Warning	0	0	Module	Hardware
922	Fan Speed Above Maximum Threshold	1	1	Critical	0	0	Module	Hardware
1257	VPort P_key Violation	1	1	Critical	0	0	Port	Security
1258	VPort Q_Key Violation	1	1	Critical	0	0	Port	Security
1300	SA Key violation	1	1	Critical	5	0	Port	Security
1301	SGID Spoofed	1	1	Critical	5	0	Port	Security
1302	SA High Rate detected	1	1	Warning	5	0	Port	Security
1303	Multicast subscriptions over limit	1	1	Critical	5	0	Port	Security
1304	Service Record subscriptions over limit	1	1	Critical	5	0	Port	Security

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
1305	Event subscriptions over limit	1	1	Warning	5	0	Port	Security
1306	Unallowed SM was detected in the fabric	1	1	Critical	0	0	Port	Fabric Notificati
1307	SMInfo SET request was received from unallowed SM	1	1	Critical	0	0	Port	Fabric Notificati
1309	SM was detected with non-matching SMKey	1	1	Critical	0	0	Port	Fabric Notificati
1310	Duplicated node GUID was detected	1	1	Critical	1	0	Device	Fabric Notificati
1311	Duplicated port GUID was detected	1	1	Critical	1	0	Port	Fabric Notificati
1312	Switch was Rebooted	1	1	Info	1	0	Device	Fabric Notificati
1315	Topo Config File Error	1	1	Critical	1	0	Grid	Fabric Notificati
1316	Topo Config Subnet Mismatch	1	1	Critical	1	0	Grid	Fabric Notificati
1321	Link Flaps	1	0	Warning	0	10	Site	Fabric Top
1322	IB Mad Keys Periodic Update	1	0	Info	0	0	Site	Fabric Notificati
1400	High Ambient Temperature	1	1	Warning	0	0	Switch	Hardware
1401	High Fluid Temperature	1	1	Warning	0	0	Switch	Hardware
1402	Low Fluid Level	1	1	Warning	0	0	Switch	Hardware

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
1403	Low Supply Pressure	1	1	Warning	0	0	Switch	Hardware
1404	High Supply Pressure	1	1	Warning	0	0	Switch	Hardware
1405	Low Return Pressure	1	1	Warning	0	0	Switch	Hardware
1406	High Return Pressure	1	1	Warning	0	0	Switch	Hardware
1407	High Differential Pressure	1	1	Warning	0	0	Switch	Hardware
1408	Low Differential Pressure	1	1	Warning	0	0	Switch	Hardware
1409	System Fail Safe	1	1	Warning	0	0	Switch	Hardware
1410	Fault Critical	1	1	Critical	0	0	Switch	Hardware
1411	Fault Pump1	1	1	Critical	0	0	Switch	Hardware
1412	Fault Pump2	1	1	Critical	0	0	Switch	Hardware
1413	Fault Fluid Level Critical	1	1	Critical	0	0	Switch	Hardware
1414	Fault Fluid Over Temperature	1	1	Critical	0	0	Switch	Hardware
1415	Fault Primary DC	1	1	Critical	0	0	Switch	Hardware
1416	Fault Redundant DC	1	1	Critical	0	0	Switch	Hardware
1417	Fault Fluid Leak	1	1	Critical	0	0	Switch	Hardware
1418	Fault Sensor Failure	1	1	Critical	0	0	Switch	Hardware
1419	Cooling Device Monitoring Error	1	0	Critical	0	0	Grid	Hardware

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
1420	Cooling Device Communication Error	1	1	Critical	0	0	Switch	Hardware
1500	New Cable Detected	1	0	Info	1	0	Site	Security
1502	Cable detected in a new location	1	0	Warning	1	0	Site	Security
1503	Duplicate Cable Detected	1	0	Critical	1	0	Site	Security
1504	SHARP Allocation Succeeded	0	0	Info	1	0	Grid	SHARP
1505	SHARP Allocation Failed	0	0	Warning	1	0	Grid	SHARP
1506	SHARP Deallocation Succeeded	0	0	Info	1	0	Grid	SHARP
1507	SHARP Deallocation Failed	0	0	Warning	1	0	Grid	SHARP
1508	Device Collect System Dump Started	1	0	Info	1	0	Device	Maintena
1509	Device Collect System Dump Finished	1	0	Info	1	0	Device	Maintena
1510	Device Collect System Dump Error	1	1	Critical	1	0	Device	Maintena
1511	Virtual Port Added	1	0	Info	1	0	Port	Fabric Notificati
1512	Virtual Port Removed	0	0	Warning	1	0	Port	Fabric Notificati

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
1513	Burn Cables Transceivers Started	0	0	Info	1	0	Device	Maintena
1514	Burn Cables Transceivers Finished	0	0	Info	1	0	Device	Maintena
1515	Burn Cables Transceivers Failed	0	1	Warning	1	0	Device	Maintena
1516	Activate Cables Transceivers FW Finished	0	0	Info	1	0	Device	Maintena
1517	Activate Cables Transceivers FW Failed	0	0	Warning	1	0	Device	Maintena
1520	Aggregation Node Discovery Failed	0	0	Critical	1	0	SHARP AM	SHARP
1521	Job Started	0	0	Info	1	0	SHARP AM	SHARP
1522	Job Ended	0	0	Info	1	0	SHARP AM	SHARP
1523	Job Start Failed	0	0	Critical	1	0	SHARP AM	SHARP
1524	Job Error	0	0	Critical	1	0	SHARP AM	SHARP
1525	Trap QP Error	0	0	Critical	1	0	SHARP AM	SHARP
1526	Trap Invalid Request	0	0	Critical	1	0	SHARP AM	SHARP
1527	Trap Sharp Error	0	0	Critical	1	0	SHARP AM	SHARP
1528	Trap QP Alloc timeout	0	0	Critical	1	0	SHARP AM	SHARP

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
1529	Trap AMKey Violation	0	0	Critical	1	0	SHARP AM	SHARP
1530	Unsupported Trap	0	0	Critical	1	0	SHARP AM	SHARP
1531	Reservation Updated	0	0	Info	1	0	SHARP AM	SHARP
1532	Sharp is not Responding	0	1	Critical	1	0	SHARP AM	SHARP
1533	Agg Node Active	0	0	Info	1	0	SHARP AM	SHARP
1534	Agg Node Inactive	0	0	Warning	1	0	SHARP AM	SHARP
1535	Trap AMKey Violation Triggered by AM	0	0	Warning	1	0	SHARP AM	SHARP
1550	Guids Were Added to Pkey	0	0	Info	1	0	Port	Fabric Configura
1551	Guids Were Removed from Pkey	0	0	Info	1	0	Port	Fabric Configura
1552	Missing HCAs Assignments to PKey	1	1	Warning	0	0	Port	Security
1553	GUID to Pkey with Index0	1	1	Warning	0	0	Port	Security
1554	XDR flag disabled	1	0	Warning	0	0	Site	Fabric Notificati
1600	VS/CC Classes Key Violation	0	0	Warning	1	0	Port	Security
1601	Tenant Access Violation	0	0	Warning	1	0	Port	Security

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
1602	PCI Speed Degradation Warning	1	1	Critical	1	0	Port	Hardware
1603	PCI Width Degradation Warning	1	1	Critical	1	0	Port	Hardware
1604	Non-optimal aggregated port bandwidth	1	1	Warning	1	0	Port	Hardware
1605	Routing Engine Action Remove	1	1	Warning	1	0	Switch	Fabric Notificati
1606	Routing Engine Action Recover	1	1	Info	1	0	Switch	Fabric Notificati
1607	Topology compare Report completed successfully	1	0	Info	0	0	Site	Fabric Notificati
1608	Topology compare Report completed with errors	1	1	Warning	0	0	Site	Fabric Notificati
1609	In Service Switch Upgrade isolate action	1	1	Warning	1	0	Switch	Fabric Notificati
1610	In Service Switch Upgrade deisolate action	1	1	Info	1	0	Switch	Fabric Notificati
1611	Different SM configuration values across the fabric	1	0	Warning	0	0	Site	Fabric Notificati

Alarm ID	Alarm Name	To Log	Alarm	Default Severity	Default Threshold	Default TTL	Related Object	Category
1612	Database File is Corrupted	1	1	Critical	0	10	Grid	Maintena

Reports

UFM reports provide summarized information about selected topics. Through the different UFM WebUI tabs, you can create reports that run a series of checks on UFM components.

The table below summarizes the types of reports and provides useful links for more information.

Report Type	Description	WebUI	REST API
UFM Health Report	A report that run a series of checks related to UFM server health, including CPU, memory, license, configurations and disk monitoring.	UFM Health Tab	Reports REST API
Fabric Health Report	Through the Fabric Health tab, you can access the fabric health reports. There are two kinds of reports: • Custom Reports - The user can generate a report that runs a series of checks on the fabric on demand. • Periodic Reports - An automatically generated report that is periodically generated by the UFM.	Fabric Health Tab	Reports REST API Periodic Fabric Health REST API
Topology Comparison Report	Reports on topology comparison, available as: • Periodic Comparison - allows users to compare the current fabric topology with a preset master topology. • Custom Comparison - compares user-defined topology with the current fabric topology.	Topology Compare Tab	Topology Compare REST API
Daily Reports	The Daily Report feature collects, analyzes, and reports the most significant issues of the fabric in the last 24 hours	Daily Reports Tab	N/A

Telemetry

UFM Telemetry allows the collection and monitoring of InfiniBand fabric port statistics, such as network bandwidth, congestion, errors, latency, and more.

UFM Telemetry Capabilities

- Real-time monitoring views
- Monitoring of multiple attributes
- Intelligent Counters for error and congestion counters
- InfiniBand port-based error counters
- InfiniBand congestion XmitWait counter-based congestion measurement
- InfiniBand port-based bandwidth data

Telemetry Session Panels Supported Actions

- Rearrangement via a straightforward drag-and-drop function
- Resizing by hovering over the panel's border

Understanding Telemetry Types

UFM collects by default two types of telemetry data, each serving different monitoring purposes:

Primary Telemetry (High-Frequency)

- Default Sample Rate: 30 seconds
- Use Cases:
    - Real-time monitoring

    - UFM dashboard charts
    - Port threshold event detection
    - Live telemetry sessions
- Counters: Collects approximately 30 key performance counters covering bandwidth, congestion, and error metrics
- Historical Data: Collected every 5 minutes and stored in UFM's SQLite database

Secondary Telemetry (Low-Frequency)

- Default Sample Rate: 300 seconds (5 minutes)
- Use Cases:
    - Historical analysis
    - Detailed diagnostics
    - Extended monitoring scenarios
- Counters: Collects approximately 120 extended counters for comprehensive fabric analysis

Telemetry Management Methods

- Legacy Mode (via UFM): In this mode, telemetry instances are invoked during UFM startup and fully managed by UFM.
- Clustered Telemetry (UTM) Mode: In this mode, telemetry instances are managed by the UFM Telemetry Manager (UTM) plugin.

UFM Clustered Telemetry

Overview

UFM Clustered Telemetry is an advanced feature that enables distributed telemetry data collection across multiple network adapters (HCAs) in your InfiniBand fabric . This feature provides improved performance and scalability for large-scale deployments through workload distribution.

Key Benefits

- Better Performance: Workload distribution across multiple instances reduces collection bottlenecks
- HCA Utilization: Leverages multiple network adapters for parallel data collection
- Scalability: Handles larger fabric deployments more efficiently
- Flexibility: Customizable instance distribution based on your infrastructure

Prerequisites

- UFM Telemetry Manager (UTM) Plugin must be deployed and enabled

Switching from Legacy to Clustered Telemetry (UTM) Mode

Follow these steps to enable Clustered Telemetry using UTM mode:

Step 1: Start UFM

Ensure UFM is running on your system.

Step 2: Deploy UTM Plugin

1. Navigate to Settings > Plugin Management in the UFM WebUI
2. Locate the UFM Telemetry Manager (UTM) plugin
3. Click Enable to activate the plugin

Step 3: Configure Telemetry Mode

1. Edit the UFM configuration file:

   ```
   vi /opt/ufm/files/conf/gv.cfg
   ```

2. Locate the `[Telemetry]` section and set the following parameters:

   ```
   [Telemetry]<p></p>primary_telemetry_legacy_mode = false<p>
   </p>secondary_telemetry_legacy_mode = false
   ```

Step 4: Restart UFM

Restart the UFM service to apply changes:

```
/etc/init.d/ufmd restart
```

Alternatively, restart only the telemetry service:

```
/etc/init.d/ufmd ufm_telemetry_stop<p></p>/etc/init.d/ufmd
ufm_telemetry_start
```

Configuration Options

Automatic Configuration (Default)

When UFM starts in UTM mode, it automatically detects available HCAs and creates a default configuration. This is the recommended approach for most deployments.

Default Behavior:

- Detects all available HCAs on the system
- Creates 1 primary and one secondary telemetry instance on the first HCA

- Configuration is stored in: `/opt/ufm/files/conf/utm/{hostname}_instances_matrix.json`

Example Auto-Generated Matrix:

```
{
  "mlx5_0": { "primary": 1, "secondary": 1 },
  "mlx5_1": { "primary": 0, "secondary": 0 }
}
```

Custom Configuration

For advanced deployments, you can customize the distribution of telemetry instances across HCAs using the `generate_telemetry_config.sh` script.

Auto-Detect and Create Configuration

Automatically detect HCAs and create a configuration file:

```
/opt/ufm/scripts/generate_telemetry_config.sh --auto-detect
/opt/ufm/files/conf/utm/$(hostname)_instances_matrix.json
```

Manual Custom Configuration

Specify custom instance counts per HCA using the format `HCA_NAME:PRIMARY_COUNT:SECONDARY_COUNT`:

```
/opt/ufm/scripts/generate_telemetry_config.sh \
  /opt/ufm/files/conf/utm/$(hostname)_instances_matrix.json \
  mlx5_0:2:1 \
  mlx5_1:0:2 \
  mlx5_2:1:0
```

This creates:

- **mlx5_0**: 2 primary instances, 1 secondary instance
- **mlx5_1**: 0 primary instances, 2 secondary instances
- **mlx5_2**: 1 primary instance, 0 secondary instances

Example Custom Matrix:

```
{
  "mlx5_0": { "primary": 2, "secondary": 1 },
  "mlx5_1": { "primary": 0, "secondary": 2 },
  "mlx5_2": { "primary": 1, "secondary": 0 }
}
```

Validate Configuration

Verify your matrix file is correctly formatted:

```
/opt/ufm/scripts/generate_telemetry_config.sh --validate
/opt/ufm/files/conf/utm/$(hostname)_instances_matrix.json
```

Get Help

Display usage information and options:

```
/opt/ufm/scripts/generate_telemetry_config.sh --help
```

> **Note**
>
> **Important**: After modifying the matrix configuration file, you must restart UFM for changes to take effect.

Advanced Configuration Parameters

The following optional parameters in `gv.cfg` allow fine-tuning of telemetry behavior. Most users should use the default values.

Parameter	Section	Default	Description
`dashboard_interval`	`[Server]`	30	Sample rate (seconds) for primary telemetry instances
`secondary_sample_rate`	`[Telemetry]`	300	Sample rate (seconds) for secondary telemetry instances
`primary_telemetry_legacy_mode`	`[Telemetry]`	true	Set to `false` to enable UTM mode for primary telemetry

Parameter	Section	Default	Description
`secondary_telemetry_legacy_mode`	`[Telemetry]`	true	Set to `false` to enable UTM mode for secondary telemetry

> **Note**
>
> Note: Changing sample rates affects data frequency and may impact system performance. Consult with NVIDIA support before modifying these values in production environments.

Telemetry Instance	Description	REST API
High-Frequency (Primary) Telemetry Instance	A default telemetry session that collects a predefined set of ~30 counters covering bandwidth, congestion, and error metrics, which UFM analyzes and reports. These counters are used for: • Default Telemetry Session - An ongoing session used by the UFM to display UFM WebUI dashboard charts information and for monitoring and analyzing ports threshold events (the session interval is 30 secs by default) • Real-Time Telemetry - allows users to define live telemetry sessions for monitoring small subsets of devices or ports and a selected set of counters. For more information, refer to Telemetry. • Historical Telemetry - based on the primary telemetry and collects statistical data from all fabric ports and stores them in an internal UFM SQLite database (the session interval is 5 mins by default)	For Default and Real-time Telemetry: Monitoring REST API For Historical Telemetry: History Telemetry Sessions REST API → History Telemetry Sessions

Telemetry Instance	Description	REST API
Low-Frequency (Secondary) Telemetry Instance	Operates automatically upon UFM startup, offering an extended scope of 120 counters. For a list of the Secondary Telemetry Fields, refer to Low-Frequency (Secondary) Telemetry Fields.	N/A

For direct telemetry endpoint access, which exposes the list of supported counters:

For the **High-Frequency (Primary) Telemetry Instance**, run the following command:

```
#curl -s 127.0.0.1:9001/csv/cset/converted_enterprise
```

For the **Low-Frequency (Secondary) Telemetry Instance**, run the following command:

```
#curl -s 127.0.0.1:9002/csv/xcset/low_freq_debug
```

Historical Telemetry Collection in UFM

Storage Considerations

UFM periodically collects fabric port statistics and saves them in its SQLite database. Before starting up UFM Enterprise, please consider the following disk space utilization for various fabric sizes and duration.

The measurements in the table below were taken with sampling interval set to once per 30 seconds.

Note

Be aware that the default sampling rate is once per 300 seconds. Disk utilization calculation should be adjusted accordingly.

Number of Nodes	Ports per Node	Storage per Hour	Storage per 15 Days	Storage per 30 Days
16	8	1.6 MB	576 MB (0.563 GB)	1152 MB (1.125 GB)
100	8	11 MB	3960 MB (3.867 GB)	7920 MB (7.734 GB)
500	8	50 MB	18000 MB (17.58 GB)	36000 MB (35.16 GB)
1000	8	100 MB	36000 MB (35.16 GB)	72000 MB (70.31 GB)

High-Frequency (Primary) Telemetry Fields

The following is a list of available counters which includes a variety of metrics related to timestamps, port and node information, error statistics, firmware versions, temperatures, cable details, power levels, and various other telemetry-related data. These metrics are collected every 30 seconds by default.

Field Name	Description
timestamp	timestamp
source_id	Node GUID
tag	N/A
node_guid	Node GUID
port_guid	Port GUID

Field Name	Description
sys_image_guid	System GUID
port_num	Port Number
aport	Aggregated Port Number
port_type	Port Type: 1 - Legacy, 2 - Plane, 4 - Aggregated
PortXmitDataExtended	Transmitted data rate per egress port in bytes passing through the port during the sample period
PortRcvDataExtended	The received data on the ingress port in bytes during the sample period
PortXmitPktsExtended	Total number of packets transmitted on the port.
PortRcvPktsExtended	Total number of packets received on the port
SymbolErrorCounterExtended	This counter provides information on error bits that were not corrected by phy correction mechanisms.
LinkErrorRecoveryCounterExtended	Total number of times the Port Training state machine has successfully completed the link error recovery process.
LinkDownedCounterExtended	Perf.PortCounters
PortRcvErrorsExtended	Total number of packets containing an error that were received on the port
PortRcvRemotePhysicalErrorsExtended	Total number of packets marked with the EBP delimiter received on the port.
PortRcvSwitchRelayErrorsExtended	Total number of packets received on the port that were discarded because they could not be forwarded by the switch relay.
PortXmitDiscardsExtended	Total number of outbound packets discarded by the port because the port is down or congested.
PortXmitConstraintErrorsExtended	Total number of packets not transmitted from the switch physical port.
PortRcvConstraintErrorsExtended	Total number of packets received on the switch physical port that are discarded.

Field Name	Description
LocalLinkIntegrityErrorsExtended	The number of times that the count of local physical errors exceeded the threshold specified by LocalPhyErrors
ExcessiveBufferOverrunErrorsExtended	The number of times that OverrunErrors consecutive flow control update periods occurred, each having at least one overrun error
VL15DroppedExtended	Number of incoming VL15 packets dropped due to resource limitations (e.g., lack of buffers) in the port
PortXmitWaitExtended	The time an egress port had data to send but could not send it due to lack of credits or arbitration - in time ticks within the sample-time window
hist[0-4]	Hist[i] give the number of FEC blocks that had RS-FEC symbols errors of value i or range of errors
infiniband_CBW	
Normalized_CBW	
NormalizedXW	
Normalized_XmitData	

Low-Frequency (Secondary) Telemetry Fields

The following is a list of available counters which includes a variety of metrics related to timestamps, port and node information, error statistics, firmware versions, temperatures, cable details, power levels, and various other telemetry-related data. These metrics are collected every 300 seconds by default.

Field Name	Description
Node_GUID	node GUID
Device_ID	PCI device ID

Field Name	Description
node_description	node description
lid	lid
Port_Number	port number
port_label	port label
Phy_Manager_State	FW Phy Manager FSM state
phy_state	physical state
logical_state	Port Logical link state
Link_speed_active	ib link active speed
Link_width_active	ib link active widthsource_id
Active_FEC	Active FEC
Total_Raw_BER	Pre-FEC monitor parameters
Effective_BER	Post FEC monitor parameters
Symbol_BER	BER after all phy correction mechanism: post FEC + PLR monitor parameters
Raw_Errors_Lane_[0-3]	This counter provides information on error bits that were identified on lane X. When FEC is enabled this induction corresponds to corrected errors. In PRBS test mode, indicates the number of PRBS errors on lane X.
Effective_Errors	This counter provides information on error bits that were not corrected by FEC correction algorithm or that FEC is not active.
Symbol_Errors	This counter provides information on error bits that were not corrected by phy correction mechanisms.
Time_since_last_clear_Min	The time passed since the last counters clear event in msec. (physical layer statistical counters)
hist[0-15]	Hist[i] give the number of FEC blocks that had RS-FEC symbols errors of value i or range of errors
FW_Version	Node FW version
Chip_Temp	switch temperature
Link_Down	Perf.PortCounters(LinkDownedCounter)

Field Name	Description
Link_Down_IB	Total number of times the Port Training state machine has failed the link error recovery process and downed the link.
LinkErrorRecoveryCounter	Total number of times the Port Training state machine has successfully completed the link error recovery process.
PlrRcvCodes	Number of received PLR codewords
PlrRcvCodeErr	The total number of rejected codewords received
PlrRcvUncorrectableCode	The number of uncorrectable codewords received
PlrXmitCodes	Number of transmitted PLR codewords
PlrXmitRetryCodes	The total number of codewords retransmitted
PlrXmitRetryEvents	The total number of retransmitted event
PlrSyncEvents	The number of sync events
HiRetransmissionRate	Recieved bandwidth loss due to codes retransmission
PlrXmitRetryCodesWithinTSecMax	The maximum number of retransmitted events in t sec window
link_partner_description	node description of the link partner
link_partner_node_guid	node_guid of the link partner
link_partner_lid	lid of the link partner
link_partner_port_num	port number of the link partner
link_partner_host_name	host name of the link partner
Cable_PN	Vendor Part Number
Cable_SN	Vendor Serial Number
cable_technology	
cable_type	Cable/module type
cable_vendor	
cable_length	
cable_identifier	
vendor_rev	Vendor revision

Field Name	Description
cable_fw_version	
rx_power_lane_[0-7]	RX measured power
tx_power_lane_[0-7]	TX measured power
Module_Voltage	Internally measured supply voltage
Module_Temperature	Module temperature
fast_link_up_status	Indicates if fast link-up was performed in the link
time_to_link_up_ext_msec	Time in msec to link up from disable until phy up state. While the phy manager did not reach phy up state the timer will return 0.
Advanced_Status_Opcode	Status opcode: PHY FW indication
Status_Message	ASCII code message
down_blame	Which receiver caused last link down
local_reason_opcode	Opcde of link down reason - local
remote_reason_opcode	Opcde of link down reason - remote
e2e_reason_opcode	see local_reason_opcode for local reason opcode for remote reason opcode: local_reason_opcode+100
PortRcvRemotePhysicalErrors	Total number of packets marked with the EBP delimiter received on the port.
PortRcvErrors	Total number of packets containing an error that were received on the port
PortXmitDiscards	Total number of outbound packets discarded by the port because the port is down or congested.
PortRcvSwitchRelayErrors	Total number of packets received on the port that were discarded because they could not be forwarded by the switch relay.
ExcessiveBufferOverrunErrors	The number of times that OverrunErrors consecutive flow control update periods occurred, each having at least one overrun error
LocalLinkIntegrityErrors	The number of times that the count of local physical errors exceeded the threshold specified by LocalPhyErrors

Field Name	Description
PortRcvConstraintErrors	Total number of packets received on the switch physical port that are discarded.
PortXmitConstraintErrors	Total number of packets not transmitted from the switch physical port.
VL15Dropped	Number of incoming VL15 packets dropped due to resource limitations (e.g., lack of buffers) in the port
PortXmitWait	The time an egress port had data to send but could not send it due to lack of credits or arbitration - in time ticks within the sample-time window
PortXmitDataExtended	Transmitted data rate per egress port in bytes passing through the port during the sample period
PortRcvDataExtended	The received data on the ingress port in bytes during the sample period
PortXmitPktsExtended	Total number of packets transmitted on the port.
PortRcvPktsExtended	Total number of packets received on the port
PortUniCastXmitPkts	Total number of unicast packets transmitted on all VLs from the port. This may include unicast packets with errors, and excludes link packets
PortUniCastRcvPkts	Total number of unicast packets, including unicast packets containing errors, and excluding link packets, received from all VLs on the port.
PortMultiCastXmitPkts	Total number of multicast packets transmitted on all VLs from the port. This may include multicast packets with errors.
PortMultiCastRcvPkts	Total number of multicast packets, including multicast packets containing errors received from all VLs on the port.
SyncHeaderErrorCounter	Count of errored block sync header on one or more lanes
PortSwLifetimeLimitDiscards	Total number of outbound packets discarded by the port because the Switch Lifetime Limit was exceeded. Applies to switches only.

Field Name	Description
PortSwHOQLifetimeLimitDiscards	Total number of outbound packets discarded by the port because the switch HOQ Lifetime Limit was exceeded. Applies to switches only.
rq_num_wrfe	Responder - number of WR flushed errors
rq_num_lle	Responder - number of local length errors
sq_num_wrfe	Requester - number of WR flushed errors
Temp_flags	Latched temperature flags of module
Vcc_flags	Latched VCC flags of module
device_hw_rev	Node HW Revision
sw_revision	Switch revision
sw_serial_number	Switch serial number
measured_freq_[0-1]	Clock frequency measurement in last 100msec
min_freq_[0-1]	Minutes of clock frequency measured. Units of 0.1 KHz
max_freq_[0-1]	Max of clock frequency measured. Units of 0.1 KHz
max_delta_freq_[0-1]	Observed max delta frequency in window of 100msec. Units of 0.1 KHz
snr_media_lane_[0-7]	SNR value on the media lane <i>. In unit scale of 1/256 dB. The SNR value represents the electrical signal-to-noise ratio on an optical lane, and is defined as the minimum of the three individual eye SNR values.
snr_host_lane_[0-7]	SNR value on the host lane <i>. In unit scale of 1/256 dB. The SNR value represents the electrical signal-to-noise ratio on an optical lane, and is defined as the minimum of the three individual eye SNR values.
tx_cdr_lol	Bitmask for latched Tx cdr loss of lock flag per lane.
rx_cdr_lol	Bitmask for latched Rx cdr loss of lock flag per lane.
tx_los	Bitmask for latched Tx loss of signal flag per lane.
rx_los	Bitmask for latched Rx loss of signal flag per lane.

Field Name	Description
phy_received_bits	This counter provides information on the total amount of traffic (bits) received.
rq_general_error	The total number of packets that were dropped since it contained errors. Reasons for this include: Dropped due to MPR mismatch.
host_name	Port's host name.

UFM Web UI

This section is constituted by the following sub-sections:

- Fabric Dashboard
- Network Map
- Managed Elements
- Telemetry - User-Defined Sessions
- System Health
- Events & Alarms
- Jobs
- Settings

Access the WebUI

UFM Web UI Supported Browsers

UFM Web UI is supported on all the following web browsers: Internet Explorer, Firefox, Chrome and Opera.

For optimal UFM Web UI performance, make sure you are using the latest version available of Google Chrome.

Launching UFM Web UI Session

Before accessing the UFM Web UI:

- If required, you can change the configuration of the connection (port and protocol) between the UFM server and the APACHE server in the file *gv.cfg*:
    - *ws_protocol* = http or https

- Setting the parameter *ws_protocol* to *http* allows unsecured access
- Setting the parameter *ws_protocol* to *https* denies unsecured access.

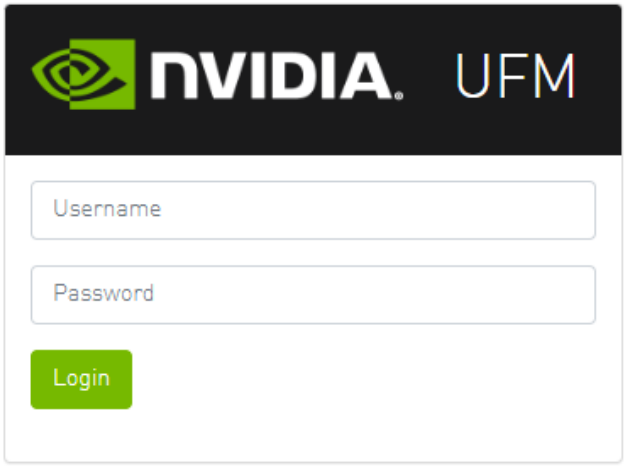
- *ws_port* = port number

To launch a UFM Web UI session, do the following:

1. Launch the Web UI by entering the following URL in your browser:

   http://<UFM_server_IP>/ufm

   https://<UFM_server_IP>/ufm



2. In the Login page, enter your **User Name** and your predefined user **Password** and click **Login**.

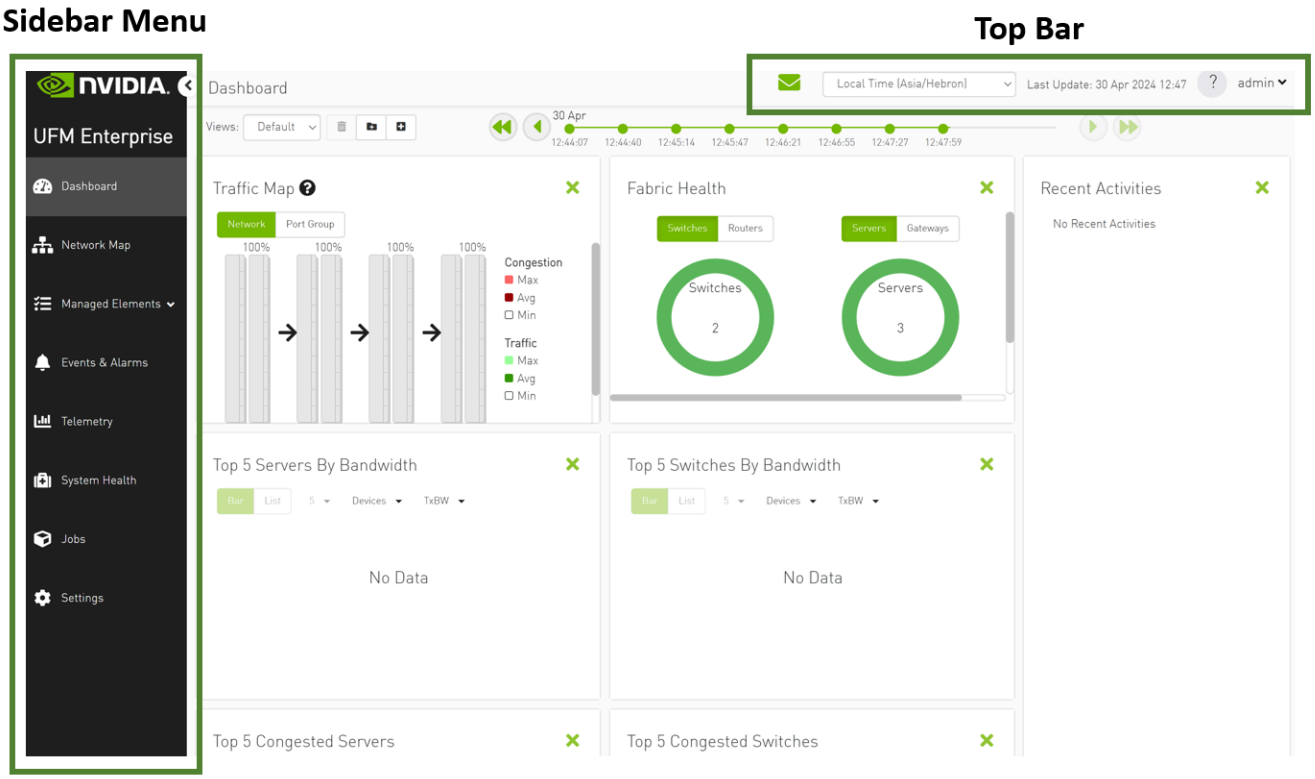
Once you have entered your user name and password, the main window shows the UFM Dashboard. For more information, see the Fabric Dashboard.

WebUI Layout

The UFM WebUI contains two main areas:

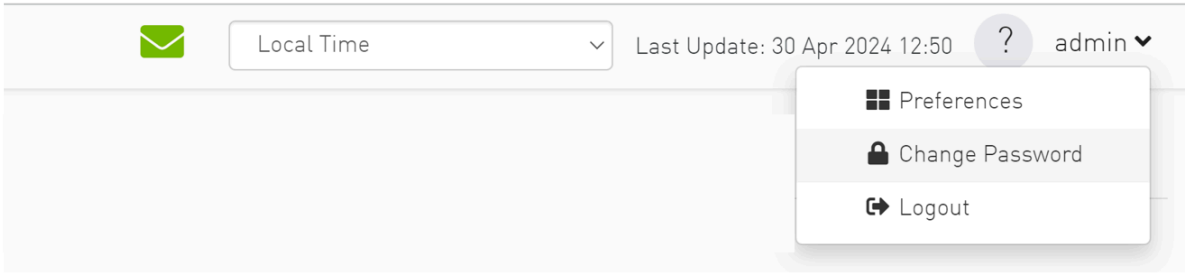
1. Top Bar - Contains local time zone and user information on the top right side of the screen.

2. Sidebar Menu - Contains a taskbar accessible from a sidebar menu on the left side of the screen. For more information on each tab, refer to UFM Web UI.



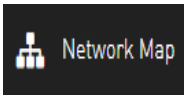
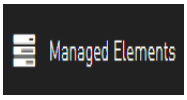
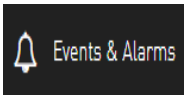
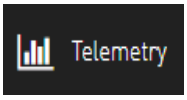
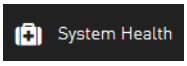
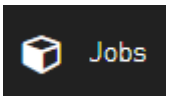
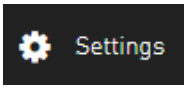
Top Bar

Each user can customize the UFM display, time zone, and date format, change their account password, and manage their preferences. For details, refer to Set User Preferences.



Sidebar Menu

Tab Icon	Description
Dashboard	Provides a summary view of the fabric status.

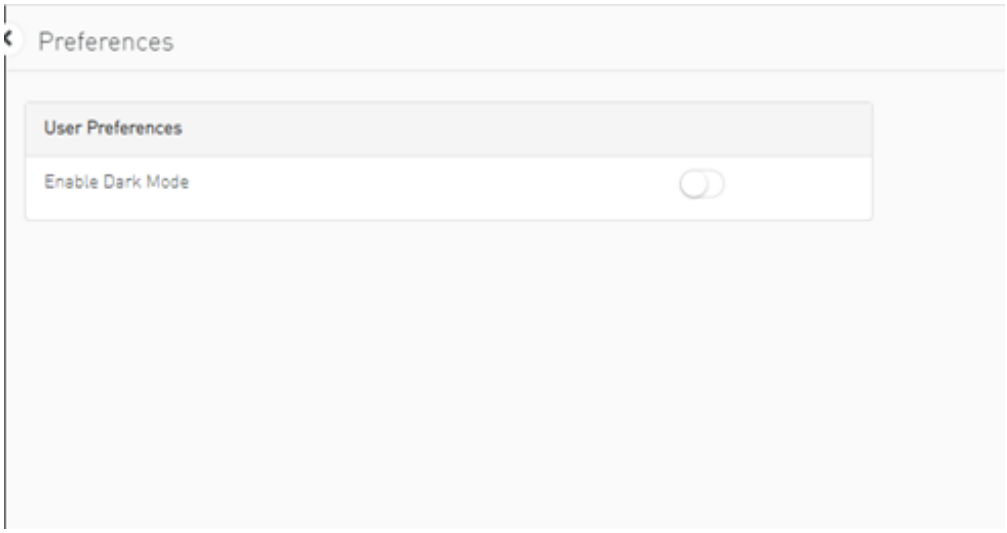
Tab Icon	Description
	Provides a hierarchical topology view of the fabric.
	Provides information on all fabric devices. This information is presented in a table format.
	Provides information on all logical servers. This information is presented in a table format.
	Provides information on the events & alarms generated by the system.
	Enables establishing monitoring sessions on devices or ports.
	Enables running and viewing fabric reports, UFM reports, and system logs. You can also back up UFM configuration files.
	Provides information on all jobs created, as a result of UFM actions.
	Enables configuring UFM server and UFM fabric settings, including events policy, device access, network management, subnet manager, and user management

Set User Preferences

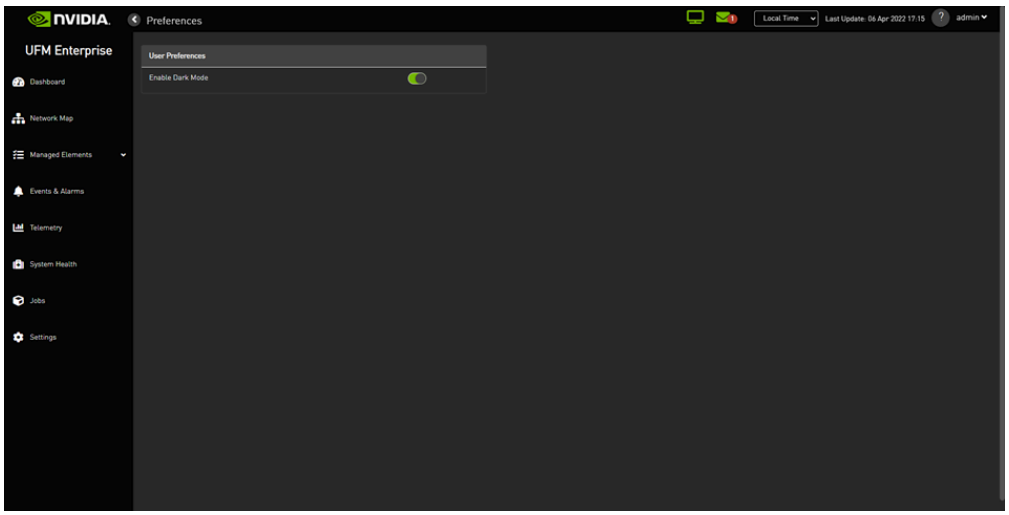
This section describes how to customize your UFM display settings and change your password,

Dark/Light Theme

1. Select **Preferences.**



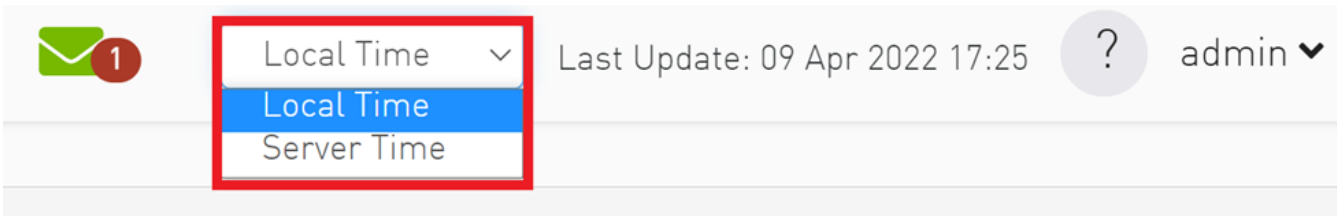
2. In **User Preferences**, enable dark mode for UFM presentation in a dark theme. The following figure shows the dark theme:

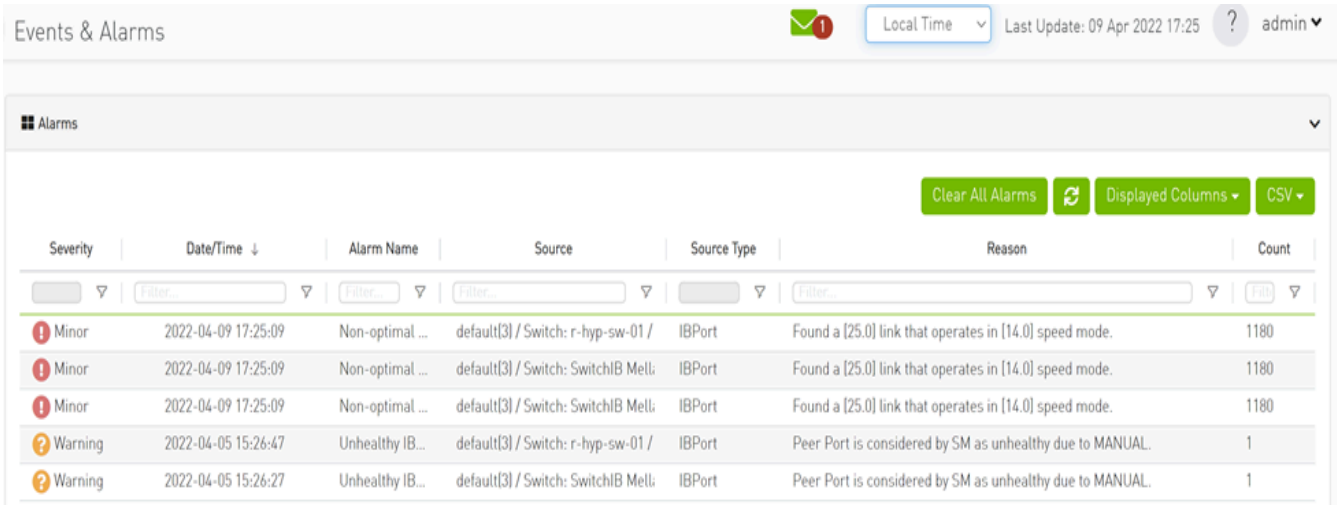


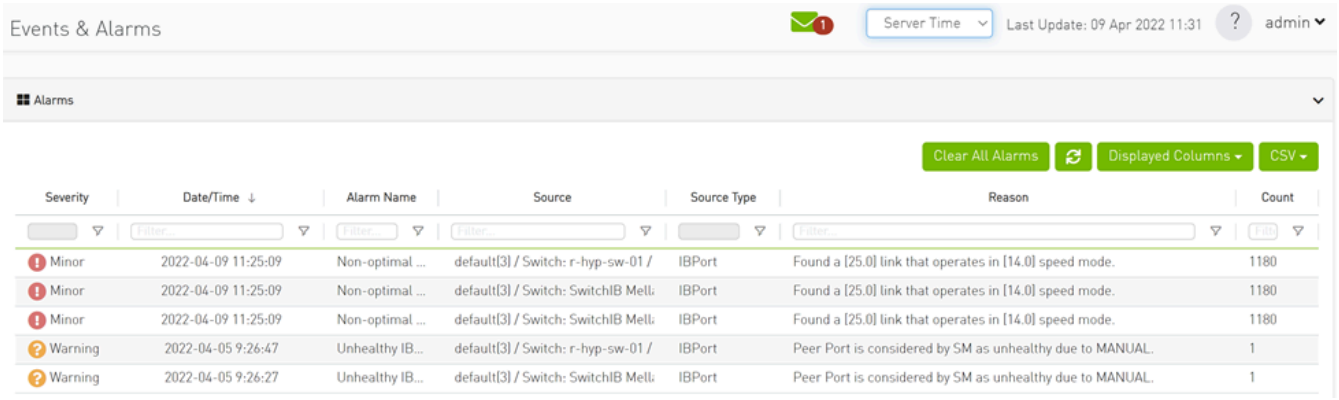
Time Zone Converter

Allows you to unify all times in UFM like events and alarms, ibdiagnet, telemetry and logs. You can switch between local and machine time.

In the status bar drop-down menu, switch between local and server/machine time.





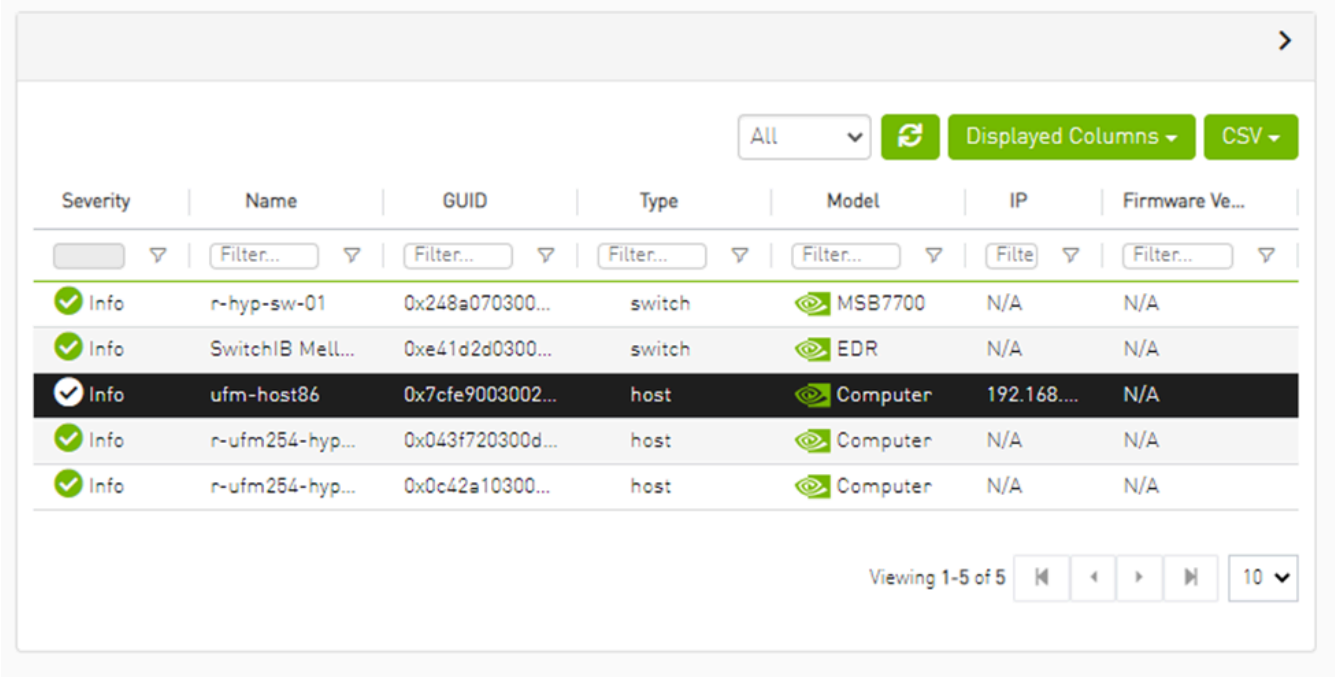


Note

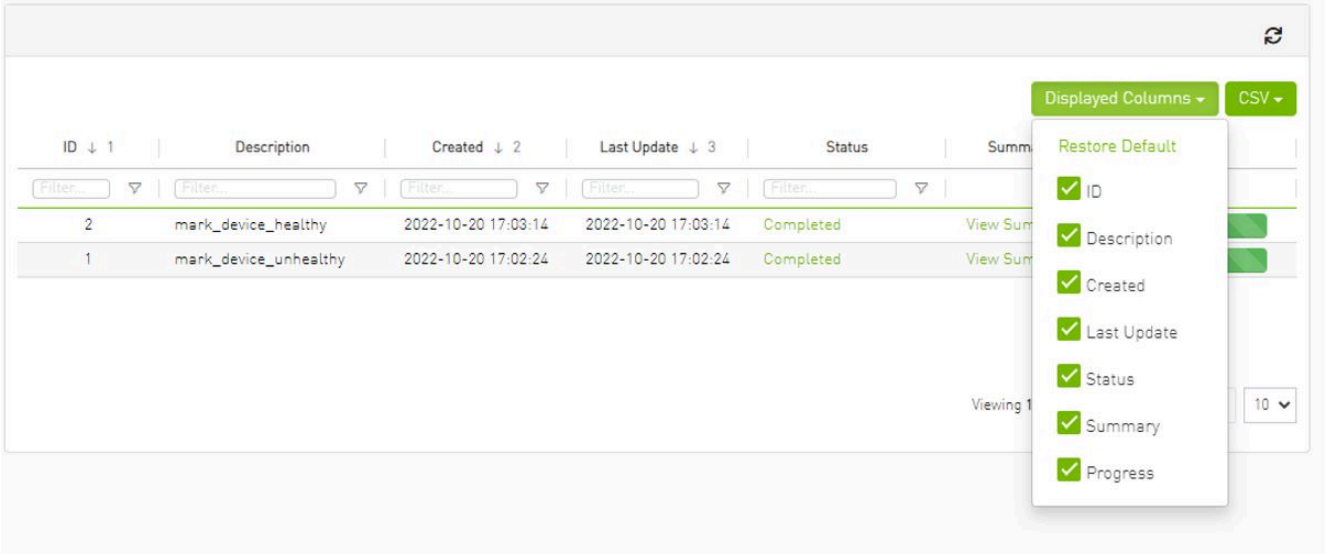
In the screenshots, the difference between Server Time and Local Time is 6 hours.

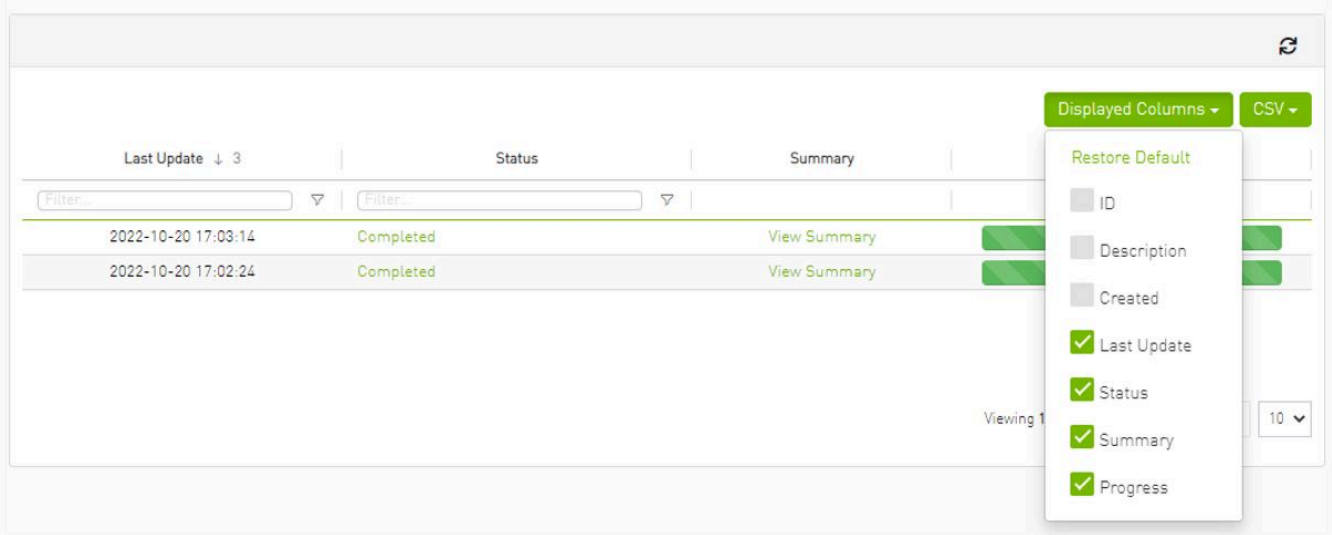
Table Enhancements

Look and Feel Improvements



Displayed Columns





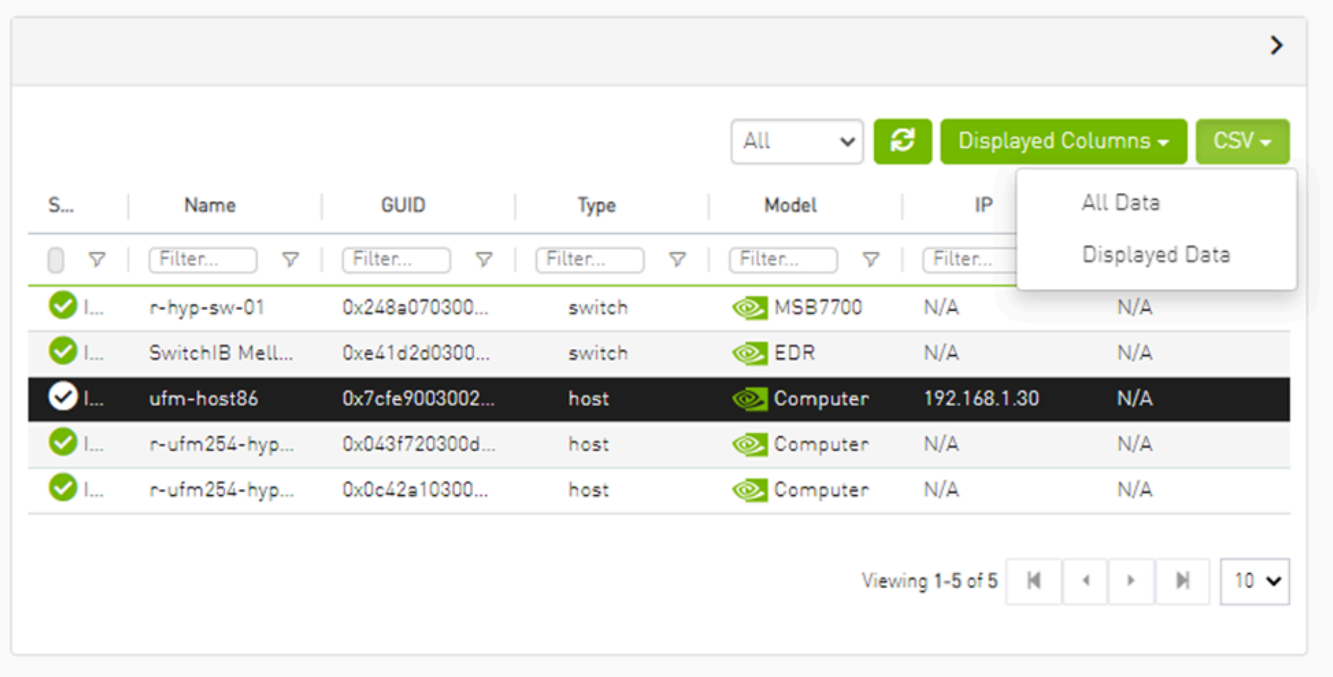
Note

Displayed columns of all tables are persistent per user, with the option to restore defaults.

Export All Data as CSV

There are two options for exporting as CSV

- All Data: all data returned from server.
- Displayed Data: only displayed rows.


All
Displayed Columns
CSV
All Data
Displayed Data
S...
Name
GUID
Type
Model
IP
Filter...
I...
r-hyp-sw-01
0x248a070300...
switch
MSB7700
N/A
N/A
I...
SwitchIB Mell...
0xe41d2d0300...
switch
EDR
N/A
N/A
I...
ufm-host86
0x7cfe9003002...
host
Computer
192.168.1.30
N/A
I...
r-ufm254-hyp...
0x043f720300d...
host
Computer
N/A
N/A
I...
r-ufm254-hyp...
0x0c42a10300...
host
Computer
N/A
N/A
Viewing 1-5 of 5
10

Multi-Subnet UFM

Overview

The Multi-Subnet UFM feature allows for the management of large fabrics, consisting of multiple sites, within a single product, namely Multi-Subnet UFM.

This feature is comprised of two layers: UFM Multi-Subnet Provider and UFM Multi-Subnet Consumer.

The UFM Provider functions as a Multi-Subnet Provider, exposing all local InfiniBand fabric information to the UFM consumer. On the other hand, the UFM Consumer acts as a Multi-Subnet Consumer, collecting and aggregating data from currently configured UFM Providers, enabling users to manage multiple sites in one place. While UFM Consumer offers similar functionality to regular UFM, there are several behavioral differences related to aggregation.

Setting Up Multi-Subnet UFM

In `/opt/ufm/files/conf/gv.cfg`, fill in the section named [`Multisubnet`] for UFM Multi-Subnet Provider and Consumer.

To set up UFM as a MultI-Subnet Provider, perform the following:

- Set `multisubnet_enabled` to `true`
- Set `multisubnet_role` to `provider`
- Set `multisubnet_site_name` (optional, if not set, it will be randomly generated); e.g., *`provider_1`*
- Start UFM

To set up UFM as a Multi-Subnet Consumer, perform the following:

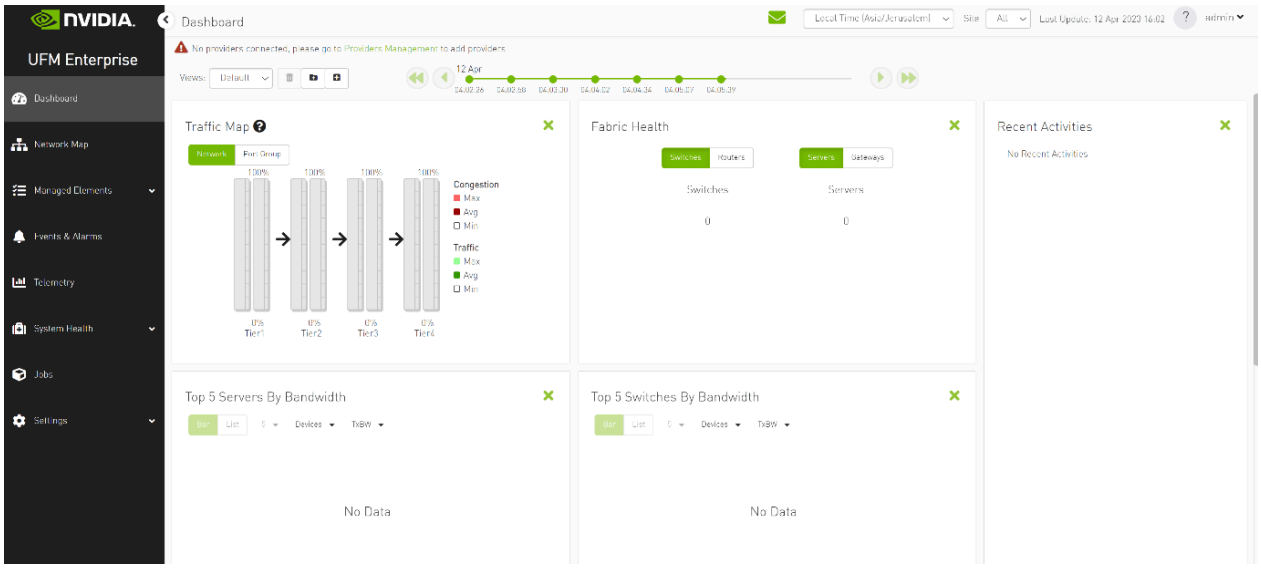
- Set `multisubnet_enabled` to `True`
- Set `multisubnet_role` to `consumer`

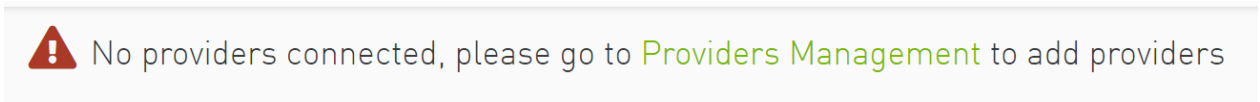
- Start UFM

It is important to note that UFM Multi-Subnet Consumer can be configured on a machine or VM without an established InfiniBand connectivity. Additionally, users may customize UFM Provider and Consumer using optional configuration parameters found in the [`Multisubnet`] section of `/opt/ufm/files/conf/gv.cfg`.

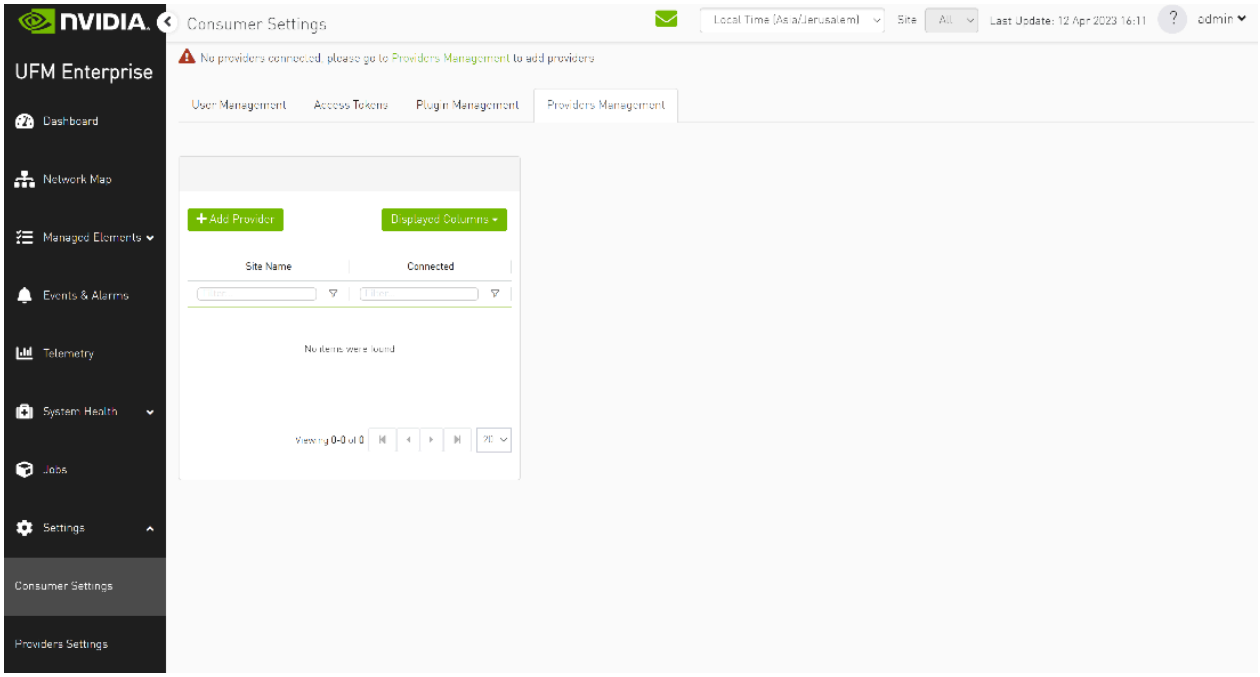
Functionality

1. Following the initial launch of the Consumer, the Dashboard view is devoid of data, and a message containing a hyperlink leading to the Provider Management section is displayed.

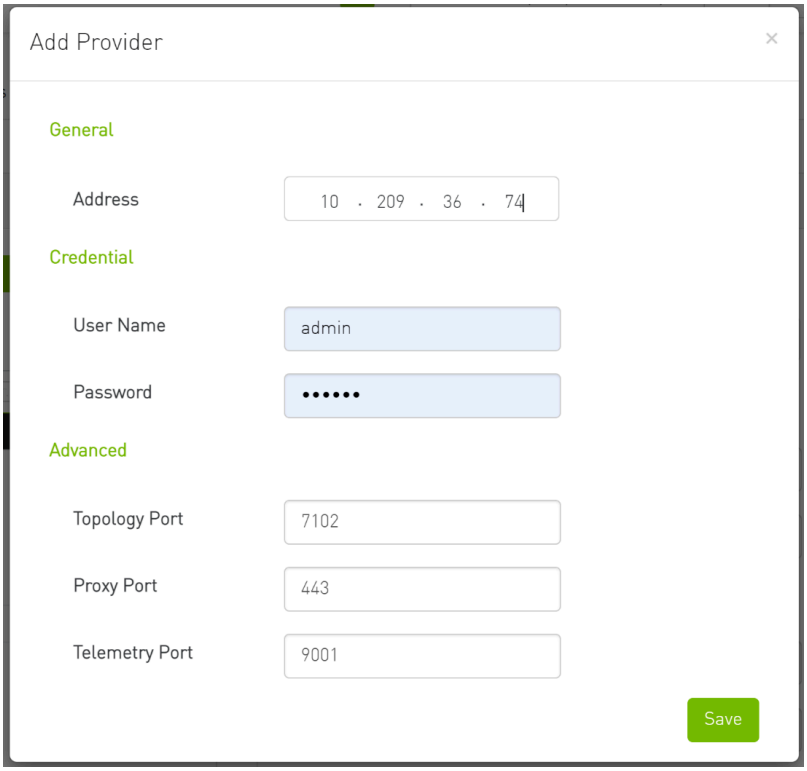




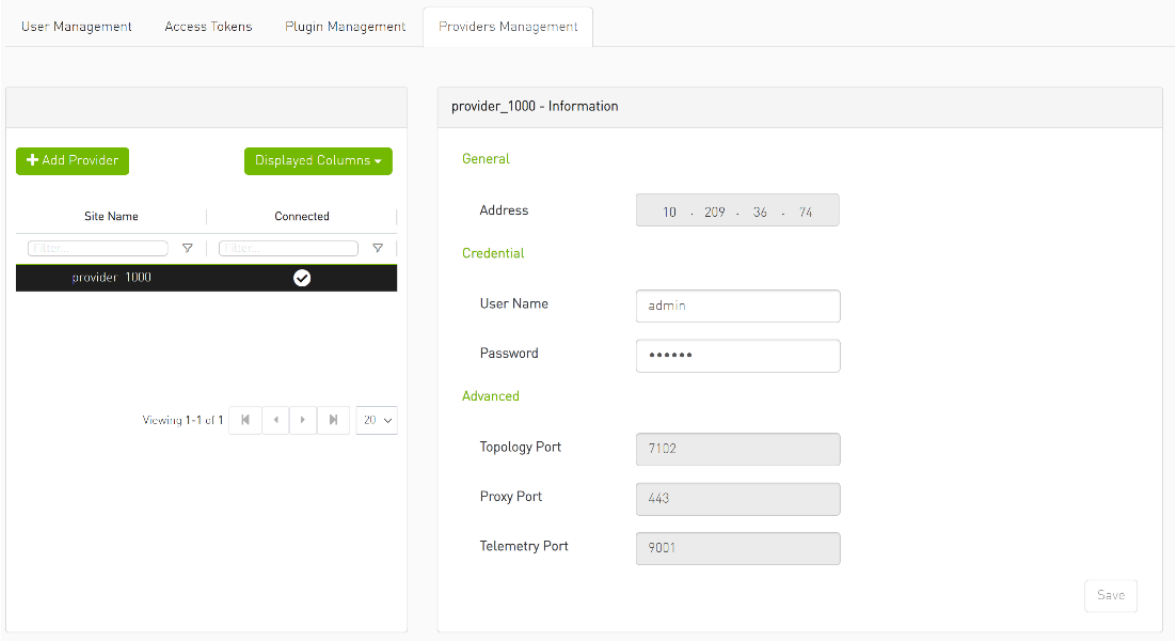
2. As shown in the below snapshot, a new section for Provider Management has been added, enabling users to configure UFM Providers.



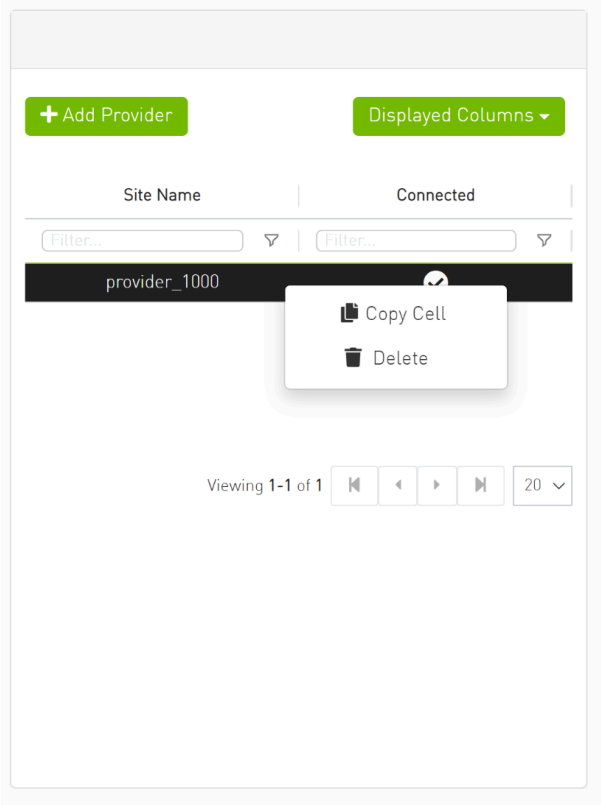
1. To add a provider, the user is required to enter its IP address and credentials. Unless there are multiple instances of UFM providers on a single machine, the advanced section parameters should be set with default values. However, if there are multiple instances, the advanced parameters may be set per Provider and then be configured in the Providers Management view.



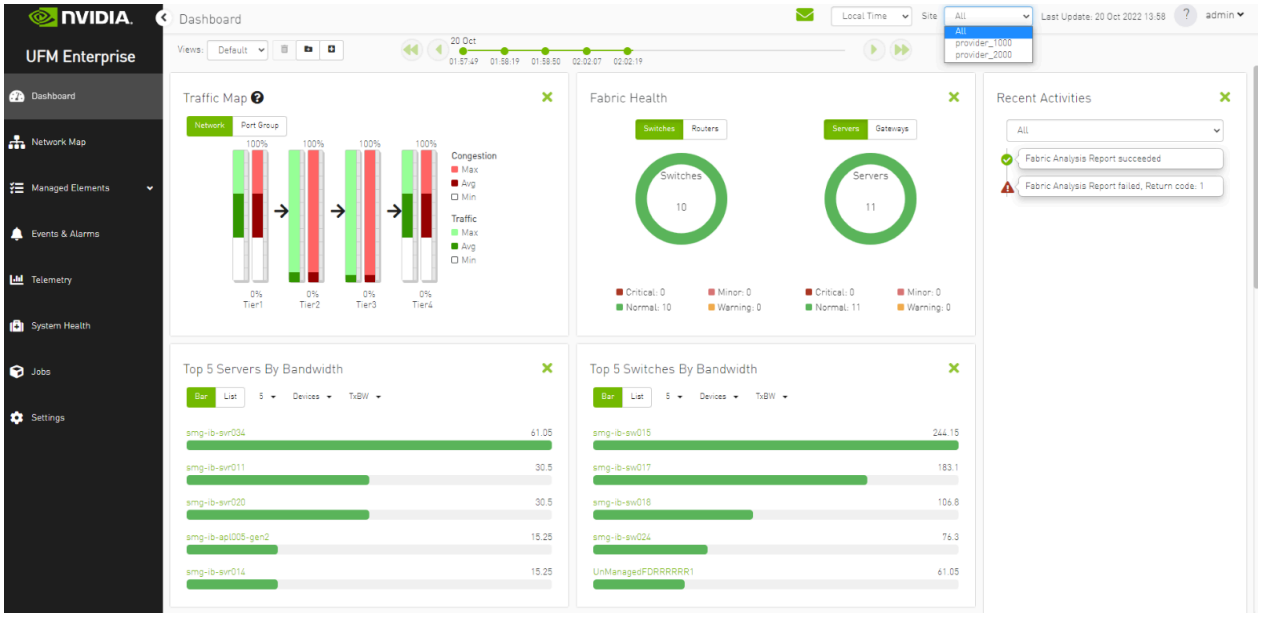
2. By editing the Provider view, you can change Provider's credentials.

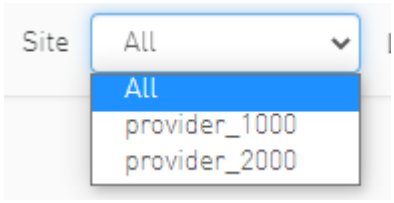


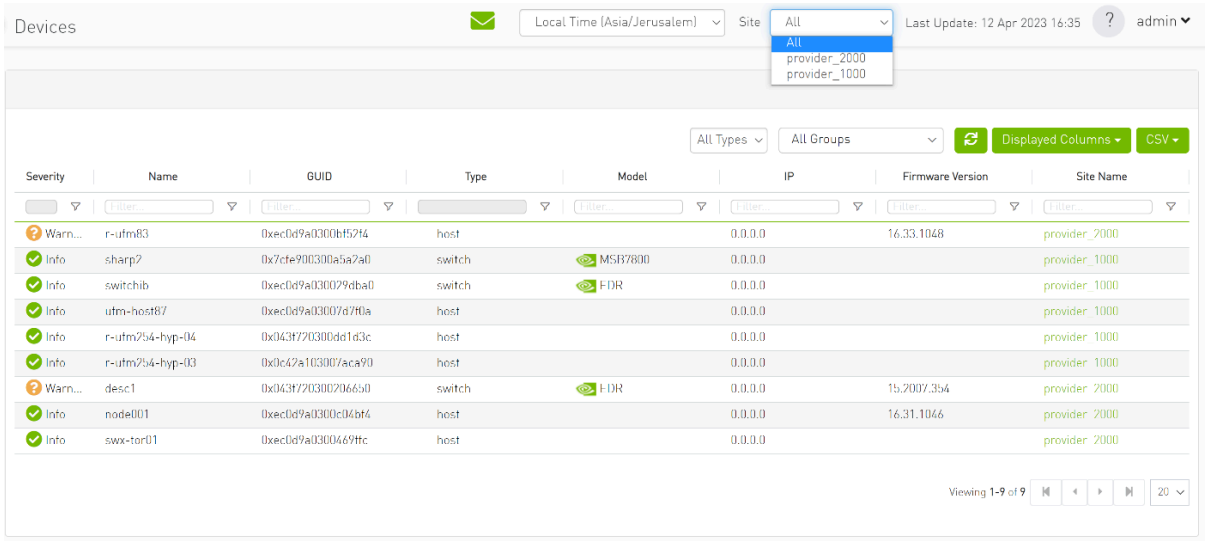
3. The "Delete Provider" function removes the selected Provider from the Consumer. Please note that this action may take some time to complete, and changes may only be reflected in the view after approximately 30 seconds.

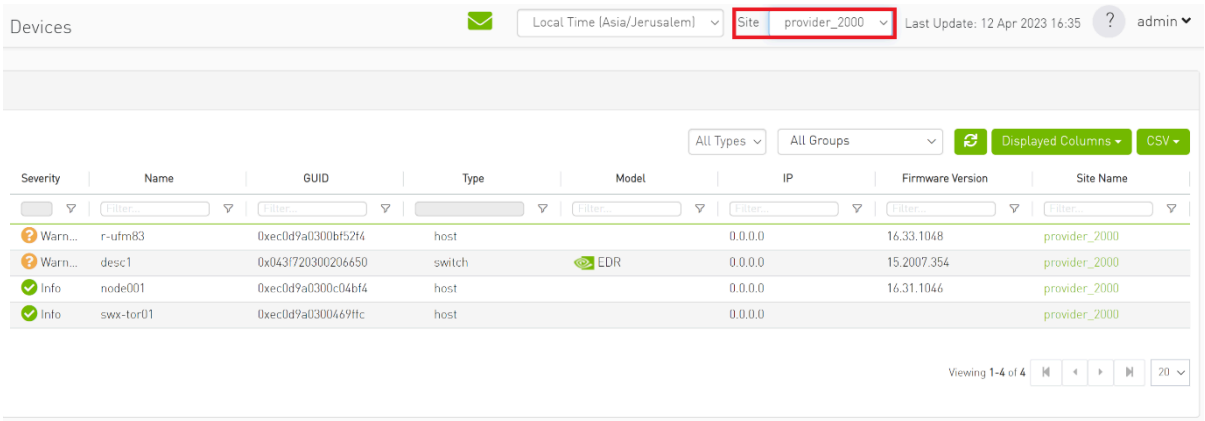


3. A general filter has been added to the top right corner of the page, enabling users to filter displayed data by site.

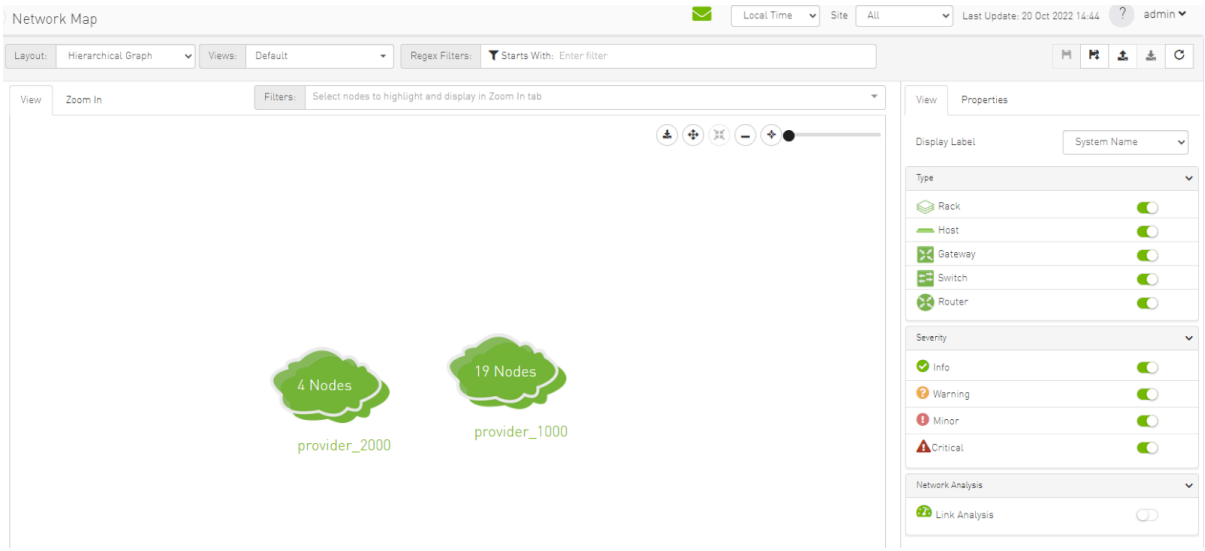

NVIDIA
UFM Enterprise
Dashboard
Network Map
Managed Elements
Events & Alarms
Telemetry
System Health
Jobs
Settings
Traffic Map
Fabric Health
Recent Activities
Top 5 Servers By Bandwidth
Top 5 Switches By Bandwidth

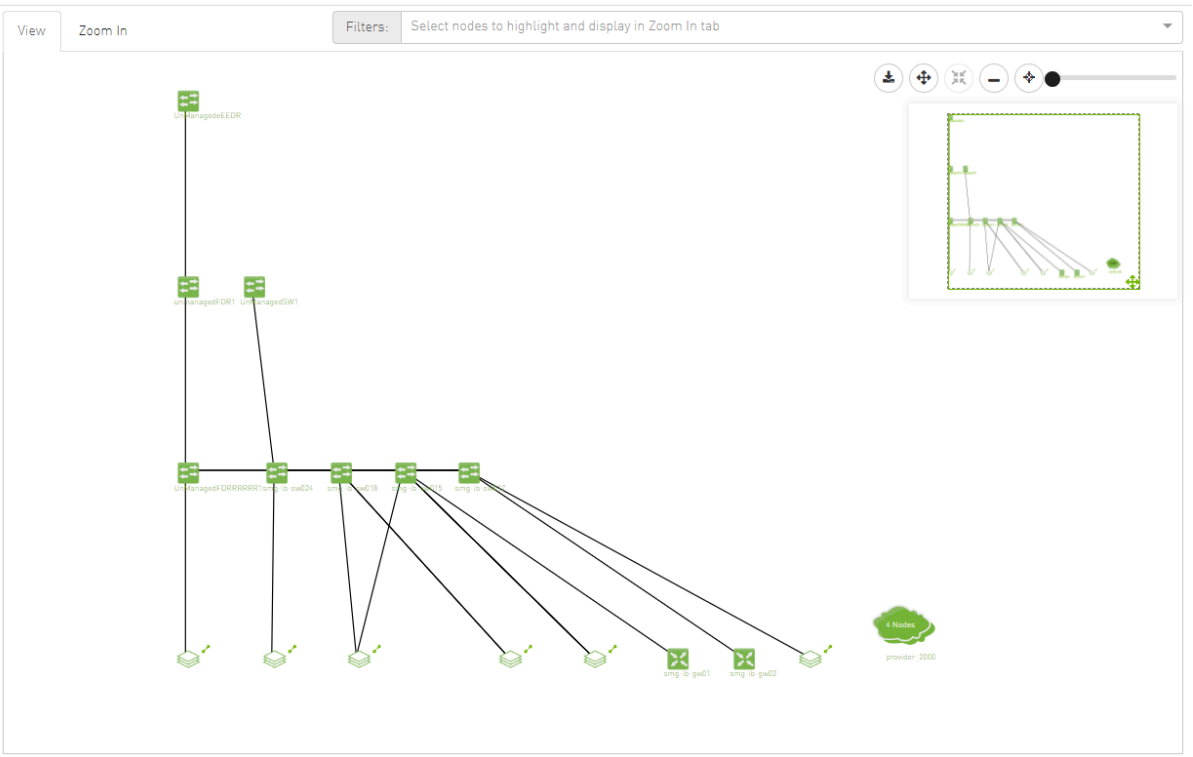

Site
All
All
provider_1000
provider_2000


Devices
Local Time (Asia/Jerusalem)
Site
All
Last Update: 12 Apr 2023 16:35
admin
All
provider_2000
provider_1000
All Types
All Groups
Displayed Columns
CSV
Severity
Name
GUID
Type
Model
IP
Firmware Version
Site Name
Warn... r-ufm83 0xec0d9a0300bf52f4 host 0.0.0.0 16.33.1048 provider_2000
Info sharp2 0x7cfe900300a5a2a0 switch MSB7800 0.0.0.0 provider_1000
Info switchib 0xec0d9a030029dba0 switch EDR 0.0.0.0 provider_1000
Info ufm-host87 0xec0d9a03007d7f0a host 0.0.0.0 provider_1000
Info r-utm254-hyp-04 0x043f720300dd1d3c host 0.0.0.0 provider_1000
Info r-utm254-hyp-03 0x0c42a103007aca90 host 0.0.0.0 provider_1000
Warn... desc1 0x043f720300206650 switch EDR 0.0.0.0 15.2007.354 provider_2000
Info node001 0xec0d9a0300c04bf4 host 0.0.0.0 16.31.1046 provider_2000
Info swx-tor01 0xec0d9a0300469ffc host 0.0.0.0 provider_2000
Viewing 1-9 of 9
20

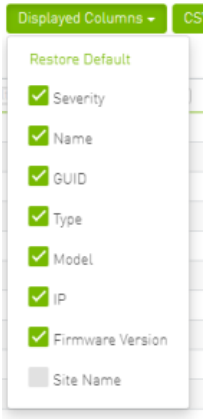


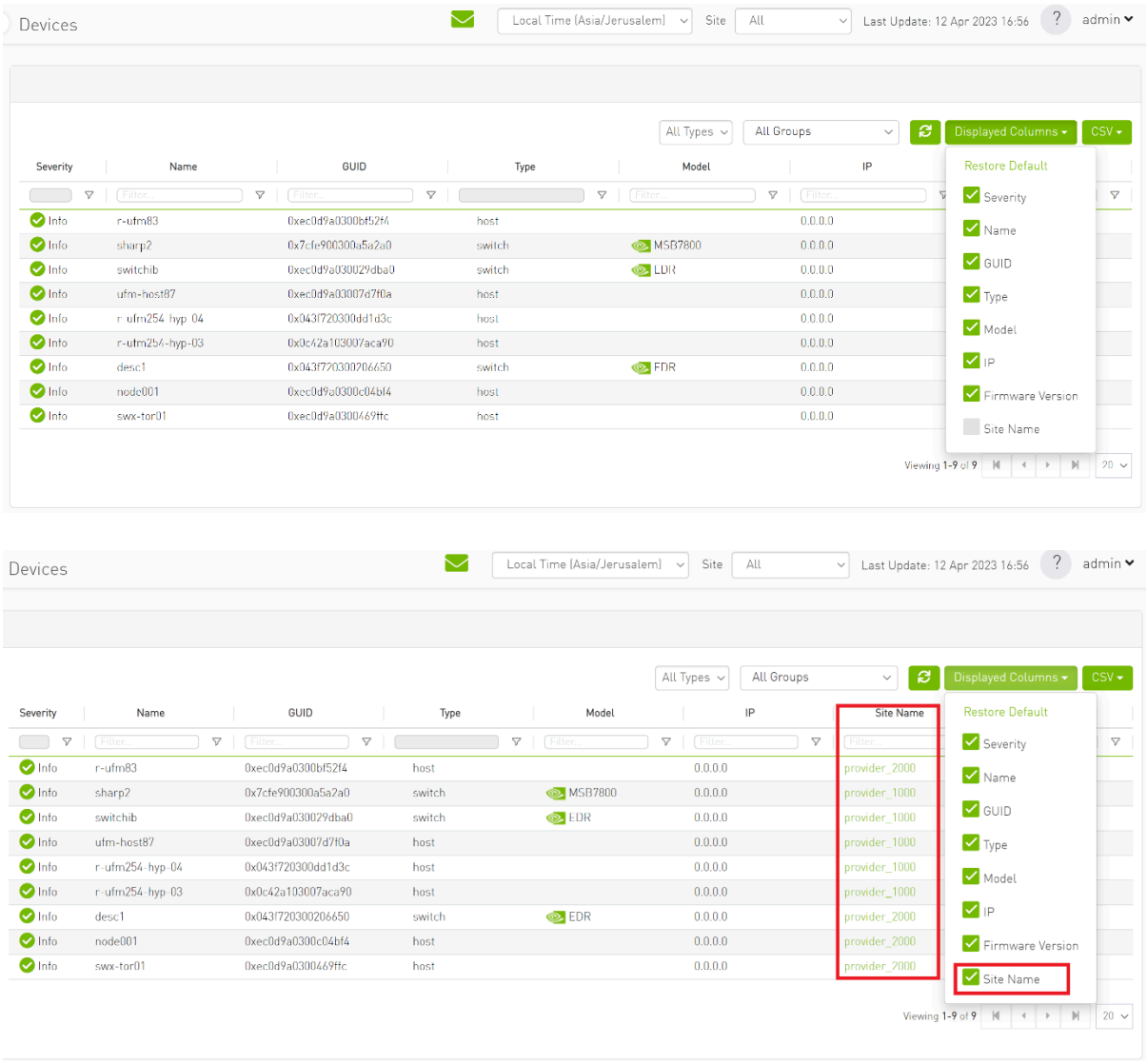
4. Network map contains “clouds” for each provider.



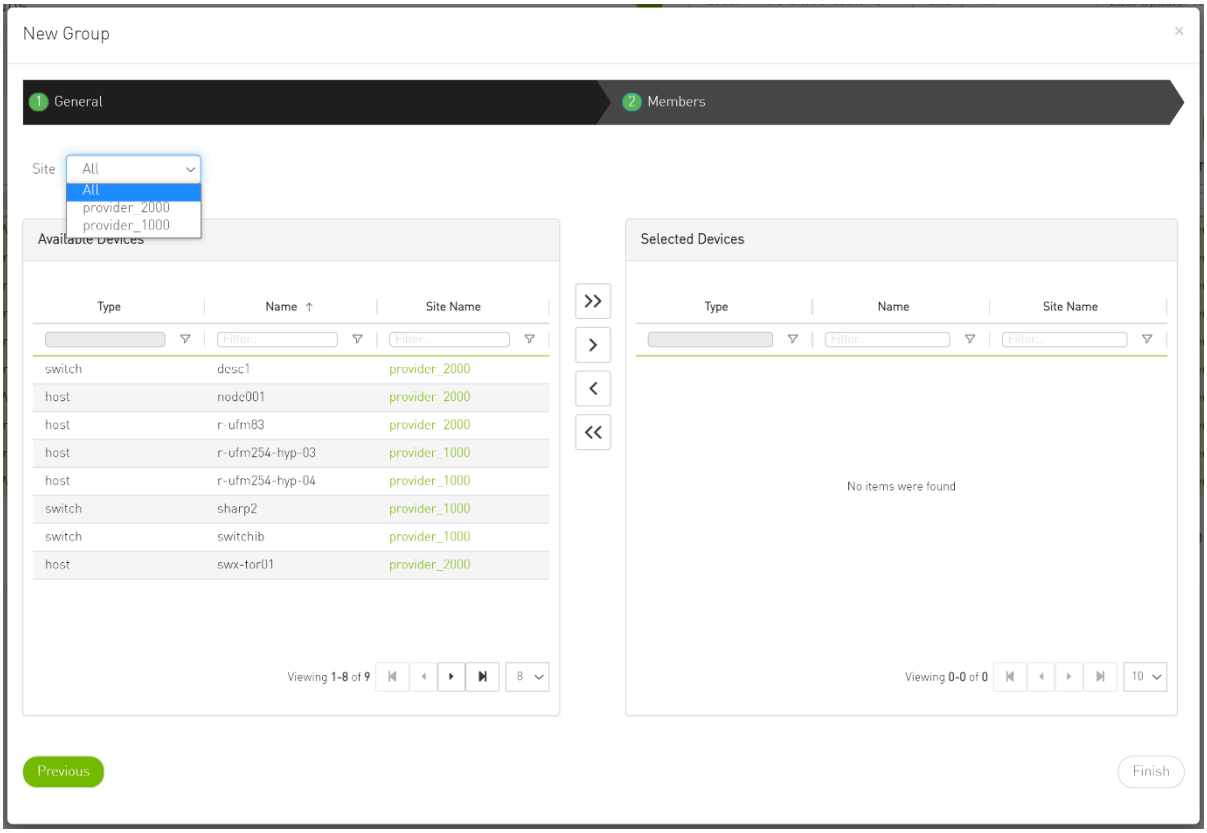


5. A "Site Name" column is present in all Managed Elements sections. The column is disabled (hidden) by default.

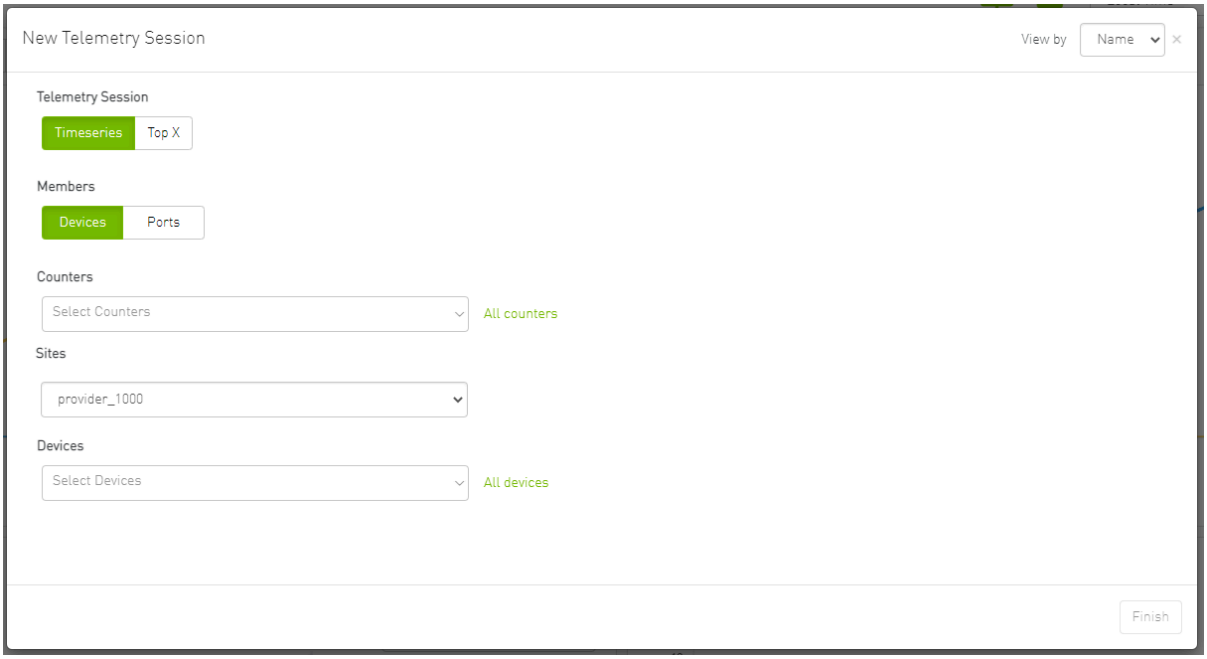


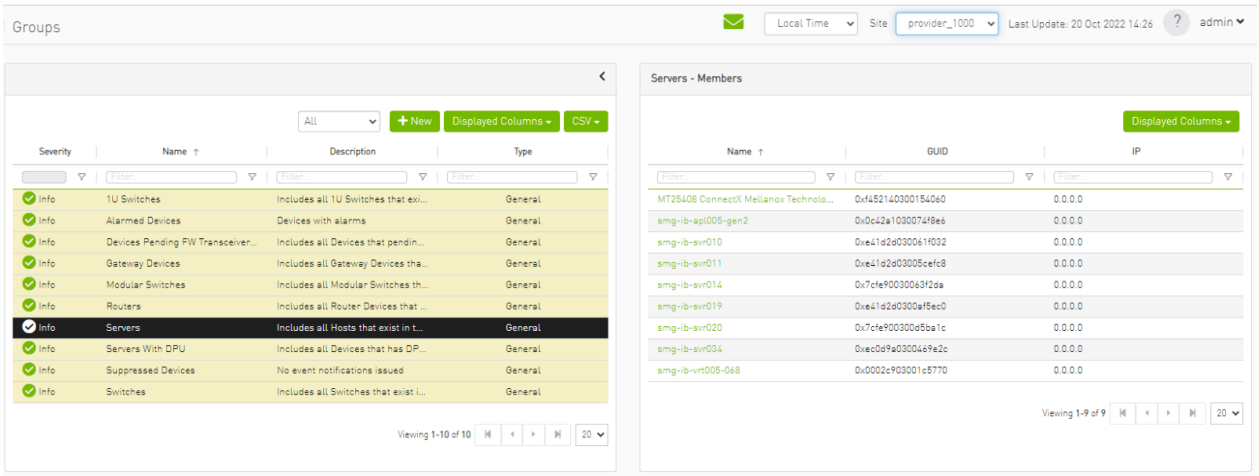


6. The "Group" and "Telemetry" sections include "Site" filters.



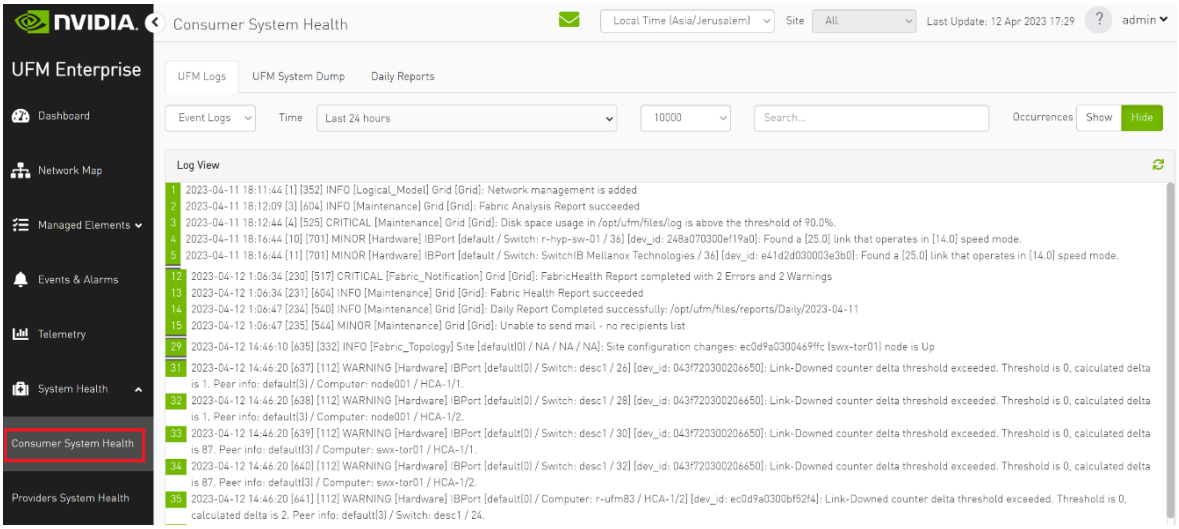
7. The filter in "Groups" impacts the Members table only.



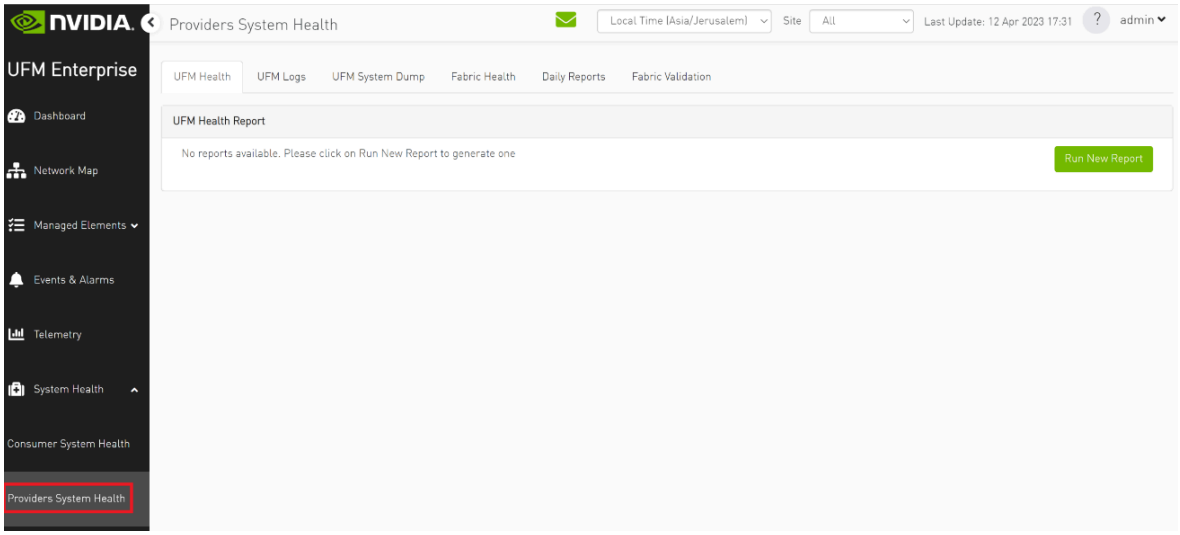


8. In the System Health tab, subsections for Consumer and Provider are available.

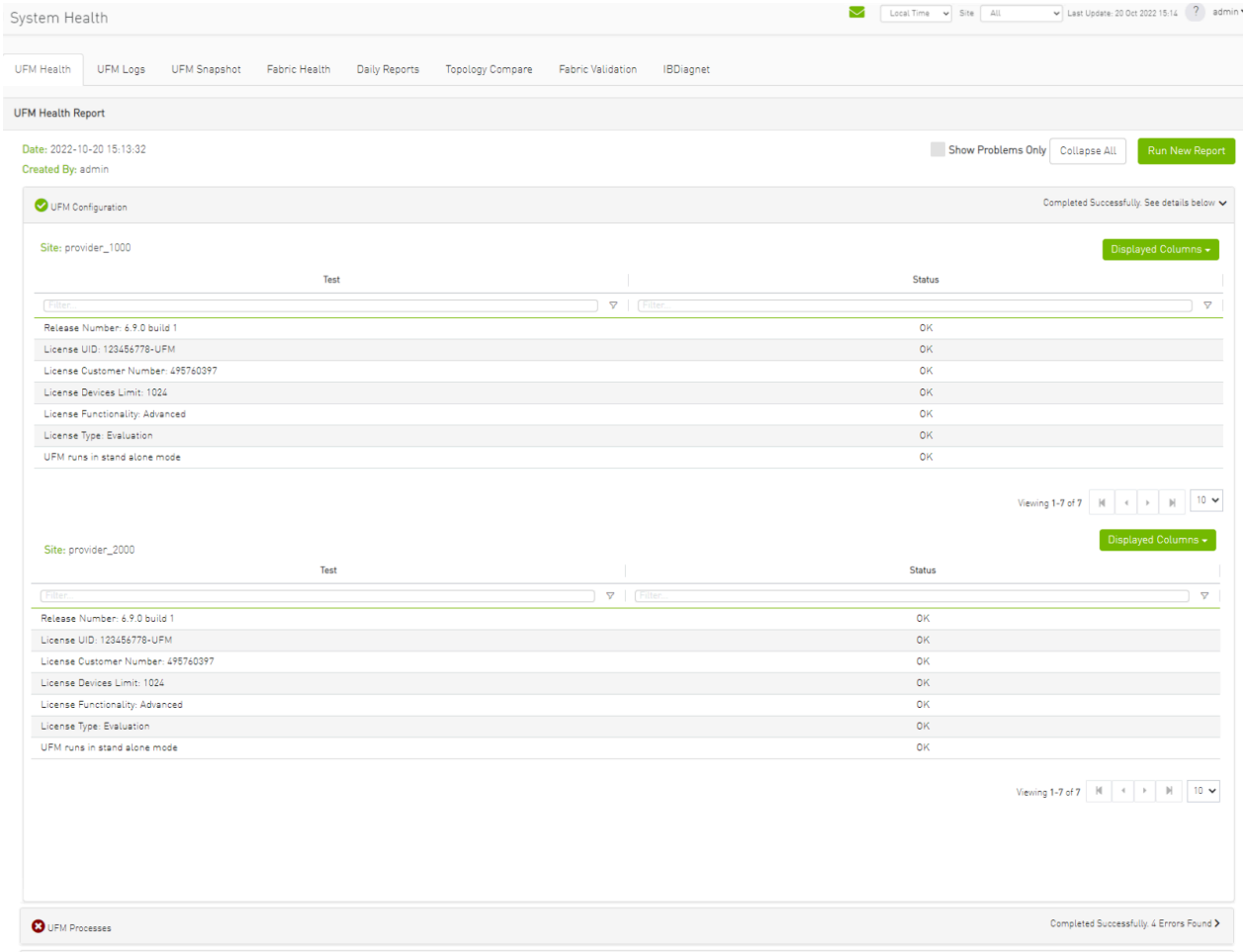
    1. Consumer System Health tab contains sections applicable to Consumer UFM specifically (e.g., logs from Consumer UFM).



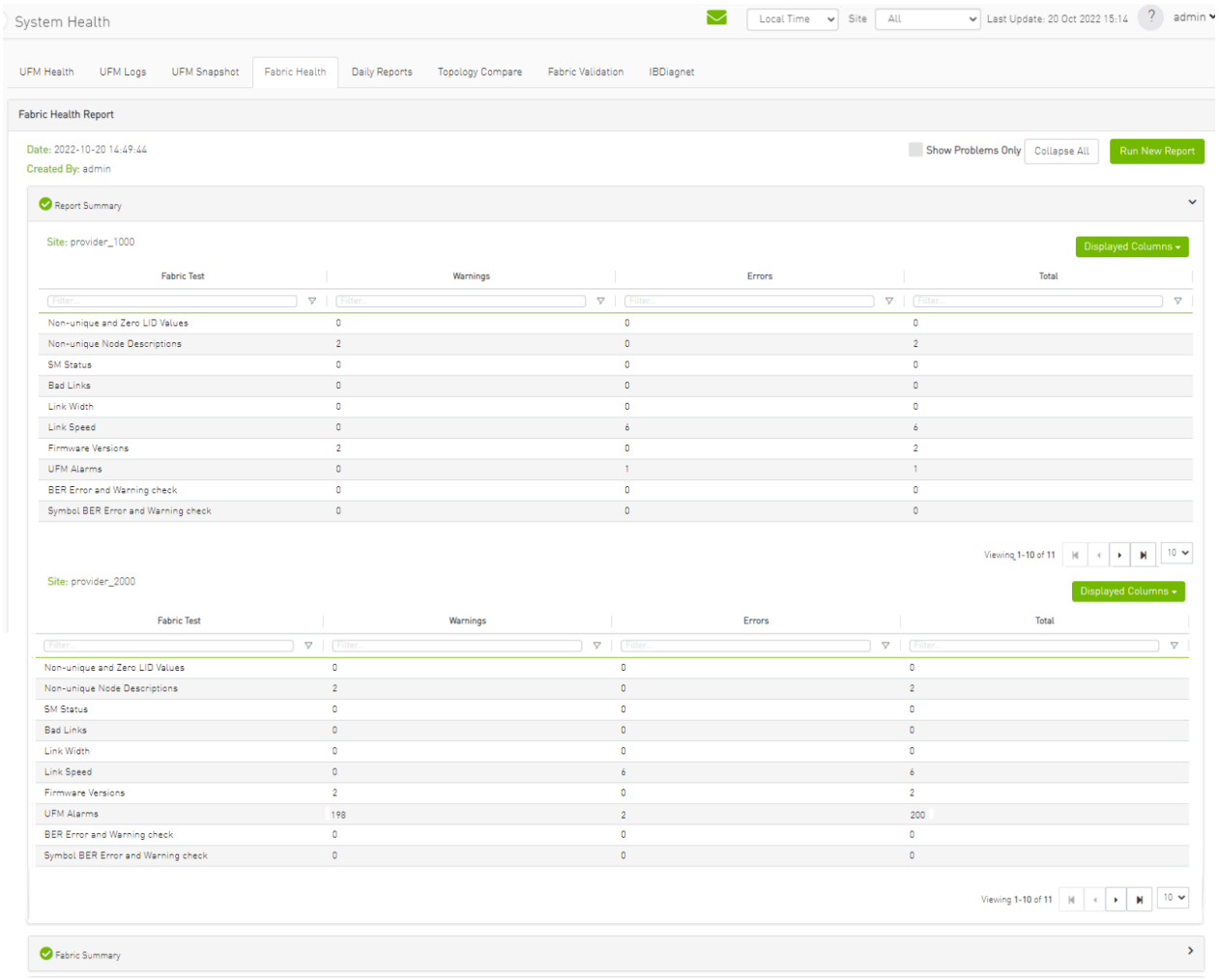
    2. Provider System Health contains sections applicable to one or multiple providers (e.g., Fabric Health Report can be triggered on multiple Providers from the Consumer).



9. UFM Health tab contains sub report tables for each provider.

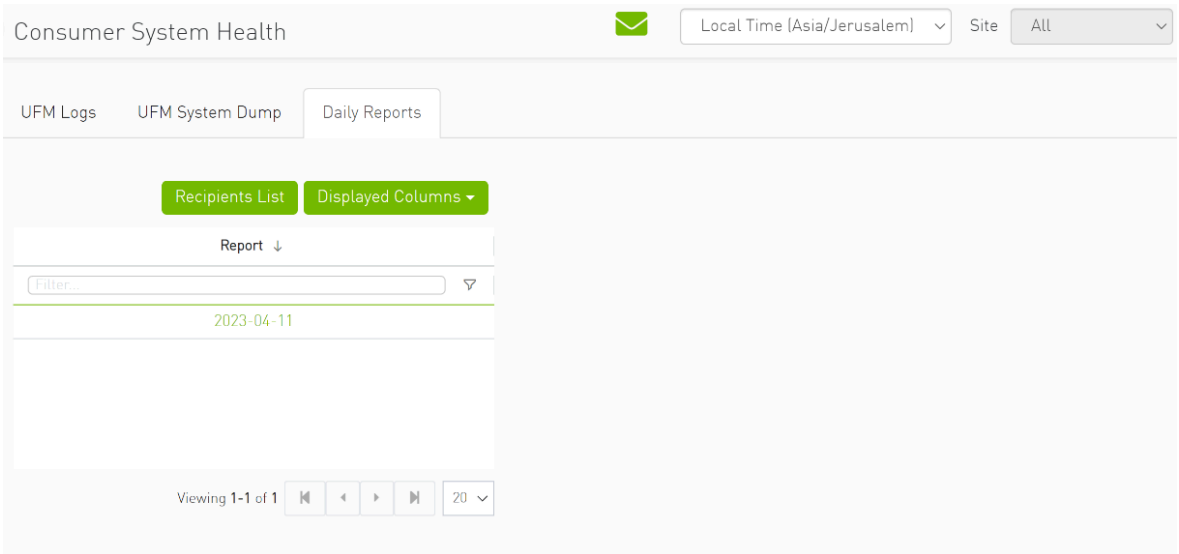


10. Fabric Health contains sub report tables for each provider.

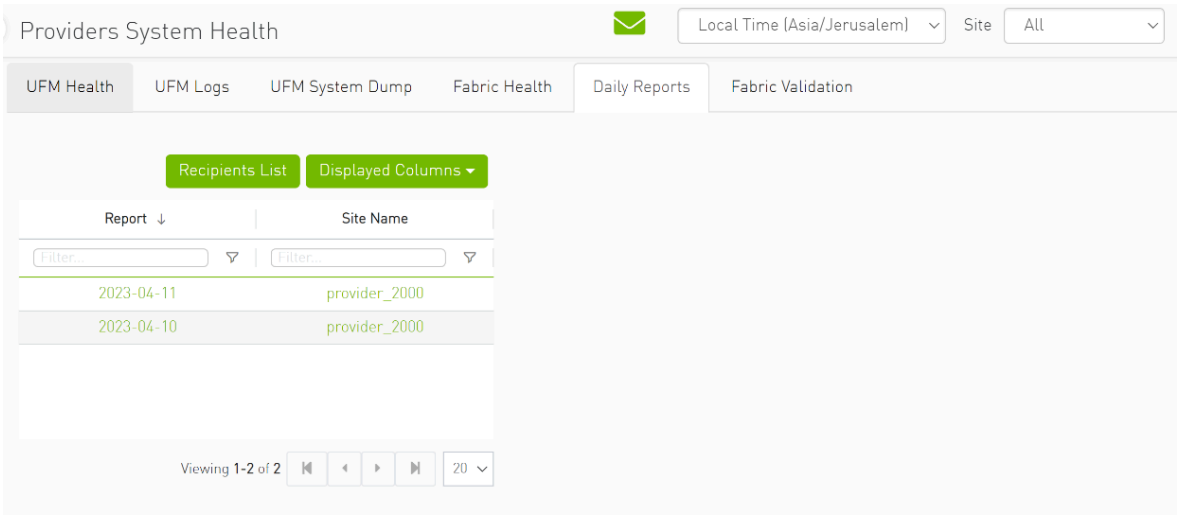


11. Daily Reports:

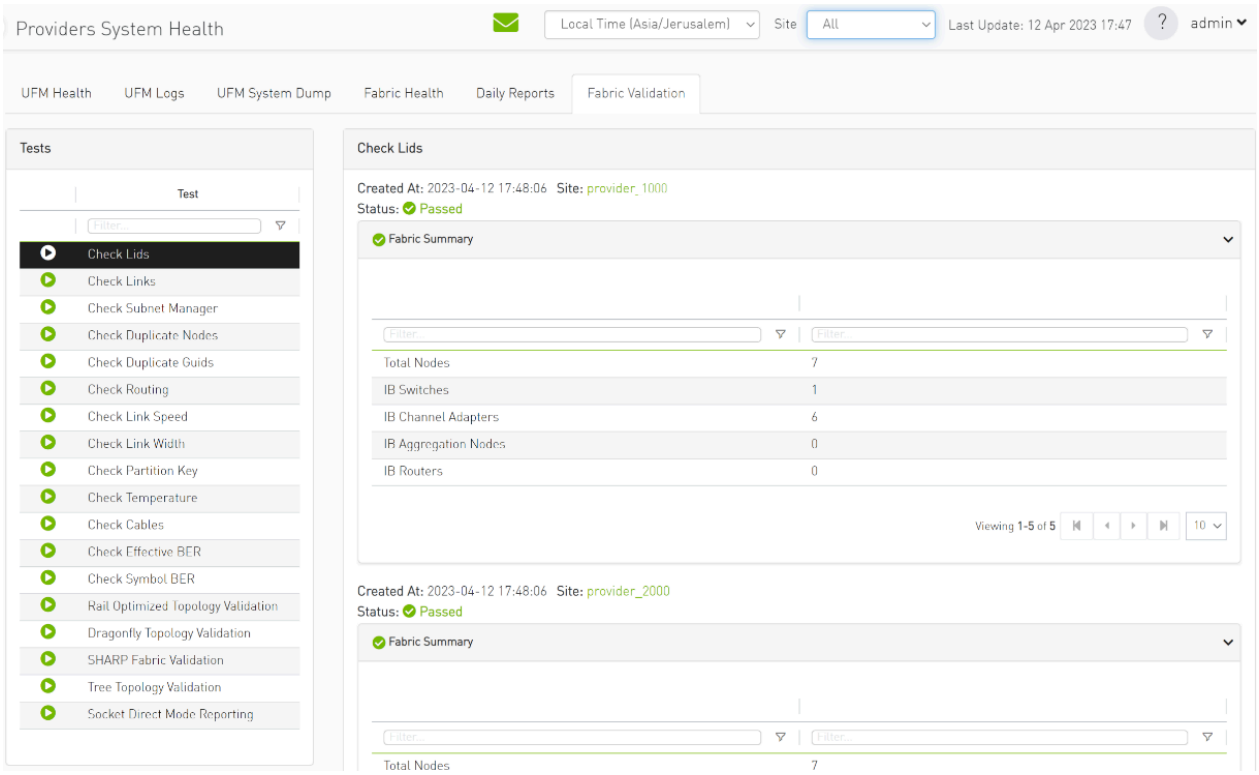
    1. Consumer Daily reports display consumer reports.



2. Providers Daily reports display reports from all providers.

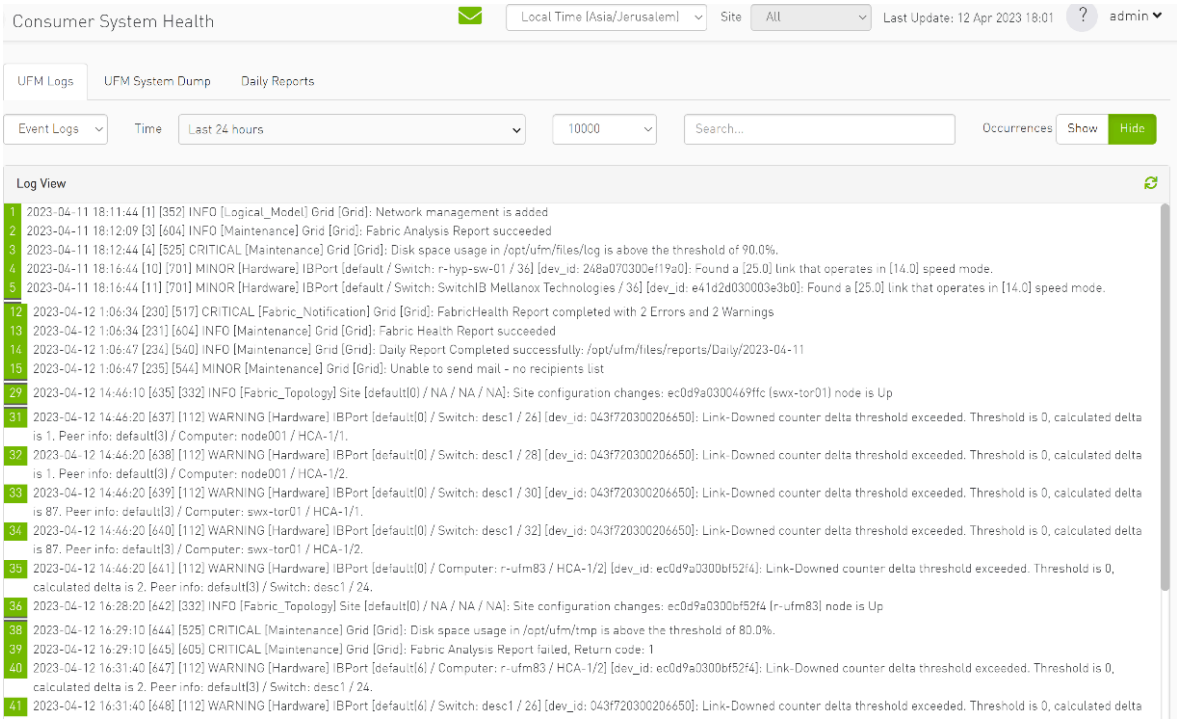


12. The "Fabric Validation" tab contains sub report tables for each provider.

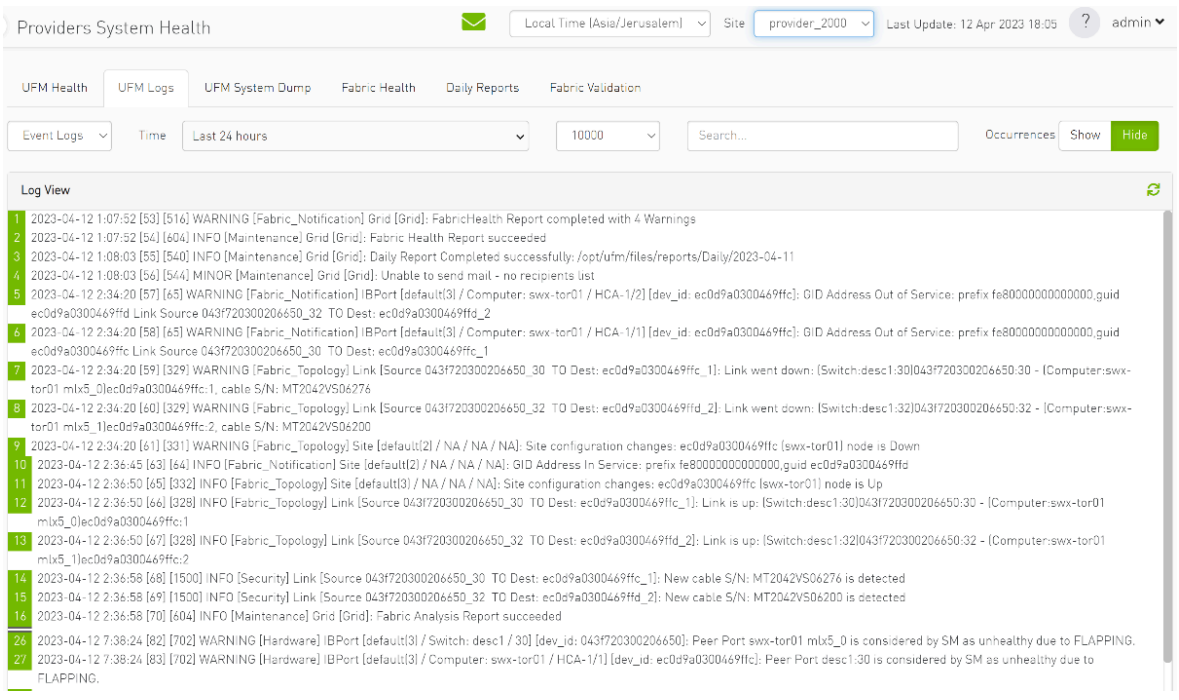


13. In "UFM Logs" Tab:

    1. Consumer logs:

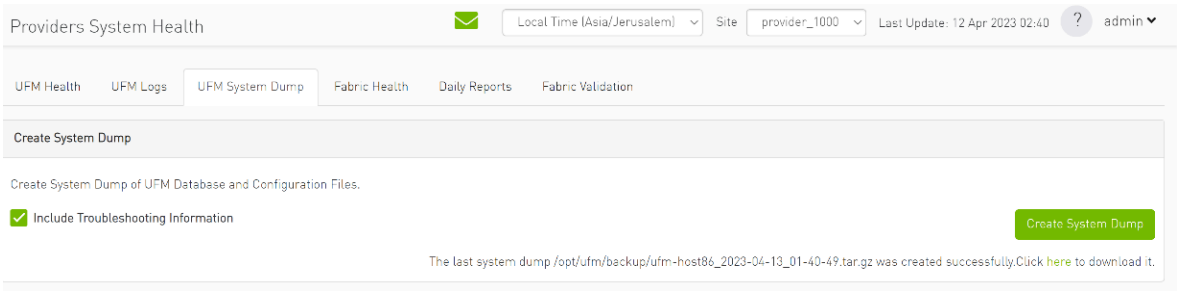


2. Providers logs display providers log separately, displaying logs for all providers is not supported.

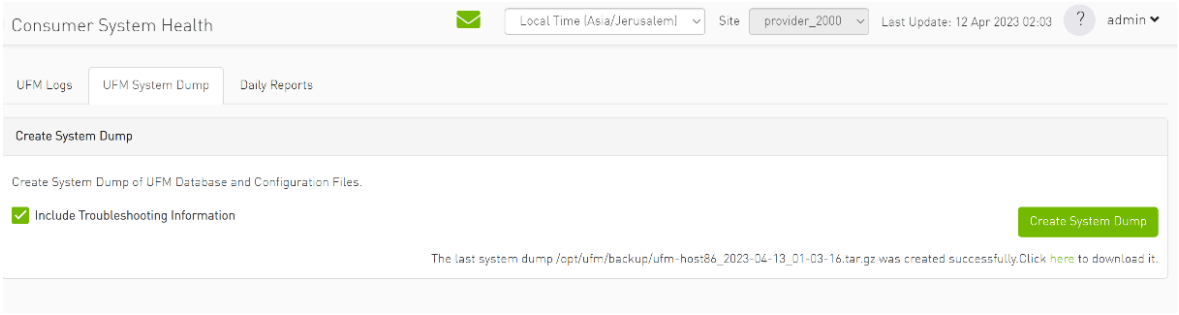


14. In the "System Dump" tab:

    1. "Consumer System Dump" collects system dump for consumer

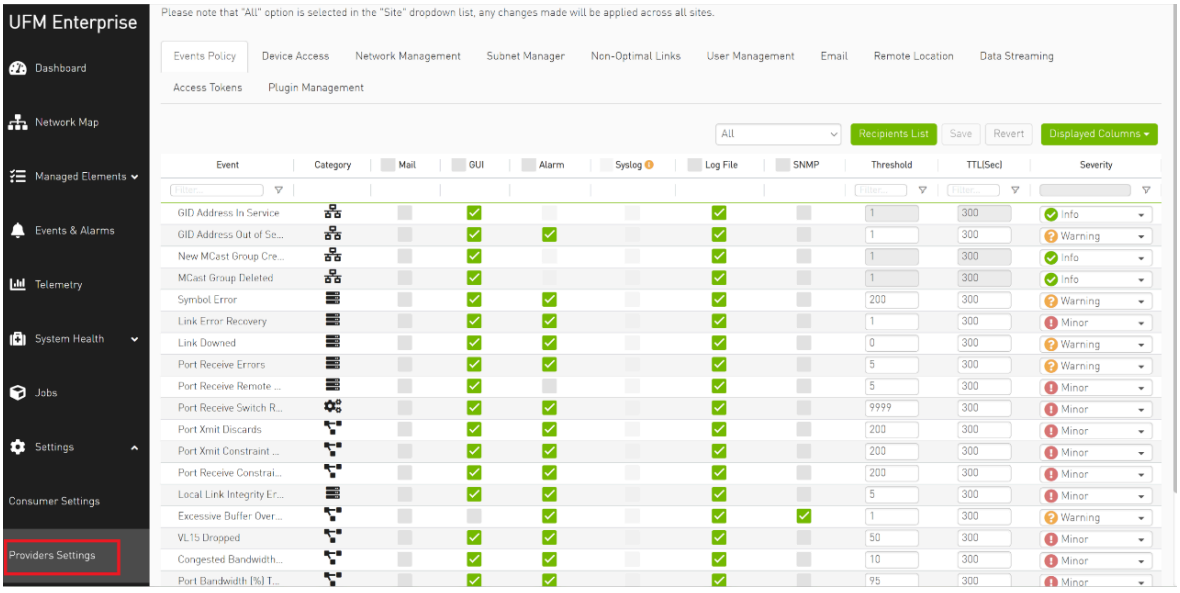


2. "Providers System Dump" collect system dumps for one or all providers and mergeS them into one folder

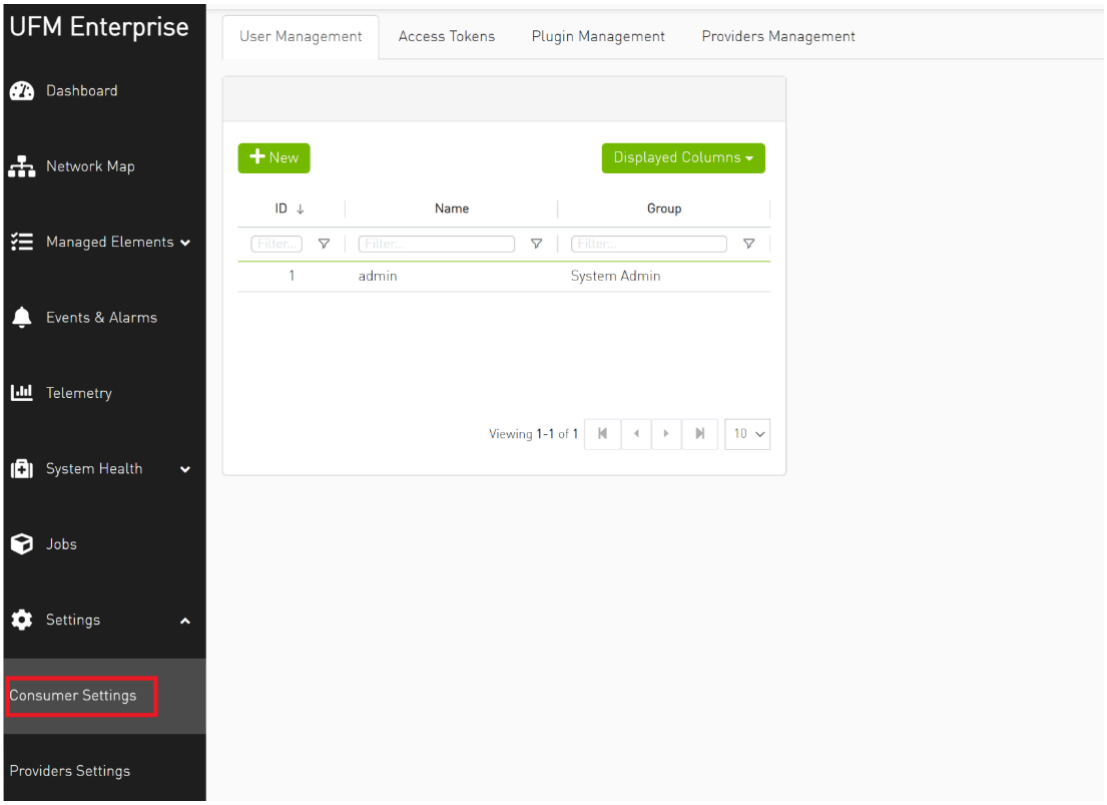


15. Under "Settings", subsections for Consumer and Provider are available.

    1. "Consumer Settings" contain sections applicable to Consumer UFM specifically (e.g., creation of access tokens for UFM consumer authentication);



    2. "Provider Settings" contain sections applicable to one or multiple providers (e.g., Event Policies can be changed for multiple Providers at once from the Consumer).


UFM Enterprise
Dashboard
Network Map
Managed Elements
Events & Alarms
Telemetry
System Health
Jobs
Settings
Consumer Settings
Providers Settings
User Management
Access Tokens
Plugin Management
Providers Management
New
Displayed Columns
ID
Name
Group
1
admin
System Admin
Viewing 1-1 of 1
10

UFM Plugins

The **UFM Plugin** (Unified Fabric Manager Plugin) refers to an add-on or extension used in conjunction with NVIDIA's **UFM software**. The table below lists the available UFM plugins and provides descriptions on their functionalities.

Plugin Deployment

To deploy UFM plugins, download the plugin image location from the **Docker Hub or NLP**. Refer to the table below for the image location per plugin.

The plugins can be deployed either via the **WebUI** or the **UFM management script**.

To deploy the UFM plugin via **WebUI**, perform the following:

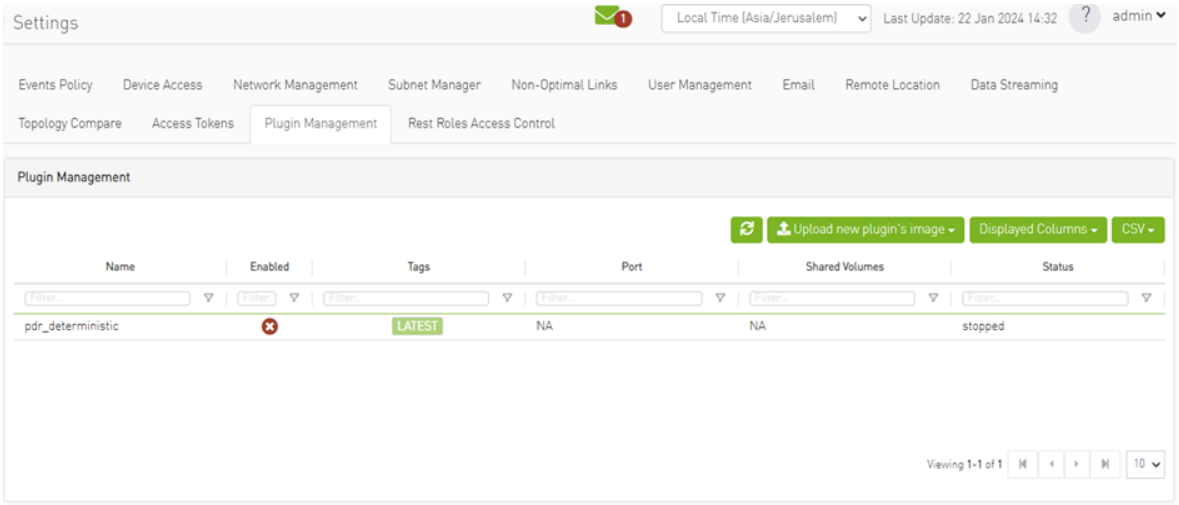
1. Load the downloaded image onto the UFM server. This can be done either by using the UFM GUI by navigating to the Settings -> Plugins Management tab or by loading the image via the following instructions:

    1. Log in to the UFM server terminal.

    2. Run:

        ```
        docker load -I <path_to_image>
        ```

    3. **Via the WebUI:** After successfully loading the plugin image, the plugin should become visible in the plugin management table within the UFM GUI. To initiate the plugin's execution, simply right-click on the respective in the table.



To deploy the UFM plugin via the **Management Script**, refer to UFM Plugins Management.

Plugin	Image Location
REST-RDMA Plugin	https://hub.docker.com/r/mellanox/ufm-plugin-rest-rdma
NDT Plugin	https://hub.docker.com/r/mellanox/ufm-plugin-ndt
UFM Telemetry Fluentd Streaming (TFS) Plugin	https://hub.docker.com/r/mellanox/ufm-plugin-tfs
UFM Events Fluent Streaming (EFS) Plugin	https://hub.docker.com/r/mellanox/ufm-plugin-efs
UFM Bright Cluster Integration Plugin	https://hub.docker.com/r/mellanox/ufm-plugin-bright
Sysinfo Plugin	N/A
SNMP Plugin	https://hub.docker.com/r/mellanox/ufm-plugin-snmp
Packet Level Monitoring Collector (PMC) Plugin	Refer to Plugins Bundle
GNMI-Telemetry Plugin	https://hub.docker.com/r/mellanox/ufm-plugin-gnmi_telemetry
UFM Telemetry Manager (UTM) Plugin	https://hub.docker.com/r/mellanox/ufm-plugin-utm
UFM Consumer Plugin	TBD
Fast-API Plugin	https://hub.docker.com/r/mellanox/ufm-plugin-fast_api

Plugin	Image Location
UFM Light Plugin	https://hub.docker.com/r/mellanox/ufm-plugin-ufm-light
Key Performance Indexes (KPI) Plugin	Refer to Plugins Bundle

Plugins Release Notes

- REST-RDMA Plugin Release Notes
- NDT Plugin Release Notes
- UFM Telemetry Fluentd Streaming (TFS) Plugin Release Notes
- UFM Events Fluent Streaming (EFS) Plugin Release Notes
- UFM Bright Cluster Integration Plugin Release Notes
- IB Link Resiliency Plugin Release Notes
- ClusterMinder Plugin Release Notes
- Sysinfo Plugin Release Notes
- SNMP Plugin Release Notes
- Packet Level Monitoring Collector (PMC) Plugin Release Notes
- GNMI-Telemetry Plugin Release Notes
- UFM Telemetry Manager (UTM) Plugin Release Notes
- UFM Consumer Plugin Release Notes
- Fast-API Plugin Release Notes
- UFM Light Plugin Release Notes
- Key Performance Indexes (KPI) Plugin Release Notes
- UFM Events Grafana Dashboard Plugin Release Notes

- Log Streamer Plugin Release Notes
- GNMI NVOS Events Plugin Release Notes
- NVLink Plugin Release Notes
- Unmanaged Switch Dump (USD) Plugin Release Notes

UFM Plugins Management

UFM Plugins Management

UFM plugin management can be done either via the `manage_ufm_plugins.sh` script or via the Web UI.

UFM Plugin Management Using themanage_ufm_plugins.sh Script

The `manage_ufm_plugins.sh` script, located in the `/opt/ufm/scripts` directory, is designed to manage UFM plugins through the command line interface.

To see the actions supported by this script, run:

```
/manage_ufm_plugins.sh  --help
usage: manage_ufm_plugins.sh <command> [<args>]
positional arguments:
  {show,get-
all,enable,start,stop,disable,add,upgrade,remove_image,remove,is-
running,is-enable,get-http-proxy-port,debug,deploy,deploy-bundle}
                        Commands
    show                Show plugins info
    get-all             get all loaded plugins
    enable              Enable plugin
    start               Start plugins
    stop                Stop all plugins
    disable             Disable plugin
    add                 Add plugin
    upgrade             Upgrade plugin
    remove_image        Remove plugin`s image
    remove              Remove plugin
        notify                          Notify plugin
    is-running          Test plugin is running
    is-enable           Test plugin is enabled
    get-http-proxy-port
                        Get plugin HTTP proxy port
    debug               Debug
    deploy              Deploy plugin image
    deploy-bundle       Deploy bundle of plugins
```

Optional Arguments:

```
  -h, --help            show this help message and exit
  -v, --version         Print version information
```

Each supported option has its own help flag, which can be received by requesting help for a specific parameter. For example:

```
./manage_ufm_plugins.sh  add --help
```

Usage:

```
manage_ufm_plugins.pyc <command> [<args>] add [-h] -p <[A-Za-z0-9_-] Name size: 32> [-t <[A-Za-z0-9._-] Name size: 128>]
```

Optional Arguments:

```
  -h, --help            show this help message and exit
  -p <[A-Za-z0-9_-] Name size: 32>, --plugin-name <[A-Za-z0-9_-] Name size: 32>
                        Plugin name
  -t <[A-Za-z0-9._-] Name size: 128>, --plugin-tag <[A-Za-z0-9._-] Name size: 128>
                        Plugin tag
```

The following table lists the supported commands and provides their description and information on the parameters.

Command	Description	Parameters
`show`	Shows information about running plugins	N/A
`get-all`	Gets information about deployed plugins in JSON format	N/A
`enable`	Enables plugin	Loaded Parameter: plugin image name
`start`	Starts UFM plugin	Loaded Parameter: plugin name

Command	Description	Parameters
`stop`	Stops UFM plugin	Loaded Parameter: plugin name
`disable`	Disables UFM plugin	Loaded Parameter: plugin name
`add`	Adds UFM plugin	Loaded Parameter: image name
`upgrade`	Upgrades UFM plugin	Loaded Parameter: plugin name
`remove_image`	Removes UFM plugin's image	Loaded Parameter: plugin image name
`remove`	Removes UFM plugin	Loaded Parameter: plugin name
`notify`	Notify UFM plugin	Loaded Parameter: plugin name
`is-running`	Tests plugin is running	Loaded Parameter: plugin name
`is-enable`	Tests plugin is enabled	Loaded Parameter: plugin name
`get-http-proxy-port`	Gets plugin HTTP proxy port	Loaded Parameter: plugin name
`deploy`	Deploys plugin image	Loaded Parameter: - `f` flag with path to UFM plugin image
`deploy-bundle`	Deploys bundle of plugins	Loaded Parameter: - `f` flag with path to tarball file, that will contain number of UFM plugins. All the plugin images from the tar ball will be deployed to UFM. Operation could take some time. The progress of the plugin images from the tar file deployment will be printed to terminal, including error messages, if exists.

UFM Plugins Upgrade

The following plugins support the upgrade option:

- GNMI-Telemtry Plugin

- Key Performance Indexes (KPI) Plugin

UFM Plugin Upgrade Procedure

Follow the steps below to upgrade a UFM plugin.

Step 1: Disable the plugin

Before upgrading, the plugin needs to be disabled to ensure a smooth upgrade process.

```
/opt/ufm/scripts/manage_ufm_plugins.sh disable -p <plugin_name>
```

Replace `<plugin_name>` with the actual name of the plugin you want to upgrade.

Step 2: Upgrade the plugin

Run the upgrade command with the version flag to specify the desired version. Replace `<plugin_name>` with the actual plugin name and `<version>` with the desired plugin version.

```
/opt/ufm/scripts/manage_ufm_plugins.sh upgrade -p <plugin_name> -
t <version>
```

Example:

```
/opt/ufm/scripts/manage_ufm_plugins.sh upgrade -p ufm-plugin-
gnmi_telemetry -t 1.1.8-3158
```

Step 3: Enable the plugin

Once the plugin has been successfully upgraded, enable the plugin again.

```
/opt/ufm/scripts/manage_ufm_plugins.sh enable -p <plugin_name>
```

Example:

1. Disable the plugin:

   ```
   /opt/ufm/scripts/manage_ufm_plugins.sh disable -p ufm-plugin-gnmi_telemetry
   ```

2. Upgrade the plugin to version 1.1.8-3158:

   ```
   /opt/ufm/scripts/manage_ufm_plugins.sh upgrade -p ufm-plugin-gnmi_telemetry -t 1.1.8-3158
   ```

3. Enable the plugin:

   ```
   /opt/ufm/scripts/manage_ufm_plugins.sh enable -p ufm-plugin-gnmi_telemetry
   ```

Plugins Bundle

The default plugin bundle contains the NDT, TFS, KPI, PMC, IB Link Resiliency, GNMI-Telemetry, UFM Telemetry Manager (UTM) and ClusterMinders plugins, packaged in a tarball file. To download this bundle, refer to NVIDIA's Licensing Portal .

To deploy the plugin bundle within the UFM ecosystem, follow these steps:

For a UFM Bare Metal Deployment

1. Download the bundle from NVIDIA's Licensing Portal.
2. For **Master/Standby High-Availability or Standalone Mode**, run the following command:

```
 /opt/ufm/scripts/manage_ufm_plugins.sh deploy-bundle -f
/opt/ufm/ufm_plugins_data/<plugins bundle file name>
```

3. Restart Apache. Run:

   For Ubuntu:

```
service apache2 restart
```

   For RedHat:

```
service httpd restart
```

For Docker Container-Based Deployment

1. Download the bundle from NVIDIA's Licensing Portal.
2. For **Standby High-Availability Mode**, run the following command:

```
/usr/bin/docker run --rm -t --name=ufm \
    --volume /dev/log:/dev/log \
    --volume /var/run/docker.sock:/var/run/docker.sock \
    --volume
/opt/ufm/ufm_plugins_data/:/opt/ufm/ufm_plugins_data/ \
    --privileged \
    --entrypoint /bin/bash \
    mellanox/ufm-enterprise \
    -c "/opt/ufm/scripts/manage_ufm_plugins.sh deploy-bundle -f
/opt/ufm/ufm_plugins_data/<plugins bundle file name>"
```

3. For **Master High-Availability or Standalone Mode**

    1. **When UFM docker is running** , run the following command:

    ```
    docker exec ufm /opt/ufm/scripts/manage_ufm_plugins.sh
    deploy-bundle -f /opt/ufm/ufm_plugins_data/<plugins
    bundle file name>
    ```

    2. **When UFM docker is not running**, run the following command:

```
/usr/bin/docker run --rm -t --name=ufm \
   --volume
/opt/ufm/files/:/opt/ufm/shared_config_files/ \
   --volume /dev/log:/dev/log \
   --volume /var/run/docker.sock:/var/run/docker.sock \
   --volume
/opt/ufm/ufm_plugins_data/:/opt/ufm/ufm_plugins_data/ \
   --privileged \
   --entrypoint /bin/bash \
   mellanox/ufm-enterprise \
   -c "/opt/ufm/scripts/manage_ufm_plugins.sh deploy-bundle -f
/opt/ufm/ufm_plugins_data/<plugins bundle file name>"
```

All plugin images from the tarball will be deployed to UFM. The process may take some time, and progress, including any error messages, will be displayed in the terminal.

REST-RDMA Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.0.0-39	N/A

Bug Fixes

Plugin Version	Bug Fix
1.0.0-39	N/A

Overview

The REST-RDMA is a tool designed for sending requests over InfiniBand to the UFM server. These REST requests can fall into three categories:

1. UFM REST API requests
2. ibdiagnet requests
3. Telemetry requests

The rest-rdma utility is distributed as a Docker container, capable of functioning both as a server and a client.

> **Note**
>
> **A general note on running `ufm_rdma.py` Inside the Docker Container**
>
> When running `./ufm_rdma.py` from inside the Docker container, make sure to set the following environment variables after entering the container:
>
> ```
> export RAPIDS_LIBUCX_PREFER_SYSTEM_LIBRARY=true
> export UCX_WARN_UNUSED_ENV_VARS=n
> ```
>
> These variables only need to be set once per container session.
>
> After that, every call to the Python script must be executed using the container's virtual environment Python:
>
> `/venv/bin/python3 ./ufm_rdma.py`

Deployment Server

Deploying the Plugin on UFM Appliance

1. Log into your UFM as admin.
2. Enter config mode. Run:

```
enable
config terminal
```

> **Note**
>
> Make sure that UFM is running with `show ufm status`. If UFM is down, then run with `ufm start`.

3. Ensure that rest-rdma plugin is disabled with the `show ufm plugin` command.
4. Pull the plugin container with `docker pull mellanox/ufm-plugin-rest-rdma:[version].`
5. Run `ufm plugin rest-rdma add tag [version]` to enable the plugin.
6. Check that plugin is up and running with `docker pull mellanox/ufm-plugin-rest-rdma:[version]`

Deploying the Plugin on Bare Metal Server

1. Verify that UFM is installed and running.
2. Pull image from docker hub:

```
docker pull mellanox/ufm-plugin-rest-rdma:[version]
```

3. To load image run:

`/opt/ufm/scripts/manage_ufm_plugins.py add -p rest-rdma`

Example with credentials:

Deployment Client

Run the following command to pull the image from the docker hub:

```
docker pull mellanox/ufm-plugin-rest-rdma:[version]
```

Verify that the `/tmp/ibdiagnet` directory exists on the client's computer. If not – create it.

To start container as client (on any host in the same fabric as UFM server) run:

```
docker run -d --network=host --privileged --name=ufm-plugin-rest-rdma --rm -v /tmp/ibdiagnet:/tmp/ibdiagnet mellanox/ufm-plugin-rest-rdma:[version] client
```

To check that plugin is up and running, run:

```
docker ps
```

How to Run

Server

In server mode `ufm_rdma.py` is started automatically and is restarted if exited. If the `ufm_rdma.py` server is not running, follow the following instructions:

- Enter the plugin's Docker Container.
- Set the environment variables:

```
export RAPIDS_LIBUCX_PREFER_SYSTEM_LIBRARY=true
export UCX_WARN_UNUSED_ENV_VARS=n
```

- Navigate to the plugin directory:

```
cd /opt/ufm/src/ufm-plugin-ufm-rest
```

- Start the server:

```
/venv/bin/python3 ./ufm_rdma.py -r server &
```

Client

There are three options to run client. Running the client from inside the Docker container, using a custom script from the hosting server to execute the client or using the "docker exec" command from the hosting server.

1. **Option 1:** Run the client from inside the Docker container
    1. Enter the docker container using `docker exec -it ufm-plugin-rest-rdma bash`
    2. Then, run `cd /opt/ufm/src/ufm-plugin-rest-rdma`
    3. Use the `-h` help option to see the available parameters

```
./ufm_rdma.py -h
```

2. **Option 2:** From the host server, the scripts can be located at `/opt/ufm/ufm-plugin-ufm-rest/` directory inside the docker container. They can copied using the following command:

> **Note**
>
> cp <containerId>:/opt/ufm/ufm-plugin-ufm-rest/[script name] /host/path/target

Example:

> **Note**
>
> cp <containerId>:/opt/ufm/ufm-plugin-ufm-rest/ufm-rest-rdma_client.sh /host/path/target

1. To see the available options, run:

```
./ufm-rest-rdma_client.sh -h
```

3. **Option 3:** From hosting server, use the `docker exec` command.

Note

To run from inside docker, run:

`docker exec ufm-plugin-rest-rdma` prior to the command.

For example:

```
docker exec ufm-plugin-rest-rdma
/venv/bin/python3 /opt/ufm/ufm-plugin-ufm-
rest/src/ufm_rdma.py -r client -u admin -p
password -t simple -a GET -w
ufmRest/app/ufm_version
```

Authentication Configuration

Telemetry and ibdiagnet request authentication options could be enabled or disabled (enabled by default – set to True) in `ufm_rdma.ini file` in [Server] section on the server. The `rest_rdma` server performs simple requests to UFM server, using supplied credentials to verify that the user is allowed to run telemetry or ibdiagnet requests.

```
[Server]
use_ufm_authentication=True
```

Remote ibdiagnet Request

The following two user scripts can run on the hosting server.

- remote_ibdiagnet_auth.sh
- remote_ibdiagnet.sh

These scripts should be copied from the container to the hosting server using the following command:

```
cp <containerId>:/opt/ufm/ufm-plugin-ufm-rest/[script name]
/host/path/target
```

Example :

```
cp <containerId>:/opt/ufm/ufm-plugin-ufm-
rest/remote_ibdiagnet_auth.sh /host/path/target
```

The `remote_ibdiagnet.sh` script does not require authentication as the server side can run on a machine which does not run UFM (which is responsible for the authentication). This means it can run from the hosting server.

```
/remote_ibdiagnet.sh [options]
```

Authenticated Remote ibdiagnet Request

Note

When running `./ufm_rdma.py` from inside the Docker container, make sure to set the following environment variables after entering the container:

```
export RAPIDS_LIBUCX_PREFER_SYSTEM_LIBRARY=true
export UCX_WARN_UNUSED_ENV_VARS=n
```

These variables only need to be set once per container session.

After that, every call to the Python script must be executed using the container's virtual environment Python:

`/venv/bin/python3 ./ufm_rdma.py`

The `remote_ibdiagnet_auth.sh` script can receive parameters as credentials for authentication with UFM server.

```
/remote_ibdiagnet_auth.sh [options]
```

To get all the options, run the following command:

```
/remote_ibdiagnet_auth.sh -h
```

Note

Important Note:

When using remote_ibdiagnet.sh, authentication is not required and the the ibdiagnet parameters should be sent in ibdiagnet format.

Example: `./remote_ibdiagnet.sh --get_phy_info`

When using the remote_ibdiagnet**_auth**.sh, the ibdiagnet parameters should be sent using the `-l` key.

Examples without credentials:

- `./remote_ibdiagnet_auth.sh -l '--get_phy_info'`
- `./remote_ibdiagnet_auth.sh -u username -p password`
- `./remote_ibdiagnet_auth.sh -u username -p password -l '--get_phy_info'`
- `./remote_ibdiagnet_auth.sh -k [token string] -l '--get_phy_info'`
- `./remote_ibdiagnet_auth.sh -s [defined for client certificate host name] -d [path to client certificate pfx file] -l '--get_phy_info'`

Please use the -h option to see the examples of credential usage.

REST Request with Username/Password Authentication

Note

When running `./ufm_rdma.py` from inside the Docker container, make sure to set the following environment variables after entering the container:

```
export RAPIDS_LIBUCX_PREFER_SYSTEM_LIBRARY=true
export UCX_WARN_UNUSED_ENV_VARS=n
```

These variables only need to be set once per container session.

After that, every call to the Python script must be executed using the container's virtual environment Python:

`/venv/bin/python3 ./ufm_rdma.py`

To get the UFM version from inside the docker:

```
export RAPIDS_LIBUCX_PREFER_SYSTEM_LIBRARY=true
export UCX_WARN_UNUSED_ENV_VARS=n
/venv/bin/python3 ./ufm_rdma.py -r client -u admin -p admin_pwd -
t simple -a GET -w ufmRest/app/ufm_version
```

To get the UFM version from hosting server using script:

```
./ufm_rest_rdma_client.sh -u admin -p admin_pwd -t simple -a GET
-w ufmRest/app/ufm_version
```

For telemetry:

```
./ufm_rdma.py -r client -u admin -p admin_pwd -t telemetry -a GET
-g 9001 -w /csv/enterprise
```

To get ibdiagnet run result using UFM REST API from inside the docker:

```
./ufm_rdma.py -r client -u admin -p admin_pwd -t ibdiagnet -a
POST -w ufmRest/reports/ibdiagnetPeriodic -l '{"general": {"name":
"IBDiagnet_CMD_1234567890_199_88", "location": "local", "running_mode": "once"}, "command_flags": {"--
pc": ""}}'
```

REST Request with Client Certificate Authentication

Note

When running `./ufm_rdma.py` from inside the Docker container, make sure to set the following environment variables after entering the container:

```
export RAPIDS_LIBUCX_PREFER_SYSTEM_LIBRARY=true
export UCX_WARN_UNUSED_ENV_VARS=n
```

These variables only need to be set once per container session.

After that, every call to the Python script must be executed using the container's virtual environment Python:

`/venv/bin/python3 ./ufm_rdma.py`

```
need to pass path to client certificate file and name of UFM
server machine:
/venv/bin/python3 ./ufm_rdma.py -r client -t simple -a GET -w
ufmRest/resources/modules -d /path/to/certificate/file/ufm-
client.pfx -s ufm.azurehpc.core.azure-test.net
for telemetry if need authentication from inside the docker
/venv/bin/python3 ./ufm_rdma.py -r client  -t telemetry -a GET -g
9001 -w csv/enterprise -d /path/to/certificate/file/ufm-
client.pfx -s ufm.azurehpc.core.azure-test.net
```

Note

Client certificate file should be located INSIDE the docker container.

REST Request with Token Authentication

Note

When running `./ufm_rdma.py` from inside the Docker container, make sure to set the following environment variables after entering the container:

```
export RAPIDS_LIBUCX_PREFER_SYSTEM_LIBRARY=true
export UCX_WARN_UNUSED_ENV_VARS=n
```

These variables only need to be set once per container session.

After that, every call to the Python script must be executed using the container's virtual environment Python:

`/venv/bin/python3 ./ufm_rdma.py`

```
need to pass token for authentication
/venv/bin/python3 ./ufm_rdma.py -r client -k
OGUY7TwLvTmFkXyTkcsEWD9KKNvq6f -t simple -a GET -w
ufmRestV3/app/ufm_version
for telemetry if need to perform authentication
/venv/bin/python3 ./ufm_rdma.py -r client -k
4rQRf7i7wEeliuJEurGbeecc210V6G  -t telemetry -a GET -g 9001 -w
/csv/enterprise
```

Note

Token could be generated using UFM UI.

> **Note**
>
> If a token is used for client authentication, `ufmRestV3` must be used.

NDT Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.1.1-24	Added support for XDR (plane ports).

Bug Fixes

Plugin Version	Bug Fix
1.1.1-24	N/A

Overview

NDT plugin is a self-contained Docker container with REST API support managed by UFM. The NDT plugin introduces the following capabilities:

1.

    1. **NDT topology comparison:** Allows the user to compare InfiniBand fabric managed by the UFM and NDT files which are used for the description of

InfiniBand clusters network topology.

- Verifies the IB fabric connectivity during cluster bring-up.
- Verifies the specific parts of IB fabric after component replacements.
- Automatically detects any changes in topology.

2. **Subnet Merger - Expansion of the fabric based on NDT topology files**

   Allows users to gradually extend the InfiniBand fabric without causing any disruption to the running fabric. The system administrator should prepare the NDT topology files, which describe the InfiniBand fabric extensions. Then, an intuitive and user-friendly UI wizard facilitates the topology extension process with a step-by-step guidance for performing necessary actions.

   - The Subnet Merger tool verifies the fabric topology within a predefined NDT file, and reports issues encountered for immediate resolution.
   - Once the verification results are acceptable by the network administrator, the tool creates a topoconfig file to serve as input for OpenSM. This allows setting the physical port states of the designated boundary ports as desired (physical ports can be set as disabled or no-discover).
   - Once the topoconfig file is deployed, the IB network can be extended and verified for the next IB extension.

Deployment

The following are the possible ways NDT plugin can be deployed:

1. On UFM Appliance
2. On UFM Software

For detailed instructions on how to deploy the NDT plugin refer to this page.

Authentication

Following authentication types are supported:

- basic (/ufmRest)
- client (/ufmRestV2)

- token (/ufmRestV3)

REST API

The following REST APIs are supported:

Topodiff

- GET /help
- GET /version
- POST /upload_metadata
- GET /list
- POST /compare
- POST /cancel
- GET /reports
- GET /reports/<report_id>
- POST /delete

Subnet Merger

- GET /merger_ndts_list
- GET /merger_ndts_list/<ndt_file_name>
- POST /merger_upload_ndt
- POST /merger_verify_ndt
- GET /merger_verify_ndt_reports
- GET /merger_verify_ndt_reports/<report_id>
- POST /merger_update_topoconfig

- POST /merger_deploy_ndt_config
- POST /merger_update_deploy_ndt_config
- POST /merger_delete_ndt
- GET /merger_deployed_ndt
- POST /merger_create_topoconfig

For detailed information on how to interact with NDT plugin, refer to the NVIDIA UFM Enterprise REST API Guide > NDT Plugin REST API.

NDT Format – Topodiff

NDT is a CSV file containing data relevant to the IB fabric connectivity. The NDT plugin extracts the IB connectivity data based on the following fields:

1. Start device
2. Start port
3. End device
4. End port
5. Link type

Switch to Switch NDT

By default, IB links are filtered by:

- Link Type is Data
- Start Device and End Device end with IBn, where n is a numeric value.

For TOR switches, Start port/End port field should be in the format **Port N**, where **N** is a numeric value.

For Director switches, Start port/End port should be in the format **Blade N_Port i/j**, where **N** is a leaf number, **i** is an internal ASIC number and **j** is a port number.

Examples:

Start Device	Start Port	End Device	End Port	Link Type
DSM07-0101-0702-01IB0	Port 21	DSM07-0101-0702-01IB1	Blade 2_Port 1/1	Data
DSM07-0101-0702-01IB0	Port 22	DSM07-0101-0702-01IB1	Blade 2_Port 1/1	Data
DSM07-0101-0702-01IB0	Port 23	DSM07-0101-0702-02IB1	Blade 3_Port 1/1	Data
DSM09-0101-0617-001IB2	Port 33	DSM09-0101-0721-001IB4	Port 1	Data
DSM09-0101-0617-001IB2	Port 34	DSM09-0101-0721-001IB4	Port 2	Data
DSM09-0101-0617-001IB2	Port 35	DSM09-0101-0721-001IB4	Port 3	Data

Switch to Host NDT

NDT is a CSV file containing data not only relevant to the IB connectivity.

Extracting the IB connectivity data is based on the following five fields:

1. Start device
2. Start port
3. End device
4. End port
5. Link type

IB links should be filtered by the following:

- Link type is "Data".
- "Start Device" or "End Device" end with **IBN**, where **N** is a numeric value.

- The other Port should be based on persistent naming convention: **ibpXsYfZ**, where **X**, **Y** and **Z** are numeric values.

For TOR switches, Start port/End port field will be in the format Port n, where n is a numeric value.

For Director switches, Start port/End port will be in the format **Blade N_Port i/j**, where **N** is a leaf number, **i** is an internal ASIC number and **j** is a port number.

Examples:

Start Device	Start Port	End Device	End Port	Link Type
DSM071081704019	DSM071081704019 ibp11s0f0	DSM07-0101-0514-01IB0	Port 1	Data
DSM071081704019	DSM071081704019 ibp21s0f0	DSM07-0101-0514-01IB0	Port 2	Data
DSM071081704019	DSM071081704019 ibp75s0f0	DSM07-0101-0514-01IB0	Port 3	Data

Other

Comparison results are forwarded to syslog as events. Example of `/var/log/messages` content:

1. Dec 9 12:32:31 <server_ip> ad158f423225[4585]: NDT: missing in UFM "SAT111090310019/SAT111090310019 ibp203s0f0 - SAT11-0101-0903-19IB0/15"

2. Dec 9 12:32:31 <server_ip> ad158f423225[4585]: NDT: missing in UFM "SAT11-0101-0903-09IB0/27 - SAT11-0101-0905-01IB1-A/Blade 12_Port 1/9"

3. Dec 9 12:32:31 <server_ip> ad158f423225[4585]: NDT: missing in UFM "SAT11-0101-0901-13IB0/23 - SAT11-0101-0903-01IB1-A/Blade 08_Port 2/13"

Minimal interval value for periodic comparison in five minutes.

In case of an error the clarification will be provided.

For example, the request “`POST /compare`” without NDTs uploaded will return the following:

- URL: https://<server_ip>/ufmRest/plugin/ndt/compare
- response code: 400
- Response:

```
{
  "error": [
    "No NDTs were uploaded for comparison"
  ]
}
```

Configurations could be found in "`ufm/conf/ndt.conf`"

- Log level (default: INFO)
- Log size (default: 10240000)
- Log file backup count (default: 5)
- Reports number to save (default: 10)
- NDT format check (default: enabled)
- Switch to switch and host to switch patterns (default: see NDT format section)

Logs could be found in "`ufm/logs/ndt.log`".

NDT Format – Subnet Merger

The Subnet Merger tool facilitates the seamless expansion of the InfiniBand fabric based on Non-Disruptive Topology (NDT) files. This section outlines the process of extending the fabric while ensuring uninterrupted operation. The tool operates through an intuitive UI wizard, guiding users step-by-step in extending the fabric topology.

The Subnet Merger tool enables the gradual expansion of the InfiniBand fabric without causing disruptions to the existing network. To achieve this, system administrators need

to prepare NDT topology files that describe the planned fabric extensions. The tool offers an intuitive UI wizard that simplifies the extension process.

Functionality

1. **NDT Topology File Verification:** The Subnet Merger tool verifies the InfiniBand fabric topology specified in a predefined NDT file. During this verification, any issues encountered are reported to the user for immediate resolution. This step ensures the integrity of the planned fabric extension.

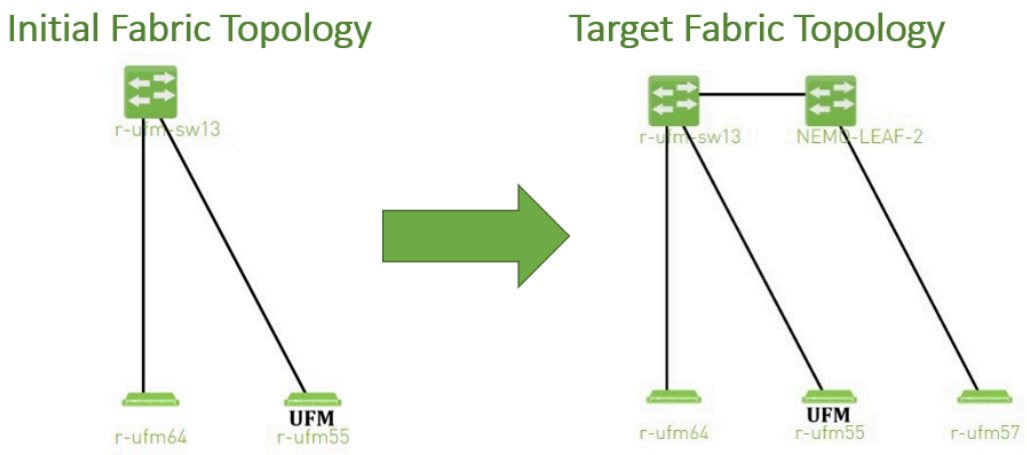
1. **Topology Extension Preparation:** Upon successful verification of the NDT topology file, the tool generates a comprehensive verification report. The network administrator reviews this report and ensures its acceptability.

1. **Topoconfig File Generation:** After obtaining acceptable verification results, the tool generates a topoconfig file. This file serves as input for OpenSM, the Subnet Manager for InfiniBand fabrics. The topoconfig file allows the network administrator to define the desired physical port states for designated boundary ports. These states include "disabled" or "no-discover."

1. **Fabric Extension and Verification:** With the topoconfig file prepared, the Subnet Merger tool initiates the deployment of the extended fabric configuration. The tool ensures that the defined physical port states are implemented. Once the extension is in place, the IB network can be extended further as needed. The fabric extension is executed while maintaining the operational stability of the existing network.

1. **Conclusion:** The Subnet Merger tool offers a reliable and user-friendly solution for expanding InfiniBand fabrics using NDT topology files. By following the steps provided in the intuitive UI wizard, system administrators can seamlessly extend the fabric while adhering to predefined physical port states. This tool ensures the smooth operation of the fabric throughout the expansion process, eliminating disruptions and enhancing network scalability.

Subnet Merger Flow



1. Create NDT, file that describes initial topology with definition of boundary ports. Boundary ports – switch ports that will be used for fabric extension. In our case it will be r-ufm-sw13 switch ports number 1 and 3. In NDT file those ports should be defined as boundary and disabled:

```
rack #,U
height,#Fields:StartDevice,StartPort,StartDeviceLocation,EndDe
height_1,LinkType,Speed,_2,Cable
Length,_3,_4,_5,_6,_7,State,Domain
,,MF0;r-ufm-sw13:MQM8700/U1,Port
1,,,,,,,,,,,,,,,Disabled,Boundary
,,MF0;r-ufm-sw13:MQM8700/U1,Port 30,,r-ufm55 mlx5_1,Port
1,,,,,,,,,,,,Active,In-Scope
,,MF0;r-ufm-sw13:MQM8700/U1,Port 29,,r-ufm55 mlx5_0,Port
1,,,,,,,,,,,,Active,In-Scope
,,MF0;r-ufm-sw13:MQM8700/U1,Port 26,,r-ufm64 mlx5_0,Port
1,,,,,,,,,,,,Active,In-Scope
,,MF0;r-ufm-sw13:MQM8700/U1,Port
3,,,,,,,,,,,,,,,Disabled,Boundary
```

2. Upload a new NDT topology file which describes the desired topology. Before deploying to UFM, the new NDT topology file should be verified against the existing topology – to find out mismatches and problems.

   After the verification, the plugin generates reports including information about:

-
    - Duplicated GUIDs
    - Misswired links
    - Non-existent links in the pre-defined NDT files

1.
    - Links that exist in the fabric and not in the NDT file

2. Following the issues detected in the plugin reports, the network administrator changes the NDT file or the fabric. The verification process can be repeated as many times as necessary until the network administrator is satisfied with the results.

3. If the NDT verification results are satisfactory, a topoconfig file is generated and can be deployed to the UFM server to be used as configuration input for OpenSM. Topoconfig file should be located at /opt/ufm/files/conf/opensm/topoconfig.cfg on UFM server. By sending SIGHUP signal to opensm it forced to read configuration and to deploy it. In topoconfig file at this stage boundary ports will be defined as **Disabled**.

   Example of topoconfig.cfg:

```
0xb83fd2030080302e,1,-,-,Any, Disabled
0xb83fd2030080302e,30,0xf452140300280081,1,Any,Active
0xb83fd2030080302e,29,0xf452140300280080,1,Any,Active
0xb83fd2030080302e,26,0xf452140300280040,1,Any,Active
0xb83fd2030080302e,3,-,-,Any, Disabled
```

4. Next stage is to extend the fabric. Prepare separately new subnet that will be added to the existing fabric and, once it is ready, connect to the boundary ports, that are defined as Disabled in configuration file, so newly added subnet will not be discovered by opensm and will not affect in any way current setup functionality.

5. Once new subnet connected to the fabric - prepare next NDT file, that contains setup, that describes current fabric with extended, when previously defined as boundary ports defined as Active and if planned to continue with extension new ports defined as boundary.

For example port number 9 of switch r-ufm-sw13:

```
rack #,U
height,#Fields:StartDevice,StartPort,StartDeviceLocation,EndDe
height_1,LinkType,Speed,_2,Cable
Length,_3,_4,_5,_6,_7,State,Domain
,,MF0;r-ufm-sw13:MQM8700/U1,Port 1,,NEMO-LEAF-2,Port
1,,,,,,,,,,,Active,In-Scope
,,MF0;r-ufm-sw13:MQM8700/U1,Port 30,,r-ufm55 mlx5_1,Port
1,,,,,,,,,,,Active,In-Scope
,,MF0;r-ufm-sw13:MQM8700/U1,Port 29,,r-ufm55 mlx5_0,Port
1,,,,,,,,,,,Active,In-Scope
,,NEMO-LEAF-2,Port 11,,r-ufm57 mlx5_0,Port
1,,,,,,,,,,,Active,In-Scope
,,MF0;r-ufm-sw13:MQM8700/U1,Port 26,,r-ufm64 mlx5_0,Port
1,,,,,,,,,,,Active,In-Scope
,,NEMO-LEAF-2,Port 1,,MF0;r-ufm-sw13,Port
1,,,,,,,,,,,Active,In-Scope
,,MF0;r-ufm-sw13:MQM8700/U1,Port 3,,NEMO-LEAF-2,Port
3,,,,,,,,,,,Active,In-Scope
,,NEMO-LEAF-2,Port 3,,MF0;r-ufm-sw13,Port
3,,,,,,,,,,,Active,In-Scope
,,MF0;r-ufm-sw13:MQM8700/U1,Port
9,,,,,,,,,,,,,Disabled,Boundary
```

6. After new subnet connected physically to the fabric, in opensm configuration file (topoconfig.cfg) boundary ports previously defined as Disabled should be set as No-discover. Example:

```
0xb83fd2030080302e,1,-,-,Any,No-discover
0xb83fd2030080302e,30,0xf452140300280081,1,Any,Active
0xb83fd2030080302e,29,0xf452140300280080,1,Any,Active
0xb83fd2030080302e,26,0xf452140300280040,1,Any,Active
0xb83fd2030080302e,3,-,-,Any,No-discover
```

7. Updated file should be deployed to UFM. In case boundary ports will be defined as No-discover – fabric, connected beyond those ports will not be discovered by opensm, but all the ibutils (ibdiagnet...) could send mads beyond those ports to newly added subnet - so NDT file verification for extended setup could be performed.

8. Upload new NDT file and run verification for this file. Fix problems detected by verification. Once satisfied with results – deploy configuration to UFM.

   Example of topoconfig file for extended setup:

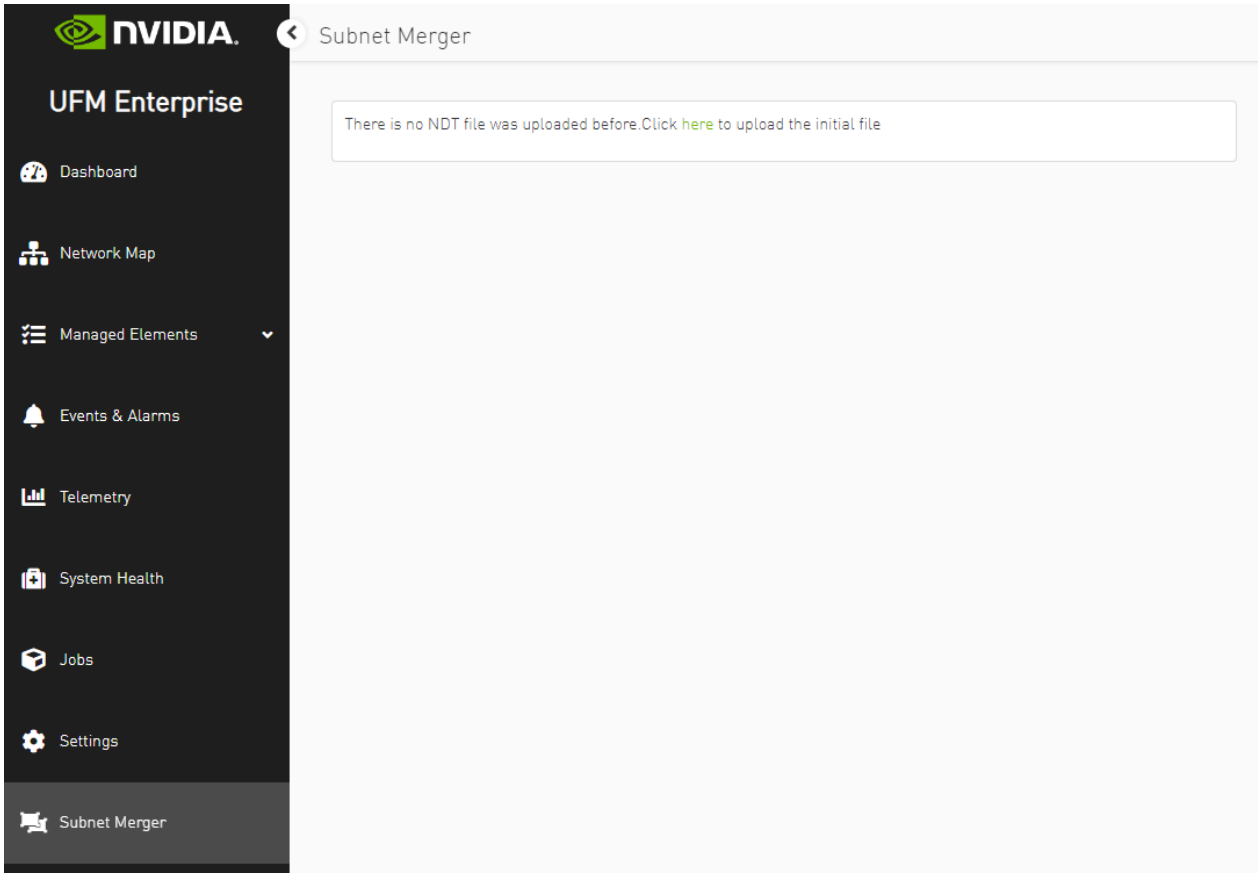
```
0xb83fd2030080302e,1,0x98039b0300867bba,1,Any,Active
0xb83fd2030080302e,30,0xf452140300280081,1,Any,Active
0xb83fd2030080302e,29,0xf452140300280080,1,Any,Active
0x98039b0300867bba,11,0x248a0703009c0066,1,Any,Active
0xb83fd2030080302e,26,0xf452140300280040,1,Any,Active
0x98039b0300867bba,1,0xb83fd2030080302e,1,Any,Active
0xb83fd2030080302e,3,0x98039b0300867bba,3,Any,Active
0x98039b0300867bba,3,0xb83fd2030080302e,3,Any,Active
0xb83fd2030080302e,9,-,-,Any,Disabled
```

9. Repeat previous steps if need to perform additional setup extension.

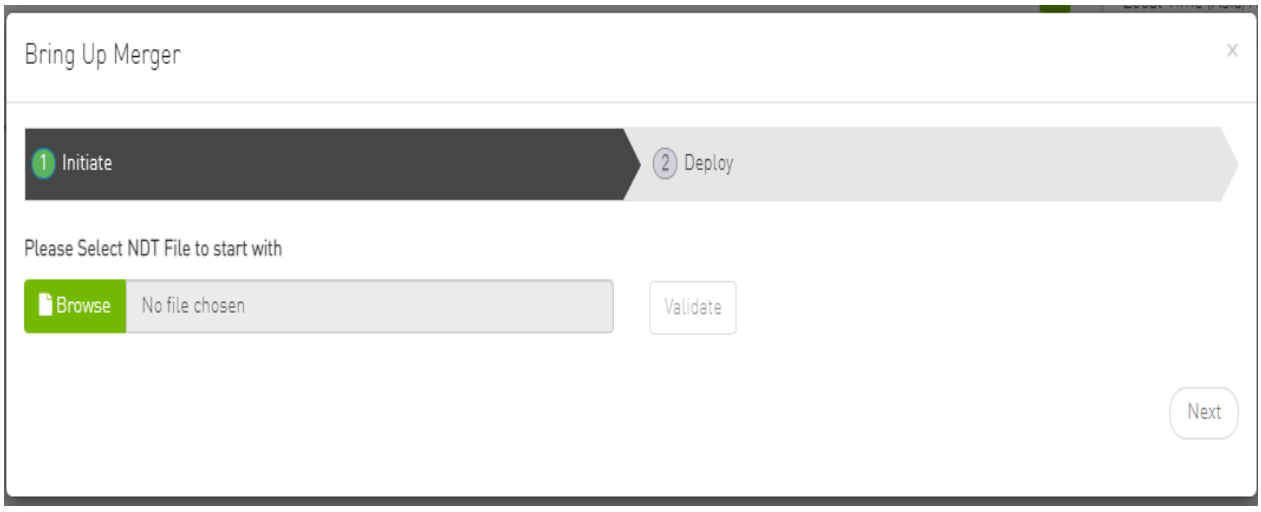
Subnet Merger UI

Bring-Up Merger Wizard

1. Add the NDT plugin to UFM by loading the plugin's image through Settings->Plugins Management. A new item will appear in the main left navigator menu of the UFM labeled "Subnet Merger".

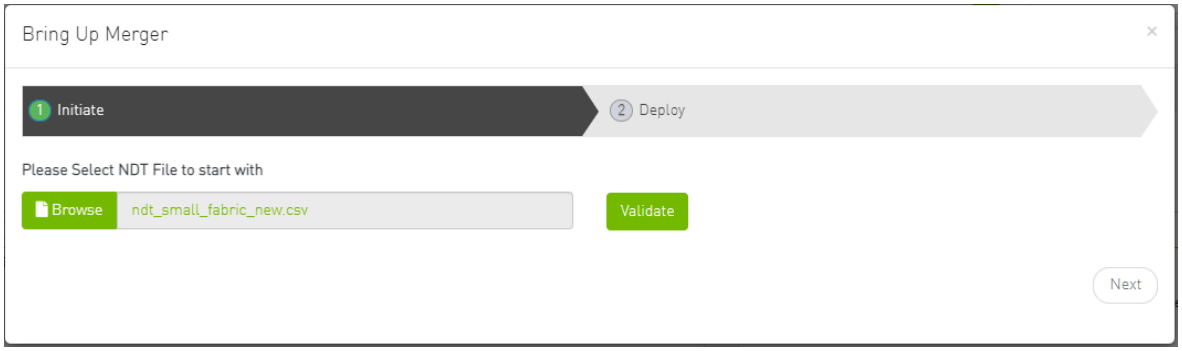


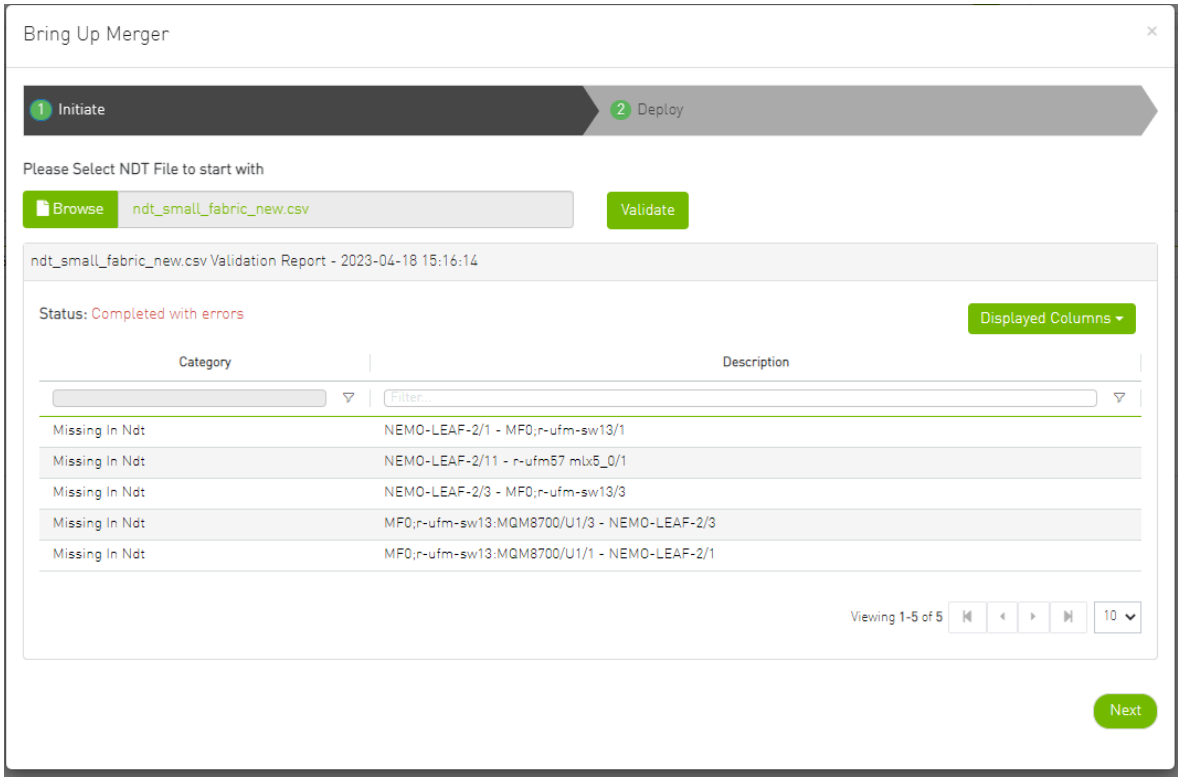
2. Access "Subnet Merger" to initiate the bring-up wizard.



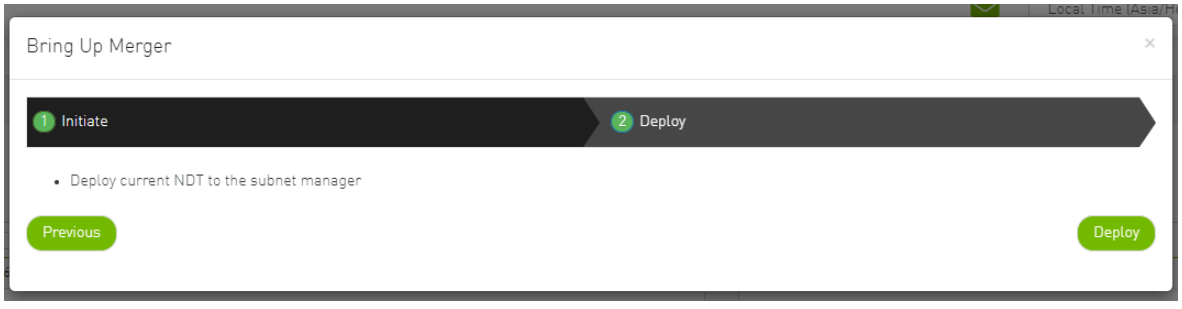
3. The wizard will guide you through the process, containing the following steps:

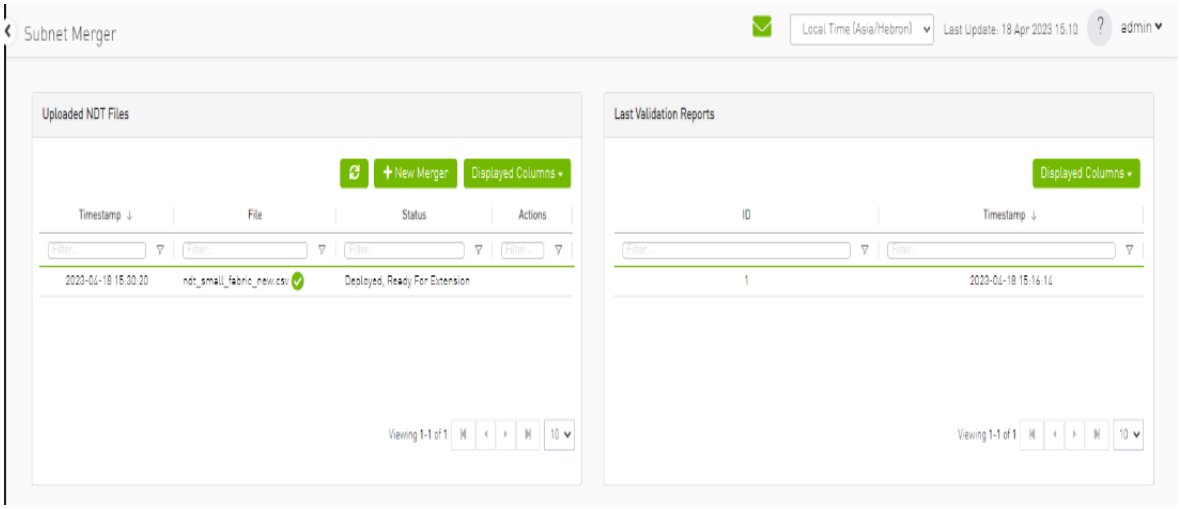
    1. Upload the initial NDT tab and validate it.





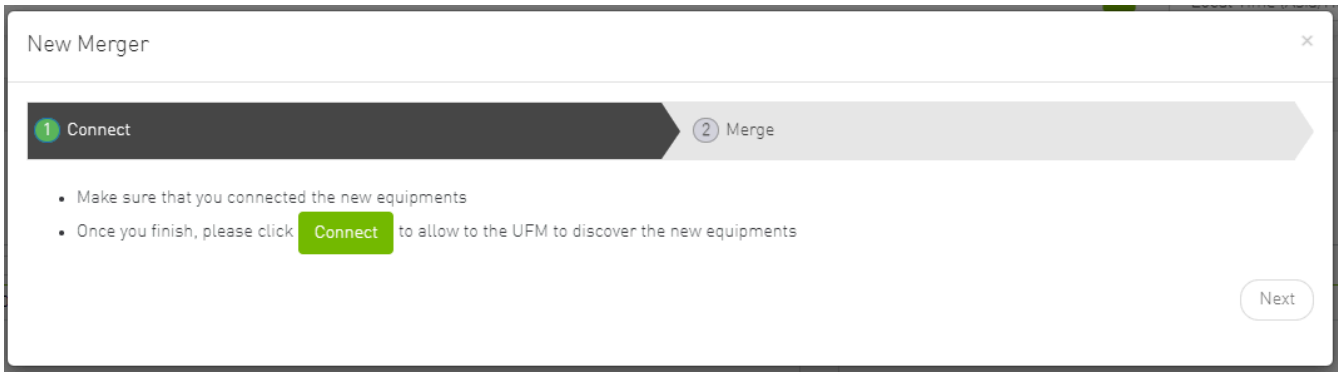
2. Once you are satisfied with the results of the validation in the previous tab, you can proceed to deploy the file.



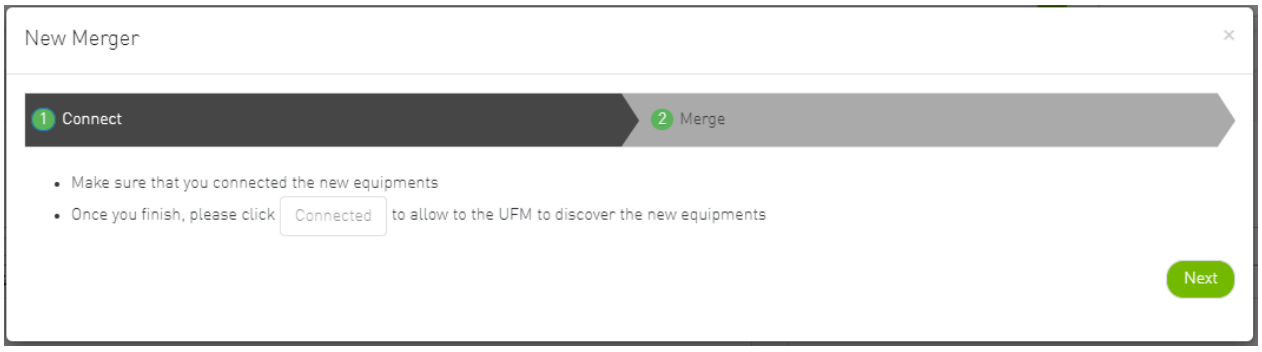


New Subnet Merger

Once you have successfully deployed the initial NDT file, you can initiate a new merger process by clicking the "New Merger" button.

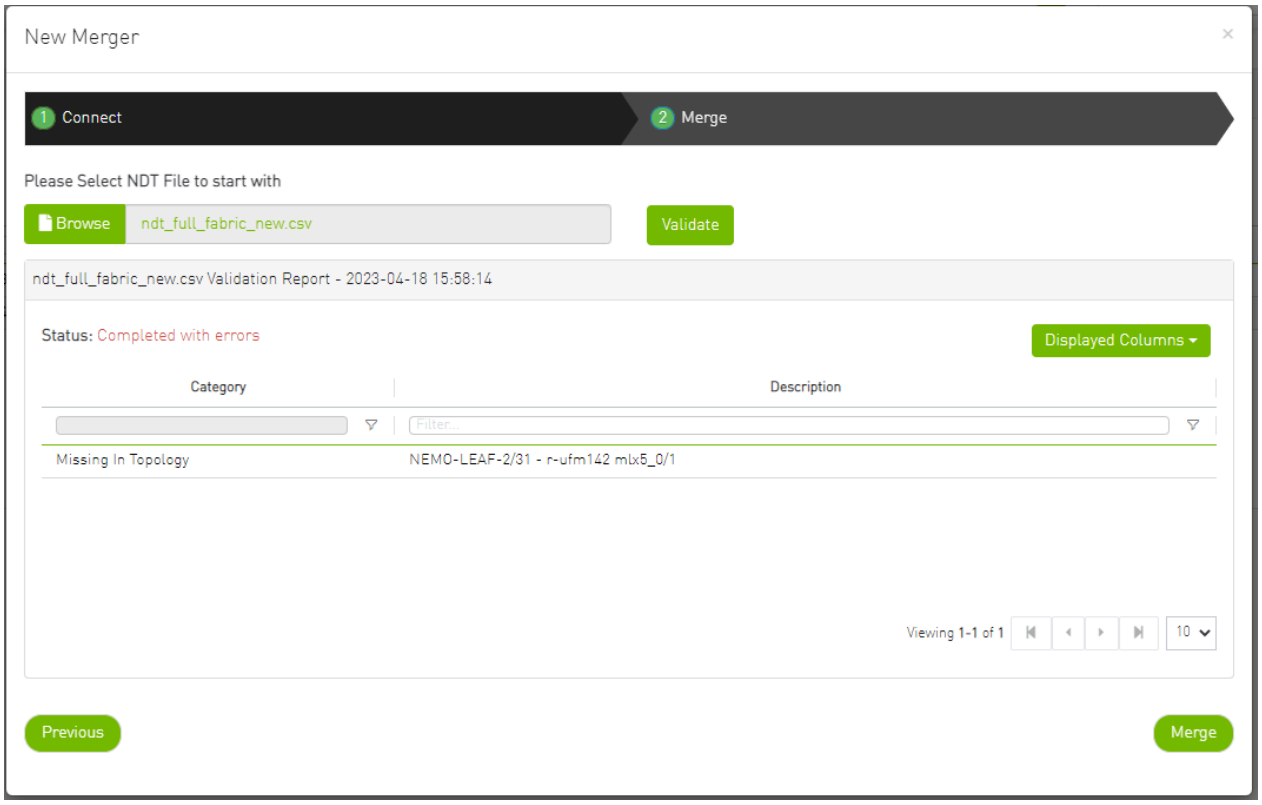


1. "Connect" Tab, it is important to physically connect the new equipment and confirm the connection. Then, click on a button which will open the boundary ports, change their state from Disabled to No-discover, and then deploy the active file again.

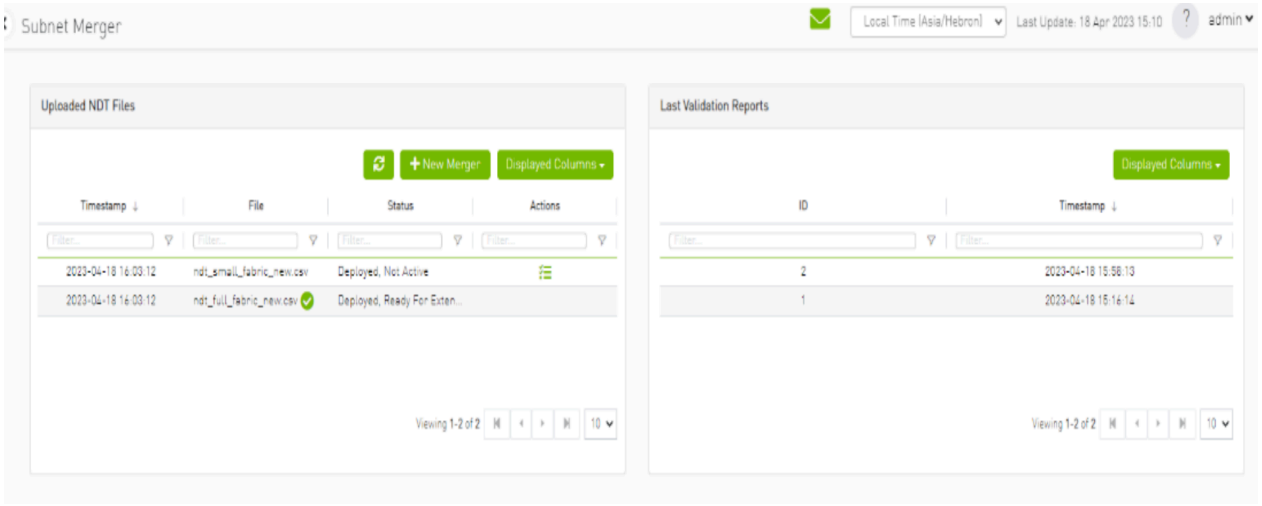


2. "Merge" Tab: Once the new equipment is connected and the boundary ports are updated, upload a new NDT file that includes both the current and newly added

equipment, along with their boundary ports for future merges. Please note that you cannot merge the file if there are duplicate GUIDs in the report's results.



3. After completing the merge wizard, and if necessary, you can further proceed to extend the IB fabric.



Extending the InfiniBand Setup via Subnet Merger

The following instructions outline the necessary steps for expanding the InfiniBand setup or fabric using subnet merging.

1. **Step 1: NDT File Upload (Repeatable)**

   Upload the NDT file, performing this action as many times as required, especially when addressing file-related issues.

2. **Step 2: NDT File Validation and Verification (Repeatable)**

   Validate the NDT file, a process that can be repeated multiple times, particularly after fixing fabric topology or NDT file errors. After initiating this call, you will obtain a validation report ID. The progress of this process is asynchronous, with the report's status initially indicated as "running." Once the report is completed, the status will change to either "Successfully completed" or "Completed with errors."

3. **Step 3: Retrieving and Monitoring the Validation Report**

   Retrieve the validation report by its corresponding ID, running this step through continuous polling until the report reaches completion.

4. **Step 4: Review and Potential Fixes**

   Inspect the report and address any necessary fixes to either the NDT file or the topology. Should changes be made to the file, upload the corrected NDT file anew. Alternatively, in case of topology has changed, repeat the verification process.

5. **Step 5: Topology Deployment to UFM**

   Deploy the verified topology to UFM once you are satisfied with the verification outcomes.

6. **Step 6: Adjusting Boundary Ports and Deployment**

   Following the physical connection of the setup extension, change the boundary ports' state from "Disabled" to "No-discover."

7. **Step 7: Uploading Updated Topoconfig File**

   Deploy the updated topoconfig file to the UFM server.

8. **Step 8: Next NDT File Upload (Combined Fabric and Extension)**

   Upload the next NDT file, which consolidates the current fabric and extension components.

9. **Step 9: NDT File Verification**

   Conduct the NDT file verification process.

10. **Step 10: Reviewing Verification Report**

    Review the verification report.

11. **Step 11: Addressing Setup or NDT File Issues**

    If necessary, make necessary adjustments to the setup or NDT file.

12. **Step 12: Final Configuration Deployment**

    Once content with the modifications, proceed to deploy the configuration to UFM.

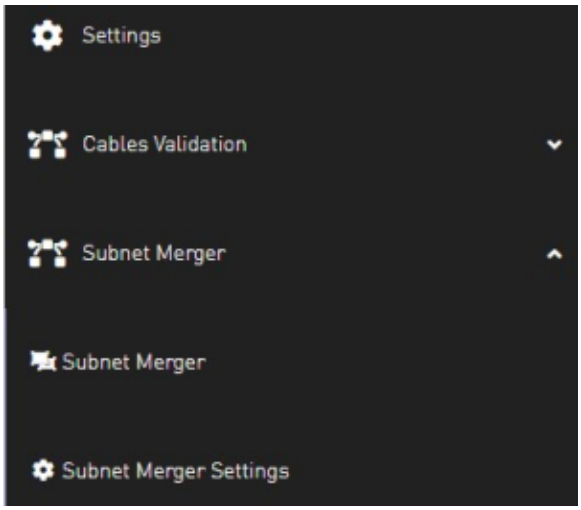
13. **Step 13: Iterative Workflow**

    Repeat this flow as many times as needed to further the expansion process.

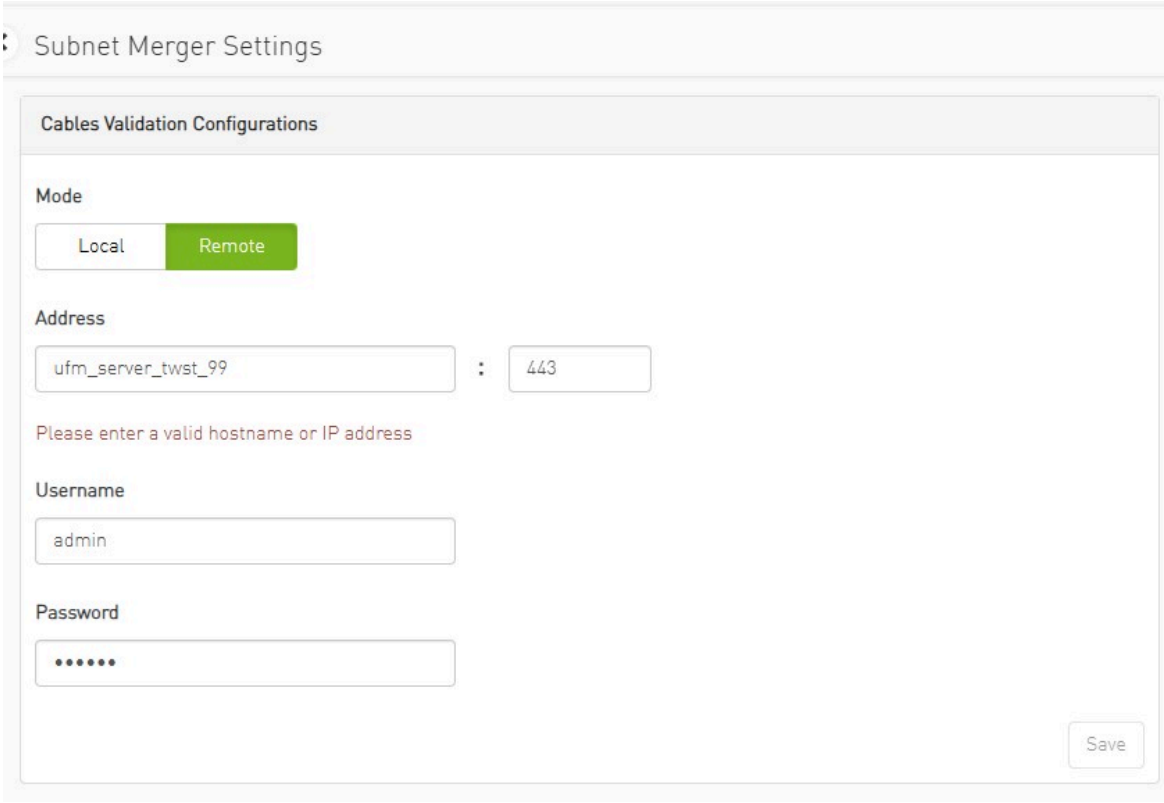
View Cable Validation Report, using NDR Merger Plugin UI

It is an option to view cable validation report (created by Cable Validation plugin) using NDT Merger UI.

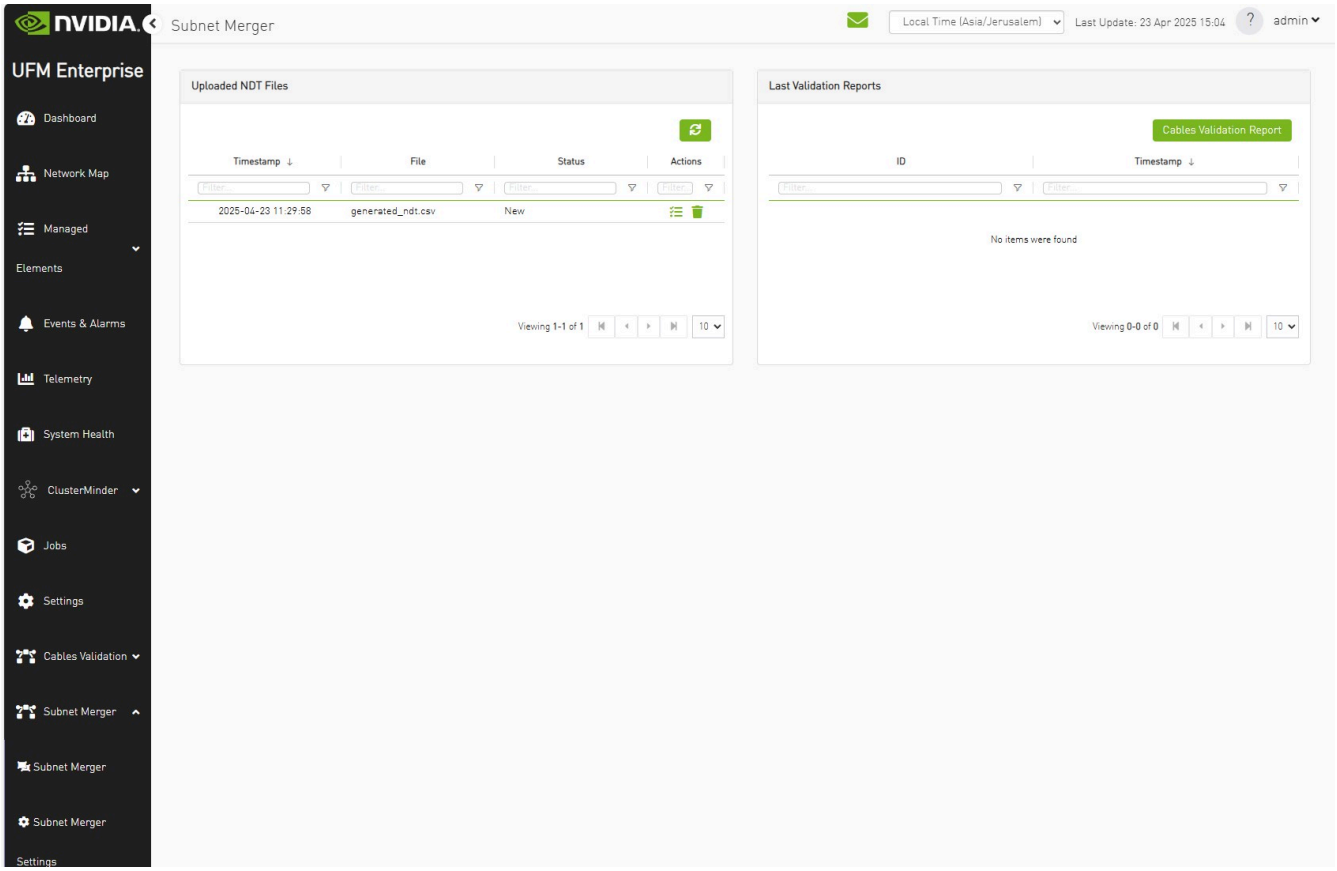
1. Define connectivity parameters to server, running Cable Validation Plugin.



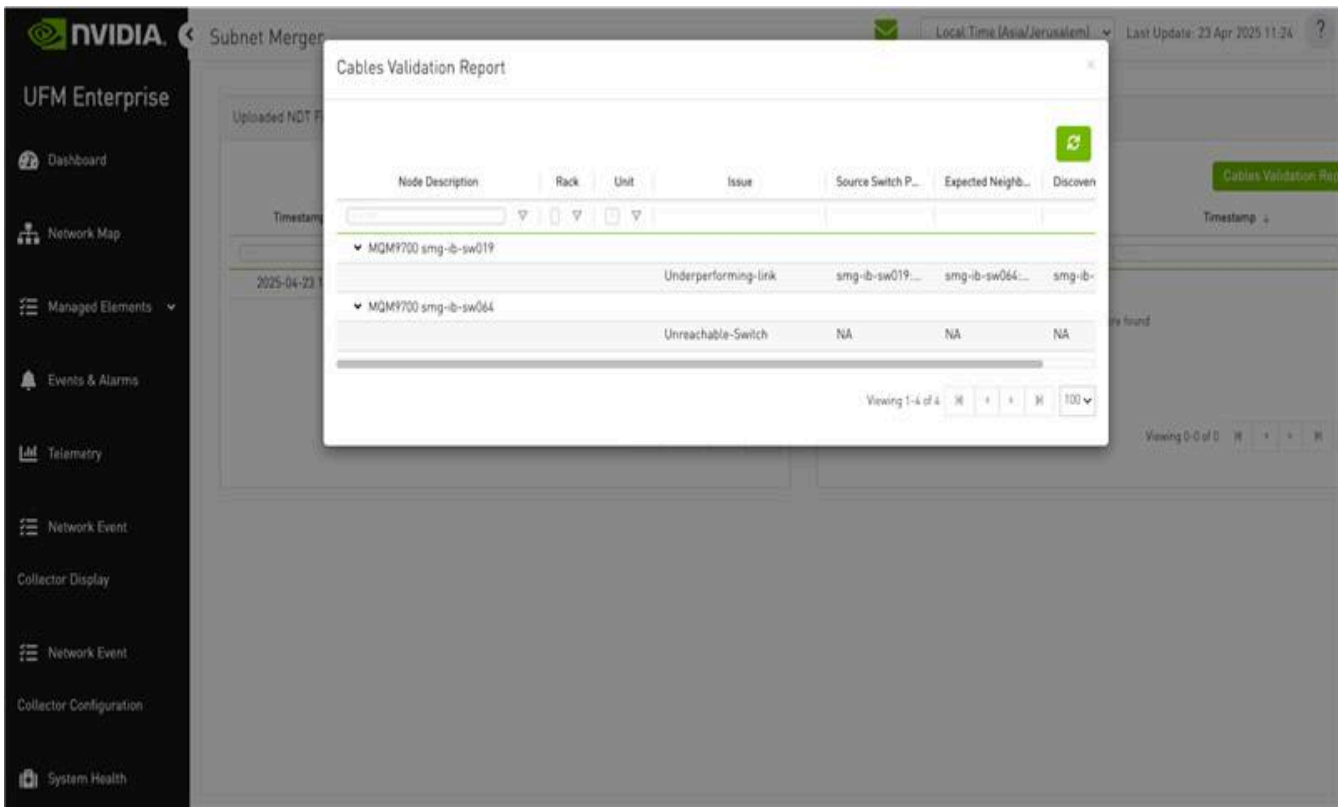
Click on Subnet Merger Settings to access Cable Validation plugin connectivity configuration window and enter parameters.



2. Using NDT Merger main screen request Cable Validation Report by clicking on Cable Validation Report button



3. View the report



UFM Telemetry Fluentd Streaming (TFS) Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.1.1-1	N/A

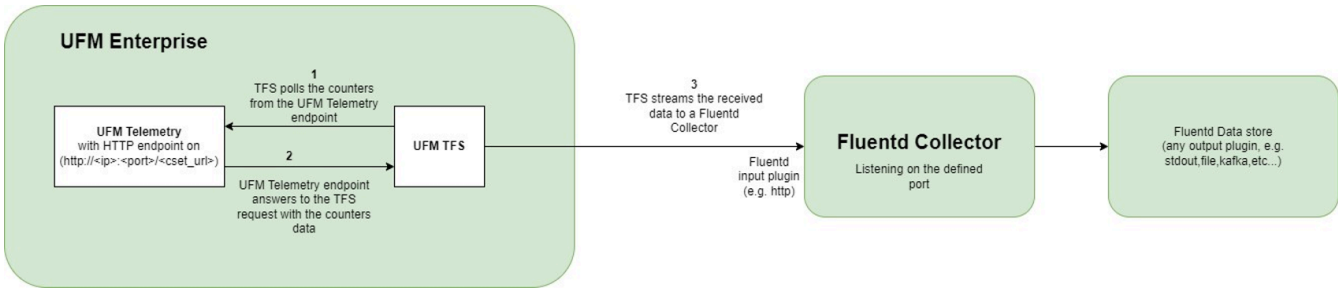
Bug Fixes

Plugin Version	Bug Fix
1.1.1-1	N/A

Overview

The TFS plugin is designed to extract UFM Telemetry counters from specified telemetry HTTP endpoints and stream them to the designated Fluentd collector destination.

The diagram below illustrates the plugin's stream flow:



Deployment

The following are the possible ways the TFS plugin can be deployed:

1. On UFM Appliance
2. On UFM Software

For complete instructions on deploying the TFS plugin, refer to UFM Telemetry endpoint stream To Fluentd endpoint (TFS).

Authentication

The following authentication types are supported:

- basic (/ufmRest)
- client (/ufmRestV2**)**
- token (/ufmRestV3)

Rest API

The following REST APIs are supported:

- POST /plugin/tfs/conf
- GET /plugin/tfs/conf
- POST /plugin/tfs/conf/attributes
- GET /plugin/tfs/conf/attributes

For detailed information on interacting with TFS plugin, refer to the NVIDIA UFM Enterprise > Rest API > TFS Plugin REST API.

UFM Events Fluent Streaming (EFS) Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.0.0-6	N/A

Bug Fixes

Plugin Version	Bug Fix
1.0.0-6	N/A

Overview

EFS plugin is a self-contained Docker container with REST API support managed by UFM. EFS plugin extracts the UFM events from UFM Syslog and streams them to a remote FluentD destination. It also has the option to duplicate current UFM Syslog messages and forward them to a remote Syslog destination. As a fabric manager, it will be useful to collect the UFM Enterprise events/logs, stream them to the destination endpoint and monitor them.

Deployment

The following are the ways EFS plugin can be deployed:

1. On UFM Appliance
2. On UFM Software

For detailed instructions on how to deploy EFS plugin, refer to UFM Event Stream to FluentBit endpoint (EFS).

Authentication

The following authentication types are supported:

- basic (/ufmRest)
- client (/ufmRestV2)

- token (/ufmRestV3)

Rest API

The following REST APIs are supported:

- PUT /plugin/efs/conf
- GET /plugin/efs/conf

For detailed information on how to interact with EFS plugin, refer to the NVIDIA UFM Enterprise > Rest API > EFS Plugin REST API.

UFM Bright Cluster Integration Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.0.0-4	N/A

Bug Fixes

Plugin Version	Bug Fix
1.0.0-4	N/A

Overview

The Bright Cluster Integration plugin is a self-contained docker container managed by UFM and is managed by the REST APIs. It enables integrating data from Bright Cluster Manager (BCM) into UFM, providing a more comprehensive network perspective. This integration improves network-centered Root Cause Analysis (RCA) tasks and enables better scoping of workload failure domains.

Deployment

The Bright Cluster Integration plugin can be deployed either on the UFM Appliance or on UFM Software.

For detailed instructions on Bright Cluster Integration plugin deployment, refer to UFM Bright Cluster Integration Plugin.

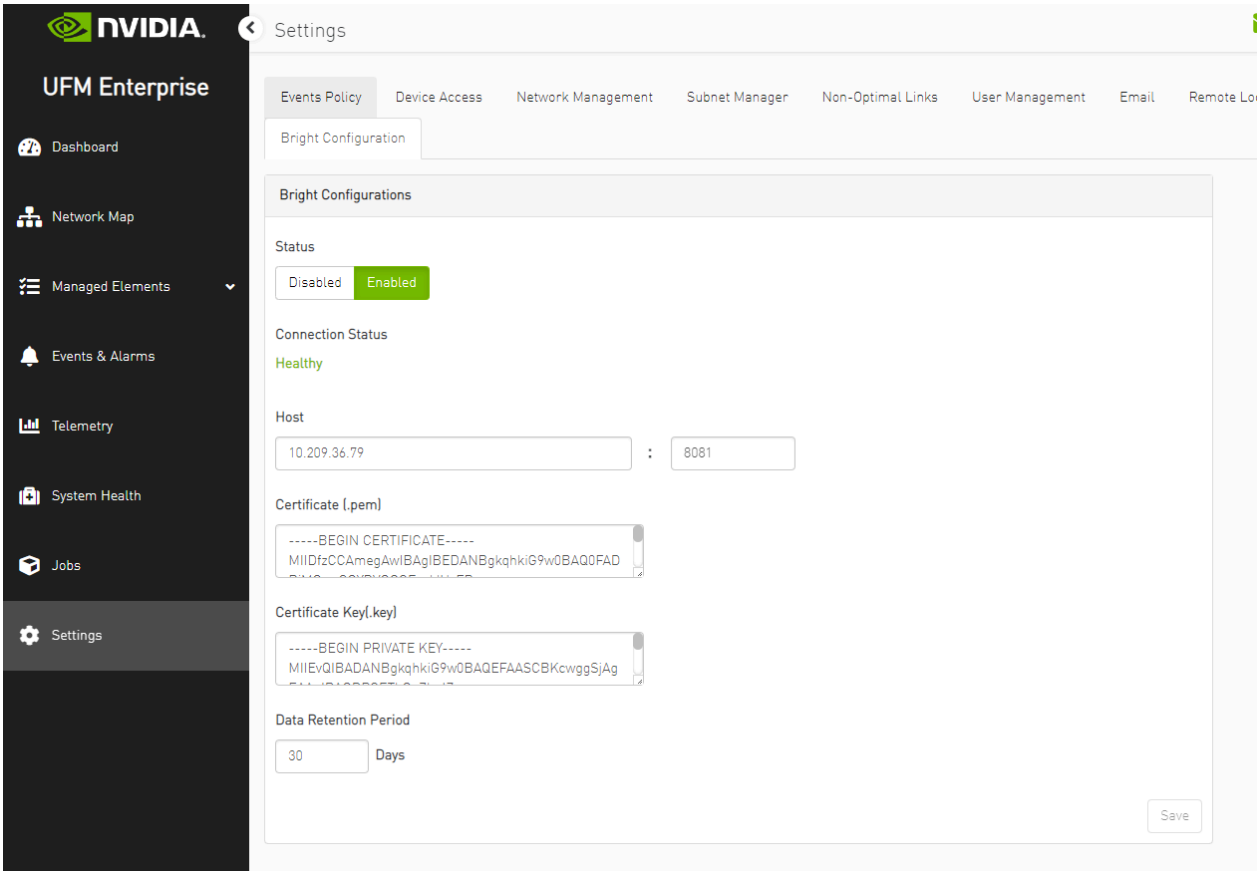
Authentication

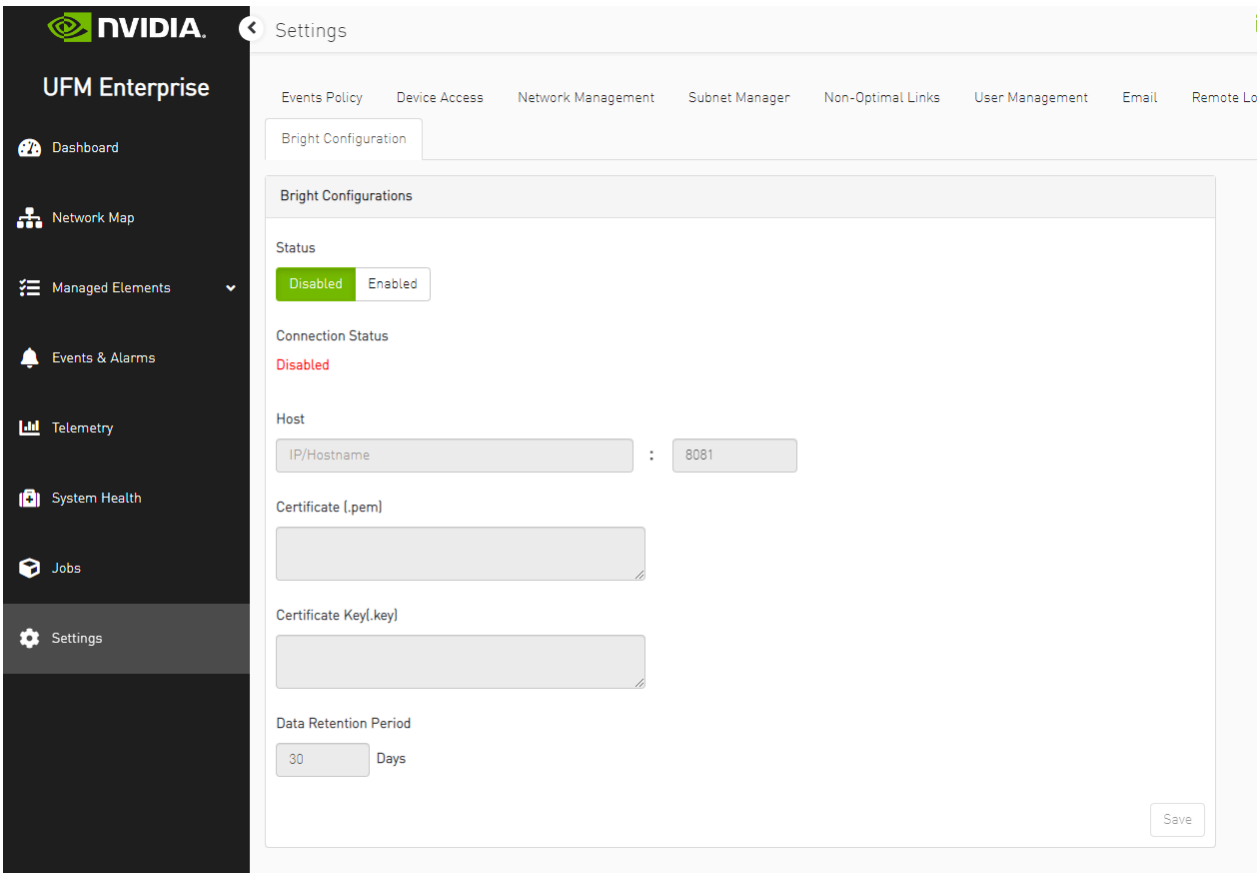
The following authentication types are supported:

- basic (/ufmRest)
- client (/ufmRestV2**)**
- token (/ufmRestV3)

Bright Cluster Integration UI

1. After the successful deployment of the plugin, a new tab is shown under the UFM settings section for bright configurations management:

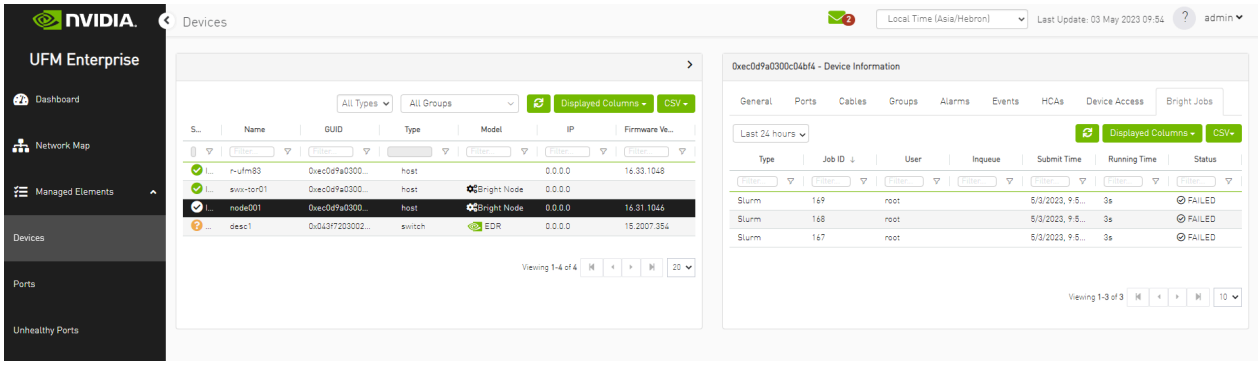

NVIDIA
UFM Enterprise
Dashboard
Network Map
Managed Elements
Events & Alarms
Telemetry
System Health
Jobs
Settings
Settings
Events Policy
Device Access
Network Management
Subnet Manager
Non-Optimal Links
User Management
Email
Bright Configuration
Bright Configurations
Status
Disabled
Enabled
Connection Status
Healthy
Host
10.209.36.79
8081
Certificate (.pem)
-----BEGIN CERTIFICATE-----
MIIDfzCCAmegAwIBAgIBEDANBgkqhkiG9w0BAQ0FAD
Certificate Key(.key)
-----BEGIN PRIVATE KEY-----
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAg
Data Retention Period
30
Days
Save


NVIDIA
UFM Enterprise
Dashboard
Network Map
Managed Elements
Events & Alarms
Telemetry
System Health
Jobs
Settings
Settings
Events Policy
Device Access
Network Management
Subnet Manager
Non-Optimal Links
User Management
Email
Bright Configuration
Bright Configurations
Status
Disabled
Enabled
Connection Status
Disabled
Host
IP/Hostname
8081
Certificate (.pem)
Certificate Key(.key)
Data Retention Period
30
Days
Save

Fill the below required configurations:

Parameter	Description
Host	Hostname or IP of the BCM server
Port	Port of the BCM server, is typically 8081
Certificate	BMC client certificate content that could be located in the BMC server machine under `.cm/XXX.pem`
Certificate key	BMC client certificate key that could be located in the BMC server machine under .`cm/XXX.key`
Data retention period	UFM erases the data gathered in the database after the configured retention period. By default, after 30 days.

2. After you ensure you have successfully completed the plugin configuration, and that you have established a healthy connection with the BMC, navigate to the UFM Web GUI -> Managed Elements -> Devices



Rest API

The following REST APIs are supported:

- PUT plugin/bright/conf
- GET plugin/bright/conf
- GET plugin/bright/data/nodes
- GET plugin/bright/data/jobs

For detailed information on how to interact with bright plugin APIs, refer to NVIDIA UFM Enterprise > Rest API > UFM Bright Cluster Integration Plugin REST API.

IB Link Resiliency Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.1.5-7	• The BER detection system monitors sampling windows and classifies each window duration as good (⩽1 marked sample), warning (2-3 marked samples), or error (⩾4 marked samples or 2 consecutive warning windows). Once an error state is triggered, the link must complete a full good window duration before returning to normal state. (Failure Detection for XDR Ports). • Extended event messages for isolated and deisolated ports with detailed isolation reason.

Bug Fixes

Plugin Version	Bug Fix
1.1.5-7	• Fixed an issue where isolation failed if at least one of the specified isolating ports was not found in the fabric, resulting in the warning *"the ports ... were not isolated."* • Fixed a configuration issue where the Supervisor `unix_http_server` could run without HTTP authentication. • Added validation for the flapping link tier threshold to prevent invalid configurations.

Overview

The primary objective of the IB Link Resiliency (IBLR) plugin is to enhance cluster availability and improve the rate of job completion.

This objective is accomplished by combining different mechanisms, both ML-based and rule-based, for identifying problematic links. Then, the plugin autonomously applies corrective measures to these links with the aim of restoring their normal function.

For cluster topologies where no redundancy exists at the level of access links, the plugin will only execute a mitigation procedure for trunk links.

The IBLR plugin execution cycle is composed of the following tasks:

1. Collects telemetry data from the UFM secondary telemetry service, which by default collects telemetry data every 5 minutes.

2. Employs ML-based prediction models and rule-based detection logic to alert on problematic ports.

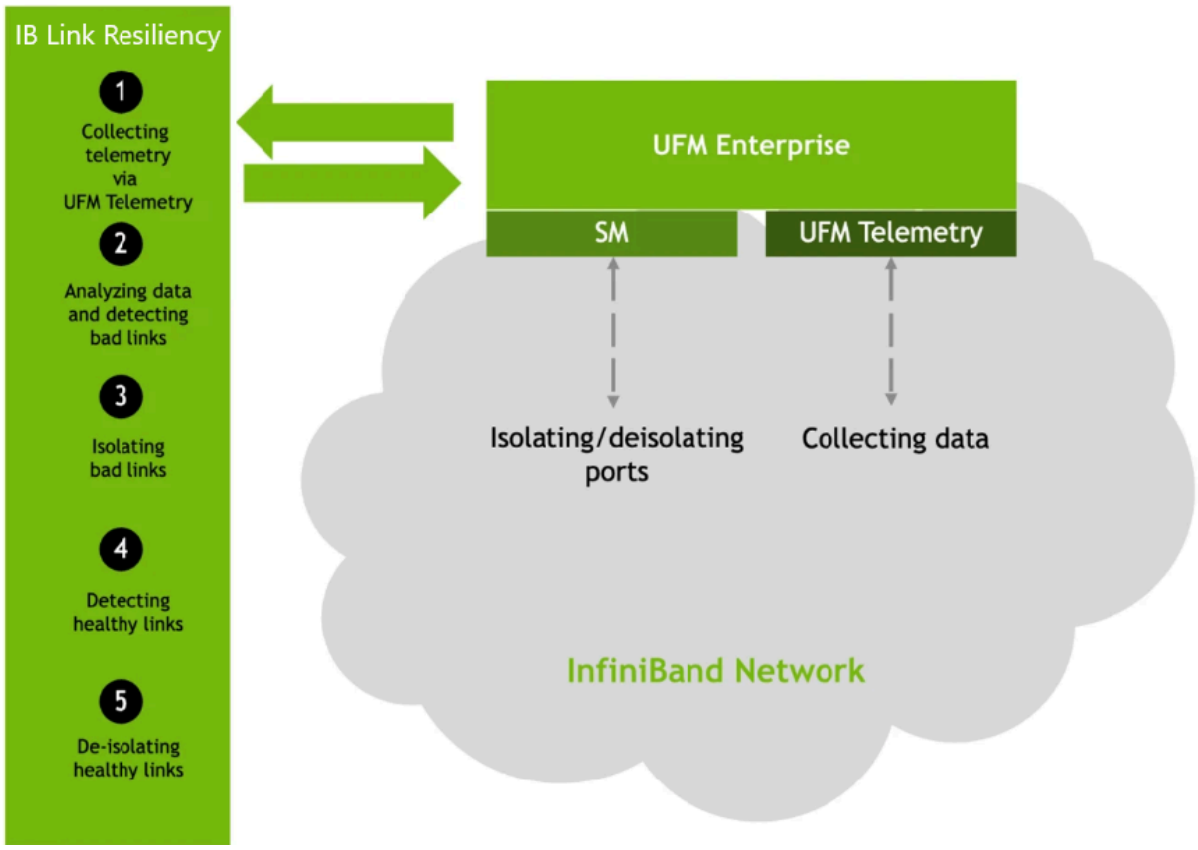
3. Based on the alerts issued by the different alert engines, determines which measures should be taken:

    1. Which links are underperforming and should be isolated from the fabric.

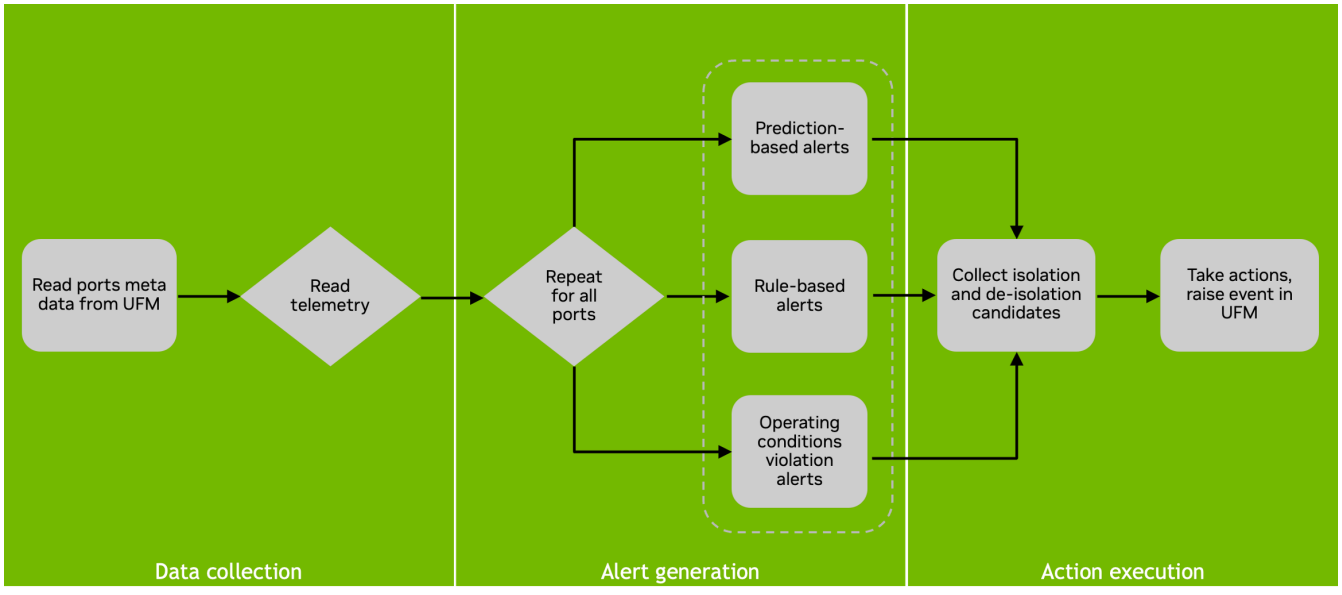
    2. Which isolated links have recovered following the mitigation process, and hence should be reinstated to the fabric.

4. Applies the required actions through the UFM and reports to the UFM events table.

Schematic Flow: External View



Schematic Flow: Internal View



Data Collection

The plugin collects its telemetry data from the secondary telemetry endpoint, a low-frequency UFM Telemetry service which collects a large set of counters. Secondary telemetry is enabled by default, and in case it is being de-activated by the user, the plugin will not be able to operate.

In addition to telemetry data, the plugin periodically reads from the UFM the cluster topology information in order to align its internal logic with any changes that has taken place in the cluster.

Main Collected Counters

Name	Description
`hist0-hist15`	FEC histogram counters. Counter hist{i} is incremented by one every time a FEC block is arrives with i bit errors.
`phy_effective_errors`	Number of FEC blocks that could not be corrected (8 or more errors for FEC8 algorithm).
`phy_symbol_errors`	Number of symbols dropped due to errors.
`PortRcvErrorsExtended`	Number of data blocks dropped due to errors.
`PortRcvDataExtended`	Number of received data blocks.
`CableInfo.Temperature`	Module temperature.
`CableInfo.diag_supply_voltage`	Module voltage.
`link_down_events`	Number of times the link was down.
`Symbol_BER`	BER after all phy correction mechanism.

Alert Generation

The alert generation block is composed of four independent engines, each employing either ML-based or rule-based logic in order to identify problematic links:

1. Failure prediction

2. Failure detection

3. Link flaps detection

4. Operating conditions violation

Per each alert engine, each port can be configured to operate in one of two modes:

- Shadow mode: the plugin will not act upon this link based on alerts from this engine.
- Active mode: the plugin is allowed to automatically execute mitigation steps upon this link in case an alert was raised from this engine.

Alert Engine Operation Mode Configurations

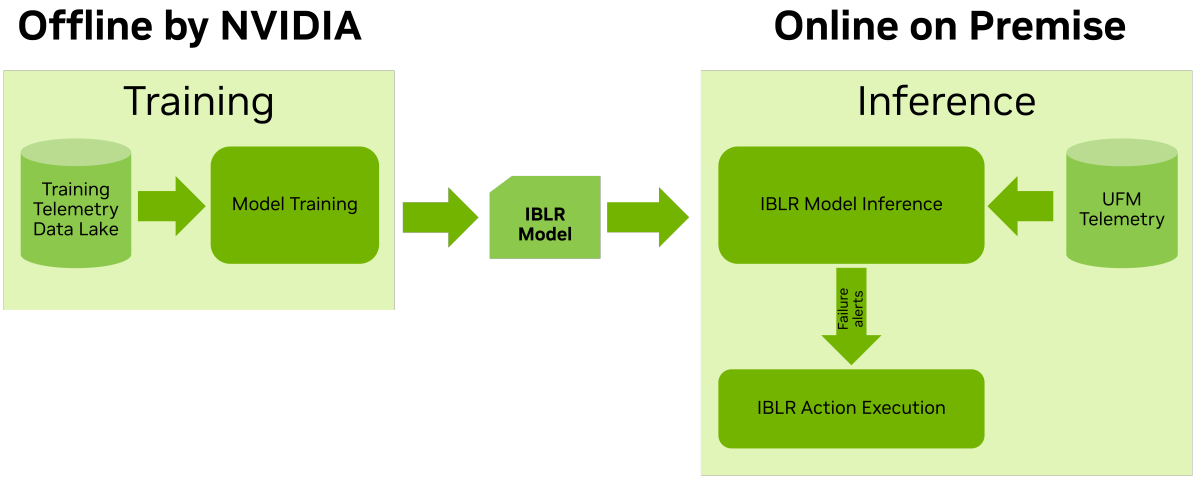
Name	Type	Values	Default Value	Description
`mode`	str	shadow, active	**shadow**	This configuration assigns the default operation mode for the respective alert engine.
`except_list`	List[str]	List of port identifiers (node guid and port number)	empty	Ports included in the exception list are operating in the mode opposite to that indicated in the `mode` configuration field. This enables to define a subset of the ports that will operate differently from the remainder of the cluster.

Failure Prediction

The port failure prediction logic is based on a binary classification ML model, trained to predict ahead of time cases where physical layer issues will eventually result in data loss.

The model relies mainly on the error histogram counters of the Forward Error Correction (FEC) block.

The model was validated to provide valuable predictions on both HDR and NDR generation clusters.



Failure Prediction Configurations

Name	Type	Values	Default Value	Description
`model_threshold`	float	(0, 1)	**0.8**	The decision threshold of the binary classification model. If the model's output probability exceeds this threshold an alert is triggered. The recommended value range is 0.65-0.8, with lower values favoring high recall and higher values favoring high precision.

Failure Detection

The failure detection module is comprised of two distinct rules, aimed at detecting based on post-FEC errors when a link is not meeting the desired performance specifications:

1. Packet drop rate: calculates the rate of dropped packets (local and remote errors) vs. successfully received packets.

2. Symbol errors: calculates the rate of symbol errors during two different time intervals (12 minutes and 125 minutes). If the threshold is exceeded for either interval, the rule triggers the failure detection alert.

   This rule is only supported in NDR clusters, and is ignored in HDR clusters.

In case either of the rules exceeds its respective threshold, a failure detection alert will be raised.

Failure Detection Configurations

Name	Type	Default Value	Description
`detect_pdr_rule_enabled`	bool	**True**	Flag indicating whether the packet drop rate rule is enabled. If disabled, alerts will not be generated based on this rule.
`detect_symbol_error_rule_enabled`	bool	**True**	Flag indicating whether the symbol errors rule is enabled. If disabled, alerts will not be generated based on this rule.
`detect_pdr_error_rate_threshold`	float	**1e-12**	(Legacy ports) The threshold for the packet drop rate rule.

Name	Type	Default Value	Description
`detect_symbol_error_short_interval_threshold`	float	**2.88**	(Legacy ports) The threshold for the symbol errors rule, 12 minutes interval.
`detect_symbol_error_long_interval_threshold`	float	**3**	(Legacy ports) The threshold for the symbol errors rule, 125 minutes interval.

Failure Detection of XDR Ports

The BER detection system monitors sampling windows and classifies each window duration as good (⩽1 marked sample), warning (2-3 marked samples), or error (⩾4 marked samples or 2 consecutive warning windows).

Once an error state is triggered, the link must complete a full good window duration before returning to normal state.

Failure Detection Configurations

Name	Type	Defa Value
`detect_xdr_ber_target`	float	**1.0E-16**
`detect_xdr_symbol_fec_enabled_ber_checks`	string	**symb**
`detect_xdr_ber_mark_sample_threshold`	int	**1**
`detect_xdr_ber_marked_samples_count_warning_threshold`	int	**2**

Name	Type	Defa Value
`detect_xdr_ber_marked_samples_count_error_threshold`	int	**4**
`detect_xdr_ber_consecutive_failed_windows_count_threshold`	int	**2**
`detect_xdr_plane_total_rate_gbps`	int	**200**

Link Flaps Detection

A heuristic for detection of link flap events is being utilized. This rule defines a link flap as a scenario where link down occurred on both ends of the link more times than defined by

the link flap threshold.

By setting the right threshold, this rule is meant to capture unintentional link flaps, and filter out links that were intentionally reset or sufferred from a rare link down event.

Link Flaps Detection Configurations

Name	Type	Default Value	Descri
`link_down_threshold_flapping_link_tier_1_or_higher`	int	**15**	The minim numbe link dc events consic a link f for tie higher
`link_down_threshold_flapping_link_tier_2_or_higher`	int	**15**	The minim numbe link dc events consic a link f for tie higher

Examples of how to set additional per tier thresholds:

- link_down_threshold_flapping_link_tier_4_or_higher = ...
- link_down_threshold_flapping_link_tier_7_or_higher = ...

Operating Conditions Violation

A module aimed at validating that the ports are within their nominal operating conditions. Configurable lower and upper thresholds are used to indicate whether the voltage and temperature of the port are exceeding the desired range.

Operating Conditions Violation Configurations

Name	Type	Default Value	Description
`port_temperature_upper_threshold`	int	**70**	The temperature upper threshold value, in °C.
`port_temperature_lower_threshold`	int	**-10**	The temperature lower threshold value, in °C.
`port_voltage_upper_threshold`	int	**3500**	The voltage upper threshold value, in mV.
`port_voltage_lower_threshold`	int	**3100**	The voltage lower threshold value, in mV.

Action Execution

The action execution block processes inputs from the alert generation module, the UFM and the internal port state table in order to determine which links should undergo isolation or de-isolation.

Then, it executes the actions via a UFM API, and finally it updates both on the actions that were and were not taken via messages to the UFM events table.

Action Execution Flow

The action execution flow is comprised of 6 main modules (below image, right side), which can be aggregated into three main steps:

1. Update of the internal port state DB, based on data collected from the alert generation module and from the UFM.

2. Based on the ports' state as captured by the port state DB, run two decision processes to determine which links should be isolated and which should be de-isolated.

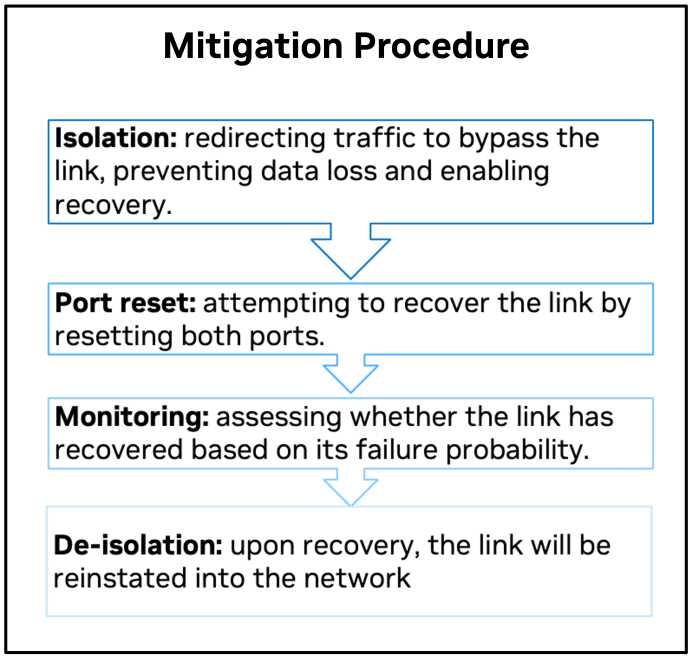
3. Apply the neccessary actions through the UFM.

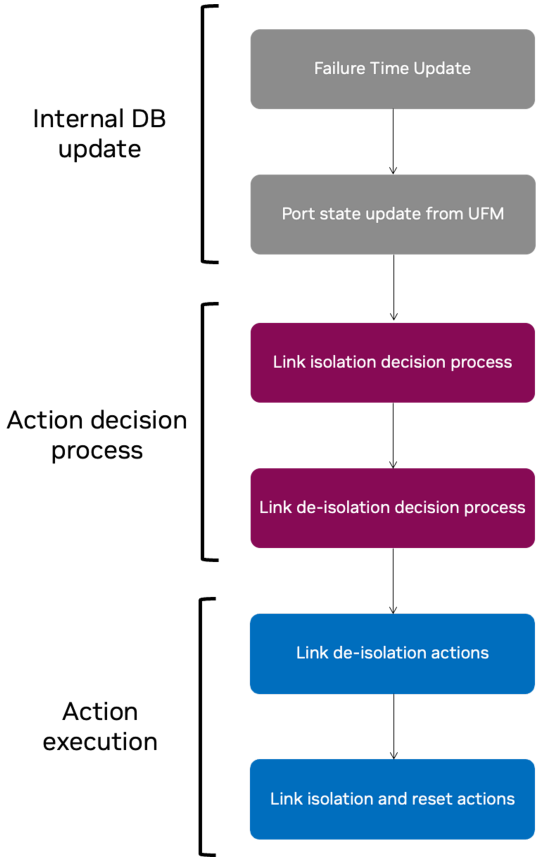
Once a link is selected for action, it enters a mitigation cycle as outlined below.

Isolation and de-isolation of links is done via updating the UFM **Unhealthy Ports** file. This file is submitted by the UFM to the SM, triggering the update of the routing tables throughout the fabric.

Notably, even while the link is isolated and no application traffic flows through it, the physical layer remains active hence we are able to keep monitoring the state of the link using the same aforementioned alert engines.

This enables the the decision process to determine if the link has successfully recovered following the mitigation procedure, and if so reinstate it back into the network.





Link State Transition Diagram

The below diagram depicts the link state mahcine managed by the IBLR plugin.

Application traffic will flow through the link only if it is in **Healthy** state. In all other states, the link is isolated and traffic is redirected to other parallel links.

The black arrows in the diagram indicate state transitions triggered by the plugin, while the red dotted arrows indicate state transitions triggered outside the plugin (e.g. by the

user).

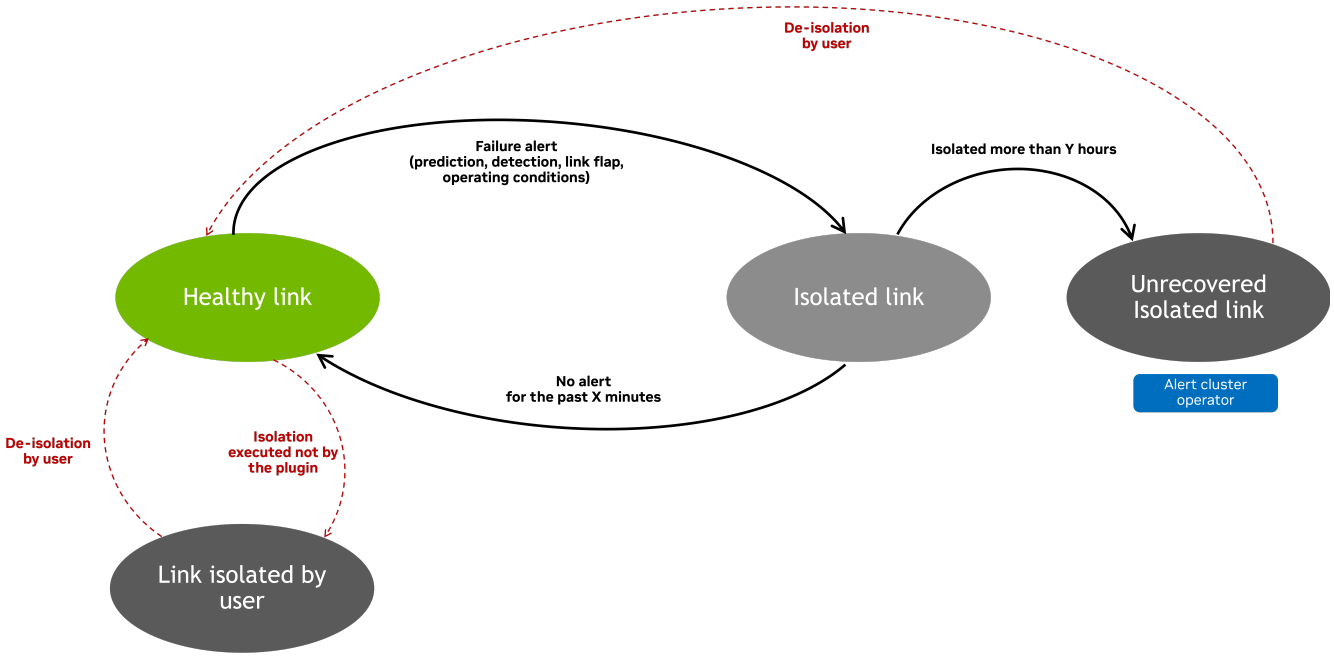
While the main flow of isolating a problematic link, waiting for it to recover and reinstating it is fully operated by the plugin, there are two additional flows that require user intervention:

1. If a link that was isolated by the plugin has not recovered within a pre-determined period, it will be moved into the **unrecoverable** state.

   This comes to suggest that the network operator should inspect the link, attempt to recover it, and if successfull de-isolate the link to signal the link was maintained.

2. If a link was isolated outside the scope of the plugin, the plugin will not de-isolate the link automatically as it has insufficient knowledge with repect to the isolation reason.

   In that case, de-isolation should also be handled by the user or external system that isolated the link.



Action Execution Checkpoints

The action execution decison process includes a set of configurable checkpoints, that provide the user the ability to determine under what conditions actions will be taken.

These configurable checkpoints can be broadly divided into the following groups:

1. Isolation and de-isolation rate limit constraints.
2. Topology-dependent constraints that prevent isolations in case this will lead to insufficient redundancy in the fabric.
3. Isolated links' time limits to qualifying for de-isolation or be considered unrecoverable.

The below tables describe the main configuration parameters used by the decision process checkpoints.

Isolation Checkpoints Configurations

Name	Type	Description
`max_per_hour/day/week/month`	int	The maximum number of distinct links allowed to be **isolated** within a specific time window.
`min_links_per_switch_pair`	int	The minimum number of healthy links between two switches, when reached no further isolations will be allowed.
`min_active_ports_per_switch`	int	The minimum number of active ports per switch, when reached no further isolations will be allowed.

De-isolation Checkpoints Configurations

Name	Type	Description
`max_per_hour`	int	The maximum number of distinct links allowed to be **de-isolated** within an hour.
`min_health_time_before_deisolation_min`	int	The minimum period in minutes that a link has to be clean of alerts in order to be qualified for de-isolation.

Name	Type	Description
`max_recovery_time_threshold_hours`	int	The maximum period in hours that a link is granted for recovery and de-isolation, before declaring it as unrecoverable.

Reporting

Customers may be looking to extend or replace the plugin's mitigation procedure with their private business logic.

For that purpose, the action execution module reports both on actions it has taken, as well as on alerts that did not result in any action due to the decision process failing at one of the checkpoints.

Reporting is done via sending event-specific messages to the UFM events table.

Indication of Action

These messages are sent following every action taken by the plugin:

Event	Message Format	Comments
Isolation	`Completed isolation of the link between ports <src_guid>_<src_num> <-> <dest_guid>_<dest_num>, isolation reasons: <alert_reasons>`	
De-isolation	`Completed deisolation of the link between ports <src_guid>_<src_num> <-> <dest_guid>_<dest_num> after no alerts were triggered for <isolation_reasons>`	Isolation reasons are the alerts that triggered the initial isolation
Declaring unrecoverable port	`Port <src_guid>_<src_num> could not be deisolated for more than <threshold value> hours due to repeating <alert_reason> alerts, hence it is marked as unrecoverable`	Threshold is configurable, the plugin will not deisolate unrecoverable ports

Indication of Alert without Action

These messages are sent following alerts that were triggered and did not results in any action (for example, due to the link being configured to run in shadow mode).

In this case, as no attempt to recover a problematic link is done by the plugin, the same alert may be repeatedly raised and result in repetitive messages sent to the UFM events table indicating the same issue.

To avoid such a scenario, we have included an option to suppress repetitive messages, defined as messages that repeat information already provided in a previous message (e.g., similar event reported for the same port due to the same alert). By setting the value of the configuration parameter `suppresion_interval_hours` (default value is 24), the user can determine for how many hours a repetitive alert message will be suppressed (setting a value of 0 disables the suppression mechanism).

Event	Message Format	Comments
Total alerts count exceeding alert threshold per iteration	`Isolation will not be performed because the number of ports alerted for isolation (<actual count>) is greater than the threshold for the maximum number of isolation alerts (<threshold value>)`	Threshold is configurable
Access link alert	`Isolation will not be performed for port <guid>_<num> because this is an access link, triggered alerts: <alert_reasons>`	
Shadow mode preventing isolation	`Isolation will not be performed for port <guid>_<num> because the port is in shadow mode, triggered alerts: <alert_reasons>`	
Shadow mode preventing de-isolation	`Deisolation will not be performed for port <guid>_<num> because the port is in shadow mode, triggered alerts: <alert_reasons>`	Only if mode changed from active to shadow while port is isolated

Event	Message Format	Comments
Exceeding isolation count per time	`Isolation will not be performed for port <guid>_<num> because the number of isolated ports in the last <time frame> (<actual count>) exceeds the threshold for max isolations allowed (<threshold value>), triggered alerts: <alert_reasons>`	Time frame is in (hour, day, week, month), each has a configurable threshold
Exceeding de-isolation count per time	`Deisolation will not be performed for port <guid>_<num> because the number of de-isolated ports in the last <time frame> (<actual count>) exceeds the threshold for max de-isolations allowed (<threshold value>)`	Time frame is hour, threshold is configurable
Insufficient redundancy in the connectivity between the switches	`Isolation will not be performed for port <guid>_<num> because the number of links between switches <src_guid> and <dest_guid> (<actual count>) will decrease below the threshold for minimum number of healthy links (<threshold value>), triggered alerts: <alert_reasons>`	Threshold is configurable
Insufficient number of active ports in the switch	`Isolation will not be performed for port <guid>_<num> because the number of active ports for switch <switch_guid> (<actual count>) will decrease below the threshold for minimum number of active ports (<threshold value>), triggered alerts: <alert_reasons>`	Threshold is configurable, validated for the two switches which the link connects

Deployment

The IB Link Resiliency plugin can be deployed using the following methods:

1. On the UFM Appliance

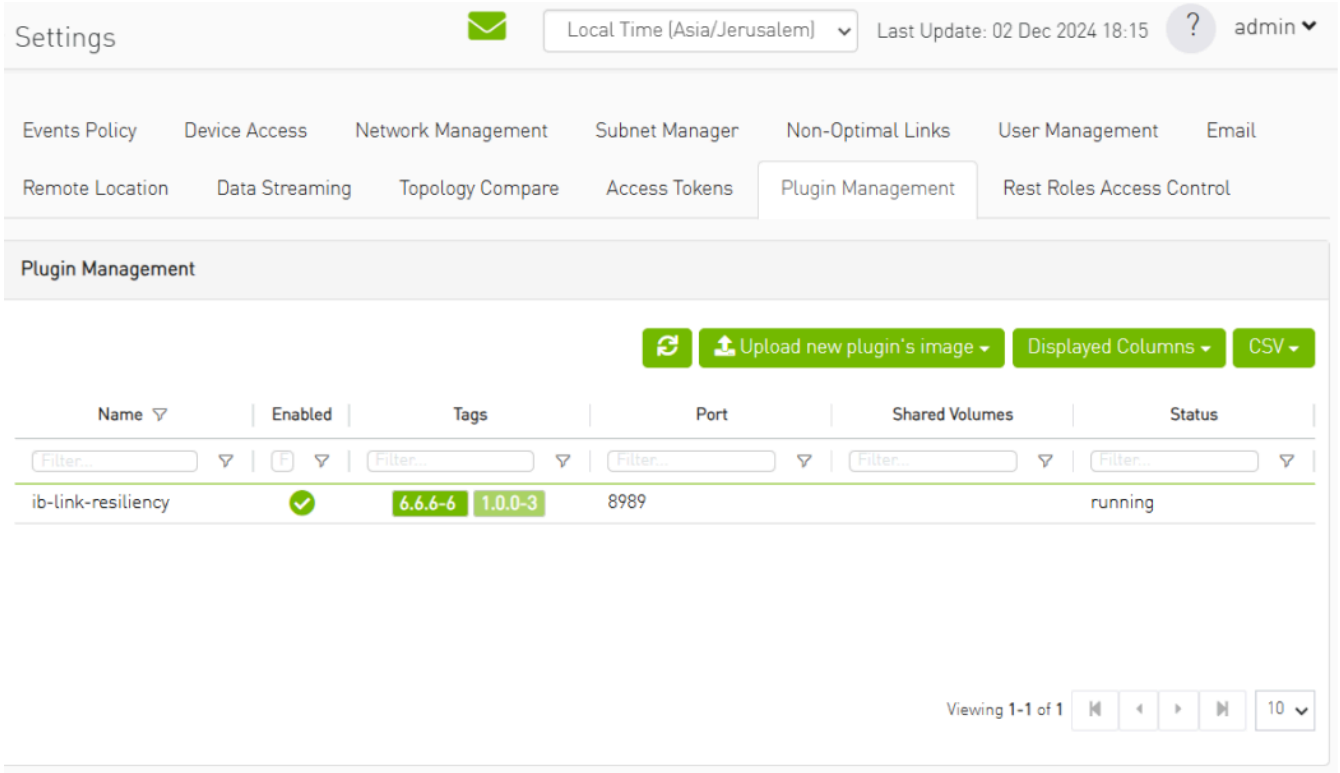
2. On the UFM Software

To deploy the plugin, follow these steps:

1. Download the ufm-plugin-ib-link-resiliency-image from DockerHub.
2. Load the downloaded image onto the UFM server. This can be done either by using the UFM GUI by navigating to the `Settings -> Plugins Management` tab or by loading the image via the following instructions:
    1. Log in to the UFM server terminal.
    2. Run:

```
docker load -i <path_to_image>
```

3. After successfully loading the plugin image, the plugin should become visible within the plugins management table within the UFM GUI. To initiate the plugin's execution, simply right-click on the respective line in the table.

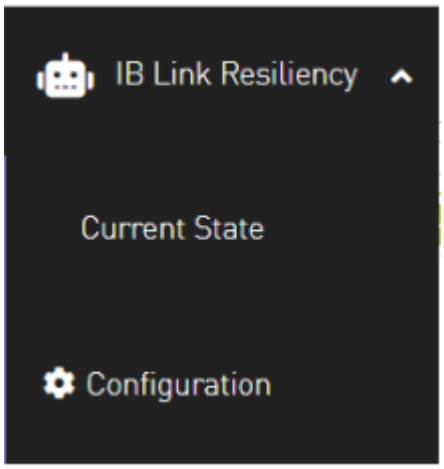


Note

The supported InfiniBand hardware technologies are HDR and NDR.

IB Link Resiliency UI

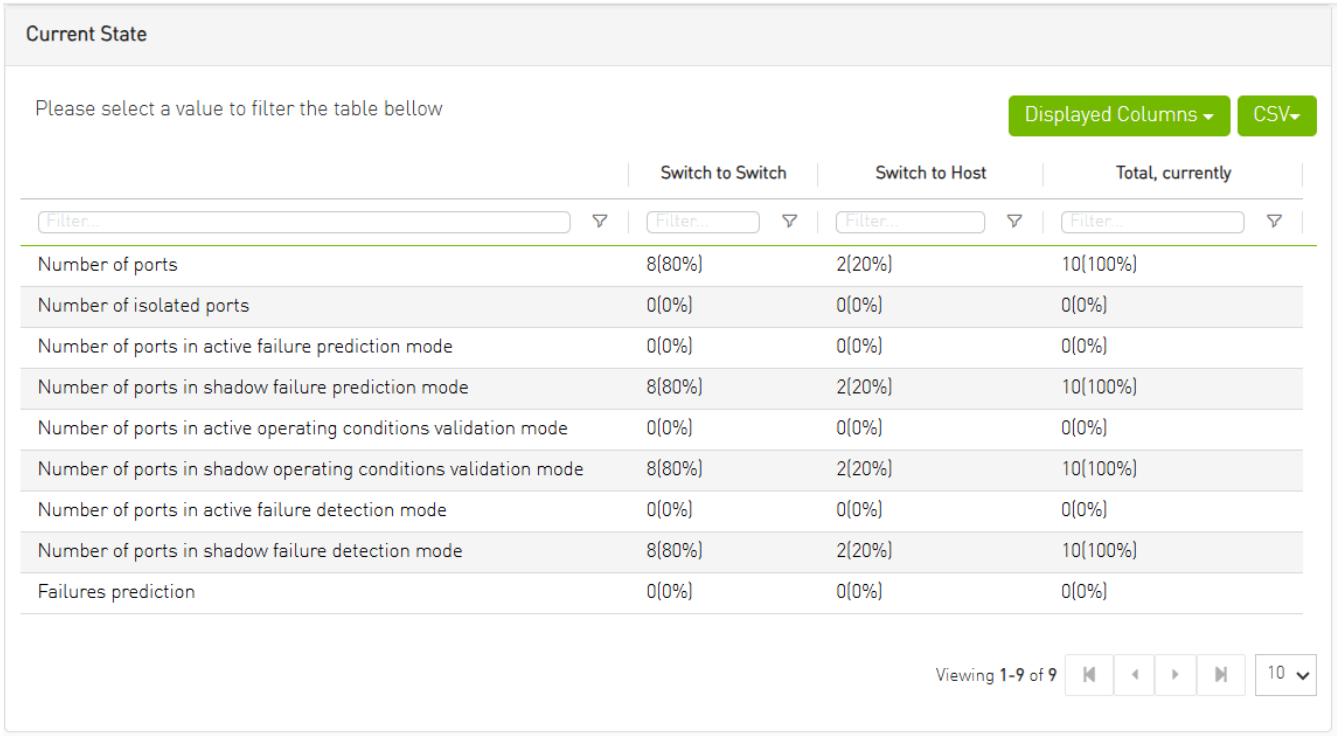
After the successful deployment of the plugin, a new item is shown in the UFM side menu for the IB Link Resiliency plugin:



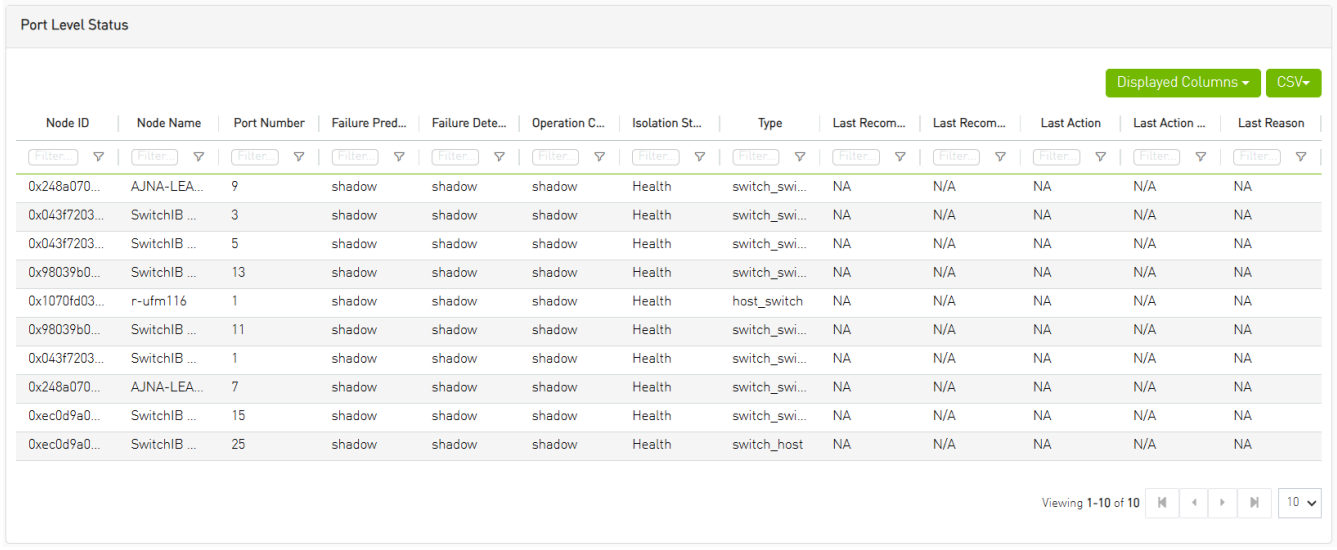
Current State

This page displays the `Current State` table presenting the cluster status, outlining the following counts:

1. Number of ports.
2. Number of isolated ports.
3. Number of ports operating in active/shadow mode, per each of the alert engines.
4. Number of predicted failures, reflecting ports which were recently flagged by the ML model as having a high probability of failure.

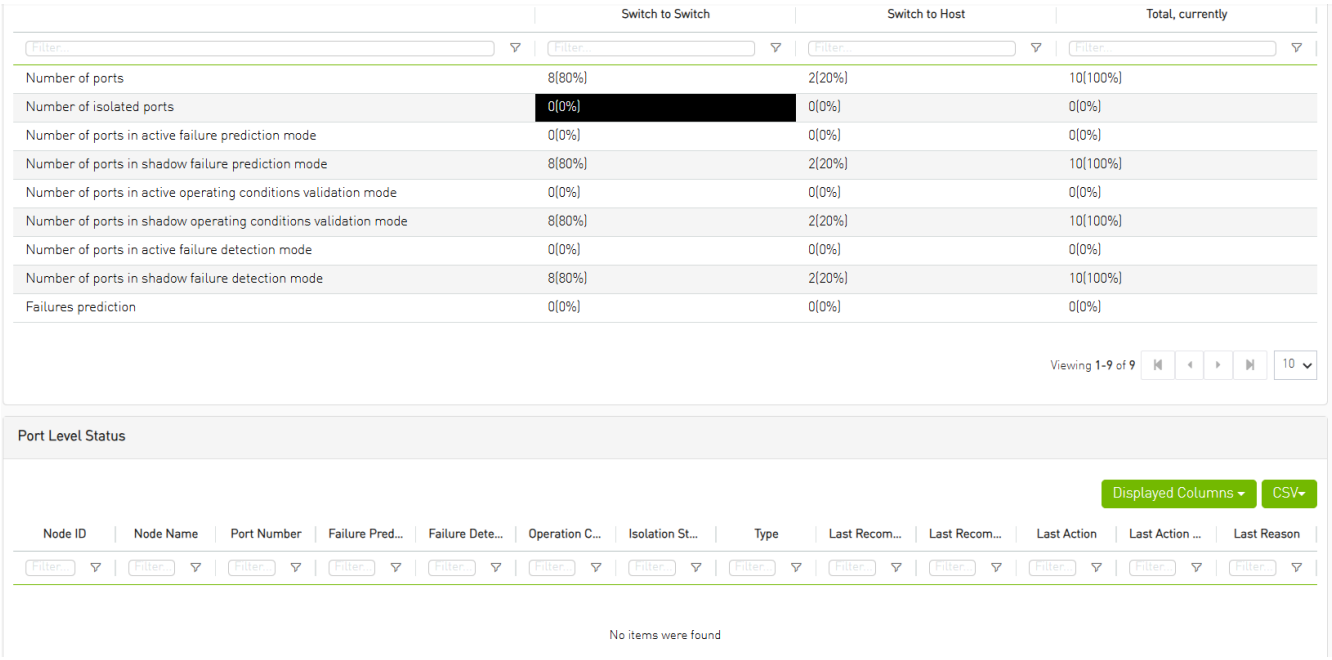


The `Port Level Status` table is displayed below and shows the status of the cluster ports, as stored in the port state DB.



The user can filter the `Port Level Status` table by clicking on any value in the `Current State` table.

For example, if the user clicks on the number of isolated switch-to-switch ports, the `Port Level Status` table will display only the isolated switch-to-switch ports.



	Switch to Switch	Switch to Host	Total, currently
Number of ports	8(80%)	2(20%)	10(100%)
Number of isolated ports	0(0%)	0(0%)	0(0%)
Number of ports in active failure prediction mode	0(0%)	0(0%)	0(0%)
Number of ports in shadow failure prediction mode	8(80%)	2(20%)	10(100%)
Number of ports in active operating conditions validation mode	0(0%)	0(0%)	0(0%)
Number of ports in shadow operating conditions validation mode	8(80%)	2(20%)	10(100%)
Number of ports in active failure detection mode	0(0%)	0(0%)	0(0%)
Number of ports in shadow failure detection mode	8(80%)	2(20%)	10(100%)
Failures prediction	0(0%)	0(0%)	0(0%)

Viewing 1-9 of 9

Port Level Status

Displayed Columns CSV

Node ID	Node Name	Port Number	Failure Pred...	Failure Dete...	Operation C...	Isolation St...	Type	Last Recom...	Last Recom...	Last Action	Last Action ...	Last Reason

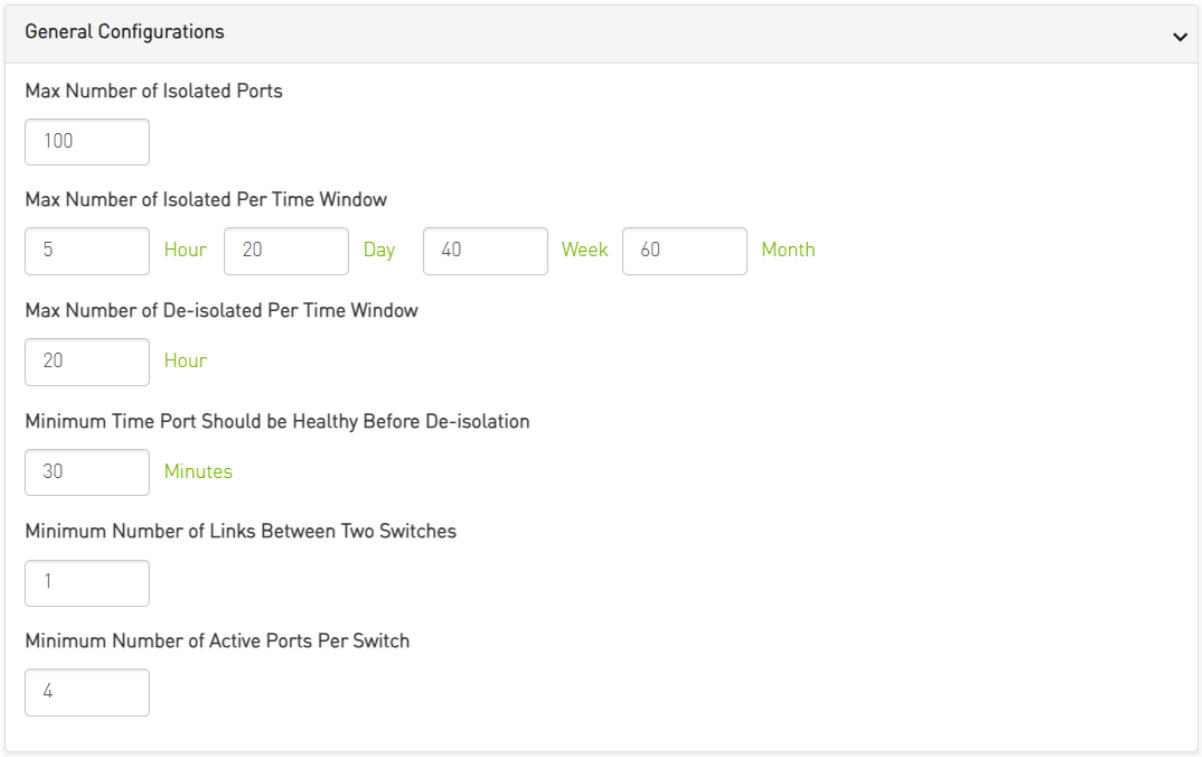
No items were found

Configuration

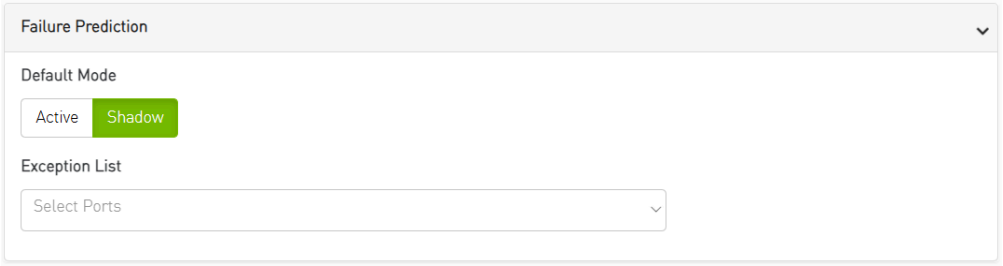
This page displays a subset of the aforementioned IBLR configuration parameters.

Below are representative screenshots showing some of the configuration parameters accessible through the UI:

- Action execution configurations:



- Failure prediction operating mode configurations:



ClusterMinder Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Changes and New Features
1.1.14	• Added alert notifications for CDU (Coolant Distribution Unit) • Added data collection from in-Row cooling units • Added data collection from SATEC power measurement devices • Analytics tab for identifying suspected errors and group comparisons • Quick view of switch firmware versions (NVOS, CUMULUS) • Support for ClusterMinder ARM64 container image • Reduced container image size for improved efficiency

Bug Fixes

Plugin Version	Bug Fix
1.1.14	• Resolved issue where the plugin did not run after HA failover (Ref #4654021) • Addressed high-severity vulnerability in libtiff6 (Ref #4646789) • Fixed upgrade process requiring manual browser cache clearing (Ref #4514314)

Note

This plugin is supported on UFM Enterprise Appliance only.

Overview

The ClusterMinder plugin collects telemetry data from multiple data sources and aggregates, streams and visualizes the backend data. The plugin can cluster/group aggregated data from multiple machines that allows operational anomaly and misconfiguration detection. The plugin provides Cluster-wide histograms of hardware

telemetry which details compute node configuration and inventory, PCIe bus, hardware information (SN and FW version) and health alerts of all relevant devices.

The plugin can be deployed as a container and supports multiple data sources

Data Source Supported Types and Devices

Type	Devices
Redfish	AMI DGX H100 Dell PowerEdge XE9680 Dell PowerEdge R760 Dell PowerEdge R750 Dell PowerEdge R760xa Supermicro SYS-421GE-TNRT3 Supermicro SYS-421GE-TNHR2-LC-TW008 Supermicro SYS-821GE-TNHR AMI ESC N8-E11 ASUSTeK COMPUTER INC
Redfish on DPU	Nvidia
Switch	MLNXOS, Cumulus, NVOS
Unmanaged Switch	UFM Telemetry
DTS (DOCA Telemetry Service) on DPU (Bluefield)	DTS: version > 1.12 DPU OS : ubuntu 20/22
DTS (DOCA Telemetry Service) on Host	
NMX	NMX-C or NMX Aggregator
DDN	DDN storage
CDU (Coolant Distribution Unit)	XDU1350, XDU450, XDU R100
InRow	Vertiv
Satec	SATEC PM 130,135,175

Note

For Redfish we auto discover (with best effort) devices if it is not supported for example: GB200/300 compute tray/switch tray, GB200 power shelf. Do note this takes some time (up to 15 minutes in the case of GB200 and GB300).

Deployment

The plugin can be deployed using the following methods:

1. On the UFM Appliance
2. On the UFM Software

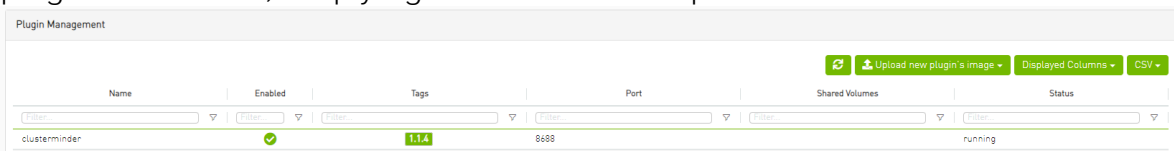
To deploy the plugin, follow these steps:

- The plugin is included in the default plugin bundle available at NVIDIA's Licensing Portal .
- Load the downloaded image onto the UFM server. This can be done either by using the UFM GUI by navigating to the Settings -> Plugins Management tab or by loading the image via the following instructions:
  - Log in to the UFM server terminal.
  - Run:

```
docker load < <path_to_image>
```

  - After successfully loading the plugin image, the plugin should become visible within the plugins management table within the UFM GUI. To initiate the

plugin's execution, simply right-click on the respective in the table.



ClusterMinder Plugin UI

After the successful deployment of the plugin, a new item is shown in the UFM side menu for the ClusterMinder plugin

Data Sources Page

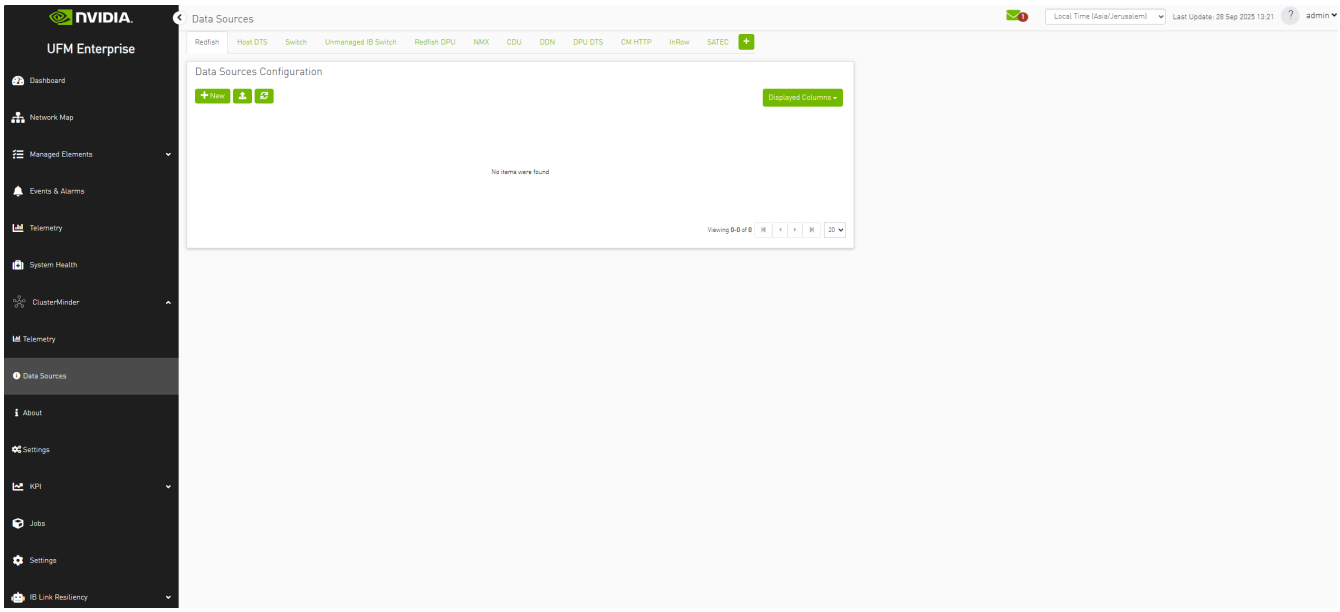
Managing data sources with ClusterMinder is intuitive and efficient. Users can easily add new data sources, update existing ones, or remove those that are no longer needed. This feature ensures your data is always current and relevant.

Example of Adding Data Source

Adding hosts is done through the "Data Sources" section. To add a data source, you start by selecting the appropriate tab, clicking + New , and filling out the information on the endpoint. Then you must first test the endpoint and if the endpoint status is "up," the add button will become clickable, allowing you to add the data source.

Note: we support adding hosts in hostlist format for exmaple: agx[01-10].

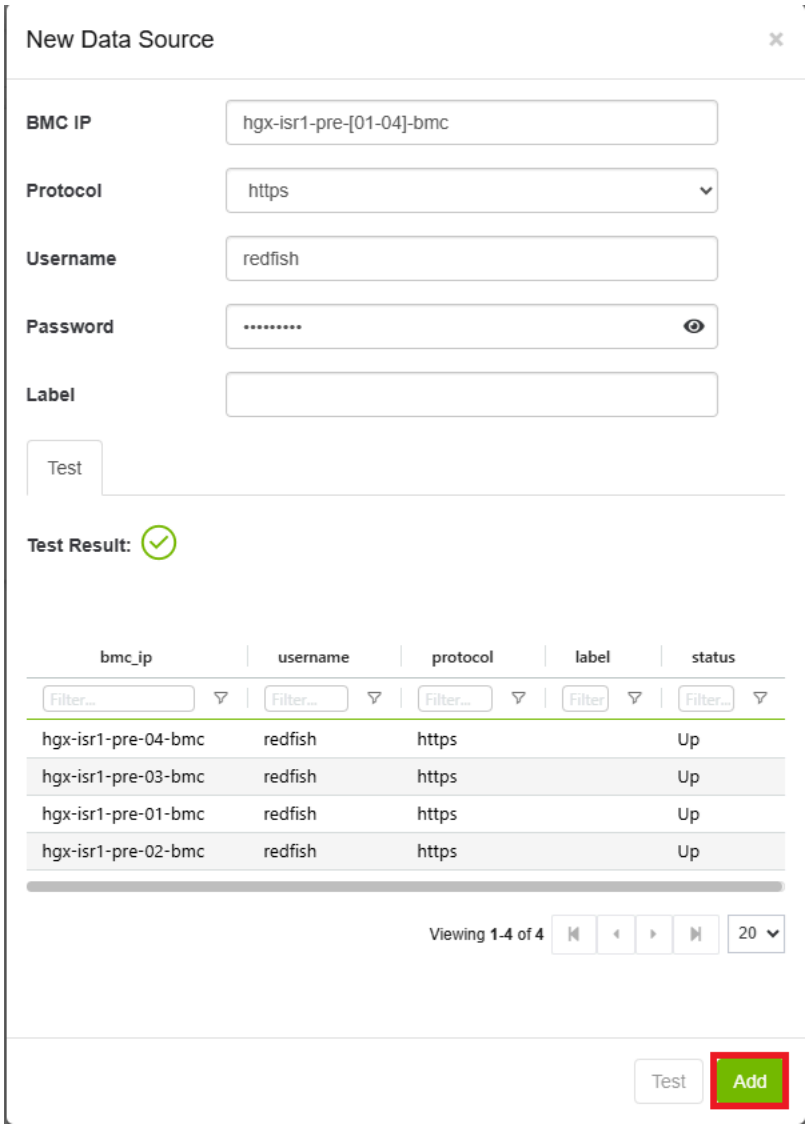
for example:



Example of Adding a Redfish Host

After pressing the + New button you will be presented with a form for inputting the following fields: " BMC IP", "Protocol","Username" and "Password". after inputting the needed information, pressing the Test button tests the connection to the host. You will then be presented with a window notifying if the connection was successful. if successful you can click the Add button to add the data source.

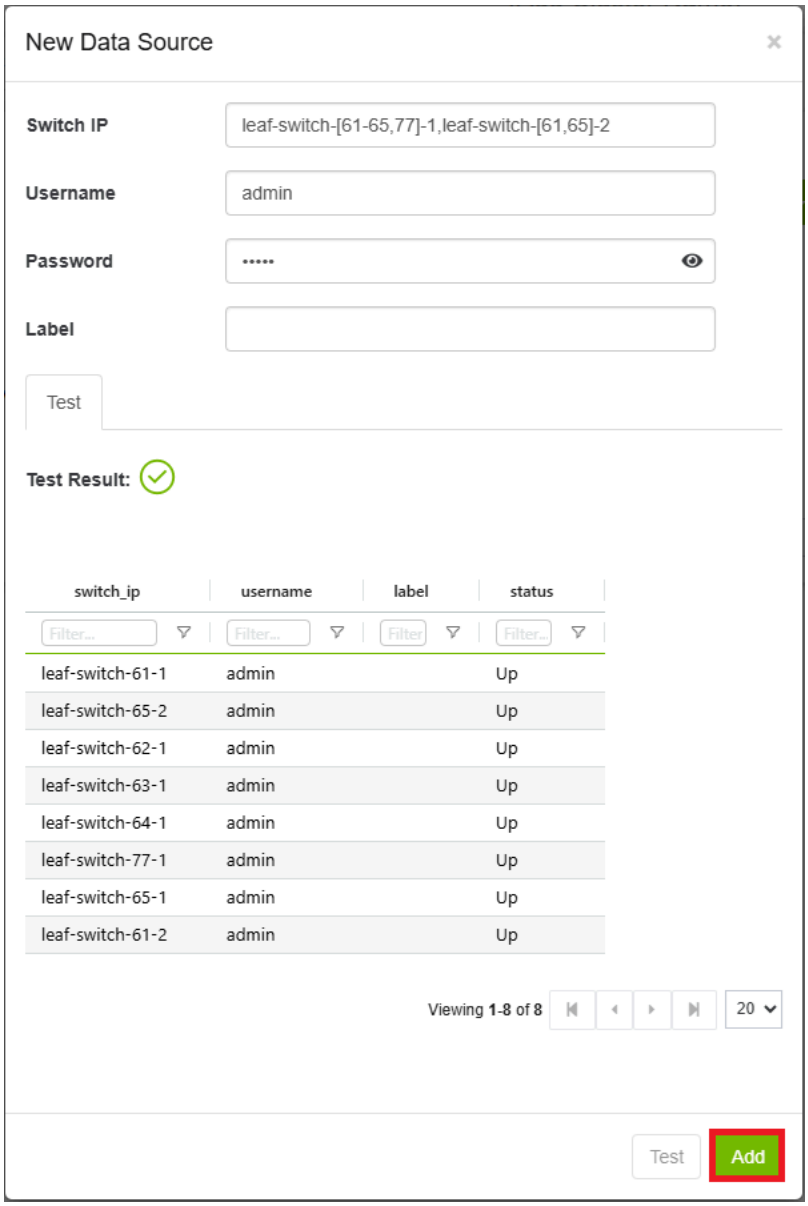
For example:



Note: when adding multiple redfish hosts, they should all be of the same model and vendor. If this is not done it can cause issues with data collection and presentation.

Example of Adding a Switch Host

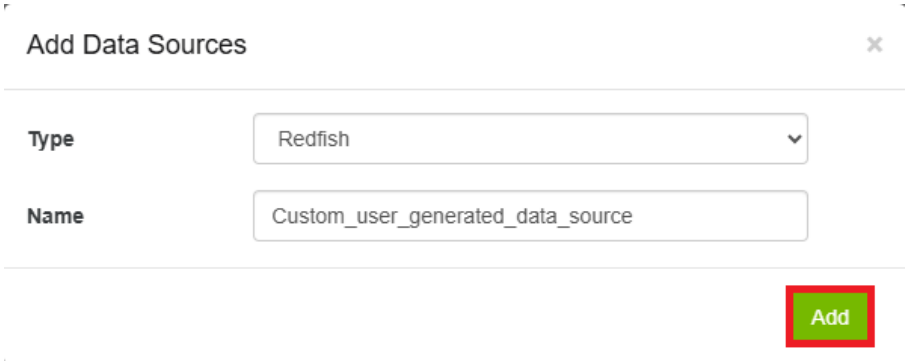
After pressing the + New button you will be presented with a form for inputting the following fields: "Switch IP", "Username", "Password". after inputting the needed information, pressing the Test button tests the connection to the host. You will then be presented with a window notifying if the connection was successful. if successful you can click the Add button to add the data source.



Note: when adding multiple switch hosts, they should all be of the same OS type. If this is not done it can cause issues with data collection and presentation.

Example of Adding a User Generated Data Source Type

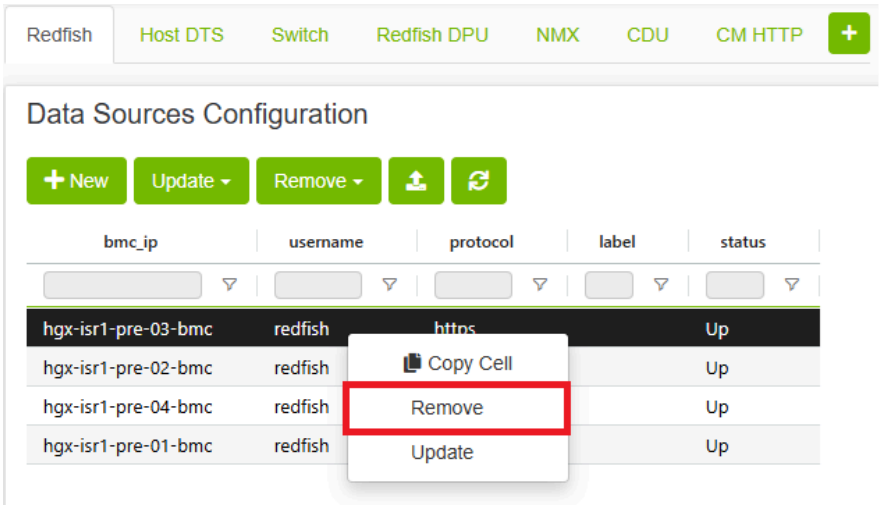
Pressing the + button in the row of data types allows the user to create custom user generated data source, you will be presented with a form for inputting the following fields: "Type", "Name". After inputting the needed information, you can click the Add button to add the data source.



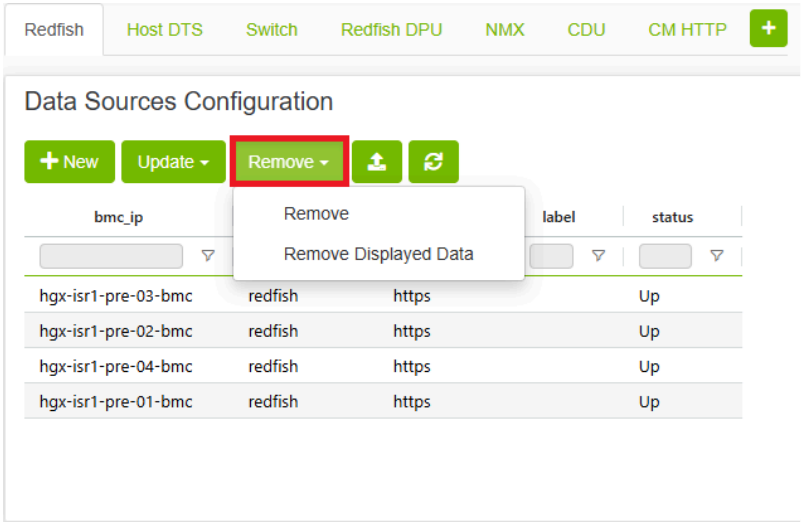
> **Note**
>
> Type dropdown has "Redfish","Switch","Host DTS", " CM Http". These are the currently supported data types. Requires a page refresh to appear in the sidebar.

Example of Removing Data Source Endpoint

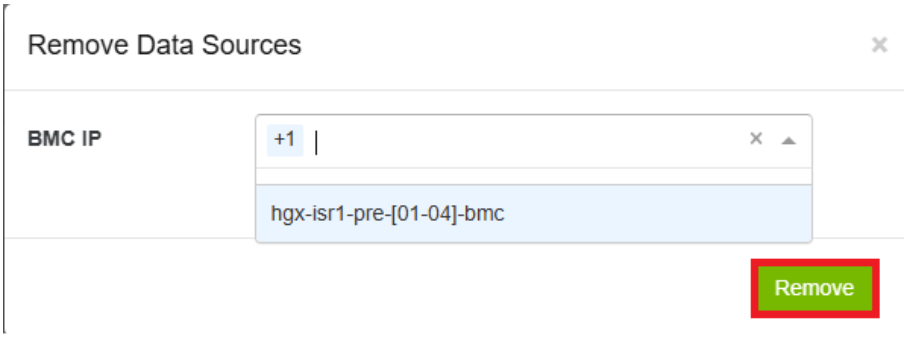
Removing hosts is done through the "Data Sources" section. here you can right click any available host and click the remove option.



You can also use the Remove button

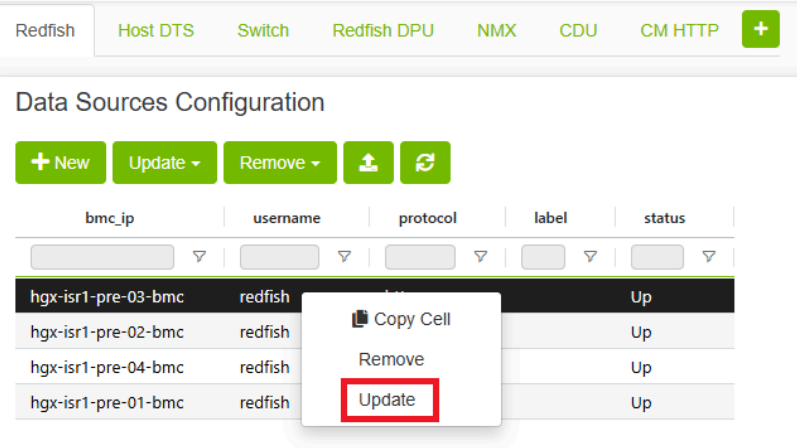


to open the Remove Data Sources form that has the IP field. After choosing the appropriate IP you can click the Remove button to remove those endpoints

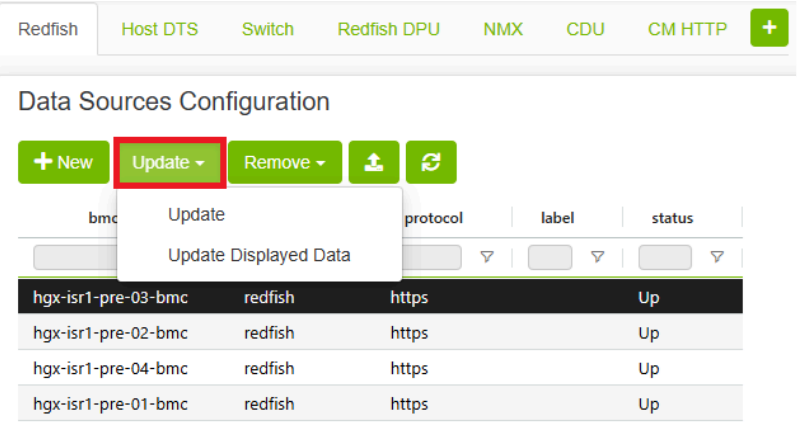


Example of Updating Data Source Endpoint

Updating hosts is done through the "Data Sources" section. here you can right click any available host and click the update option.



You can also use the Update button to open the Update Data Sources form that thar has the IP field. After filling the appropriate fields, pressing the Test button tests the connection to the host. You will then be presented with a window notifying if the connection was successful. if successful you can click the Add button to update the endpoint.

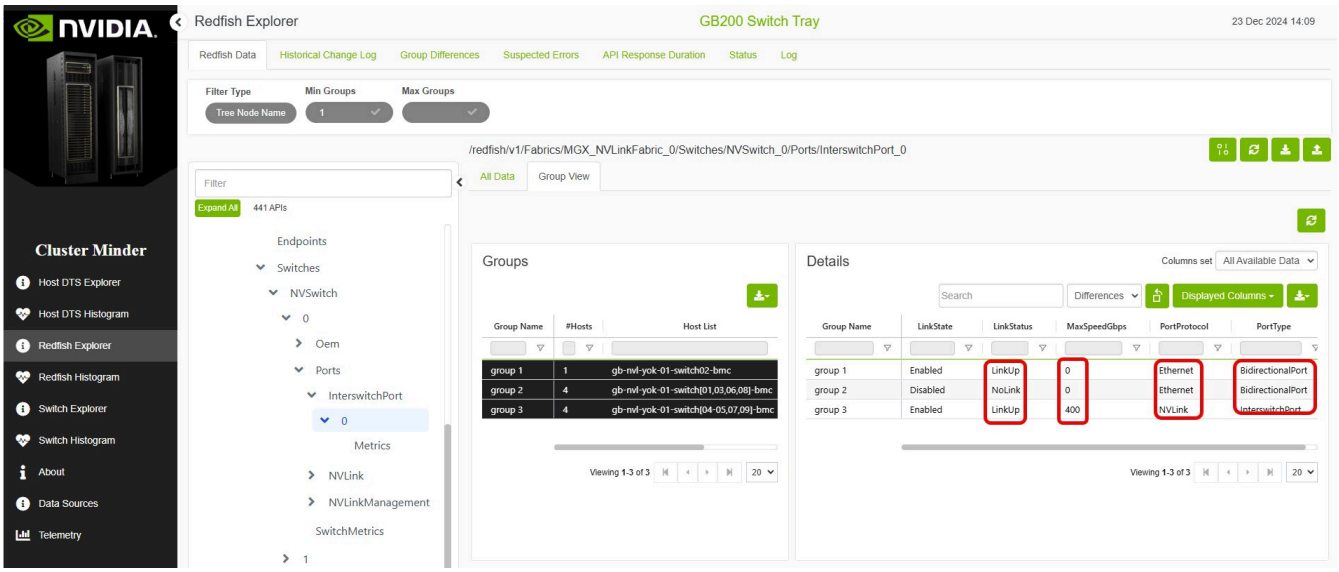


Explorer Page

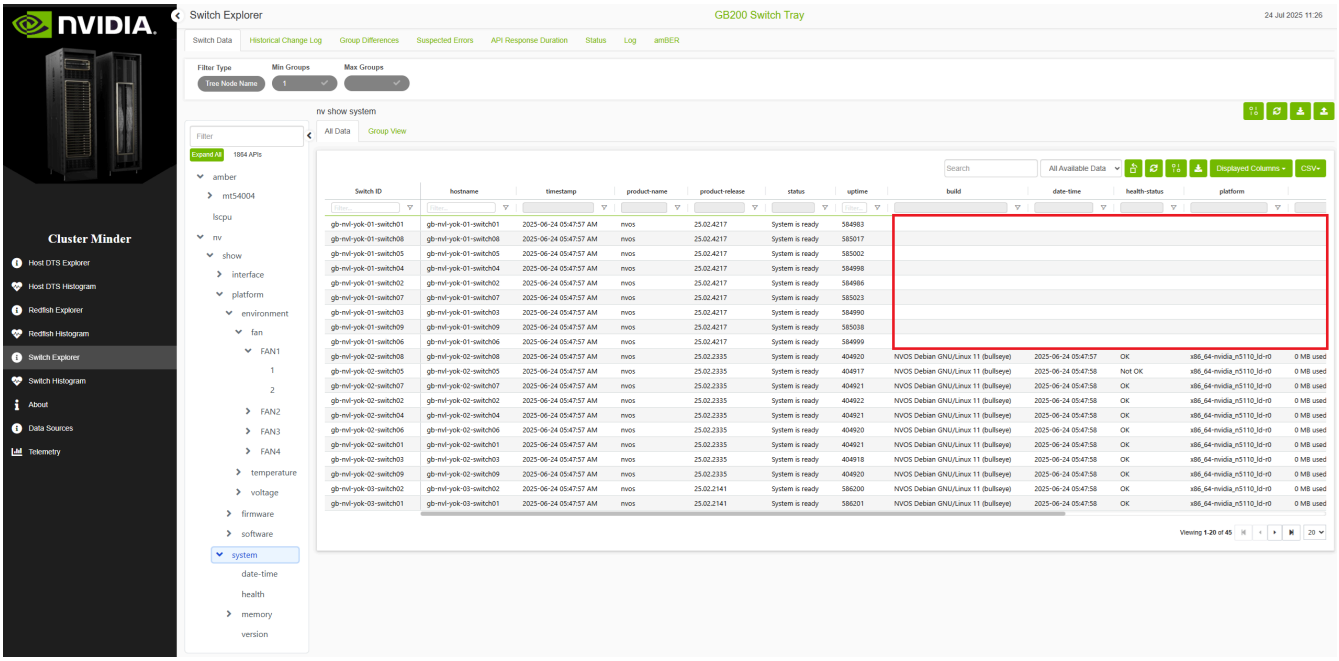
Data Tab

The Data tab provides a comprehensive view of all your data in one place, organized in an easy-to-navigate tree or table view. Additionally, there is a group view tab that allows users to see differences in host groups, helping to identify and understand misconfigurations within their data. This feature allows users to quickly access and navigate through their data, making it easier to perform analyses and derive insights.

Redfish Data Example



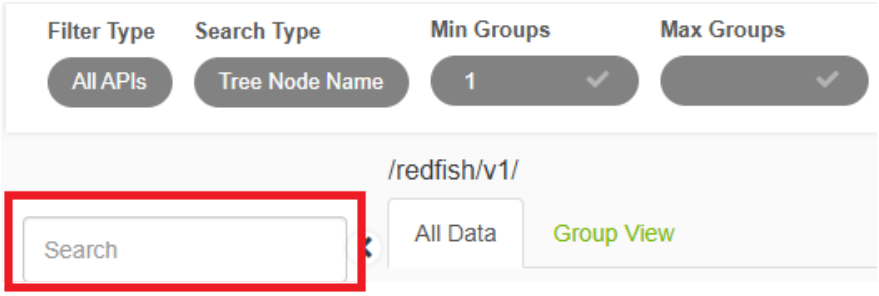
Switch Data Example



Additional Features

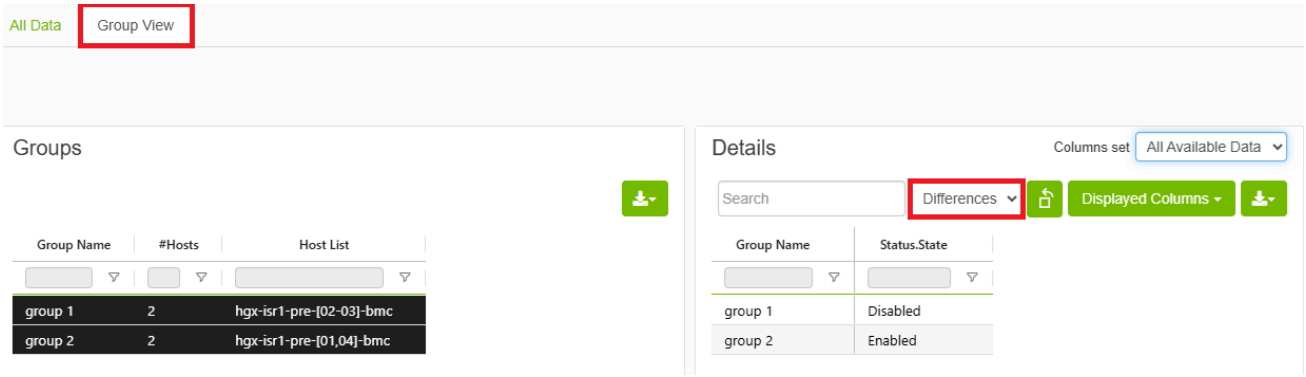
Filter Bar

The filter allows to choose specific API based on frequency level. The search box allows to search for substring in the API or a specific data field. You can also choose to only show APIs with a minimum or maximum number of groups .



Group View

The group view inner tab allows to view endpoints divided to groups based on differences in data.



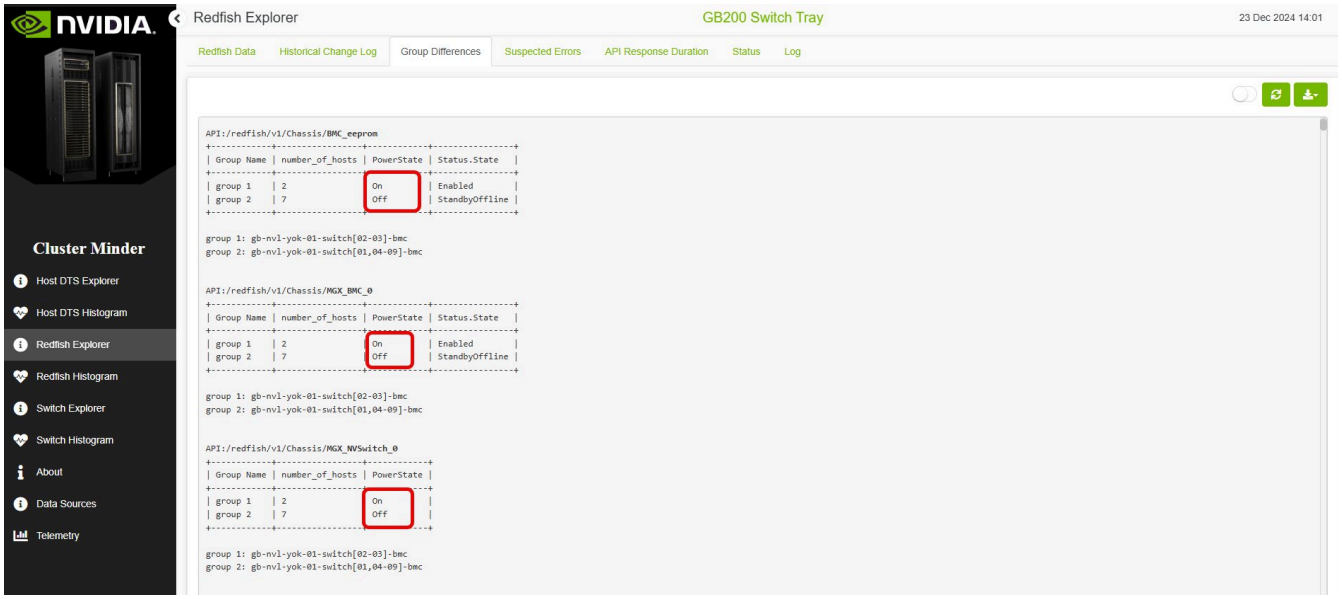
Group Differences Tab

The “Group differences” tab helps users identify and understand misconfigurations within their data. By comparing different data groups, users can easily spot discrepancies and

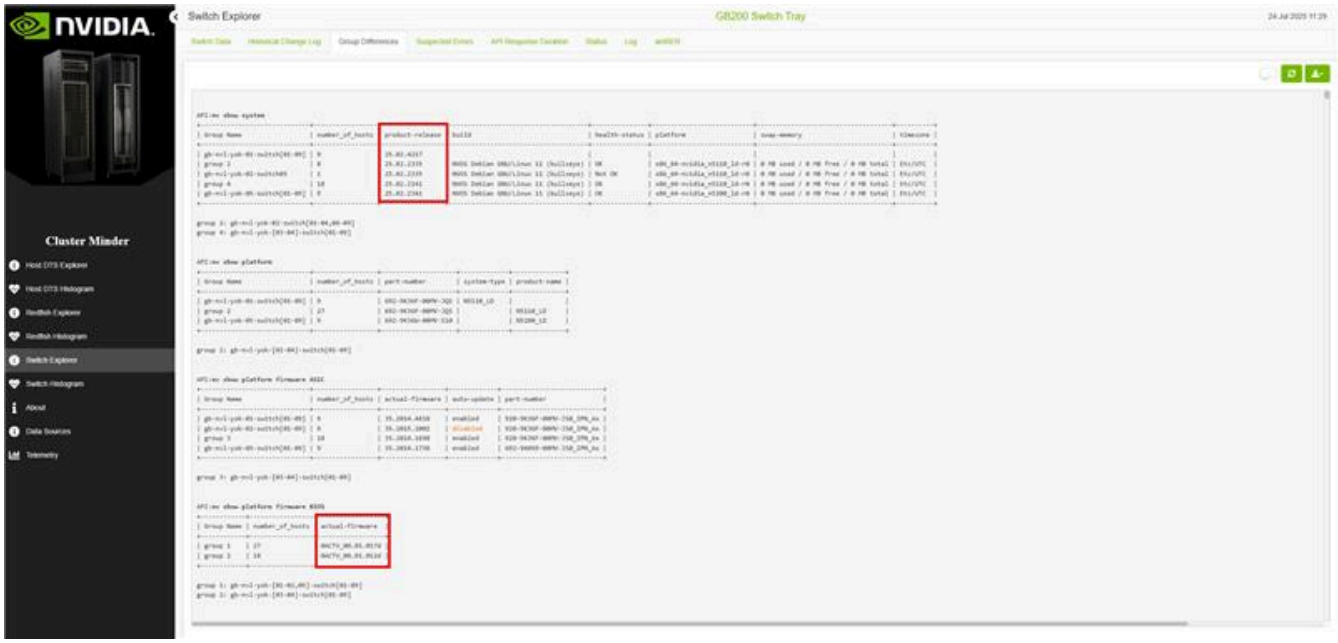
take corrective actions. At the start of the report, there is a summary table that lists the most problematic hosts in descending order based on the number of appearances. This table provides a quick overview of the hosts that require the most attention, allowing users to prioritize their troubleshooting efforts effectively. Additionally, each table in the report (besides the summary table) includes the API that provided the data, a column for the number of hosts per group, and the fields where their values differed, ensuring transparency and traceability of the information presented.

> **Note**
>
> There is a switch button that enables component view.

Redfish Group Differences Example



Switch Group Differences Example

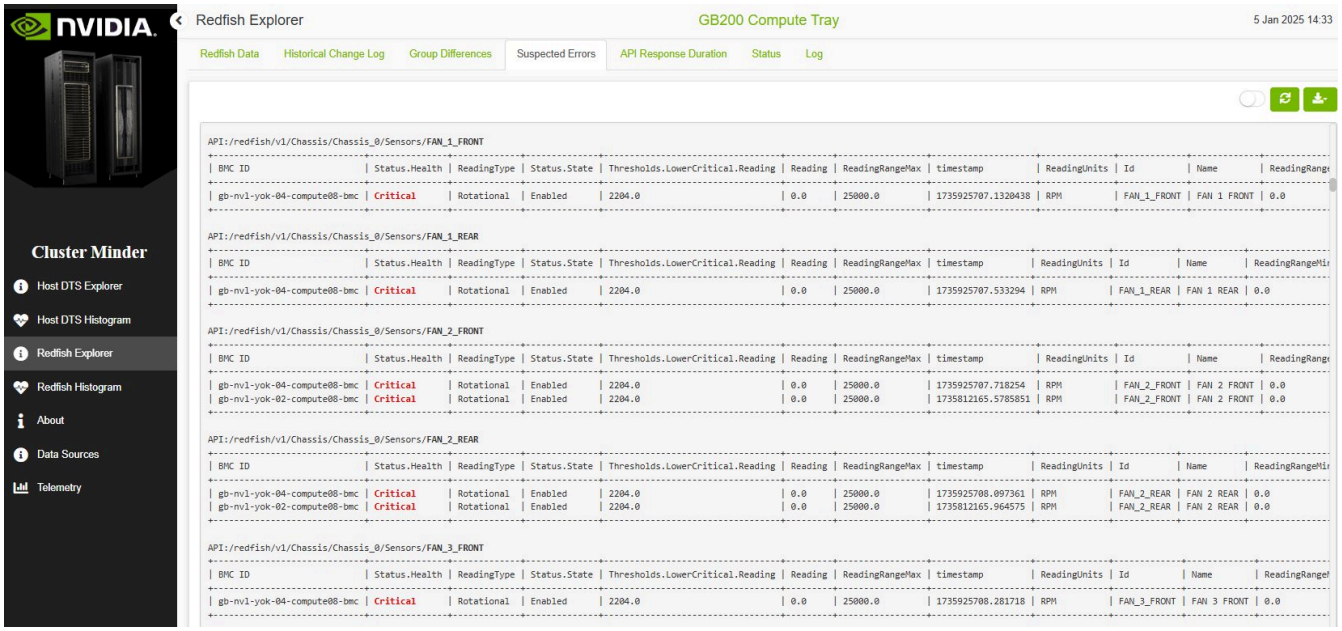


Suspected Errors Tab

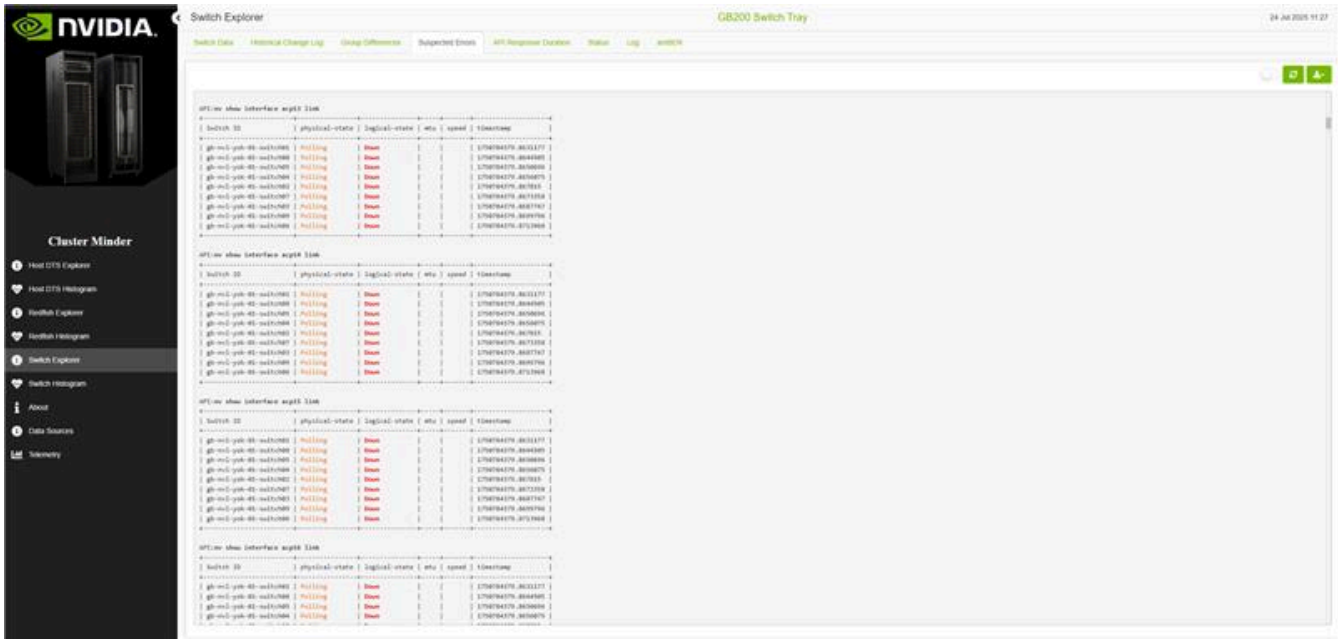
The “Suspected Errors” tab provides a comprehensive report on APIs that have returned values flagged as potential issues. This report is crucial for maintaining the reliability and integrity of your data analysis. At the start of the report, a summary table lists the number of hosts with errors compared to the total number of hosts, giving users a quick snapshot of the overall health of their data environment. This summary helps prioritize troubleshooting efforts and allocate resources effectively. Each entry in the report details the host ID, the specific fields where values were problematic, and the problematic values themselves. These values are highlighted in red or orange to indicate the severity of the issue, with red denoting more critical problems and orange indicating less severe ones. This color-coding allows users to quickly assess the urgency of each issue and take appropriate corrective actions.

> **Note**
>
> There is a switch button that enables component view

Redfish Suspected Errors Example



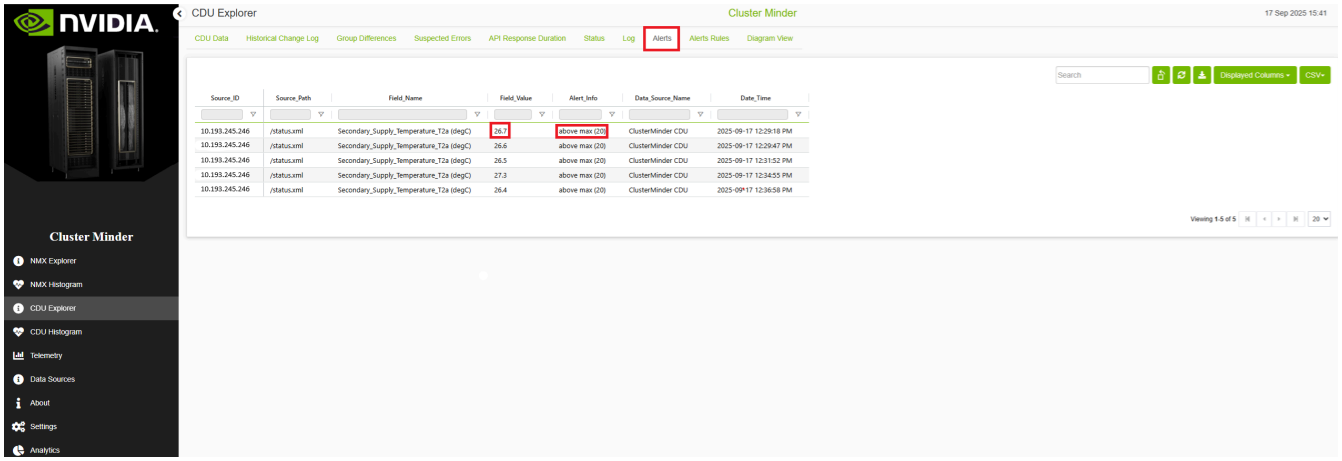
Switch Suspected Errors Example

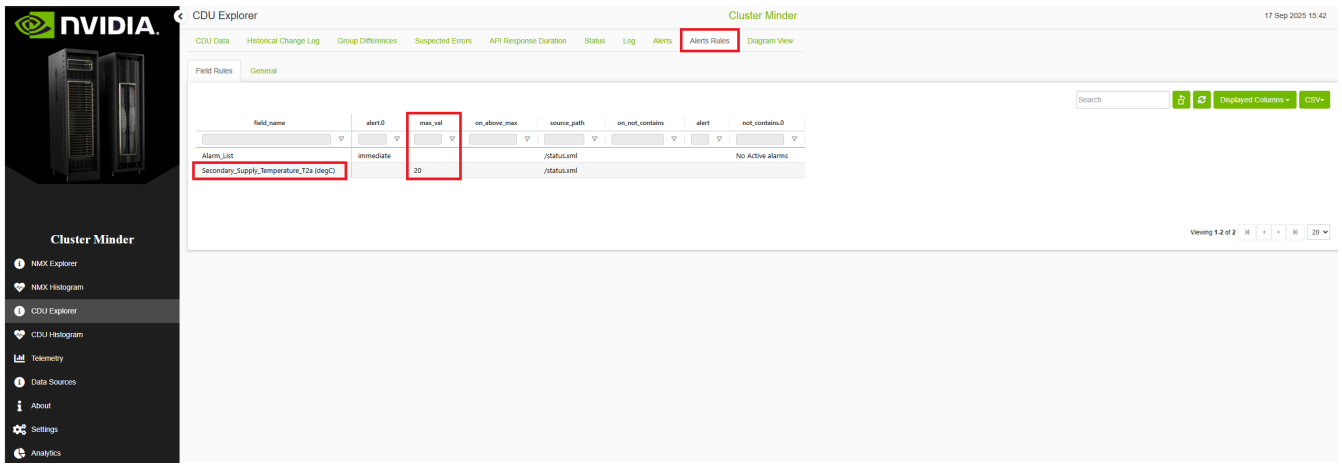


Alerts and Alerts Rules Tab

The Alert tab displays all fields whose values meet or violate specific thresholds defined by the user. The Alert Rules tab provides an overview of all the rules the user has created.

Currently only supported on CDU.





Additional Features

Historical Change log Tab

Displays fields that have had their values changed, sorted by SRC ID and API.

API Response Duration Tab

Displays the duration of each APIs data collection.

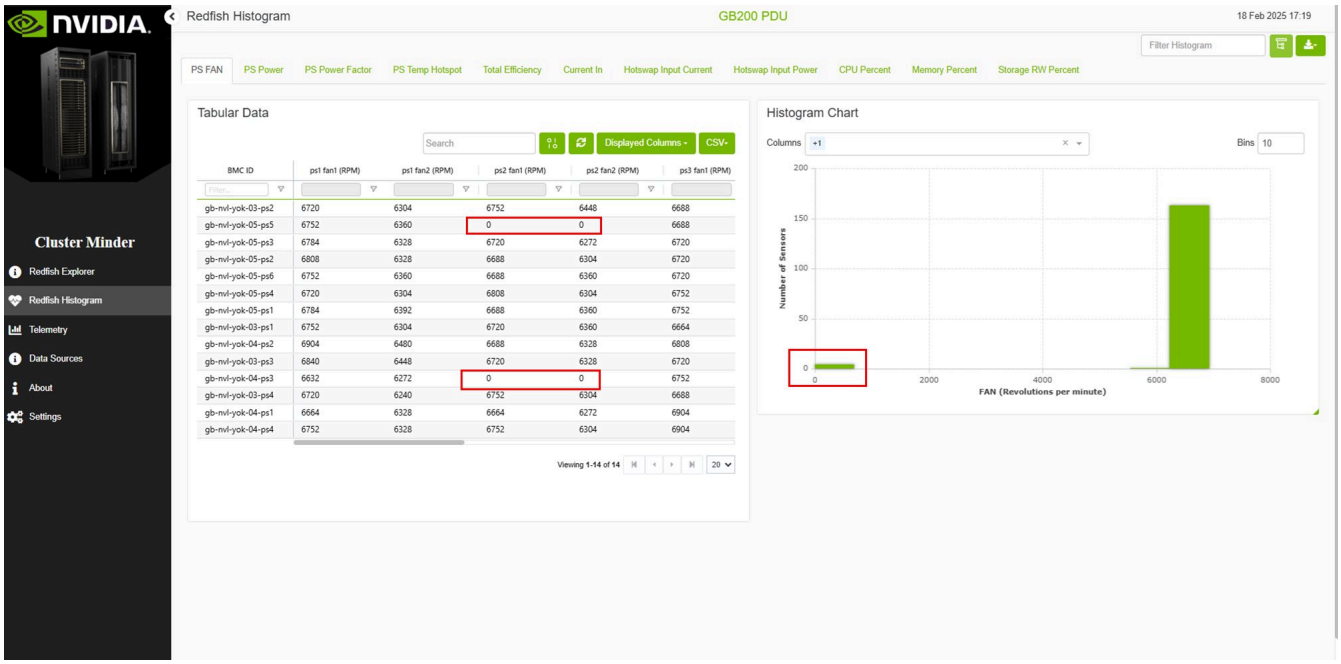
Log Tab

The log page in Cluster Minder provides a comprehensive view of all product logging activities. This feature is essential for tracking changes, monitoring system performance, and diagnosing issues.

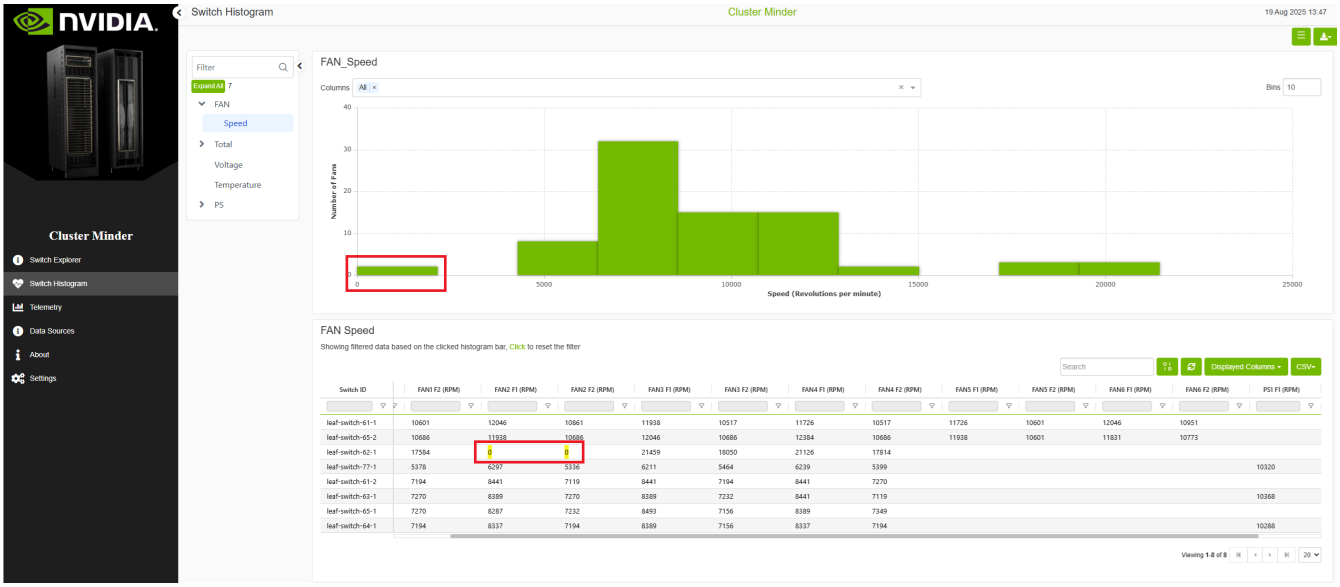
Histogram Page

Visualizing data is made simple with ClusterMinder's histogram feature. Histograms are premade depending on the type of data source as well as the make and model. The histograms range from rpm speeds, voltages, wattage, amperage, temperatures (Celsius) and more. By clicking the histograms, users can filter specific hosts or values, allowing for a more targeted analysis. Additionally, using the dropdown menu, users can further filter the histogram bars to hone in on specific data subsets and attributes, enhancing their ability to derive meaningful insights.

Redfish Histogram Example



Switch Histogram Example

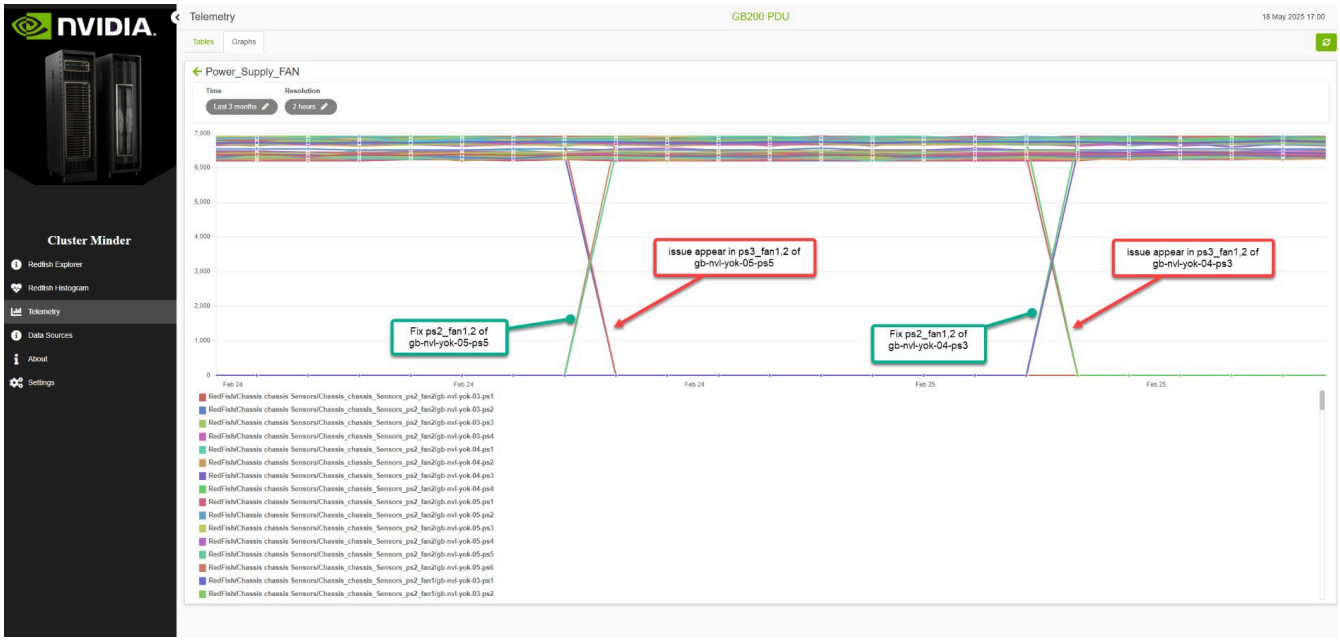


Telemetry Page

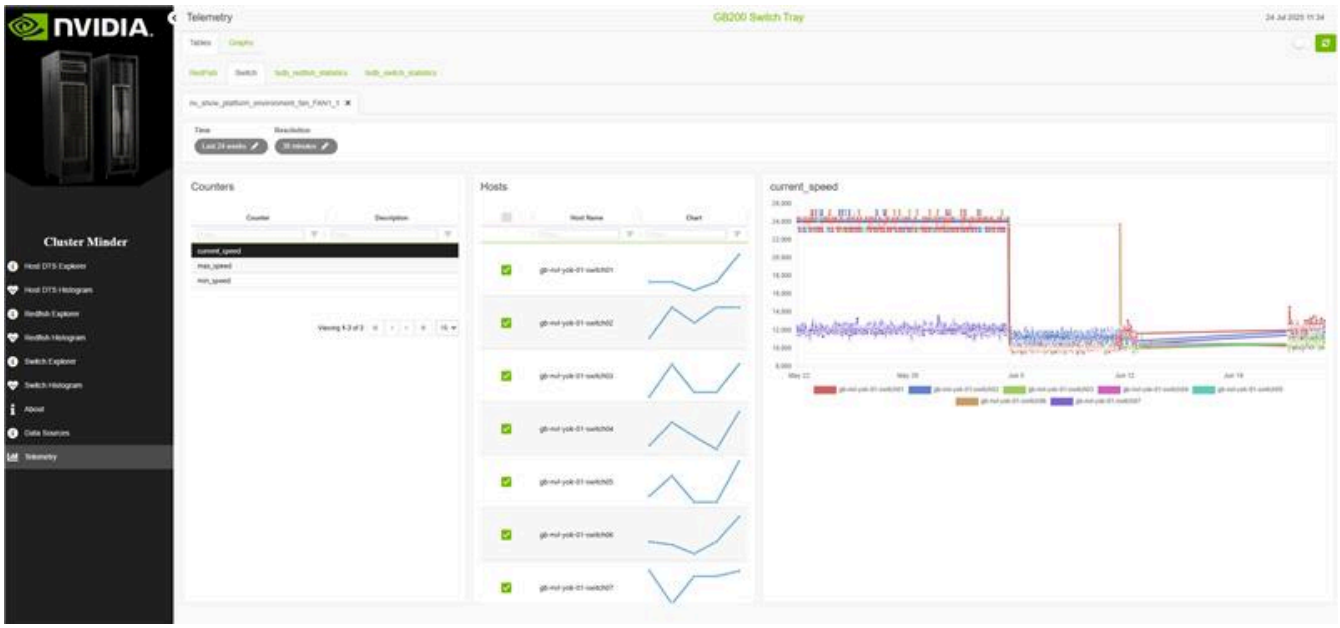
ClusterMinder's telemetry page allows users to create custom graphs based on their data. Additionally, there are premade graphs available depending on the telemetry data the data source provides. This feature provides flexibility in how data is displayed and analyzed, enabling users to tailor their graphs to meet specific needs.

> **Note**
>
> There is a switch button that enables tree view.

Redfish Telemetry Example



Switch Telemetry Example



Additional Features

Graphs Tab

Allows user to create a custom graph. To do so users should navigate to the “Graphs” tab.

Here, they can press the + New Graph button to initiate new graph creation. Next, users select a provider, device, and counter to add to the graph. Finally, they name the graph and complete the process by pressing the finish button. Graphs offer various options such as time frame and resolution, as well as built-in filtering by pressing the specific counter, allowing for more precise and customizable data visualization.

Dashboards

Dashboards in this project enable users to collect and organize specific graphs they create, grouping related visualizations within a single dashboard for easier access and comparison. This allows users to save sets of graphs that serve a common purpose and effortlessly switch between multiple dashboards to manage different groups of visualizations tailored to various needs.

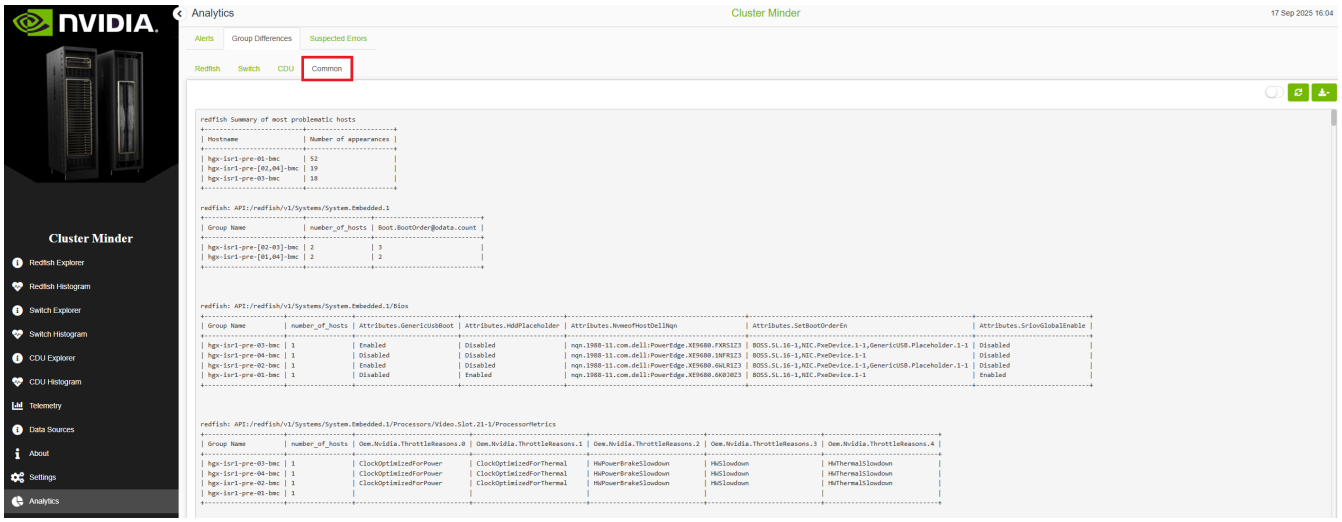
Dashboard example

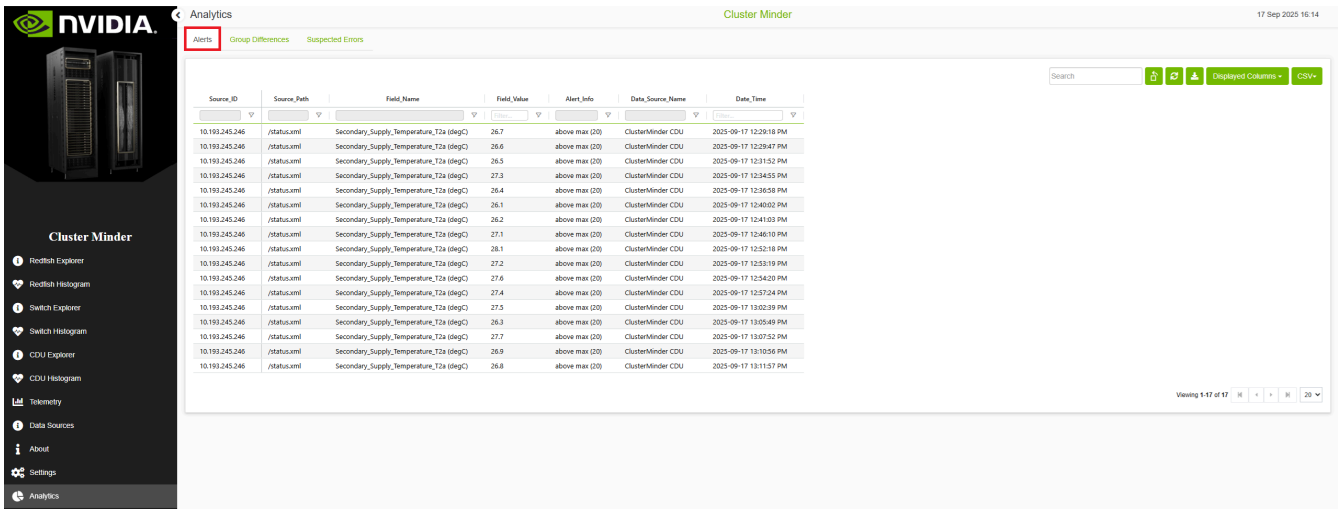


Analytics Page

ClusterMinder's analytics page allows users to view suspected errors and group differences reports in one aggregated location as well as viewing common suspected errors and common group differences. Analytics page also provides alerts (currently only CDU supported) for various user determined scenarios. This page provides ease of use viewing of all the text reports in a singular location as well as important user tailored alerts.

> **Note**
>
> There is a switch button that enables component view.





Sysinfo Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.1.1	N/A

Bug Fixes

Plugin Version	Bug Fix
1.1.1	N/A

Overview

The Sysinfo plugin is a Docker container that is managed by UFM and comes with REST API support. Its purpose is to allow users to run commands and extract information from managed switches. This feature enables users to schedule runs at regular intervals and execute commands on switches directly from UFM.

The plugin takes care of managing sessions to the switches and can extend them if necessary. It also enables users to send both synchronous and asynchronous commands to all the managed switches. Additionally, it can intersect the given switches with the running UFM to ensure that only those switches that are on the UFM are activated.

Deployment

The following are the possible ways plugin plugin can be deployed:

1. On UFM Appliance
2. On UFM Software.

3. Authentication

Following authentication types are supported:

- basic (/ufmRest)
- client (/ufmRestV2**)**
- token (/ufmRestV3)

REST API

The following REST APIs are supported:

- GET /help
- GET /version
- POST /query
- POST /update
- POST /cancel
- POST /delete

Sysinfo Query Format

The Sysinfo plugin is responsible for extracting basic data needed to create a query. This is done using the following five fields:

1. Switches - An array of switch IP addresses. If this field is left empty, the plugin will gather all switches from the running UFM.
2. Callback - The URL location to which the answers should be sent.
3. Commands - An array of commands that need to be executed.
4. Schedule_run - An optional field used to set intervals for running the commands. The interval can be specified in seconds and can be set to run until a certain duration or end time. The start time can also be controlled.

There are additional flags for a configurable query:

- `ignore_ufm=True:` Does not check the UFM for switches or intersect it with given switches
- `username`: Overrides the switches' default username
- `password`: Overrides the switches' default password
- `is_async`: Rather than attempting to execute all commands simultaneously at the switch, the commands are executed one after the other in sequence.
- `one_by_one=False:` Instead of sending results from each switch as soon as information is obtained, all data is sent at once to the callback. This change eliminates multiple small sends and replaces them with a single large send.

For detailed information on how to interact with Sysinfo plugin, refer to the NVIDIA UFM Enterprise > Rest API > Sysinfo Plugin REST API.

SNMP Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.0.0-10	N/A

Bug Fixes

Plugin Version	Bug Fix
1.0.0-10	N/A

Overview

The SNMP plugin is a self-contained Docker container that includes REST API support and is managed by UFM. Its primary function is to receive SNMP traps from switches and forward them to UFM as external events. This feature enhances the user experience by providing additional information about switches in the InfiniBand fabric via UFM events and alarms.

Deployment

There are two potential deployment options for the SNMP plugin:

- On UFM Appliance
- On UFM Software

For detailed instructions on how to deploy the SNMP plugin, refer to this page.

Authentication

The following authentication types are supported:

- basic (/ufmRest)
- client (/ufmRestV2**)**
- token (/ufmRestV3)

REST API

The following REST API are supported:

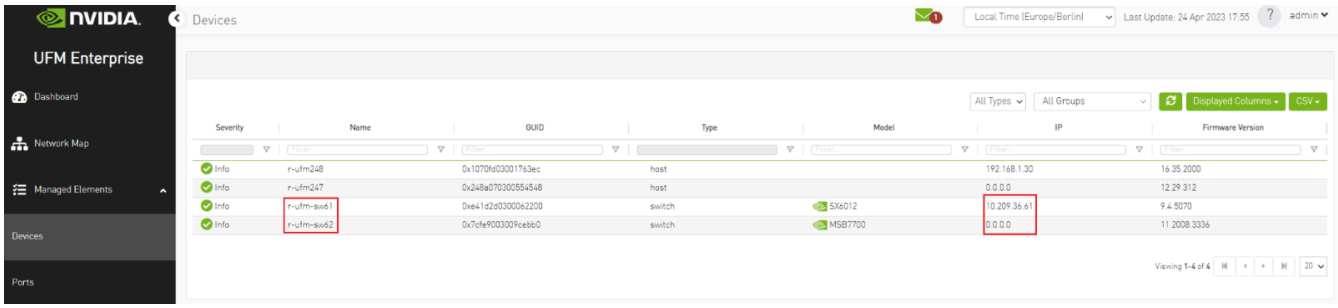
- GET /switch_list
- GET /trap_list
- POST /register
- POST /unregister
- POST /enable_trap
- POST /disable_trap
- GET /version

For more information, please refer to UFM Enterprise Documentation → UFM REST API → SNMP Plugin REST API.

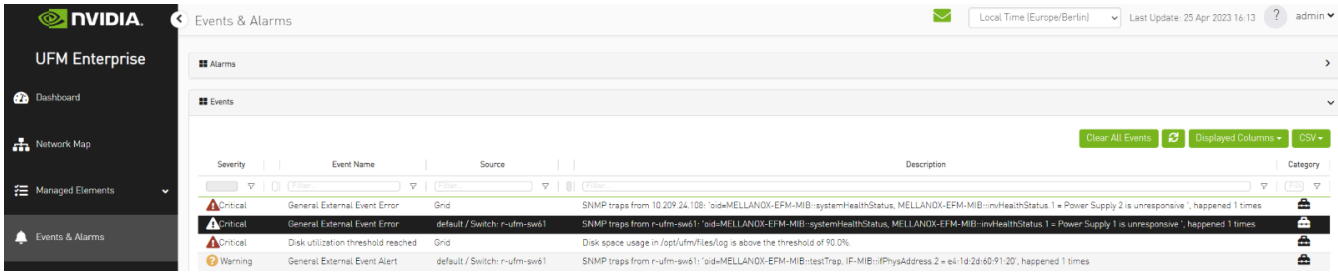
Usage

By default, upon initialization, the SNMP plugin captures traps from all switches within the fabric. However, this behavior can be modified through configuration settings utilizing the "snmp_mode" option, with available values of "auto" or "manual".

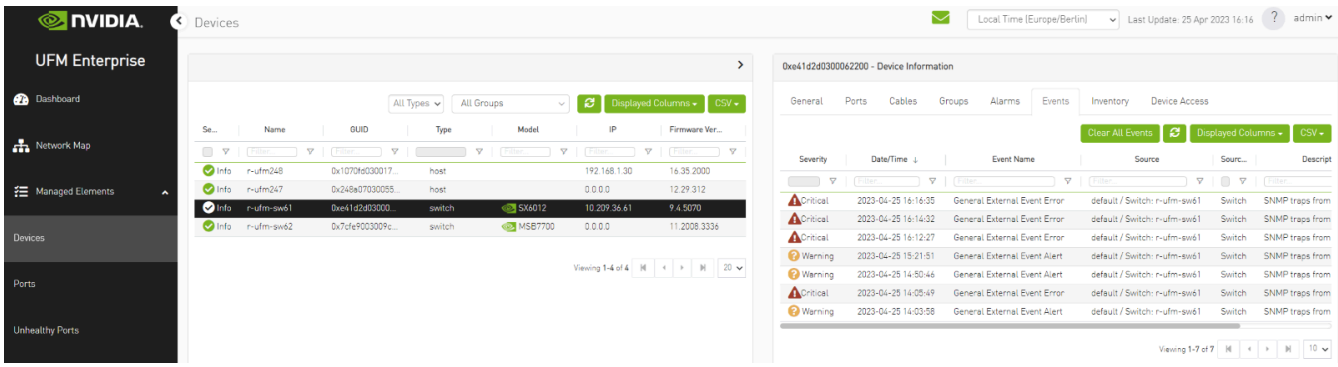
It is important to ensure that the switch is visible to UFM and has a valid IP address. As illustrated in the following example, switch traps will only be received from "r-ufm-sw61".



The following is an instance of a trap received by the SNMP plugin and displayed as a UFM event:



Additionally, there is an option to verify events/alarms for a particular switch:



The SNMP plugin performs a periodic check of the fabric every 180 seconds, allowing for prompt receipt of traps from new switches or updated IP addresses of existing switches in under 180 seconds. This interval may be adjusted via the "ufm_switches_update_interval" option. To manually register or unregister a switch, please refer to the UFM Enterprise Documentation → UFM REST API → SNMP Plugin REST API.

The SNMP plugin employs the most up-to-date SNMP v3 protocol, which incorporates advanced security measures such as authentication and encryption. The "snmp_version" option enables the selection of SNMP versions “1” or “3”. It is essential to note that only switch-exposed traps will be transmitted to UFM as events.

OID	Name	Description	Status	Severity
MELLANOX-EFM-MIB::testTrap	send-test	A test trap ordered by the system administrator	Enabled	Warning
MELLANOX-EFM-MIB::asicChipDown	asic-chip-down	ASIC (Chip) Down	Enabled	Critical
MELLANOX-EFM-MIB::cpuUtilHigh	cpu-util-high	CPU utilization has risen too high	Enabled	Warning
MELLANOX-EFM-MIB::diskSpaceLow	disk-space-low	Filesystem free space has fallen too low	Enabled	Warning
MELLANOX-EFM-MIB::expectedShutdown	expected-shutdown	Expected system shutdown	Enabled	Info
MELLANOX-EFM-MIB::systemHealthStatus	health-module-status	Health module Status	Enabled	Critical
MELLANOX-EFM-MIB::insufficientFans	insufficient-fans	Insufficient amount of fans in system	Enabled	Warning
MELLANOX-EFM-MIB::insufficientFansRecover	insufficient-fans-recover	Insufficient amount of fans in system recovered	Enabled	Info
MELLANOX-EFM-MIB::insufficientPower	insufficient-power	Insufficient power supply	Enabled	Warning
RFC1213::linkdown	interface-down	An interface's link state has changed to down	Enabled	Minor

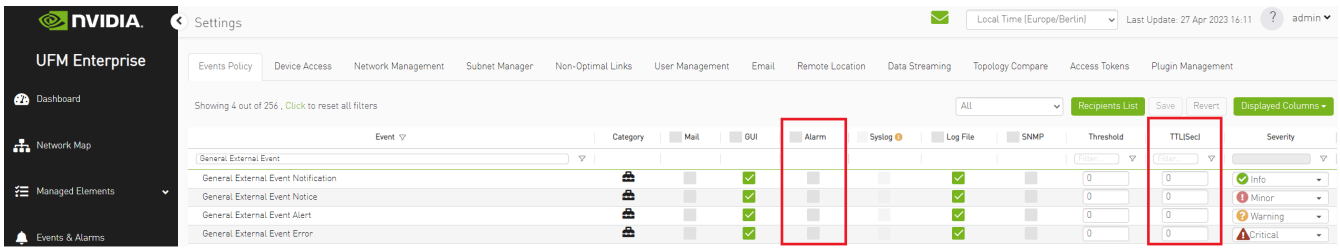
OID	Name	Description	Status	Severity
RFC1213::linkup	interface-up	An interface's link state has changed to up	Enabled	Info
MELLANOX-EFM-MIB::unexpectedShutdown	unexpected-shutdown	Unexpected system shutdown	Enabled	Minor
SNMPv2-MIB::coldStart	cold-start	SNMP entity reinitialized	Enabled	Info

To learn more about how to enable or disable a specific trap, please refer to the UFM Enterprise Documentation → UFM REST API → SNMP Plugin REST API.

If some traps are not included in the default list, they may be added using the "snmp_additional_traps" option. The SNMP plugin will consider these traps as "enabled" and transmit them to UFM as events with an "Info" severity level.

To ensure the uninterrupted reception of traps from switches within a large fabric, changes must be made to the UFM configuration in the [/opt/ufm/conf/gv.cfg] file's [Events] section. Specifically, the "max_events" option should be raised from 100 to 1000, while "medium_rate_threshold" and "high_rate_threshold" should both be set to 500. To implement configuration adjustments, disable and then enable the plugin.

In case of an event storm, it is necessary to adjust the Event Policy settings such that General Events are non-alarmable and the TTL is set to zero, as illustrated in the following screenshot:



Other

Additional configurations are located in "/opt/ufm/conf/plugins/snmp/snmp.conf". To implement configuration adjustments, disable and then enable the plugin. For instructions on modifying the appliance, please refer to the UFM-SDN Appliance Command Reference Guide .

Logs for the SNMP plugin are stored in "/opt/ufm/logs/snmptrap.log". For guidance on accessing logs on the appliance, please refer to the UFM-SDN Appliance Command

Reference Guide .

Packet Level Monitoring Collector (PMC) Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.19.33	N/A

Bug Fixes

Plugin Version	Bug Fix
1.19.33	• Fixes mirrored packets being sent only from a single ASIC on XDR switches.

Overview

The Packet Monitoring Collector/Controller plugin facilitates the configuration capture and display of a variety of events, enabling users to conduct real-time monitoring of network events. The PMC plugin is included in the plugins bundle, which can be downloaded from NVIDIA's Licensing Portal.

Supported triggers are pFRN, Congestion, Fast Recovery, CQE and PHY Error Links.

Network events are stored as UFM events and are archived in files for later retrieval. Additionally, they can be observed through the PMC user interface. Events can be streamed externally via UFM REST API in the same way that UFM events are streamed. The REST APIs are described in the UFM Enterprise REST API Guide.

pFRN

- pFRN Notifications - Enables/Disables mirroring on pFRN trigger for entire network or list of GUIDs

Fast Recovery

- Fast Recovery Notifications - Enables/Disables mirroring on Fast Recovery trigger for entire network or list of GUIDs
- Notifications Level - Specifies threshold for Fast Recovery mirroring. (Thresholds are configured in SM configuration)

PHY Error Links

- PHY Error Links Notifications - Enables/Disables mirroring on PHY Link Error trigger for entire network or list of GUIDs
- Specifies threshold for PHY Link Error mirroring. (Thresholds are configured in SM configuration)

CQE

- CQE Notifications - Enables/Disables mirroring on CQE Notifications trigger for entire network or list of GUIDs

Congestion

- Congestion Notifications - Enables/Disables mirroring on Congestion Notifications trigger for entire network or list of GUIDs
    - Mirrored packets (%) - Specifies the percent of congested packets to be mirrored.

- High threshold - High threshold percentage for InfiniBand switch egress port queue size. Values are in the [1,1023] range.
- Low threshold - Low threshold percentage for InfiniBand switch egress port queue size. Values are in the [1,1023] range.

Note

When a packet enters an InfiniBand switch, its data is stored at an ingress port buffer. A pointer to the packet's data is inserted into the egress port's queue, from which the packet will be exiting the switch. At that point, the threshold given by this command line argument is compared to the egress queue data size. If the queue data size exceeds the threshold, a congestion event is reported. The threshold is given in percent of the ingress port size.

An egress port queue can point data coming from multiple ingress port buffers, therefore the threshold can be bigger than 100%.

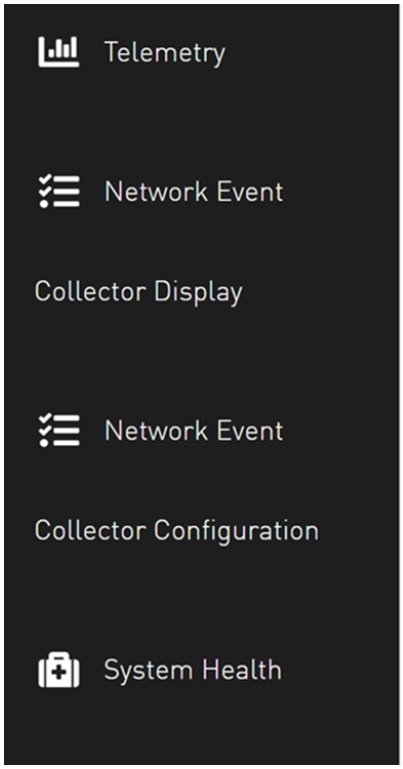
Deployment

Installation

Load the image on the UFM server; either using the UFM GUI -> Settings -> Plugins Management tab, or by loading the image via the following command:

1. Login to the UFM server terminal.
2. Run

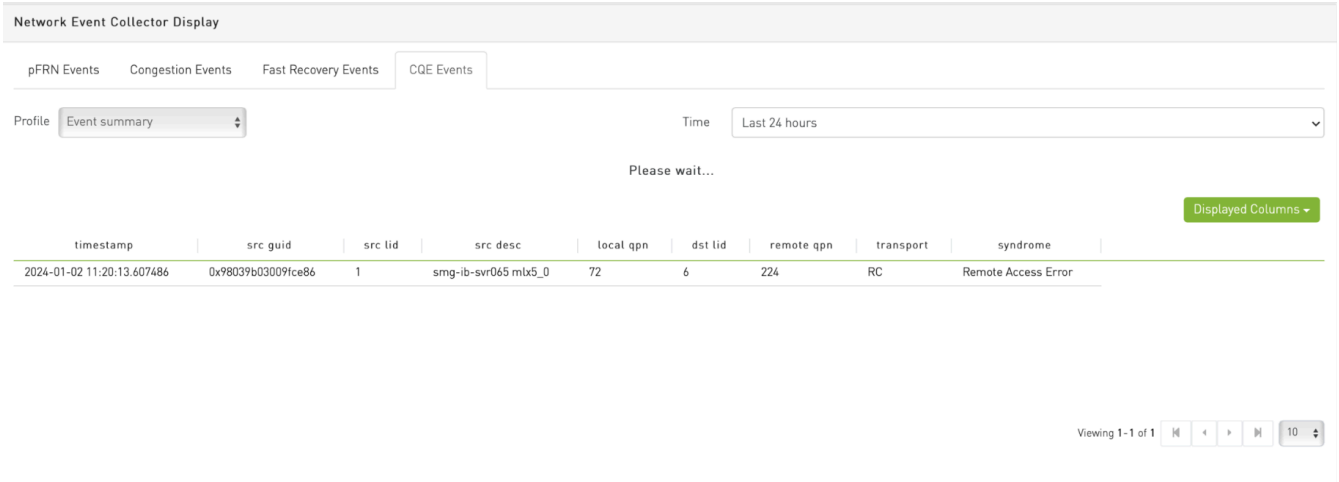
```
docker load -I <path_to_image>
```



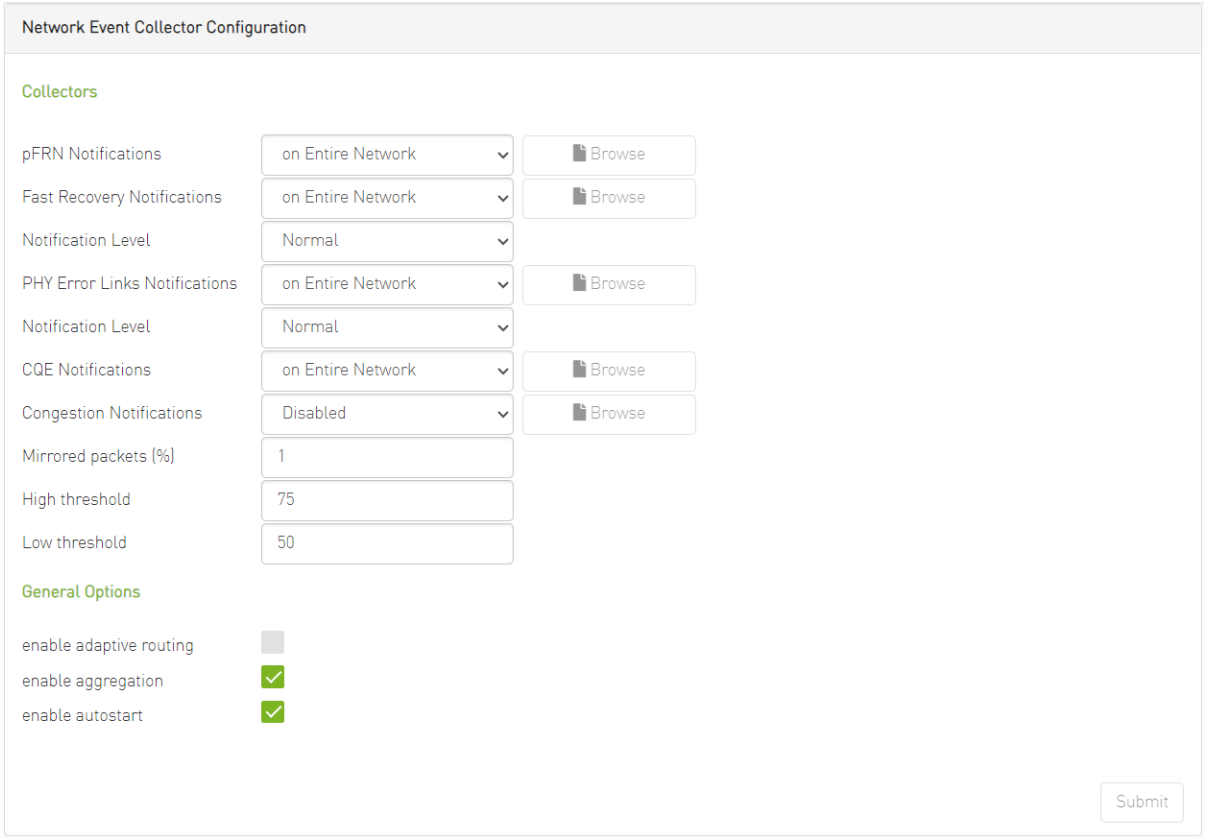
Upon completion of the plugin addition and subsequent refresh of the UFM GUI, the left navigation bar will display two new menu items. These two tabs can be observed in the following GUI screenshots

PMC UI

Network Event Collector Display



Network Event Collector Configuration



GNMI-Telemetry Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Changes and New Features
1.3.8-5	• Added the capability for UFM to stream events in JSON format through gNMI with the option to include the device information (events related to the device or the port). For more information, refer to GNMI-Server Configurations (Added the `include_dev_details_in_events` parameter). To activate the ability to stream the device information, ensure you have UFM version 6.23.1 is installed.
1.3.8-4	N/A
1.3.8-3	• Added an option to disable the HTTP server through configuration. Introduced the `enable_http_server=true` option under the `[HTTP-Server]` section in `gnmi_telemetry.ini`. • Enhanced general logging and improved default configuration settings. • Optimized dynamic XCSET group loading in the gNMI service, significantly reducing start-up time and removing unclear log entries. The gNMI service now fetches only the configured endpoints and `ib.port_counters`, instead of retrieving all counter sets.

Bug Fixes

Plugin Version	Bug Fixes
1.3.8-5	N/A
1.3.8-4	• Fixed a bug that caused the gNMI service to crash due to excessive memory consumption. • Removed the data snapshot mechanism during data retrieval, which was causing high memory consumption. • Cleaned the orphaned goroutines that held locks indefinitely.

Plugin Version	Bug Fixes
1.3.8-3	• Fixed telemetry fetch overlap that occurred when telemetry notifications were received. • Resolved the “port in use” error by adding a fallback mechanism for the HTTP client’s static port. • Added queue depth protection — implemented queue depth monitoring with a 50-message limit to prevent out-of-memory (OOM) conditions. • Added HTTP connection leak prevention. • Improved resource cleanup, including enhanced handling of client references, message queues, and goroutines. • Enhanced logging and monitoring for better observability. • Fixed stale port TTL timing calculation to ensure ports are only reported as down after the correct grace period. • Resolved lock contention issues and prevented potential deadlocks during concurrent data access.

Known Issues

Plugin Version	Known Issues
1.3.8-5	N/A
1.3.8-4	The gNMI plugin Docker container requires a **minimum of 12 GB of RAM on the host system** for successful deployment and stable operation.

Overview

The GNMI Telemetry Plugin is a server that uses the gNMI protocol to stream data from UFM telemetry. Users can select the data to stream, specify intervals, and choose to include only deltas (on-change mode).

The server supports three functions: Capability, Get, and Subscribe.

Data Streaming: The streamed data is delivered in CSV format. Headers are provided in the first message and included in subsequent messages. Data is presented in hex format to conserve space for unchanged data. Values are displayed as an array of strings, each

representing a unique identifier (GUID) and port. Depending on the mode, values may have missing rows if there are no changes in the GUID and port.

Metadata Streaming: The plugin can stream UFM's metadata, providing an inventory of it. For convenience, examples use the gNMIc client, but any gNMI client can be used.

Configuration and Polling Intervals: The polling intervals for each server cache are configurable with the following defaults:

- Telemetry: every 5 minutes
- Inventory: every minute
- Events: every minute
- Switch rank: every 6 hours
- UFM Health KPI: every 5 minutes

The service supports telemetry from switch-level data (fset) and port-level data (xcset), querying `low_freq_debug` xcset by default. Multiple telemetries can be polled simultaneously.

Data Sharding: The service supports sharding the cache data on request, allowing many clients to request the same data while each receives a different part.

Deployment

To deploy the plugin with UFM (SA or HA):

1. Install the latest version of UFM.
2. Run UFM with `/etc/init.d/ufmd start`.
3. Pull the plugin image from the Docker Hub.
4. Run
   ```
   /opt/ufm/scripts/manage_ufm_plugins.sh add -p gnmi_telemetry -t <version>
   ```
   to enable the plugin, or use the UFM UI to add the plugin via Settings → Plugin Management → Right Click on GNMI-telemetry → Add → select version → Add.
5. Check that the plugin is running with `docker ps`.

6. If the gNMI default port is unavailable, change the configuration file `gnmi_telemetry.ini` and restart the plugin.

Configurations

The `/opt/ufm/files/conf/plugins/gnmi_telemetry/gnmi_telemetry.ini` file centralizes the configuration of the GNMI-Telemetry Plugin, allowing users to customize logging, server behavior, telemetry intervals, security settings, and more. Below is an overview of the available configuration parameters, their default values, and their purpose within the plugin.

Common Configurations

Parameter	Description	Default Value
`log_level`	Sets the logging level for the plugin (e.g., INFO, DEBUG).	INFO
`log_file_max_size_MB`	Maximum size of the log file before it rotates, measured in MB.	10
`log_file_backup_count`	Number of backup log files to retain after rotation.	5
`log_file_name`	Full path of the log file	/log/gnmi_streaming.log

GNMI-Server Configurations

Parameter	Description	Default Value
`grpc_port`	Port on which the gRPC server listens for incoming connections.	9339
`cpu_list`	Specifies which CPU cores the server can use (comma-separated for multiple cores, e.g., 3,5,6).	3
`memory_usage_threshold`	Specifies the server memory usage threshold for forcing garbage collection in MB	500

Parameter	Description	Default Value
`data_directory`	data directory, this directory used by the server and it will contain all the necessary files that generated by the server. DON'T MODIFY IT in case the gNMI server is running as ufm-plugin container	/data
`include_port_sharding`	Determines whether the port number is included in the sharding algorithm. Ports of the node may not be in the same shard	false
`include_old_data_on_change`	Includes previous data when sending notifications in `on_change` mode.	true
`strict_collected_counters`	Enforces strict rules for collecting counters based on client requests.	false
`disable_events_inventory`	Disables the collection and delivery of events and inventory data.	false
`include_dev_details_in_events`	If true, the GNMI server will Include device details in events notification response (node_guid, node_name, node_type, peer details, etc.). **Note** This feature is available in UFM version 6.23.1 or later.	true

Parameter	Description	Default Value
`force_heartbeat_notification`	In Subscribe on-change mode, setting this flag to `true` ensures that the server sends a heartbeat notification at every interval, even if no updates occur during the heartbeats.	false
`include_headers_on_every_update`	In Subscribe mode, headers are included only in the first update by default. If set to true, headers will be included in every update.	false
`serial_telemetry_fetch`	If true fetches telemetry serially (one by one) from all the telemetry endpoints.	false
`serial_telemetry_interval`	The gNMI telemetry fetch interval when applying serial telemetry fetch.	300s
`rest_retry_attempts`	Number of retry attempts for failed REST API calls	3
`rest_retry_backoff`	Exponential backoff duration between REST call retry attempts (e.g. 2s, 4s, 6s)	2s
`disable_cache_cleanup`	If false, the cache will be cleaned up according to the cleanup policy. If true, the cache will not be cleaned up.	false

Parameter	Description	Default Value
`port_stale_time_multiplier`	The multiplier for the stale time of a port. The stale time is the time after which the port is considered stale and will be removed from the cache. The stale time is calculated as the maximum of the telemetry interval from all the endpoints multiplied by the stale time multiplier, which means the number of telemetry fetch iterations wait before the port is considered stale and will be removed from the cache.	3

HTTP Server Configurations

The gNMI plugin includes a built-in HTTP Server that enables event-driven data synchronization between UFM Telemetry endpoints and the gNMI server. This real-time communication complements the existing periodic telemetry fetching mechanism controlled by the `telemetry_interval` parameter.

Parameter	Description	Default Value
`enable_http_server`	If true, the HTTP server will be enabled	true
`http_port`	Port for the HTTP server to listen on	9338

Telemetry Configurations

Parameter	Description	Default Value
`notification_server_path`	Path for the telemetry notification HTTP endpoint	/telemetry/notify/

Parameter	Description	Default Value
`throtling_interval`	Throttling interval - minimum time between processing notifications (in seconds)	10s

For more details about the real-time data synchronization between UFM Telemetry and gNMI server and how to enable it in Telemetry, please refer to section UFM Telemetry Notification Subscription.

Time-Intervals Configurations

Parameter	Description	Default Value
`events_interval`	Time interval for collecting events, specified in seconds (e.g., `60s`).	60s
`inventory_interval`	Time interval for collecting inventory data, specified in seconds (e.g., `60s`).	60s
`minimum_sample_rate`	Minimum sampling interval for telemetry notifications, specified in seconds.	10s
`timeout_rest_call`	Timeout for REST API calls, specified in seconds.	30s
`switch_rank_interval`	Interval for monitoring changes in the switch rank datasource file, specified in hours.	6h
`ufm_health_kpi_interval`	Interval for monitoring system health metrics,specified in seconds.	300s

Telemetry Cluster-Specific Configurations

Each cluster should have its own section named `cluster-config-$cluster_name`.For example, `[cluster-config-low_freq_debug]`. you can add multiple sections in case you need to collect data from multiple clusters/telemetries simultaneously.

By default, the plugin comes with a single cluster to collect data from `http://127.0.0.1:9002/csv/xcset/low_freq_debug`.

Parameter	Description	Default Value
`telemetry_endpoint_url`	URL for the telemetry endpoint for the cluster.	`http://127.0.0.1:9002/csv/xcset`
`default_id_cols`	Default identifier columns applied to all rows (ports) for telemetry data.	Node_GUID,Port_Number
`id_type_col`	Telemetry column used to determine identification schemes based on its value.	N/A
`id_cols_<type>`	Identifier columns for rows where `id_type_col` has a specific value (e.g., `id_cols_1`).	N/A
`per_slvl`	Enables `per_slvl` counters for telemetry if supported by the configuration.	false

Parameter	Description	Default Value
`telemetry_interval`	Interval for sending telemetry data to the endpoint, specified in seconds.	300s
`filtered_columns`	Columns to exclude from telemetry data, separated by commas.	port_guid

UFM Configurations

Parameter	Description	Default Value
`default_inventory`	Default inventory values if UFM is unavailable, specified as a JSON string.	{"Servers":8,"Switches":4,"HCAs":4,"A
`ufm_ip`	IP address of the UFM instance used for inventory data.	127.0.0.1
`ufm_access_token`	Access token for authenticating with UFM if running on a different host.	N/A

Parameter	Description	Default Value
`ufm_users_cache_ttl`	Interval for refreshing the UFM users and roles cache. Increase this value if changes to UFM user or roles are infrequent.	10 minutes
`switch_rank_datasource_file`	File path for the switch rank datasource used by UFM.	/opt/ufm/files/log/opensm-smdb.du

GNMI-Security Configurations

Parameter	Description	Default Value
`secure_mode_enabled`	Enables secure mode for gNMI.	true
`client_cert_subject_identifier`	Specifies the certificate subject identifier (SAN or CN).	SAN
`authorized_roles`	Comma-separated list of UFM authorized users roles that can access gNMI, if empty, all users are allowed to access	
`initial_check_time`	Time of day for the initial certificate validation check, specified as HH:mm.	01:30
`check_interval`	Interval for periodic certificate validation checks, specified in hours.	12h

XDR Configurations

Parameter	Description	Default Value
`xdr_mode`	Enables XDR Mode Setup	false
`xdr_ports_types`	Types of XDR ports to collect, separated by commas (e.g., `legacy,aggregated,plane`).	legacy,aggregated,plane

Authentication

The server's authentication is determined by the gNMI protocol. Two configurable items require authentication: the UFM Telemetry URL and the UFM inventory IP.

- Authentication is not necessary for the UFM telemetry URL. Therefore, only the telemetry URL is required.
- The inventory is sourced from the UFM of the local host, but can be changed to a different machine in the config file. To do so, token access to that machine is necessary.

Secure Server using mTLS and Certificate Subject Identifier

The gNMI server can be secured using certificate. To secure the server, set the "`secure_mode_enabled`" flag to "true" in the configuration (default is true).

The certificate must be placed under the `/opt/ufm/files/conf/webclient` folder and can be changed by modifying the shared volume. The gNMI server periodically checks its certificates for updates, ensuring they remain up-to-date. The client certification naming convention must align with the DNS name (SAN) as the UFM.

The gNMI plugin supports certificate subject identifier (the default value is SAN). Configure the certificate subject identifier under the gNMI-security section to be SAN or CN (Common Name). For example: `client_cert_subject_identifier=CN.`

Role-Based Access Control

The UFM gNMI plugin supports Role-Based Access Control (RBAC) to enable granular, user-based authorization for gNMI operations. By leveraging existing UFM user management and certificate-based (mTLS) authentication, the plugin enforces access policies according to user role.

Authentication and User Mapping

- The gNMI server uses mTLS certificates for secure connections, leveraging the UFM certificate infrastructure.
- Each UFM user is associated with an mTLS certificate by mapping the certificate subject identifier (SAN or CN) to a UFM username. The association information is stored in `/opt/ufm/files/conf/webclient/ufm_client_authen.db`
- Refer to Client-Based Authentication for configuring client authentication and user association in UFM.

Role Cache and Management

- On plugin's startup, the gNMI server queries the UFM Users API to obtain the current list of users and their assigned roles (groups).
- The user-role mapping is cached to reduce API load, The refresh interval is configurable via the `ufm_users_cache_ttl` parameter (default: 10 minutes).
- Any changes to UFM users or roles are reflected after the next cache refresh.

Configuring RBAC

- Use the `authorized_roles` configuration option to control which UFM roles are permitted to access the gNMI server. Example: `authorized_roles=System_Admin,Monitoring_Only,Custom_Role1`
- Only users whose role matches one of the authorized roles will be allowed to access or execute gNMI operations.

Access Control Flow

1. **Connection Attempt**: A gNMI client initiates a connection using an mTLS certificate.
2. **Certificate Validation**: The gNMI server validates the client certificate.
3. **Subject Identifier Extraction**: The server extracts the subject identifier (as configured—SAN or CN) from the certificate.
4. **User Association**: Uses the identifier to look up the UFM user in `ufm_client_authen.db`
5. **Role Lookup**: The user's role is fetched from the cached UFM user-role mapping.
6. **Authorization Check**: Access is granted if the user's role matches one in `authorized_roles`; otherwise, it is denied.

Supported API Requests

The service supports the following requests:

- **Capability**: Describes the YANG files the service supports (UFM telemetry).
- **Get**: Requires legal paths; receives the cache data from the service.
- **Subscribe**: Requires legal paths and an interval; receives cache data at the specified interval. The first message contains headers extracted from the path, and subsequent messages include only the headersID. In on-change subscribe mode, a heartbeat interval is provided instead of an interval. During the heartbeat interval, if no data changes, no notification is sent; A full notification message, similar to the first message, is sent. If some data changes a notification of the change is sent; No heart message is send.

Capability Request

The capability request provides information about the YANG files that the server supports, including their versions. This request can be fulfilled without requiring a connection to the telemetry or inventory.

Request Example:

```
gnmic -a localhost:9339 capability
```

Response Example:

```
gNMI version: 1.3.0-2
supported models:
  - nvidia-ib-amber, Nvidia IB, 1.0.0
  - nvidia-ib-amber-ext, Nvidia IB, 1.0.0
  - nvidia-ib-amber-inventory-counters, Nvidia IB, 1.0.0
  - nvidia-ib-amber-port-counters, Nvidia IB, 1.0.0
supported encodings:
  - JSON
  - JSON_IETF
```

Supported Paths

Telemetry Request Path Construction

To construct a path for a telemetry request, follow these steps:

1. Begin with "`nvidia/ib`".
2. Specify sharding if desired. For example, to partition the data into 10 pieces and take the second partition, use 2/10.
3. Specify the `node_guid` to select, using an asterisk (*) to select all nodes.
4. Specify the desired ports for the selected nodes, using an asterisk (*) to select all ports.
5. Select "`amber`" for amBER telemetry.
6. Specify the desired counters group. If unknown, this step can be skipped.

7. Specify the counter, using an asterisk (*) to select all the counters in the cache. If a counters group is used, it will return all counters in the specified group.

Other Information Requests (Events, Inventory)

1. Begin with "`nvidia/ib`".
2. Specify `inventory` or `events`.

Switch Rank Information Path Construction

To construct a path for switch rank information, follow these steps:

1. Begin with `"nvidia/ib"`.
2. Specify the `node_guid` to select, using an asterisk (*) to select all nodes.
3. Select `"amber"` for amBER telemetry.
4. Use `Switch_rank` as the counter name.

Telemetry Messages - Data Format

Telemetry messages consist of two key components: Headers and Values, both representing telemetry data in a CSV format.

- **Headers**: Initially provided in a full mode, but transition to a string hash format after the second message when using a subscribe request to reduce message size.
- **Values**: Each value begins with a timestamp, followed by the `node_guid` and port number, and then the counter value in the same order as the headers. If a counter is not present for a node, it will be empty in the message.

In on-change subscribe messages, only nodes with changes and their corresponding modified values are included. All other counters for that node will remain empty.

Request Example:

```
gnmic -a localhost:9339 --insecure sub --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/port_counters/hist
--path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/port_counters/hist
-i 30s
```

Response Example:

```
[
   {
      "source": "localhost:9339",
      "subscription-name": "default-1690282472",
      "timestamp": 1690282475124352000,
      "time": "2023-07-25T13:54:35.124352063+03:00",
      "updates": [
         {
            "Path": "nvidia/ib/amber/reply/sample",
            "values": {
               "nvidia/ib/amber/reply/sample": {
                  "Headers": "timestamp,guid,port,hist0,hist1",
                  "HeaderID": "5246201354",
                  "Values": [
                     "240771222771818,0x8168793592c6a790,1,,2",
                     "240771222771818,0x47a67159c915493f,1,1,2",
                     "240771222771818,0x667203ac69f3f2bf,1,2,",
                     "240771222771818,0x113cd807bfed3853,1,0,"
                  ]
               }
            }
         }
      ]
   }
]
```


The second message on the headers will be set to hash values.

GET Request

The Get request retrieves data at a specified path. If the telemetry is devoid of information, the server will respond with an empty response. Otherwise, it will respond with counters it can locate.

Example:

```
gnmic -a localhost:9339 --insecure get --path
nvidia/ib/guid[guid=0x5255456]/port[port_number=2]/amber/port_coun
```

The request retrieves data from node_guid `0x5255456`, specifically in port number 2, with the request counter set to hist0.

Example 2:

```
gnmic -a localhost:9339 --insecure get --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/port_counters/his
```

The request retrieves the data from all the ports and the node_guids, with the request counter set to hist0.

Example 3:

```
gnmic -a localhost:9339 --insecure get --path
nvidia/ib/guid[guid=0x5255456]/port[port_number=2]/amber/*
```

The request retrieves the data from node_guid `0x5255456`, port 2, with the request counters set to "all".

Example for multi path:

```
gnmic -a localhost:9339 --insecure get
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/CableInfo.transmi
--path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/sel_gctrln_en_5_la
--path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/num_plls_7nm --
path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/rcal_fsm_done --
path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/LinkErrorRecovery(
--path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/sel_enc2_ib0_lane2
--path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/lockdet_err_cnt_ur
```

Response Example:

```
[
   {
      "source":  "localhost:9339",
      "timestamp":  1719232374915165200,
      "time":  "2024-06-24T15:32:54.915165166+03:00",
      "updates":  [
         {
            "Path":  "nvidia/ib/amber/reply",
            "values":  {
               "nvidia/ib/amber/reply":  {
                  "Headers":  [
                     "timestamp",
                     "Node_GUID",
                     "Port_Number",
                     "CableInfo.transmitter_technology",
                     "sel_gctrln_en_5_lane0",
                     "num_plls_7nm",
                     "rcal_fsm_done",
                     "LinkErrorRecoveryCounterExtended",
                     "sel_enc2_ib0_lane2",
                     "lockdet_err_cnt_unlocked_sticky"
                  ],
                  "Values":  [

"1719232345757948,0x91f87bf42deb3e03,1,5091,7826,6290,8615,4247,8586,6214",

"1719232345757948,0x7b8c2e08907250ce,1,2891,3293,5774,4398,3681,3548,7408",

"1719232345757948,0x48b60e6f3670eaca,1,9477,3847,1184,5527,4783,2102,8192",

"1719232345757948,0xabccdad7f8a3eda6,1,7976,6143,8257,3770,6166,6690,2835",

"1719232345757948,0x6d9ec4bb5fa45736,1,9051,2982,7145,3604,9256,1061,2638",

"1719232345757948,0x028cf9e0f9ed7c32,1,5623,7483,2263,2265,6890,4875,5564",
```

```
"1719232345757948,0x92a984c1a491b72a,1,6732,7795,6411,8569,3370,705,5536" ,

"1719232345757948,0x8b4b404acd2f34da,1,7610,7128,10064,1880,4834,3411,6724" ,

"1719232345757948,0x20f92ed58991d56c,1,6805,1632,5407,2038,1865,7279,8350" ,

"1719232345757948,0x1dac004a426bb5f5,1,8351,5757,7925,6181,3260,3081,1554"
                ]
             }
          }
       }
    ]
  }
]
```

Subscribe Stream Request

The Subscribe request, similar to the get request, provides data from the specified path. When the telemetry is empty, the server responds with an empty result. If data is available, the server responds with the retrieved counters. The stream delivers information at the specified interval. If no interval is specified, the server transmits the information at the default server rate, which is configurable and defaults to 10s.

Example:

```
gnmic -a localhost:9339 --insecure sub --path
nvidia/ib/guid[guid=0x5255456]/port[port_number=2]/amber/port_coun
-i 30s
```

This request retrieves data from the node_guid `0x5255456`, port 2, where the request counter is hist0, and the interval is configured for 30 seconds. If the user wishes to test the stream, the stream mode can be configured to "once," and following a single response, the stream will be stopped.

Example:

```
gnmic -a localhost:9339 --insecure sub --path
nvidia/ib/guid[guid=0x5255456]/port[port_number=2]/amber/port_coun
-i 30s --mode once
```

This request retrieves the data from node_guid `0x5255456`, port 2, where the request counter is hist0. The stream shuts down after one response, similar to a Get request.

Example:

```
gnmic -a localhost:9339 --insecure sub --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/* -i 10s
```

The server responds for the first two notifications, as follows:

```
{
   "source": "localhost:9339",
   "subscription-name": "default-1719233128",
   "timestamp": 1719233128171946500,
   "time": "2024-06-24T15:45:28.171946518+03:00",
   "updates": [
      {
         "Path": "nvidia/ib/amber/reply/sample",
         "values": {
            "nvidia/ib/amber/reply/sample": {
               "HeaderID": "970426048",
               "Headers": [
                  "timestamp",
                  "Node_GUID",
                  "Port_Number",
                  "Counter1",
                  "Counter2",
                  "Counter3",
                  "Counter4",
                  "Counter5",
                  "Counter6",
                  "Counter7"
               ],
               "Values": [
                  "1719232345757948,0x91f87bf42deb3e03,1,5091,7826,6290,8615,4247,8586,6214",
                  "1719232345757948,0x7b8c2e08907250ce,1,2891,3293,5774,4398,3681,3548,7408",
                  "1719232345757948,0x1dac004a426bb5f5,1,8351,5757,7925,6181,3260,3081,1554",
                  "1719232345757948,0x48b60e6f3670eaca,1,9477,3847,1184,5527,4783,2102,8192",
                  "1719232345757948,0xabccdad7f8a3eda6,1,7976,6143,8257,3770,6166,6690,2835",
                  "1719232345757948,0x6d9ec4bb5fa45736,1,9051,2982,7145,3604,9256,1061,2638",
                  "1719232345757948,0x028cf9e0f9ed7c32,1,5623,7483,2263,2265,6890,4875,5564",
                  "1719232345757948,0x92a984c1a491b72a,1,6732,7795,6411,8569,3370,705,5536",

"1719232345757948,0x8b4b404acd2f34da,1,7610,7128,10064,1880,4834,3411,6724",
                  "1719232345757948,0x20f92ed58991d56c,1,6805,1632,5407,2038,1865,7279,8350"
```

```
                ]
            }
        }
    }
]
}
{
    "source": "localhost:9339",
    "subscription-name": "default-1719233128",
    "timestamp": 1719233138173907700,
    "time": "2024-06-24T15:45:38.173907825+03:00",
    "updates": [
        {
            "Path": "nvidia/ib/amber/reply/sample",
            "values": {
                "nvidia/ib/amber/reply/sample": {
                    "HeaderID": "970426048",
                    "Values": [
                        "1719232345757948,0x20f92ed58991d56c,1,6805,1632,5407,2038,1865,7279,8350",
                        "1719232345757948,0x1dac004a426bb5f5,1,8351,5757,7925,6181,3260,3081,1554",
                        "1719232345757948,0x48b60e6f3670eaca,1,9477,3847,1184,5527,4783,2102,8192",
                        "1719232345757948,0xabccdad7f8a3eda6,1,7976,6143,8257,3770,6166,6690,2835",
                        "1719232345757948,0x6d9ec4bb5fa45736,1,9051,2982,7145,3604,9256,1061,2638",
                        "1719232345757948,0x028cf9e0f9ed7c32,1,5623,7483,2263,2265,6890,4875,5564",
                        "1719232345757948,0x92a984c1a491b72a,1,6732,7795,6411,8569,3370,705,5536",

"1719232345757948,0x8b4b404acd2f34da,1,7610,7128,10064,1880,4834,3411,6724",
                        "1719232345757948,0x91f87bf42deb3e03,1,5091,7826,6290,8615,4247,8586,6214",
                        "1719232345757948,0x7b8c2e08907250ce,1,2891,3293,5774,4398,3681,3548,7408"
                    ]
                }
            }
        }
    ]
}
```

Subscribe On-Change Request

The subscribe on-change request, similar to the standard subscribe request, provides data from the specified path. If the telemetry lacks data, the server responds with an empty result. When data is available, the server responds with the located counters.

The stream delivers information at the specified interval. If no changes occurred between heartbeats, all cached data will be transmitted**.** However, if a change occurred and was pushed to the client, no data will be transmitted during the heartbeat.

The path construction follows the same pattern as the get request and includes inventory and event paths. Only updated data will be included in the response, while all other parts remain empty but retain the specified format. Similarly, only the nodes that have been updated will be included in the response.

Example:

```
gnmic -a localhost:9339 --insecure  sub --path
nvidia/ib/guid[guid=0x5255456]/port[port_number=2]/amber/port_coun
 --stream-mode on-change --heartbeat-interval 1m
```

This request retrieves data from node_guid `0x5255456`, port 2, with the request counters set to hist0. It periodically checks for changes every minute, and when changes are detected, it promptly sends the updated values.

Example:

```
gnmic -a localhost:9339 --insecure  sub --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/port_counters/*
--stream-mode on-change --heartbeat-interval 1m
```

This request involves all nodes and ports, aiming to retrieve all counters from the telemetry. It periodically checks for changes every minute, and when changes are detected, it promptly sends the updated values.

The below is an example of the response to a particular GUID, which represents an on-change request for a few counters. However, only specific counters have been updated, those who have not updated have a value of 0. Because the flag `include_old_data_on_change` default is true

```
1706532307824,0x0002c903007e5220,1,0,0,0,41447490564,617155163,41423305825,617155163
```

The same example with the flag set to `false` will give this:

```
1706532307824,0x0002c903007e5220,1,,,,41447490564,617155163,41423305825,617155163,24
```

Only the values that have changed return while the others are empty values. To get this format of data, one need to change the `include_old_data_on_change` in the config file to false.

Example:

```
gnmic -a localhost:9339 --insecure sub --path
nvidia/ib/guid[guid=*]/port[port_number=*]/amber/* --stream-mode
on-change --heartbeat-interval 24h
```

The server responds for the first 2 notifications are the following (where `include_old_data_on_change` is true), one can see the last two columns have not changed but still return the data before, the second message was send due to some rows have changed, those rows

```
{
    "source": "localhost:9339",
    "subscription-name": "default-1719236764",
    "timestamp": 1719236764654659600,
    "time": "2024-06-24T16:46:04.654659517+03:00",
    "updates": [
        {
            "Path": "nvidia/ib/amber/reply/onchange",
            "values": {
                "nvidia/ib/amber/reply/onchange": {
                    "HeaderID": "912200528",
                    "Headers": [
                        "timestamp",
                        "Node_GUID",
                        "Port_Number",
                        "Counter1",
                        "Counter2",
                        "Counter3",
                        "Counter4",
                        "Counter5",
                        "Counter6",
                        "Counter7"
                    ],
                    "Values": [

"1719236753818594,0x7e680fb8f81a1950,1,100531,107250,100999,107455,109258,3716,5329",

"1719236753818594,0x0176438fe4ee507c,1,104269,108884,104887,108502,105366,4540,6673",

"1719236753818594,0x2e36224302959e79,1,101228,100555,105616,102767,108899,87,9953",

"1719236753818594,0x8e62a55d7571a9b8,1,100684,108124,106670,102400,106689,2910,4203",

"1719236753818594,0x0be75a9e97016f5e,1,102227,102735,108903,103547,108705,2629,1830",
```

```
"1719236753818594,0x8307bfad0672adbd,1,106033,103906,106185,107450,105736,2567,6914",

"1719236753818594,0x2cbe66ec0b1af84c,1,105958,106959,100349,107704,105073,8330,4962",

"1719236753818594,0x6b6da39a9ec4bbfc,1,104340,106752,109134,103796,103500,7136,3493",

"1719236753818594,0x6d122dbdd99cfb60,1,104941,107630,104190,105392,109582,5480,7934",

"1719236753818594,0xeed4bd9cd3b7f325,1,102416,100164,106731,102033,103807,3048,6316"
                ]
            }
        }
    }
]
}
{
  "source": "localhost:9339",
  "subscription-name": "default-1719236764",
  "timestamp": 1719237054620929500,
  "time": "2024-06-24T16:50:54.620929561+03:00",
  "updates": [
    {
      "Path": "nvidia/ib/amber/reply/onchange",
      "values": {
        "nvidia/ib/amber/reply/onchange": {
          "HeaderID": "912200528",
          "Values": [

"1719237054172043,0xeed4bd9cd3b7f325,1,117416,115164,121731,117033,118807,3048,6316",

"1719237054172043,0x2e36224302959e79,1,116228,115555,120616,117767,123899,87,9953",

"1719237054172043,0x8e62a55d7571a9b8,1,115684,123124,121670,117400,121689,2910,4203",

"1719237054172043,0x7e680fb8f81a1950,1,115531,122250,115999,122455,124258,3716,5329",
```

```
"1719237054172043,0x0176438fe4ee507c,1,119269,123884,119887,123502,120366,4540,6673"
            ]
          }
        }
      }
   ]
}
```

Inventory Requests

Inventory messages are conveyed in separate updates, presenting the inventory details of the UFM associated with the provided IP. These messages display comprehensive information, including the total count of various components within the UFM, such as switches, routers, servers, and more, along with details about active ports and the total number of ports, including disabled ones. Moreover, inventory requests include the size of the telemetry, which is not always the same as the active ports. In cases where the plugin is unable to establish contact with the UFM, it will revert to using default values defined in the configuration file. It is worth noting that the path for inventory requests differs from the conventional path structure, as they do not rely on specific nodes or ports. Consequently, inventory requests are initiated after "`nvidia/ib`."

Example:

```
 gnmic -a localhost:9339 --insecure get –path
nvidia/ib/inventory/*
```

Response:

```
[
  {
    "source": "localhost:9339",
    "timestamp": 1698824237536878000,
    "time": "2023-11-01T09:37:17.536878067+02:00",
    "updates": [
      {
        "Path": "nvidia/ib/inventory",
        "values": {
          "nvidia/ib/inventory": {
            "ActivePorts": 4,
            "Cables": 2,
            "Gateways": 0,
            "HCAs": 2,
            "Routers": 0,
            "Servers": 2,
            "Switches": 1,
            "TotalPorts": 38,
            "TelemetrySize": 4,
            "timestamp": 1698824211535069000
          }
        }
      }
    ]
  }
]
```


Events Requests

Events messages are provided in separate updates, offering insights into the events occurring within the UFM associated with the specified IP. Given that the event metadata remains consistent, even when numerous events are part of a request, the message format adopts a CSV-like structure. The Headers section contains essential metadata regarding UFM events, while the Values section contains the raw event data. Users can

subscribe to these events with the on-change feature enabled, receiving only the events triggered within the subscription interval. Notably, the path structure for event requests differs from the typical node or port-based structure and is requested after "`nvidia/ib`."

Example:

```
gnmic -a localhost:9339 --insecure get –path nvidia/ib/events/*
```

Response:

```
[
   {
      "source" :  "localhost:9339",
      "timestamp" :  1698824809647515600,
      "time" :  "2023-11-01T09:46:49.647515575+02:00",
      "updates" :  [
         {
            "Path" :  "nvidia/ib/events",
            "values" :  {
               "nvidia/ib/events" :  {
                  "Headers" :  [
                     "id",
                     "object_name",
                     "write_to_syslog",
                     "description",
                     "type",
                     "event_type",
                     "severity",
                     "timestamp",
                     "counter",
                     "category",
                     "object_path",
                     "name"
                  ],
                  "Values" :  [
                     "7718,Grid,false,Disk space usage in /opt/ufm/files/log is above the threshold of
90.0%.,Grid,525,Critical,2023-11-01 07:25:54,N/A,Maintenance,Grid,Disk utilization threshold reached",
                     "7717,Grid,false,Disk space usage in /opt/ufm/files/log is above the threshold of
90.0%.,Grid,525,Critical,2023-11-01 07:24:54,N/A,Maintenance,Grid,Disk utilization threshold reached",
                     "7716,Grid,false,Disk space usage in /opt/ufm/files/log is above the threshold of
90.0%.,Grid,525,Critical,2023-11-01 07:23:54,N/A,Maintenance,Grid,Disk utilization threshold reached",
                     "7491,ec0d9a0300d42e54,false,Mcast group is deleted: ff12601bffff0000,
00000002,Computer,67,Info,2023-10-31 06:39:21,N/A,Fabric Notification,default / Computer: r-
ufm59,MCast Group Deleted"
                  ]
```

```
            }
          }
        }
      ]
    }
]
```

Switch Rank Requests

Switch rank updates are conveyed in separate messages, presenting the rank of the switches in the UFM. This data is derived from a file in the UFM and is updated by the server every 6 hours by default. The `switch_rank` counter is associated only with switch-level data, so there is no need to specify a port in the path. However, this counter is not connected to the telemetry cache of switch-level data. **Note that if the `ufm_ip` is changed, the switch_rank information will not be available.**

Example:

```
gnmic -a localhost:9339 --insecure get --path
nvidia/ib/guid[guid=*]/amber/switch_rank
```

Response:

```
{
  "source": "localhost:9339",
  "timestamp": 1719296207323383300,
  "time": "2024-06-25T09:16:47.323383222+03:00",
  "updates": [
    {
      "Path": "nvidia/ib/guid[guid=*]/amber/amber/switch_rank",
      "values": {
        "nvidia/ib/guid/amber/amber/switch_rank": {
          "Headers": "Timestamp,Node_GUID,switch_rank",
          "Values": [
            "1719296205612,0x0002c903007e5220,0"
          ]
        }
      }
    }
  ]
}
```


UFM Health KPI Requests

UFM Health KPI messages are provided in separate updates, offering insights into the UFM Health metrics occurring within the UFM associated with the specified IP. The response value is Prometheus formatted, as a one big string. Users can subscribe to these UFM Health KPI with the on-change feature enabled, receiving the whole UFM Health metrics if there is a change in one item. Notably, the path structure for UFM health KPI requests differs from the typical node or port-based structure and is requested after "`nvidia/ib`."

Example:

```
gnmic -a localhost:9339 --insecure get --path
nvidia/ib/ufm_health_kpi/*
```

Response:

```
{
   "source": "localhost:9339",
   "timestamp": 1719296207323383300,
   "time": "2024-06-25T09:16:47.323383222+03:00",
   "updates": [
      {
         "Path": "nvidia/ib/ufm_health_kpi",
         "values": {
            "nvidia/ib/ufm_health_kpi": {
               "value": "# HELP server_cpu_usage_percent_avg Average of
Server CPU usage percent
                                                  # TYPE
server_cpu_usage_percent_avg gauge

server_cpu_usage_percent_avg{duration="Last 5 minutes"} 1.5545454545454547

server_cpu_usage_percent_avg{duration="Last 1 hour"} 1.4975206611570255

server_cpu_usage_percent_avg{duration="Last 24 hour"} 1.505277777777778
...

events_history_counter{duration="Last week",event_name="Director Switch is
Down"} 0.0

events_history_counter{duration="Last week",event_name="Node is Up"} 0.0

events_history_counter{duration="Last week",event_name="Node is Down"} 0.0

events_history_counter{duration="Last week",event_name="Link is Up"} 0.0

events_history_counter{duration="Last week",event_name="Link is Down"}
0.0",
            }
         }
      }
```

```
    ]
}
```

UFM Telemetry Notification Subscription

The gNMI plugin includes a built-in Telemetry Notification Server that enables event-driven data synchronization between UFM Telemetry endpoints and the gNMI server. This real-time communication complements the existing periodic telemetry fetching mechanism controlled by the `telemetry_interval` parameter. See Telemetry Configurations.

The Telemetry Notification Server allows UFM Telemetry to push updates to the gNMI server immediately when new data becomes available, reducing latency and enhancing responsiveness compared to periodic polling alone.

The gNMI plugin provides a telemetry notification server that facilitates event-driven data updates, enabling real-time synchronization between UFM Telemetry endpoints and the gNMI server's in addition to the normal periodic fetching that's based on ` `telemetry_interval`.

UFM Telemetry Integration

To enable UFM Telemetry to notify the gNMI server when new data is ready, you must add the below configuration line to the UFM Telemetry `.ini` file (For example: /opt/ufm/files/conf/secondary_telemetry_defaults/launch_ibdiagnet_config.ini)

```
plugin_env_UFM_TELEMETRY_NOTIFY_ENDPOINTS=http://localhost:9338/telemetry/notif
```

Replace `<Telemetry_HTTP_PORT>` with the actual HTTP port of the telemetry endpoint (e.g., `9002` for a secondary telemetry instance).

After updating the configuration, restart the UFM Telemetry service to apply the changes.

```
# For UFM Bare-metal
/etc/initd/ufmd ufm_telemetry_restart
# For UFM Docker
docker exec ufm /etc/initd/ufmd ufm_telemetry_restart
```

Troubleshooting

#	Use Case	Result	Root Cause
1	UFM restarts	No gNMI data is streamed using gnmic. The notification include only headers with no data.	The gNMI inner cache is empty due to unresponsive telemetry

UFM Telemetry Manager (UTM) Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.21.4	N/A

Bug Fixes

Plugin Version	Bug Fix
1.21.4	N/A

Overview

Managed telemetry is a mode of high availability and improved performance of UFM Telemetry processes.

Governed by UFM Telemetry Manager (UTM) several UFM Telemetry Instances (TIs) run on one or more machines, each collecting a subset of the cluster fabric.

UTM manages the following aspects:

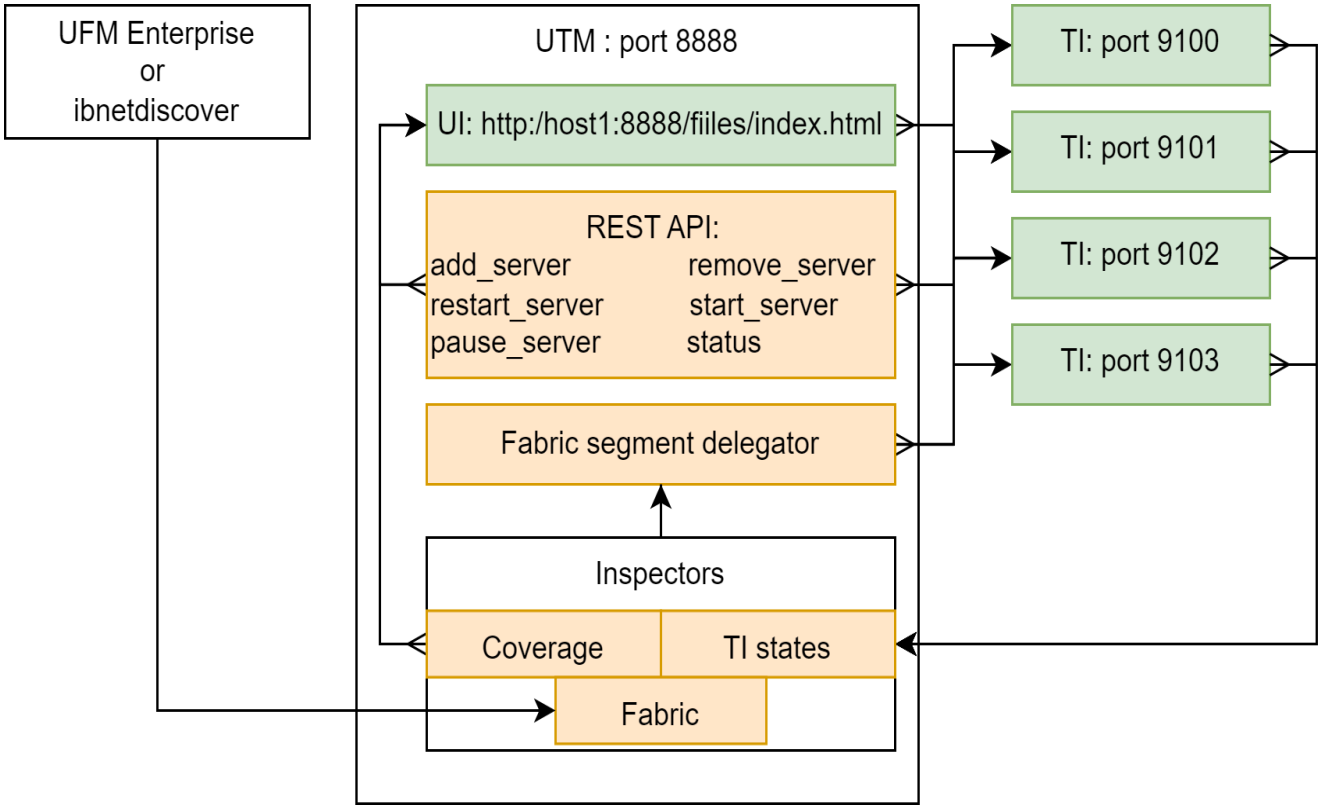
- monitoring of TI states: down, initializing, running, paused
- TI management commands: add, remove, pause, start, restart
- partitioning of fabric based on TIs health and fabric changes
- assigning fabric segments to TIs
- telemetry coverage check of a cluster

The UFM Telemetry Manager (UTM) Plugin facilitates managed telemetry in high availability mode, enhancing the performance of UFM Telemetry operations.

Under the governance of UFM Telemetry Manager (UTM), multiple UFM Telemetry Instances (TIs) are executed on one or more machines, with each TI responsible for collecting a specific portion of the cluster fabric.

Key functionalities managed by UTM include:

- Monitoring TI statuses: down, initializing, running, paused
- Execution of TI management commands: add, remove, pause, start, restart
- Fabric partitioning based on TI health and fabric changes
- Assigning fabric segments to TIs
- Verification of telemetry coverage across the cluster



Deployment

The UTM plugin is designed to operate either as a UFM plugin or in standalone mode.

UTM plugin mode is deployed via UFM UI. Telemetry instances might be deployed by UFM or manually by deployment scripts.

Standalone mode deploys the whole setup (UTM, host TI list, switch telemetry image) with the deployment scripts.

As a first step, get the UTM image. If it runs in UFM mode, upload it to the UFM machine.

```
docker pull mellanox/ufm-plugin-utm
```

UFM Plugin Mode

The UTM plugin can be added either via the Command Line Interface or Web-UI.

CLI Deployment

To add the plugin, run:

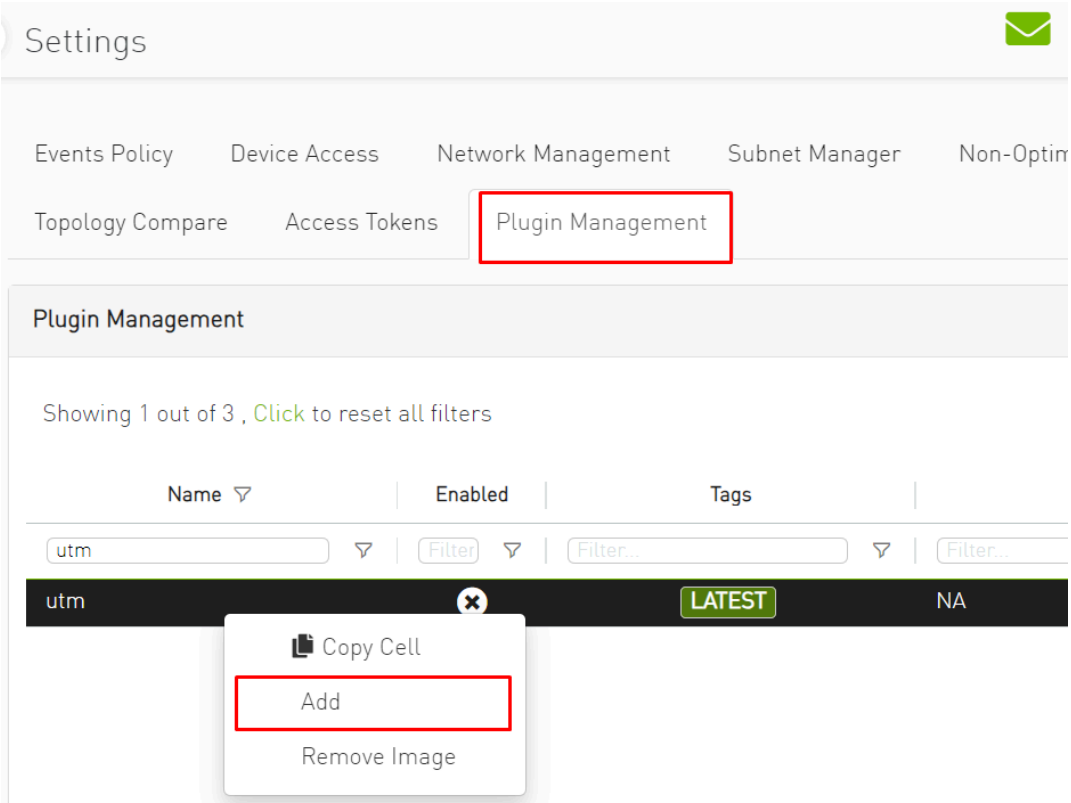
```
/opt/ufm/scripts/manage_ufm_plugins.sh add -p utm
```

To remove the plugin, run:

```
/opt/ufm/scripts/manage_ufm_plugins.sh remove -p utm
```

Web-UI Deployment

1. Navigate to the UFM web UI and click on `Settings` in the left panel.
2. Go to the "Plugin Management" tab.
3. Right-click on the UTM plugin row and select "Add."



4. Go to the option on the left called "Telemetry Status" to see the UTM UI page.

5. Operate with several options:

    1. The default UFM TIs. Depending on UFM configuration, TIs might run in legacy mode or within UTM.

    2. Start telemetry instances manually using UTM deployment scripts. See section "Manual Deployment".

To stop the UTM plugin, go to "Plugin Management", right-click on the UTM plugin line and click on `disable`.

> **Note**
>
> If non-default UFM credentials are used, UTM may fail to access the UFM REST API. To resolve this, configure the `ufm` section of the `utm_config.ini` file with `ufm_user=` and `ufm_pass=` to restore the connection between UTM and UFM.

Default UFM Telemetry Monitoring in Legacy Mode

UFM Telemetry has high and low-frequency (Primary and Secondary, respectively) TIs that are running by default.

To enable meaningful monitoring:

1. Set `plugin_env_CLX_EXPORT_API_SHOW_STATISTICS=1` in the config files:

```
/opt/ufm/files/conf/telemetry_defaults/launch_ibdiagnet_config
/opt/ufm/files/conf/secondary_telemetry_defaults/launch_ibdiag
```

2. Restart telemetry instances with the new config. If UFM Enterprise runs as a docker container, this command should be executed inside the container.

```
/etc/init.d/ufmd ufm_telemetry_restart
```

3. Give TIs some time to update performance metrics. The time depends on the update interval of default TIs.

Default UFM Telemetry instances in UTM

If legacy mode is disabled in UFM configuration, UTM will run Primary and Secondary telemetries automatically.

Manual Deployment

Additional telemetry instances for UFM plugin mode or the whole standalone setup (UTM and TIs) are deployed using UTM Deployment scripts.

UTM Deployment Scripts

Get deployment scripts and examples by mounting the local folder `UTM_DEPLOYMENT_SCRIPTS` (`/tmp/utm_deployment_scripts` in this example) and running `get_deployment_scripts.sh` :

```
$ export UTM_DEPLOYMENT_SCRIPTS=/tmp/utm_deployment_scripts
$ docker run -v "$UTM_DEPLOYMENT_SCRIPTS:/deployment_scripts" --rm --name utm-deployment-scripts -ti mellanox/ufm-plugin-utm:latest /bin/sh /get_deployment_scripts.sh
```

The content of the script folder consists of:

- `Examples` - Contains run/stop scripts for both standalone and UFM plugin modes. Each example script is an example of actual deployment script usage.
- `hostlist.txt` - Specifies the hosts, ports, and HCAs for TIs to be deployed

- `Scripts` - Contains actual deployment scripts. Entry-point script `deploy_managed_telemetry.sh` triggers the rest two scripts, depending on input arguments.

```
$ cd $UTM_DEPLOYMENT_SCRIPTS
$ tree
.
    examples
        run_standalone.sh
        run_with_plugin.sh
        stop_standalone.sh
        stop_with_plugin.sh
    hostlist.txt
    README.md
    scripts
        deploy_bringup.sh
        deploy_managed_telemetry.sh
        deploy_ufm_telemetry.sh
```

Note

All example/deployment scripts should run from the `UTM_DEPLOYMENT_SCRIPTS` folder.

Hostlist File

Please note the following:

- The `hostlist.txt` file should be set before running any script.

- The hostname and port will be used for communication and HCA for telemetry collection.
- UTM only supports a single fabric for managed TIs, even if different HCAs on the same machine are connected to different fabrics.
- Both local and remote hosts are supported for TI deployments.

```
$ cat hostlist.txt

# List lines in the following format:
# host:port:hca
#
# where:
#  - host is IP or hostname. Use localhost or 127.0.0.1 for local
deployment
#  - port to run telemetry on.
#  - hca is the target host device from which telemetry collects.
Run `ssh $host ibstat`
#       to find the active device on the target host.

localhost:8123:mlx5_0
localhost:8124:mlx5_0
```

Main Deployment Script

For a more customizable setup beyond what the example scripts offer, users have the option to manually run `./scripts/deploy_managed_telemetry.sh`. This primary deployment script can deploy multiple TIs and optionally UTM as well.

Use `deploy_managed_telemetry.sh --help` to get help.

```
./deploy_managed_telemetry.sh --help
./deploy_managed_telemetry.sh options:     mandatory:

    mandatory:
        --hostlist-file=     Path to a file that lists
hostname:port:hca lines
    mandatory run options (use only one at the same time):
        -r,  --run           Deploy and run managed telemetry
setup
        -s,  --stop          Stop all processes and cleanup
    mandatory telemetry deployment options (use only one at the
same time):
        -t=, --ufmt-image=   UFM telemetry docker image or
tgz/tar.gz-image
      or:
        --bringup-package=   Bringup tar.gz package
    optional:
        -m=, --utm-image=       UTM docker image or tgz/tar.gz-
image. Runs UTM only if it is set. Configures UTM according
hostlist file
        --utm-as-plugin=        if UTM runs as a plugin, set this
flag
        -d=, --data-root=       Root directory for run data   |
Default: '/tmp/managed_telemetry/'
        --switch-telem-image=   Switch telemetry image (tar.gz-file
or docker image). UTM will be able to deploy it to managed
switches if set
        --common-data-dir=      Common data folder for TIs
    -h,  --help              Print this message
```

Start TIs for UFM Plugin Mode:

1. Prepare TI setup using `utm_deployment_scripts` example scripts:

2. Change directory:

```
cd $UTM_DEPLOYMENT_SCRIPTS
```

3. Open and configure `hostlist.txt`

4. Deploy and run TIs according to `hostlist.txt` and set these TIs to be monitored by UTM:

```
sudo ./examples/run_with_plugin.sh
```

5. To stop and cleanup TIs setup and unset TIs to be monitored by UTM:

```
sudo ./examples/stop_with_plugin.sh
```

> **Note**
>
> This script does not stop UTM plugin!

Standalone Mode

In standalone mode, UTM periodically tracks fabric changes by itself and does not require UFM Enterprise.

Deploy via example scripts:

1. Change directory

```
cd $UTM_DEPLOYMENT_SCRIPTS
```

2. Open and configure `hostlist.txt`

3. Deploy and run TIs according to `hostlist.txt` and run UTM:

```
sudo ./examples/run_standalone.sh
```

4. To stop and cleanup TIs setup and UTM, run:

```
sudo ./examples/stop_standalone.sh
```

Deployment without Scripts

This section provides detailed instructions for manually deploying UTM and managed TIs to ensure coverage of all potential corner cases where the convenience script may not be effective.

UTM Deployment

UTM can be started with two docker run commands.

1. Set `utm_config`, `utm_data`, `utm_log`, and `utm_image` variables.

2. Initialize UTM config:

```
docker run -v $utm_config:/config \
           -v $utm_data:/data \
           --rm --name utm-init \
           --device=/dev/infiniband/ \
           $utm_image /init.sh
```

3. Run UTM

```
docker run -d --net=host  \
           --security-opt seccomp=unconfined --cap-
add=SYS_ADMIN \
           --device=/dev/infiniband/ \
           -v $utm_config:/config \
           -v $utm_data:/data \
           -v $utm_log:/log \
           --rm --name utm $utm_image
```

Managed/Standalone TIs Manual Deployment

TI can be represented either as a UFM Telemetry docker container or as a UFM Telemetry bring-up package.

To run the docker container in managed mode, `launch_ibdiagnet_config.ini` should have the following flags enabled:

```
plugin_env_CLX_EXPORT_API_SHOW_STATISTICS=1
plugin_env_UFM_TELEMETRY_MANAGED_MODE=1
```

To run UFM Telemetry with Distributed Telemetry, enable its receiver and specify `HCA` to work on:

```
plugin_env_CLX_EXPORT_API_RUN_DT_RECEIVER=1
plugin_env_CLX_EXPORT_API_DT_RECEIVER_HCA=$HCA
```

To run bringup in managed mode, create `enable_managed.ini` file with the same flags and use `custom_config` option of `collection_start`:

```
collection_start custom_config=./enable_managed.ini
```

UTM Configuration File

The UTM configuration file `utm_config.ini` is placed under the configuration folder (which is referred to as `UTM_CONFIG` later on this document).

In the case of UFM plugin mode, `UTM_CONFIG = /opt/ufm/files/conf/plugins/utm/`.

In the case of standalone mode, the default value is `UTM_CONFIG =/tmp/managed_telemetry/utm/config` and can be changed via `--data-root` argument of deployment script.

When changes are made to the configuration file, UTM initiates a restart of its main process to apply the updated configuration.

Users may wish to adjust timeout and update rate configurations based on their specific setups. However, it is important to note that the remaining configurations are tailored to enable UTM to function as a UFM plugin and should not be modified.

Distributed Telemetry

To enable distributed telemetry set `dt_enable=1` in the corresponding section.

Note

Distributed Telemetry requires Switch Telemetry docker image tagged as `switch-telemetry:{version}` and placed under `$UTM_CONFIG/telem_files/` as `switch-telemetry_{version}.tar.gz`

UTM scans this file at its start.

Example deployment scripts handle it for both UFM plugin and standalone modes.

For more details refer to NVIDIA UFM Telemetry Documentation→ Distributed Telemetry - Switch Telemetry Agent

GUI

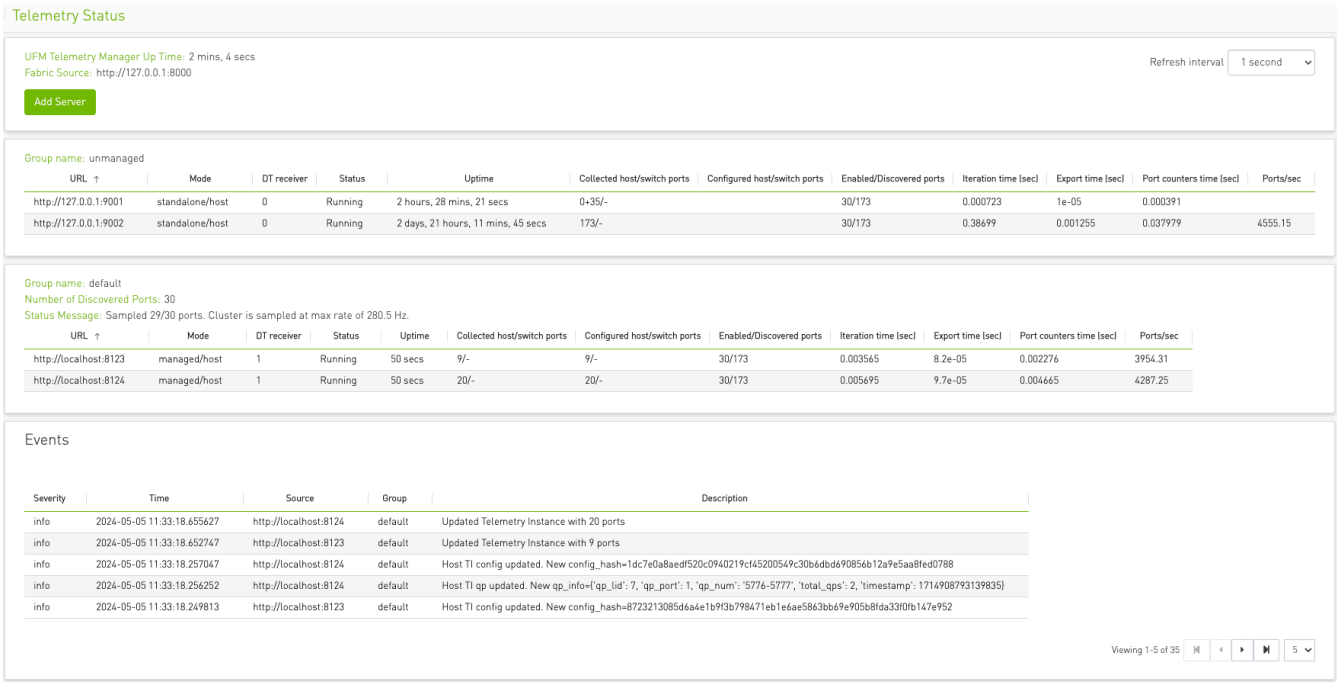
To access the GUI within the UFM web UI, navigate to the Telemetry status section in the left panel.

The UI is accessible whether it is running as a part of UFM Enterprise or standalone via the endpoint: `http://127.0.0.1:8888/files/index.html.`

The GUI comprises several zones:

- The top pane displays general information and provides options to add a server name/IP and port for monitoring. Users can set the GUI refresh interval in the top right corner.
- The middle panes showcase TI groups, with the default group being basic. Unmanaged (standalone) TIs can be monitored and are placed in the "Unmanaged" group.
- Each group pane presents monitoring information for each TI.

- The bottom pane exhibits system events. Utilize the bottom right menu to navigate through the events history.



TI Management

In managed mode, UTM can dispatch commands to TIs. By right-clicking on the TI line, users can:

- Pause a currently running TI. This action redistributes fabric sharding among the active TIs.
- Resume a paused TI.
- Exclude a TI from monitoring. Although the TI remains on the machine, it enters a paused state and is removed from its group. It's important to note that empty TI groups are automatically removed.

Telemetry Status Fields

The table below lists each column of a Telemetry Group panel:

Field Name	Description
URL	TI URL in format `http://{hostname}:{port}` or `http://{IP}:{port}`
Mode	standalone or managed / platform
DT receiver	With or without a Distributed Telemetry receiver. If 0, cannot receive DT data from a switch TI
Status	Down, Running, Initializing, Paused, or Restarting
Uptime	TI uptime in human-readable format
Collected host/switch ports	Ports collected from the host/switch. By default data that did not change from the last sample is not being re-exported. Such data is shown in the host part ad +num_old_ports.In the screenshot above. first TI of the "unmanaged" group sampled 0 new data samples and found 35 old ones.Nothing is being sampled from Distributed Telemetry, because this TI runs without DT receiver. The resulting format is: `0+35/-`
Configured host/switch ports	Ports configured to be sampled from a host and corresponding switches in total. For more details refer to Distributed Telemetry documentation.
Enabled/discovered ports	Enabled and discovered ports of the Fabric.
Iteration time	Total iteration time of UFM Telemetry data collection
Export time	Export time in the last iteration of UFM Telemetry data collection. Included to `Iteration time`
Port counters time	Time spent only on port counters telemetry collection. Included to `Iteration time`
Ports/sec	Speed of new port counters data collection during the last iteration of UFM Telemetry.

REST API

All the GUI features including TI management and monitoring can be accessed via REST API.

Accessing UTM API Commands Based on Operating Mode

The method to access UTM API commands varies depending on the mode:

- **In UFM Plugin Mode:** Use the UFM REST API proxy:

```
curl -s -k https://{UFM_HOST_IP}/ufmRest/plugin/utm/{COMMAND} -u {user}:{pass}
```

- **In Standalone Mode:** Access the UTM HTTPS endpoint on the default port `8888`:

```
curl -s -k https://{UTM_HOST_API}:8888/{COMMAND}
```

Command List

For simplicity, the following commands are provided for standalone mode.

- Get the list of supported user endpoints:

```
curl -s -k https://127.0.0.1:8888/help
```

- Get the status of the monitored TIs in JSON format:

```
curl -k https://127.0.0.1:8888/status
```

- Add TI http://127.0.0.1:8123 to the `my_group` monitoring group:

```
curl -k 'https://127.0.0.1:8888/add_server?url=http://127.0.0.1:8123&group=my_group'
```

- Add TI http://127.0.0.1:8123 to `default` monitoring group:

```
curl -k https://127.0.0.1:8888/add_server?
url=http://127.0.0.1:8123
```

- Remove TI from monitoring (running TI will be paused):

```
curl -k https://127.0.0.1:8888/remove_server?
url=http://127.0.0.1:8123
```

- Pause running TI:

```
curl -k https://127.0.0.1:8888/pause_server?
url=http://127.0.0.1:8123
```

- Resume paused TI:

```
curl -k https://127.0.0.1:8888/start_server?
url=http://127.0.0.1:8123
```

UFM Consumer Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.0.0-17	N/A

Bug Fixes

Plugin Version	Bug Fix
1.0.0-17	N/A

Overview

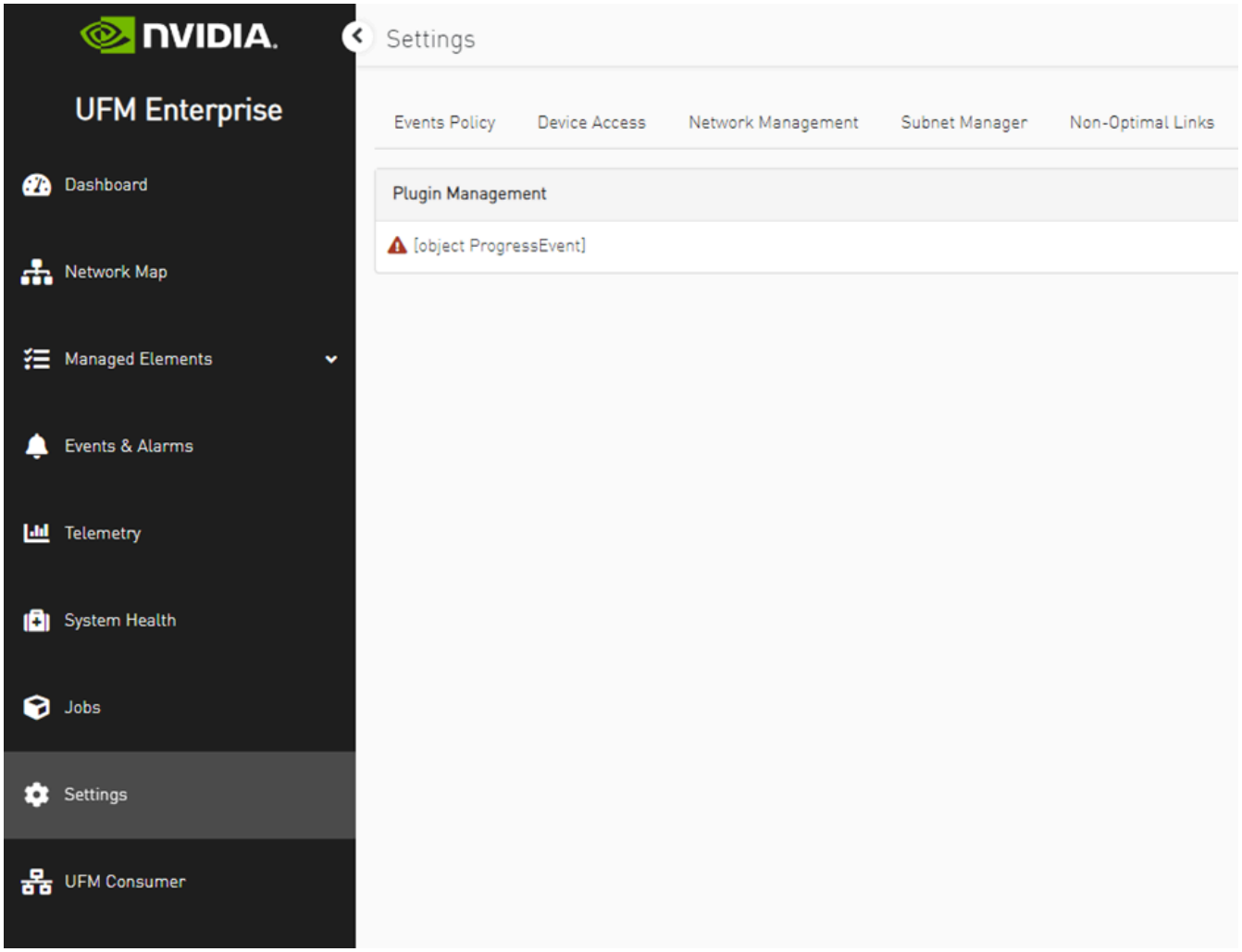
The UFM consumer plugin is a self-contained Docker container with with REST API support, under UFM management. It serves as a Multi-Subnet consumer within UFM, offering all the functionalities available for Multi-Subnet UFM.

The Multi-Subnet UFM feature allows the management of large fabrics spanning multiple sites within a single product, specifically Multi-Subnet UFM.

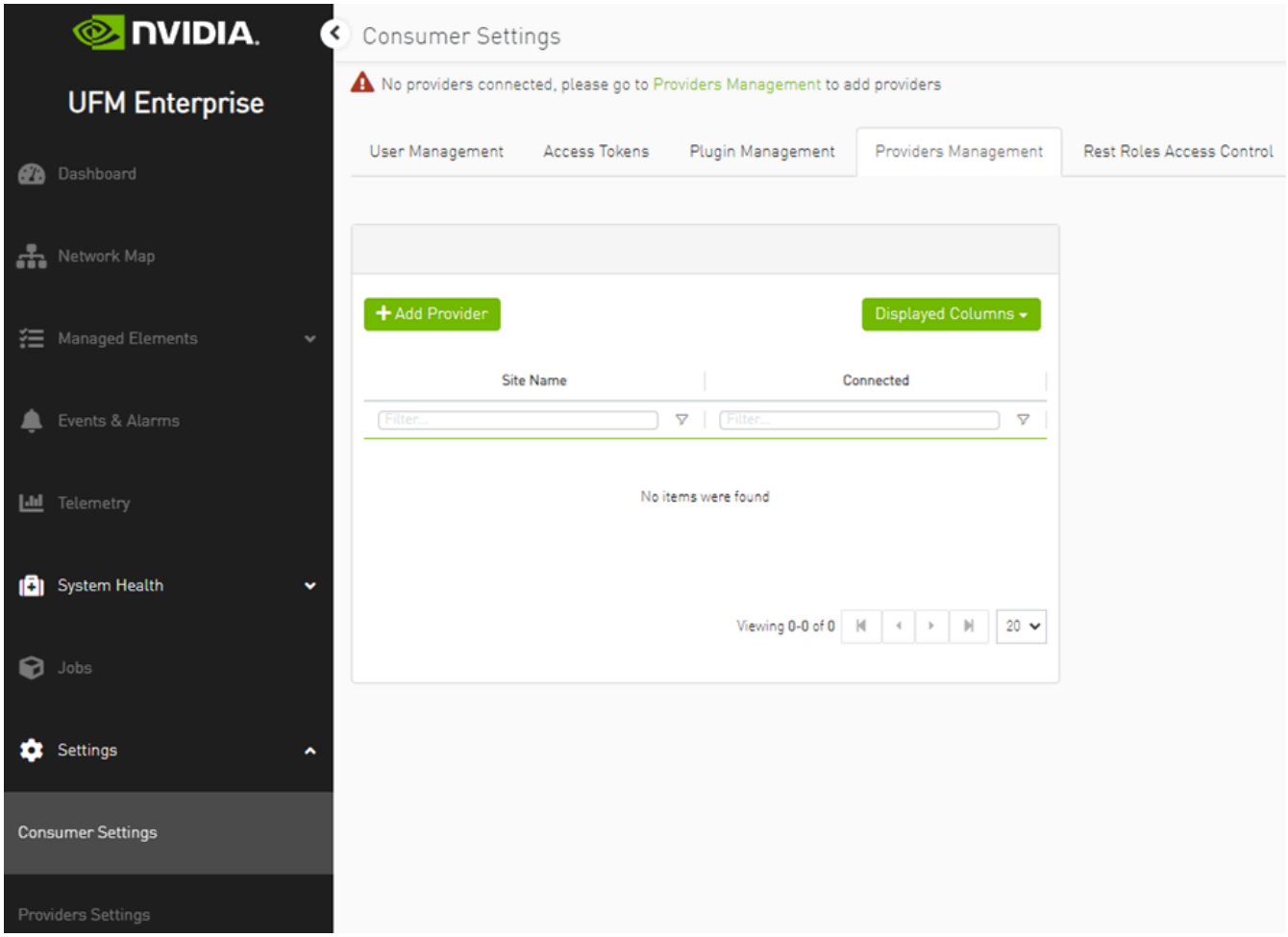
This feature comprises two layers: UFM Multi-Subnet Provider and UFM Multi-Subnet Consumer (UFM consumer plugin).

The UFM Provider acts as a Multi-Subnet Provider, exposing all local InfiniBand fabric information to the UFM consumer plugin. On the other hand `, the UFM consumer plugin functions as a Multi-Subnet Consumer, gathering and consolidating data from configured UFM Providers. This enables users to manage multiple sites conveniently in one location. While the UFM Consumer offers similar functionality to regular UFM, there are behavioral differences concerning aggregation.

Upon deployment of the UFM consumer plugin, an additional tab is incorporated into the UFM interface.



After clicking on UFM consumer plugin UI, a new browser tab with UFM Multi-Subnet consumer functionality options is presented. Upon the initial launch of the UFM consumer plugin UI and in case there are no configured providers, a screen for adding providers (Provider Management) will be displayed.



The functionality of the UFM consumer plugin is similar to that of Multi-Subnet UFM. For further details, refer to Multi-Subnet UFM.

Fast-API Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.0.5-2	N/A

Bug Fixes

Plugin Version	Bug Fix
1.0.5-2	N/A

Overview

The Fast-API plugin is a new component that runs in parallel to the UFM model and provides a more scalable and efficient way to manage the inventory and topology of the fabric. It uses a Redis database to store the data it receives from the SM client consumer and the `ibdiagnet` collector. It also exposes REST APIs for querying the ports, systems, links, switches, and sharp reservations. The Fast-API plugin also supports persistent inventory, which means it keeps track of all the devices, ports, and links the UFM ever encountered, even if they are disabled or removed. The Fast-API plugin is designed to be backward compatible with the UFM model, except for some minor differences in the API parameters and the handling of generic node names.

Deployment

- As a first step, get the Fast API image:

```
docker pull mellanox/ufm-plugin-fast_api
```

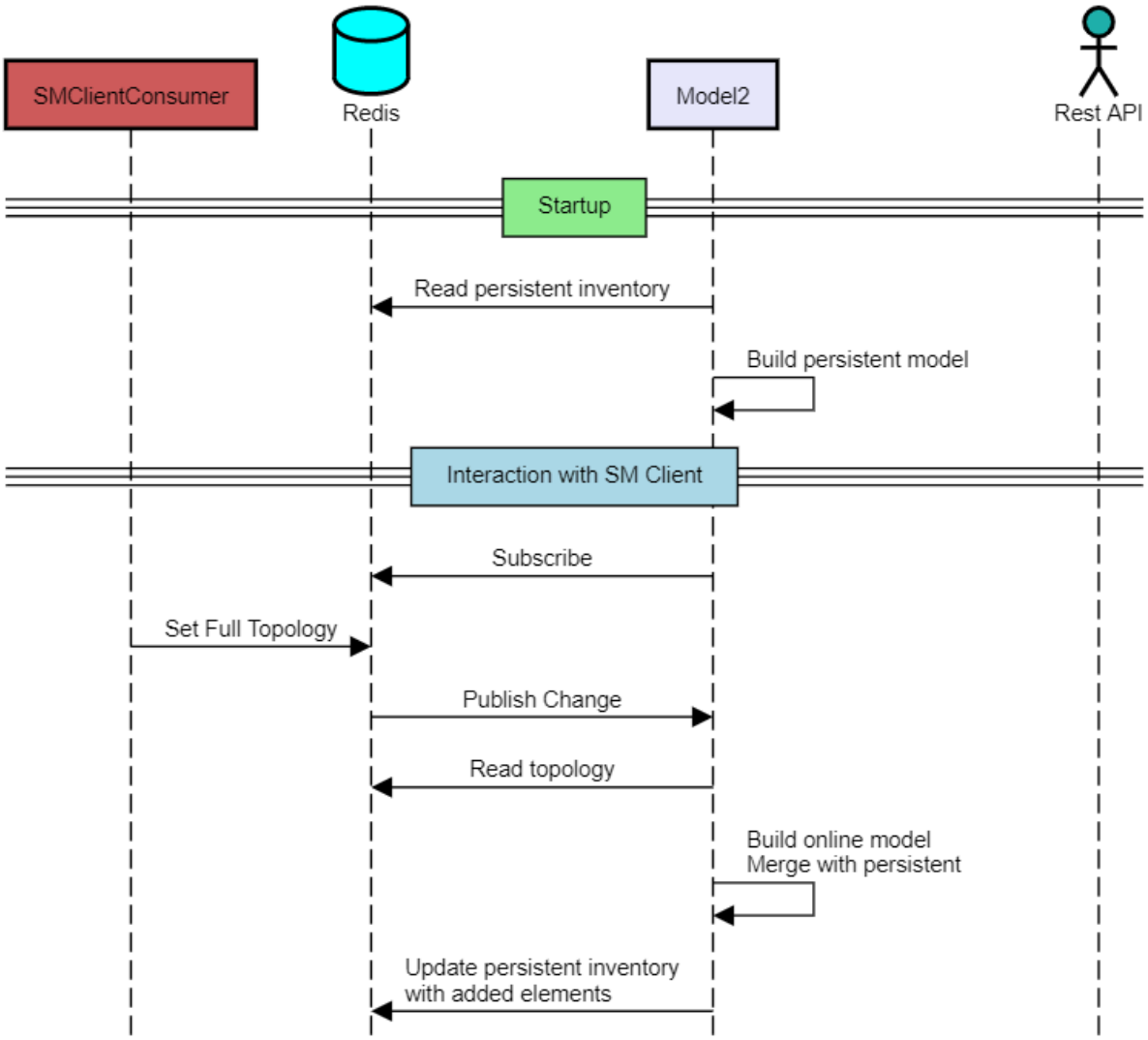
- Load the downloaded image onto the UFM server. This can be done either by using the UFM GUI by navigating to the Settings -> Plugins Management tab or by loading the image via the following instructions:

    1. Log in to the UFM server terminal.
    2. Run:

```
docker load -I <path_to_image>
```

- After successfully loading the plugin image, the plugin should become visible in the plugin management table within the UFM GUI. To initiate the plugin's execution, simply right-click on the respective in the table.

Sequence Diagram



Supported APIs

The following APIs are supported:

Ports API - Retrieve all the ports in the cluster or filter by the system name. The following attributes are retrieved:

- GET https://<UFM_IP>/ufmRestV2/plugin/fast_api/resources/ports
- GET /resources/ports/ - deviceID referred as sys_guid for switches and hostname for hosts
    - `name`: The name of the port.

- `lid`: The local ID of the port.
- `logical_state`: The logical state of the port.
- `physical_state`: The physical state of the port.
- `number`: The number of the port.
- `decimal_guid`: The decimal GUID of the port.
- `mtu`: The maximum transmission unit of the port.
- `active_speed`: The active speed of the port.
- `active_width`: The active width of the port.
- `peer_address`: The peer address of the port.
- `peer_port`: The peer port of the port.
- `tier`: The tier of the port.
- `label`: The label of the port.

This API retrieves all the ports in the cluster, including cable information. The following attributes are supported for cable information:

- GET https://<UFM_IP>/ufmRestV2/plugin/fast_api/resources/ports/?cable_info=true
  - `part_number`: The part number of the cable.
  - `serial_number`: The serial number of the cable.
  - `revision`: The revision of the cable.
  - `identifier`: The identifier of the cable.
  - `length`: The length of the cable.
  - `technology`: The technology of the cable.

- `fw_version`: The firmware version of the cable.

Links API - Retrieve all the links in the cluster. The following attributes are retrieved:

- GET https://<UFM_IP>/ufmRestV2/plugin/fast_api/resources/links
  - `source_guid`: The GUID of the source port.
  - `source_port`: The source port.
  - `destination_guid`: The GUID of the destination port.
  - `destination_port`: The destination port.
  - `source_port_dname`: The display name of the source port.
  - `source_port_name`: The name of the source port.
  - `destination_port_dname`: The display name of the destination port.
  - `destination_port_name`: The name of the destination port.
  - `width`: The width of the link.
  - `severity`: The severity of the link.
  - `source_port_node_description`: The description of the node connected to the source port.
  - `destination_port_node_description`: The description of the node connected to the destination port.
  - `name`: The name of the link.
  - `active`: The status of the link.

This API retrieves information about the systems in the cluster. It can retrieve all the systems or filter by the system name. The following attributes are retrieved:

- GET https://<UFM_IP>/ufmRestV2/plugin/fast_api/resources/systems

- GET https://<UFM_IP>/ufmRestV2/plugin/fast_api/resources/systems/<system_name>
- GET https://<UFM_IP>/ufmRestV2/plugin/fast_api/resources/switches
    - `name`: The display name of the system.
    - `system_name`: The name of the system.
    - `guid`: The GUID of the system.
    - `sys_guid`: The system GUID.
    - `ports`: A list of ports associated with the system.
    - `hostname`: The hostname of the system.
    - `vendor`: The vendor of the system.
    - `temperature`: The temperature of the system.
    - `fw_version`: The firmware version of the system.
    - `technology`: The technology used by the system.
    - `level`: The level of the system.
    - `uptime`: The uptime of the system.
    - `sw_version`: The software version of the system.
    - `system_type`: The type of the system.
    - `description`: The description of the system.
    - `psid`: The PSID of the system.
    - `active`: The status of the system.
    - `state`: The state of the system.

- `is_managed`: Indicates whether the system is managed.

UFM Light Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.1.0-2	N/A

Bug Fixes

Plugin Version	Bug Fix
1.1.0-2	N/A

Overview

Overview

UFM Light is a C++ UFM plugin designed to create a reduced UFM model and deliver a high-performance REST API. While it builds a model similar to the legacy UFM, there are notable differences:

- The model in UFM Light is streamlined to meet the specific requirements of the UFM Light REST API functionality.
- UFM Light builds its model more quickly.

UFM Light offers the same REST API as the legacy UFM, but with enhanced performance.

Deployment

Prerequisites

1. UFM is running
2. Fast-API plugin is running

UFM Light builds its model from the Redis and requires the Fast-API plugin as Redis provider. Writing to Redis must be enabled in UFM.

Installation

1. Get UFM Light image:

```
docker pull mellanox/ufm-plugin-ufm-light
```

2. Load UFM Light plugin:

```
sudo /opt/ufm/scripts/manage_ufm_plugins.sh add -p ufm-light
-t <version>
```

3. Configure the UFM Light plugin if needed (see "Configuration" section).

Note: it is recommended to configure parameter --http.threads to “4” for 16-Core systems, “16” for 64-Core systems.

Configuration

UFM Light runs with its default configuration. To update the configuration, edit:

```
sudo vim /opt/ufm/files/conf/plugins/ufm-light/ufm-light.cfg
```

If `ufm-light.cfg` is updated, the UFM Light plugin must be stopped and started again manually to apply the changes:

```
sudo /opt/ufm/scripts/manage_ufm_plugins.sh stop -p ufm-light

sudo /opt/ufm/scripts/manage_ufm_plugins.sh start -p ufm-light
```

Configuration Options:

```
  --model.srcmode arg (=redis:full)     base source of model:
redis:full, redis:invnt, grpc
  --model.period arg (=30)              set farbric request
period in seconds
  --redis.address arg (=localhost)      set Redis address
  --redis.port arg (=6379)              set Redis port
  --grpc.address arg (=localhost)       set gRPC address
  --grpc.port arg (=2510)               set gRPC port
  --telemetry.address arg (=localhost)  set Telemetry address
  --telemetry.port arg (=9001)          set Telemetry port
  --telemetry.url arg (=/csv/cset/converted_enterprise) set
Telemetry url
  --http.address arg (=127.0.0.1)       set http address
  --http.port arg (=8950)               set http port
  --http.threads arg (=4)               set number of http server
threads, min 1
  --logging.level arg (=INFO)           set logging level:
CRITICAL, ERROR, WARNING, INFO, DEBUG, TRACE
  --logging.log_dir arg (=/log)         set logging directory
  --logging.file.max_size_MB arg (=10)  set MAX log file size
limit in MBs, min 1
  --logging.file.backups arg (=10)      set number of log file
backups
```

For normal operation, keep the parameter `model.srcmode = redis:full` at its default value. Other values (`redis:invnt`, `grpc`) can be set, but it is not guaranteed that the model will be built correctly. Parameters `grpc.address` and `grpc.port` will have no effect.

Note: it is recommended to configure parameter --http.threads to “4” for 16-Core systems, “16” for 64-Core systems.

Supported REST API

Telemetry API:

- GET monitoring/session/0/data

  Access methods:

  -
    - **Direct access:**

      ```
      curl http://127.0.0.1:8950/monitoring/session/0/data
      ```

    - **Over plugins API:**

      ```
      curl -k https://<lab_ip>/ufmRest/plugin/ufm-light/monitoring/session/0/data -u <user>:<password>
      ```

Topology API:

- GET resources/systems

  Access methods:

  -
    - **Direct access:**

      ```
      curl http://127.0.0.1:8950/resources/systems
      ```

    - **Over plugins API:**

      ```
      curl -k https://<lab_ip>/ufmRest/plugin/ufm-light/resources/systems -u <user>:<password>
      ```

- GET resources/ports

  Access methods:

-
  - **Direct access:**

    ```
    curl http://127.0.0.1:8950/resources/ports
    ```

  - **Over plugins API:**

    ```
    curl -k https://<lab_ip>/ufmRest/plugin/ufm-light/resources/ports -u <user>:<password>
    ```

- GET resources/modules?type=hca

  Access methods:

-
  - **Direct access:**

    ```
    curl http://127.0.0.1:8950/resources/modules?type=hca
    ```

  - **Over plugins API:**

    ```
    curl -k https://<lab_ip>/ufmRest/plugin/ufm-light/resources/modules?type=hca -u <user>:<password>
    ```

Troubleshooting

If the response to `GET monitoring/session/0/data` contains an empty dictionary, ensure that the Fast-API plugin is running.

UFM Light logs are located in:

```
/opt/ufm/files/log/plugins/ufm-light/
```

The default log level is `INFO`. To change the log level to `DEBUG`, edit:

```
sudo vim /opt/ufm/files/conf/plugins/ufm-light/ufm-light.cfg

logging.level = DEBUG
```

After config update, ufm-light plugin must be stopped and started again manually in order to obtain config changes.

Key Performance Indexes (KPI) Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.0.9-0	Added new telemetry KPIs

Bug Fixes

Plugin Version	Bug Fix
1.0.9-0	N/A

Overview

The KPI plugin periodically collects telemetry metrics and topology data from one or multiple UFM Telemetry and UFM clusters to calculate high-level Key Performance Indicators (KPIs). It can operate as a standalone Docker container or as a UFM plugin.

The calculated KPIs and collected telemetry metrics are stored in a Prometheus time-series database using the remote-write Prometheus protocol.

KPIs

NICs Connectivity

- **Name**: `connected_endpoints`
- **Description**: This KPI shows the percentage of NICs connectivity, namely, from all NICs available, how many are connected. The desired value is 100%, meaning all NICs are connected.

  The default threshold for "bad" values is ⩽ 95%.

  > **Note**
  >
  > The complete list of NICs includes all those detected at least once since the plugin is started.

Topology Correctness

- **Name**: `topology_correctness`
- **Description**: This KPI reports the number of wrongly connected links. The ability to spread traffic over multiple minimal-distance paths is critical in utilizing the bandwidth provided by the network.

  Therefore, a pass/fail criteria for the topology not being broken by miss-connections is key for the network to provide reasonable bandwidth to running applications. The

threshold for failure is > 0.

Link Stability

- **Name**: `stability`
- **Description:** For each UID (Unique Identifier), the KPI checks for changes in the link down counter. A change may indicate a link recovery, suggesting there was a period with no link. The statistics are tracked per UID, and the ideal value for link down errors is 0, indicating no problematic links.

Overheated Components

- **Name**: `operating_conditions`
- **Description**: This KPI displays the total number of elements (ports and devices) experiencing operating condition violations. Each hardware element has a predefined normal operating temperature range, with a common default threshold set at 70°C (158°F) or higher.

Bandwidth Loss due to Congestion

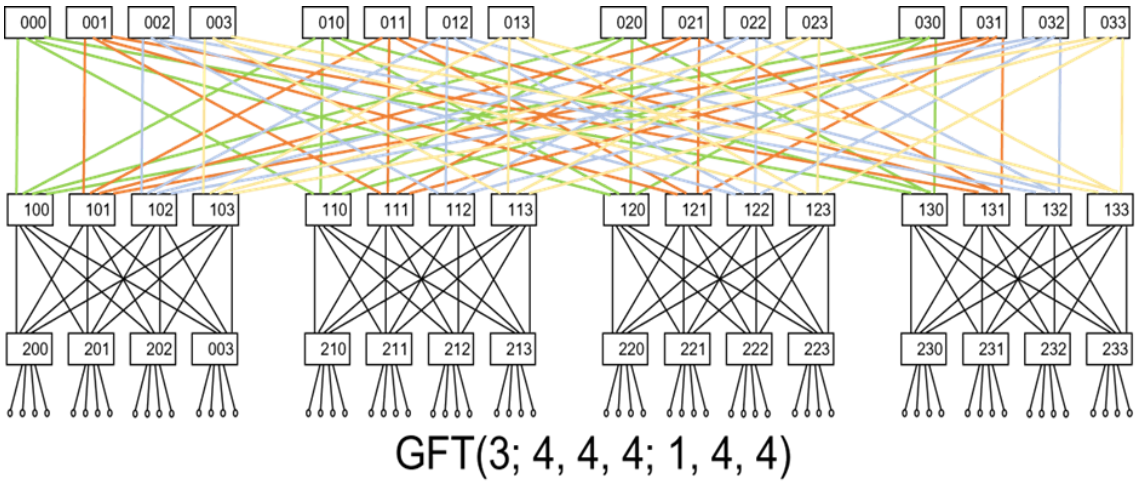
- **Name**: `bw_loss_by_congestion`
- **Description**: An Infiniband (IB) network is lossless and can therefore experience loss congestion spread. Features like congestion control aim to minimize this issue. This KPI reports the percentage of bandwidth loss due to congestion for each layer and direction, as measured by the `port-xmit-wait` counter.

  The reported percentage is calculated by averaging the `xmit-wait` equivalent time with the time between samples across all links in the specified layer-direction group.

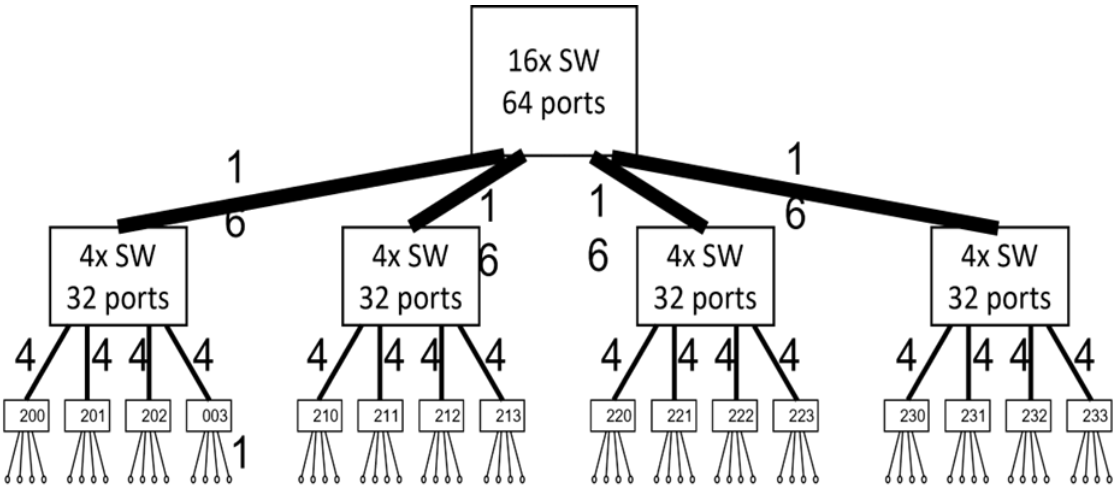
Fat Tree (Single Root) to Tree Conversion

- **Name**: `tree_over_subscription`
- **Description**: Fat Trees were constructed by Leiserson in order to enable building a tree structure using fixed radix K and capacity switches T. Consider a regular K-1 tree (each switch has one parent and K-1 children), which have a single root switch service and after H-1 levels (K-1)^(H-1) leaf switches connecting (K-1)^H hosts each with bandwidth B. Consider the worst case traffic pattern where each leaf switch send traffic to other leaf switches that must cross the top. The connection from each leaf switch up should carry B(K-1) traffic and the capacity of that switch is bi-directional B(K-1) too. One level up the capacity and up link capacity required is B(K-1)^2, etc. But that requires exponentially growing capacity from links and switches.

For example, the below diagram is of a three level fat tree:



The Single Root Tree:



The solution to that problem was to split each single switch in the original Single Root Tree (SRT) into Multi Rooted Tree - which is a Fat Tree. In each level there are exactly same

number of switches connected up and down except for the top level which carry half the number of switches - connecting only down.

The Fat Tree formulations define which switches connect to which other switches but it does not break the basic concept that a Fat Tree can be collapsed back into a simple tree by merging switches into much larger ones. A pair of leaf switches can still be classified by their distance - or the level of their common parents - just like in the original SRT.

When we want to evaluate the damage to the perfect tree structure due to link faults, it is very hard to get an exact maximum flow bandwidth without lengthy N^2 complex algorithm. However, if we examine the SRT obtained by collapsing the Fat Tree back into a tree, we can get a upper bound to the available bandwidth between different branches of the SRT.

This metric evaluates Fat Tree clusters topology and extracts their original Tree structure. Then it evaluates the over-subscription of each sub-tree. This way, the impact of the exact set of missing links can be evaluated in terms of the bandwidth taken out between sub-tress of the topology.

Top-of-rack (ToR) to ToR Max Flow (Bandwidth) Matrix

- **Name**: `tor_to_tor_bw`
- **Description**: This KPI provides a simple metric for the impact of link failures. One such network property is the available bandwidth for pairs of ToRs. This provides a meaningful yet traffic independent metric. So no prior knowledge of the exact set of instantaneous traffic patterns is required.

  It is most important to look at the TOR up-ports since on Fat Trees the number of possible paths rows exponentially when going up the topology towards the roots. So the impact of link faults is decaying exponentially with their level towards the roots.

  It would have been nice if we could just count the number of missing up links on each of the TORs in the pair and claim that TOR X lost x uplinks and TOR Y lost y uplinks, the lost bandwidth is linkBW*min(x,y). But in reality the worst case lost bandwidth is linkBW*(x+y). This is an artifact of the specific links lost and the network structure.

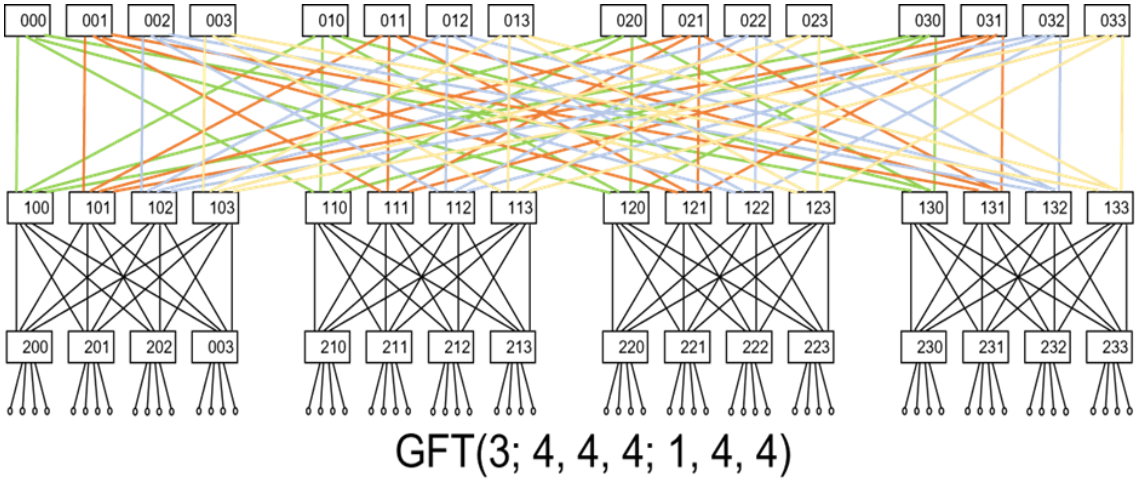
  The code we provide performs a Fat Tree specific computation of the exact available bandwidth for each TOR's pair.

The algorithm provides unique identifiers for sub-trees and consider the subtree each link of the TORs connect to. Then it sums the min number of links connected from each of the TORs to each of the subtrees.

A more straight forward approach utilizing networkx or direct implementation of Dinic's max-flow algorithm yielded much higher runtime, yielding this metric impractical to use.

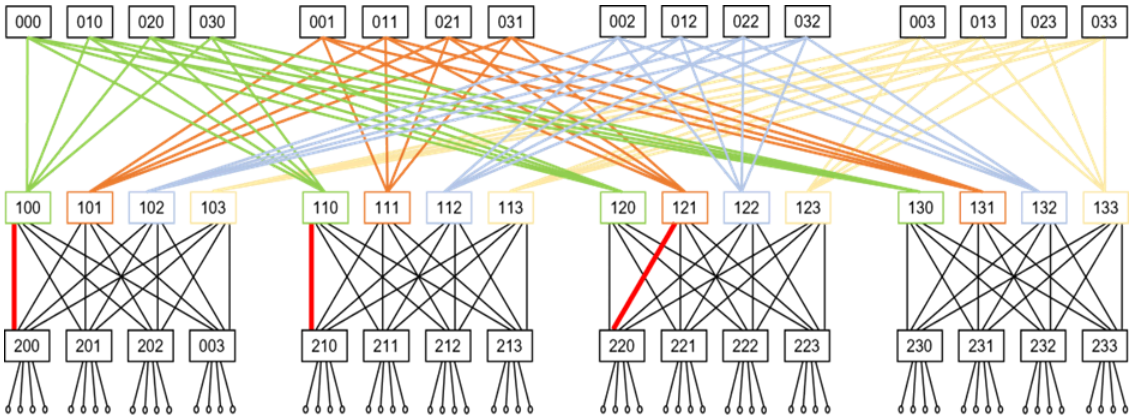
We use an example to demonstrate these concepts:

Consider the Fat Tree GFT(3;4,4,4;1,4,4) depicted in the diagram below:



GFT(3; 4, 4, 4; 1, 4, 4)

We re-arrange the top row of switches (cores) such that the full-bipartites between switch levels 2 and 3 are close together:



GFT(3; 4, 4, 4; 1, 4, 4) Cores Reordered

The re-ordered picture highlights that if we imagine the red links as missing links, then:

1. TORs 200 and 210 missing links are connected to the same bipartite and thus they only lose 1 out of the 4 paths between them.

2. However, TORs 200 and 220, missing links connect to 2 different L2-L3 bipartites and thus 2 different paths out of the 4 possible are lost.

The resulting TOR to TOR max-flow matrix, providing the maximal bandwidth between each pair, in units of single link bandwidth is provided below:

	100	101	102	103	110	111	112	113	120	121	122	123	130	131	132	133
100		3	3	3	3	3	3	3	2	3	3	3	3	3	3	3
101	3		4	4	3	4	4	4	3	4	4	4	4	4	4	4
102	3	4		4	3	4	4	4	3	4	4	4	4	4	4	4
103	3	4	4		3	4	4	4	3	4	4	4	4	4	4	4
110	3	3	3	3		3	3	3	2	3	3	3	3	3	3	3
111	3	4	4	4	3		4	4	3	4	4	4	4	4	4	4
112	3	4	4	4	3	4		4	3	4	4	4	4	4	4	4
113	3	4	4	4	3	4	4		3	4	4	4	4	4	4	4
120	2	3	3	3	2	3	3	3		3	3	3	3	3	3	3
121	3	4	4	4	3	4	4	4	3		4	4	4	4	4	4
122	3	4	4	4	3	4	4	4	3	4		4	4	4	4	4
123	3	4	4	4	3	4	4	4	3	4	4		4	4	4	4
130	3	4	4	4	3	4	4	4	3	4	4	4		4	4	4
131	3	4	4	4	3	4	4	4	3	4	4	4	4		4	4
132	3	4	4	4	3	4	4	4	3	4	4	4	4	4		4
133	3	4	4	4	3	4	4	4	3	4	4	4	4	4	4	

The Algorithm

The following steps are conducted in order to provide that table:

1. DFS from top to bottom collecting the list of parents for each switch. We require all TORs be accessed so we try all top level switches until we find one that is connected to all TORs. After we are done we can ask for every TORs pair what is the level of the lowest common parent. This is required since routing will only use shortest paths, and thus go up to that level only.

2. Establish for each TOR and up the unique bottom-up subtree it is part of. The same sub-tree number should be used for all parents of each switch that are on the same level. Since the topology may be missing some links, it is not simple to check if there was previous allocation of such number, so first all parents are scanned, and then a consolidation step decides which levels are missing a number. If any needs to be set then another step applies the consolidated value.

3. For TOR switch, we sum the total number of links that connect to switches with same bottom-up tree (recognized by their assigned number). We do that for every parent level.

4. Now that we have that data for every TOR, we compute the TOR to TOR max-flow by first looking up the level of their common parent and then computing the the minimal number of flows they connect to each bottom-up tree they connect to.

Congestion Point Indication

- **Name**: queue_depth_average, queue_depth_variance
- **Description**: This is a post-processing of the Egress Queue Depth Histogram feature.

  The egress queue depth is the length of the queue the packet pointers must wait in for them to be transmitted via the port.

  A “healthy” behavior of the queue depth is that it is somewhat constant and short – longer queues imply higher xmit-wait and latency, and jittering queue depth means the congestion on the port changes rapidly and unpredictably.

  The egress queue depth is the length of the queue the packet pointers must wait in for them to be transmitted via the port.

  Following that definition of a healthy queue depth, we would like to know the average of the queue depth, and variance of the queue depth.

  The reported values are calculated per port.

- **Required telemetry configuration**:

  plugin_env_CLX_EXPORT_API_SKIP_PERFORMANCE_HISTOGRAM_BUFFER_DATA=0

  in /opt/ufm/files/conf/secondary_telemetry_defaults/launch_ibdiagnet_config.ini

Lost Bandwidth Indication

- **Name**: lost_bandwidth_percentage
- **Description**: This is a post-processing of the PortRcvData Histogram feature.

  PortRcvData Histogram samples the amount of data at the ingress at short timescales, where the behavior should be 100% BW usage or 0% BW usage.

Anything in between might give a sign for “unhealthy” port, as described in the NCCL pickle research.

The reported values are calculated per port.

- **Required telemetry configuration**:

  plugin_env_CLX_EXPORT_API_SKIP_PERFORMANCE_HISTOGRAM_PORTS_DATA=0

  in /opt/ufm/files/conf/secondary_telemetry_defaults/launch_ibdiagnet_config.ini

Bitflip Indication

- **Name**: bitflips_generated
- **Description**: A bitflip is an expected occurrence in every digital part because of noise coming from diverse sources.

  Usually, these sources across the cluster will have similar effect on each node, but in some cases, because of localized problems (manufacturing defects, elevated temperatures, power supply issues etc.), a few nodes might experience extremely high bitflip rates, thus making packet processing difficult and causing many packets to drop.

  We would like to narrow down on the affected nodes, in these cases, and the ICRC mechanism should be able to detect them.

  Bit flips can only be detected within the ingress ICRC counter of nearby connected switches and not within the affected switch itself.

  The reported values are calculated per switch.

- **Required telemetry configuration**:

  plugin_env_CLX_EXPORT_API_SKIP_PORT_GENERAL_COUNTERS=0

  in /opt/ufm/files/conf/secondary_telemetry_defaults/launch_ibdiagnet_config.ini

Deployment

The plugin could be deployed as a standalone application or as a UFM plugin.

Deploy the KPI as a UFM plugin

1. Pull/load the latest image of the plugin:

```
docker pull mellanox/ufm-plugin-kpi
```

2. Pull/load the latest image of the plugin:

```
/opt/ufm/scripts/manage_ufm_plugins.sh add -p kpi -t <TAG>
```

Deploy the KPI as a Standalone application

1. Pull/load the latest image of the plugin:

```
docker pull mellanox/ufm-plugin-kpi
```

2. Pull/load the latest image of the plugin:

```
docker run --network host -v
/opt/ufm/files/conf/plugins/kpi:/config -v
/opt/ufm/files/logs/plugins/kpi:/logs --rm -dit $IMAGE
```

Configurations

The configurations can be managed through a configuration file.

- **Within the container**, the configurations file can be found under `/config/kpi_plugin.conf`
- **On the host**, the shared volume location of the /config is `/opt/ufm/files/conf/plugins/kpi/kpi_plugin.conf`

Clusters Configurations

The default KPI configurations KPI include one default cluster called `unknown`, once the plugin starts, the default cluster is configured to collect the secondary telemetry endpoint http://localhost:9002/csv/metrics and the local UFM topology data.

To change the parameters of the default cluster, or to add additional clusters, please refer to the following configuration under the `kpi_plugin.conf`:

```
### Set name to "cluster-config-$cluster_name". Add section per each cluster
[cluster-config-unknown]

### uncomment and set 2 following options:
#host_list = host_name[1-1024],another_host_name[1-100]
### threshold to distinguish between hosts and switches
(inclusive)
#host_max_ports = 4

### OR

### If hostlist format is not possible, another option is per-
level regular expressions
### That is a list of regular expressions to detect nodes level
in topology (0 is lowest - hosts)
### Key structure is 'level.<level number>.<per-level running index>'
#level.0.0=some-host-pattern-\d+
#level.0.1=another-host-pattern-\d+
#level.1.0=leaf-pattern-l\d+
#level.2.0=spine-pattern-s\d+
#level.3.0=core-pattern-c\d+


### If running as standalone app please uncomment and set next 3
lines
#ufm_ip=0.0.0.0
#ufm_access_token=1234567890abcdefghijklmnopqrst
#telemetry_url=http://0.0.0.0:9100


### If running as UFM plugin and UFM port has changed
#ufm_port=1234
```

Property	Description	Default Value
`cluster.host_list`	Set of hosts in hostlist format. Used to detect the topology leafs.	None
`cluster.host_max_ports`	The maximal number of ports in a server. Used as threshold to classify a node as server / switch.	4
`cluster.level.X.Y`	Regular Expression to capture topology levels. The first index (X) is the level and the following index (Y) is a running index for level X	None
`cluster.ufm_ip`	UFM IP that used to collect the topology data	'127.0.0.1'
`cluster.ufm_port`	UFM port that used to collect the topology data in case the cluster is local	'8000'
`cluster.ufm_access_token`	The UFM access token that should be provided in case the cluster is collecting data from a remote UFM	None
`cluster.telemetry_url`	UFM telemetry endpoint URL that used to collect the telemetry metrics.	'http://127.0.0.1:9002'

Property	Description	Default Value
`cluster.telemetry_metrics_push_delta_only`	If True, only changed telemetry metrics are pushed to Prometheus after each pulling interval, with a fallback to push unchanged metrics if they remain static for over an hour. Otherwise, all fetched metrics will be pushed to the Prometheus after each pulling interval	True

KPI Configurations

The configurations are related to the KPI plugin itself. please refer to the below configuration section in order to manage the KPI generic configurations:

```
[kpi-config]
### Optional Comma separated list that contains list of the
disabled KPIs that we don't want to store them in Prometheus DB
### Available KPIs:
###
connected_endpoints,topology_correctness,stability,operating_condi
disabled_kpis=telemetry_metrics
```

Property	Description
`disabled_kpis`	Optional Comma-separated list that contains the list of the disabled K The available KPIs are: connected_endpoints,topology_correctness,stability,operating_condit

Prometheus Configurations

The plugin includes a local Prometheus server instance that is used by default, the following parameters are used to manage the Prometheus configurations:

```
[prometheus-config]
prometheus_ip=0.0.0.0
prometheus_port=9090
prometheus_db_data_retention_size=120GB
prometheus_db_data_retention_time=15d
```

Property	Description	Default Value
`prometheus_ip`	IP of Prometheus server	0.0.0.0
`prometheus_port`	Port of Prometheus server	9090
`prometheus_db_data_retention_size`	Data retention policy by size (used only for the local Prometheus server)	120GB
`prometheus_db_data_retention_time`	Data retention policy by time (used only for the local Prometheus server)	15d

The data storage path of the local Prometheus DB is under `/opt/ufm/files/conf/plugins/kpi/prometheus_data`.

Global Time Interval Configurations

The following is used by all clusters globally, each property could be overridden by adding it under the cluster's section.

```
[time-interval-config]
telemetry_interval=300
ufm_interval=60
```

Property	Description	Default Value
`telemetry_interval`	Polling interval for the telemetry metrics data in seconds	300
`ufm_interval`	Polling interval for the UFM APIs in seconds	60
`disabled_kpis`	Polling interval for the connected_endpoints KPI specifically	60

Logs Configurations

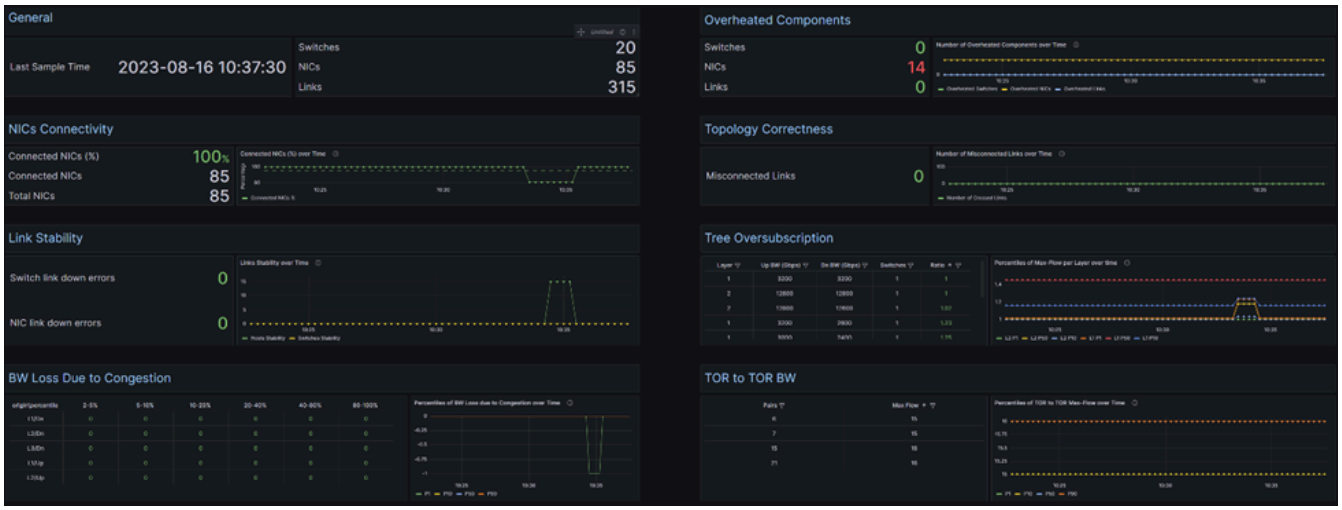
The below configurations are to manage the `kpi_plugin.log` file

```
[logs-config]
logs_level = INFO
logs_file_name = /log/kpi_plugin.log
log_file_max_size = 10485760
log_file_backup_count = 5
```

Grafana Dashboard Templates

The KPI plugin provides several Grafana dashboard templates that the Grafana users can import, these dashboards display the KPIs panels and graphs. The main dashboard is a cluster view that shows the cluster's KPIs. The dashboard should be connected to the KPI Prometheus server.

The dashboard JSON template can be found under `/opt/ufm/files/conf/plugins/kpi/grafana/`:



Logs

The following logs are exposed under `/opt/ufm/files/logs/plugins/kpi/` in case of UFM plugin mode, and under `/log` inside the container in case of Standalone mode.

- `kpi_plugin.log` - The application logs.
- `kpi_plugin_stderr/stdout` - The application service logs
- `prometheus.log` - The local Prometheus logs.

REST APIs

Get List of Configured Clusters

- URL:
  - UFM Plugin: https://<IP>/ufmRest/plugin/kpi/api/cluster/__names__
  - Standalone: http://<IP>:8686/api/cluster/__names__
- Method: GET

- Response: List of cluster names strings

Get KPIs Information

- URL:
  - UFM Plugin: https://<IP>/ufmRest/plugin/kpi/files/kpi_info
  - Standalone: http://<IP>:8686/files/kpi_info
- Method: GET
- Response: List of KPI names and HTML descriptions.

Get KPIs Values

- URL:
  - UFM Plugin: https://<IP>/ufmRest/plugin/kpi/api/cluster/<cluster_name>
  - Standalone: http://<IP>:8686/api/cluster/<cluster_name>
- Method: GET

Response: A list of KPIs with the relative calculates graph values to be used in the UI. Generally, 2 main components are expected: the table data (key “2d_matrix_data”) and the graph data (key “graph_data”). Each value corresponds to rules expected by the UI:

- `display name` – KPI display name
- `2d_matrix_data` – Latest data in a table-like format. Every list item is a dict that makes a table row - corresponding to the columns of that row.
  - `data` – List of dict, each dict is a column in the row. May contain:
    - `percentage` – Value in percentage.
    - `value` – Raw value.

        - `name` – Column header
    - `direction` – For arrows icon (up / down)
    - `name` – Row header
- `graph_data` – dictionary with the following items:
    - `multi_line_graph` – time series graph for one or more series.
        - `title` – Graph title.
        - `x_label` – X axis label.
        - `x_values` – X axis values.
        - `y_labels` – Y axis labels.
        - `y_values` – Y axis values.
    - `info` – string with information of the graph data.

Get KPI Plugin Overview

- URL:
    - UFM Plugin: https://<IP>/ufmRest/plugin/kpi/overview?time_window=<value> (in hours)
    - Standalone: http://<IP>:8686/overview?time_window=<value> (in hours)
- Method: GET

Response:

```
{
    "clusters_telemetry_info": [
        {
            "cluster_name": "unknown",
            "interval": 60,
            "url": "http://pjazz:9002/csv/metrics"
        }
    ],
    "prometheus_info": {
        "db_statistics": {
            "appended_samples_rate_per_sec": 11.46,
            "bytes_rate_per_sample": 9.227,
            "total_compressed_blocks_size_bytes": 272329313.0,
            "total_head_chunks_size_bytes": 1937238.0,
            "total_wal_size_bytes": 2989807.0
        },
        "url": "http://pjazz:9191"
    }
}
```


Where:

`clusters_telemetry_info`: contains a list of the configured clusters' information, each cluster has the following properties:

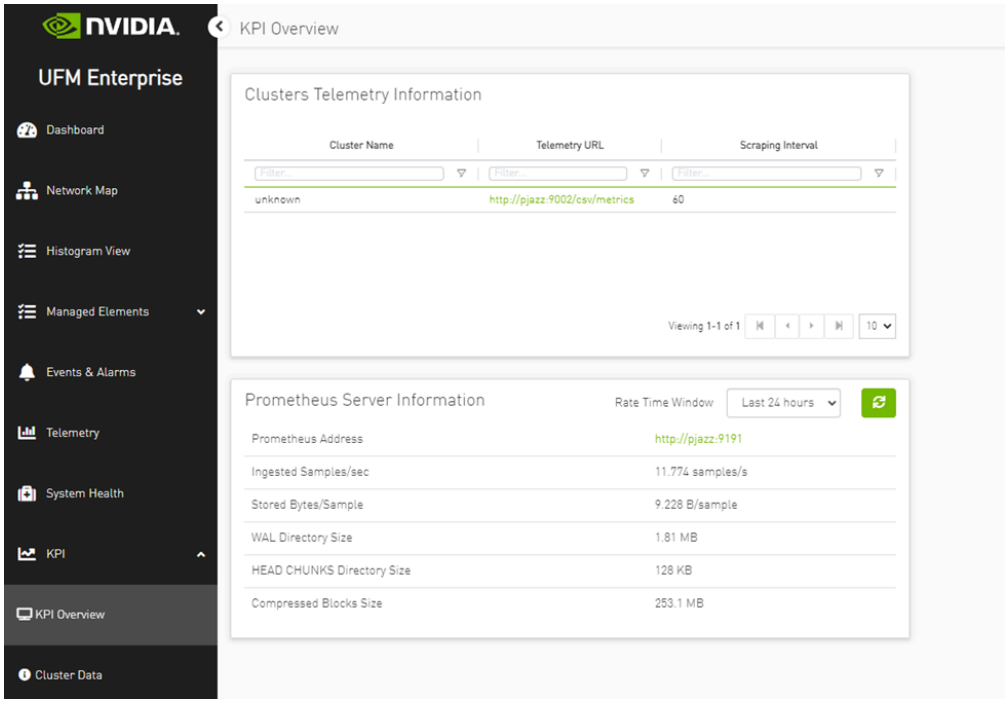
Property	Description	Default Value
`cluster_name`	Cluster's name	One cluster with name 'unknown'
`interval`	Cluster's telemetry pulling time interval	60
`url`	The cluster's URL telemetry	http://0.0.0.0:9002/csv/metrics

prometheus_info: Contains the general Prometheus configurations (e.g. the URL) and statistics about the collected samples (for the local Prometheus mode)

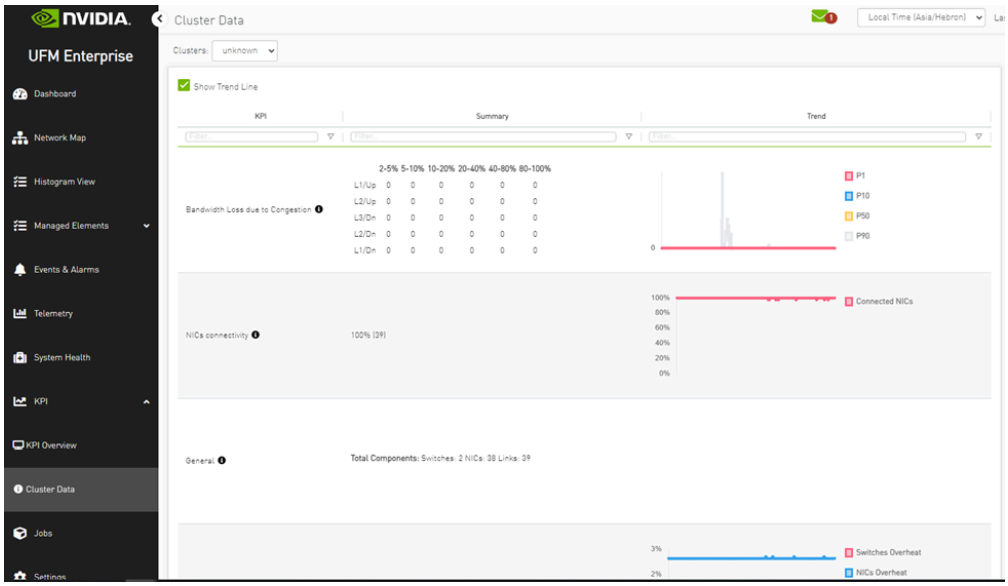
Property	Description
`db_statistics.appended_samples_rate_per_sec`	Prometheus rate of the total appended sar following Prometheus expression: ***rate(prometheus_tsdb_head_samples_ap***
`db_statistics.bytes_rate_per_sample`	Prometheus rate of ingested bytes per sar Prometheus expression: ***rate(prometheus_tsdb_compaction_chun / rate(prometheus_tsdb_compaction_chu***
`db_statistics.total_compressed_blocks_size_bytes`	Total compressed blocks size in bytes that the following Prometheus expression: ***prometheus_tsdb_storage_blocks_bytes***
`db_statistics.total_head_chunks_size_bytes`	Total HEAD chunks blocks size in bytes tha by the following Prometheus expression: ***prometheus_tsdb_head_chunks_storage_***
`db_statistics.total_wal_size_bytes`	Total WAL size bytes, calculating by the fol ***prometheus_tsdb_wal_storage_size_byte***
`prometheus_info.url`	The promethues's URL telemetry

UI Views

KPI Overview



Cluster KPIs Data



UFM Events Grafana Dashboard Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.0.2-0	**Limitations:** • Not supported on UFM Gen 2.0 • FluentD fails with RHEL9 OS .

Bug Fixes

Plugin Version	Bug Fix
1.0.2-0	N/A

Overview

This plugin monitors and parses UFM events from the `event.log` file using `Fluentd`. It pushes the logs to a Loki server and visualizes them through Grafana dashboards.

Installation

1. **Install Plugin** on UFM.
2. **Access Grafana Dashboard** at: `http://<IP>:3002` (default credentials: `admin/admin`).

Components

- **Fluentd**: Parses `event.log` file, extracts relevant fields, and pushes logs to Loki.
- **Loki**: Aggregates and stores logs.

- **Grafana**: Visualizes logs with predefined panels (e.g., Events Stats, Topology Changes, Hardware Events).

Configuration Files

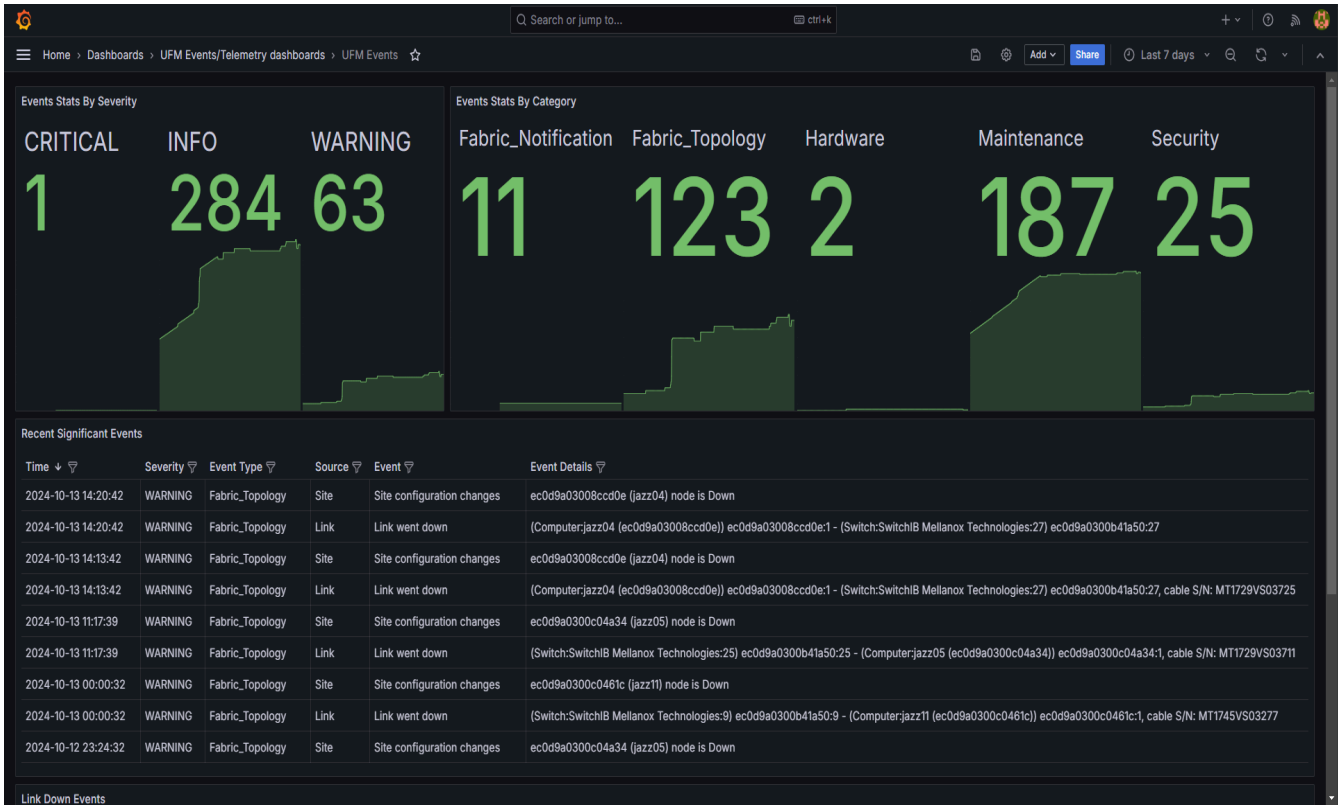
- Fluentd:
  `/opt/ufm/files/conf/plugins/ufm_events_grafana_dashboard/fluentd/`
- Loki:
  `/opt/ufm/files/conf/plugins/ufm_events_grafana_dashboard/loki/lok`
  `local-config.yaml`
- Grafana:
  `/opt/ufm/files/conf/plugins/ufm_events_grafana_dashboard/grafana/`

Logs

Plugin logs are stored in `/opt/ufm/files/log/plugins/ufm_events_grafana_dashboard/`.

Usage

Navigate to Grafana -> Dashboards -> UFM Events to view predefined panels or create custom queries under the Explore section.



Log Streamer Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.0.1-2	N/A

Bug Fixes

Plugin Version	Bug Fix
1.0.1-2	N/A

Overview

The Log Streamer feature allows users to stream external logs (such as OpenSM, SHARPAM, etc.) to a remote syslog server.

Prerequisites

- UFM plugin system
- Access to UFM configuration files
- A remote syslog server to receive logs
- UFM Enterprise version 6.21.0 or later

Installation

The Log Streamer plugin is installed as part of the UFM plugin system. No additional installation steps are required.

Basic Configuration

The plugin is configured in the `gv.cfg` file under the Logging section. Key configuration parameters include:

- `syslog` – set to `true` to enable streaming, `false` to disable
- `syslog_addr` – IP address of the remote syslog server, optionally with port (`<IP>:<PORT>`)
- `files_to_stream` – list of log files to stream, separated by commas. Files must be located in `/opt/ufm/files/log/`

Ensure that `imudp` is enabled on the remote server by removing the `#` character and restarting the `rsyslogd` service:

```
# Provides UDP syslog reception
$ModLoad imudp
$UDPServerRun 514
```

Usage

Enabling the Plugin

- Load the Docker image
- Configure the `gv.cfg` file with the required settings
- Enable the plugin via plugin management

Example Configuration

The following is an example configuration:

```
[Logging]
syslog_addr = 192.168.0.1:515
files_to_stream =
opensm.log,secondary_telemetry/ibdiagnet2_port_counters.log
syslog = true
```

- Stream logs from `opensm.log` and `secondary_telemetry/ibdiagnet2_port_counters.log`
- Send them to the remote syslog server at `192.168.0.1`
- Use the UDP protocol on port `515`

Log File Locations

All files selected for streaming must be located under `/opt/ufm/files/log/`

Removing the Plugin

Disable the plugin through the UFM interface.

GNMI NVOS Events Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.0.1-8	N/A

Bug Fixes

Plugin Version	Bug Fix
1.0.1-8	Improved Logging

The GNMI NVOS Events plugin is a standalone Docker container managed by UFM. Its main role is to collect GNMI events from NVOS switches and relay them to UFM as external events. This capability improves the user experience by delivering more detailed information about switches within the InfiniBand fabric through UFM events and alarms.

Deployment

There are two potential deployment options for the GNMI NVOS Events plugin:

- On UFM Appliance

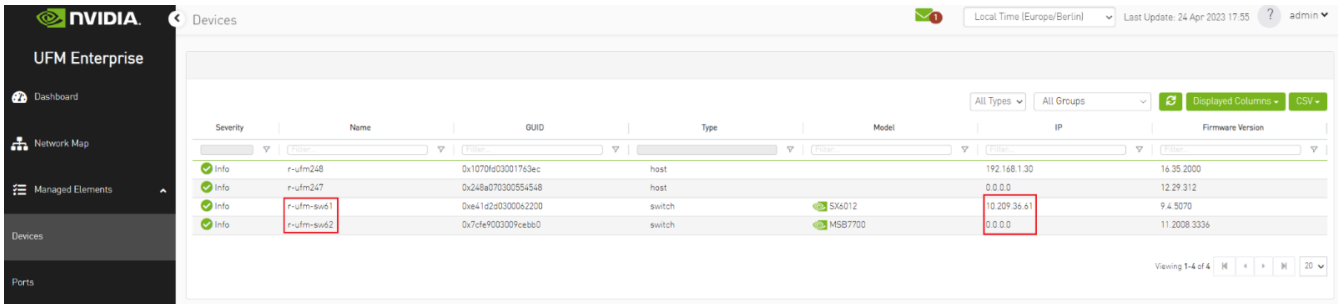
- On UFM Software

For detailed instructions on how to deploy the GNMI NVOS Events plugin, refer to this page.

Usage

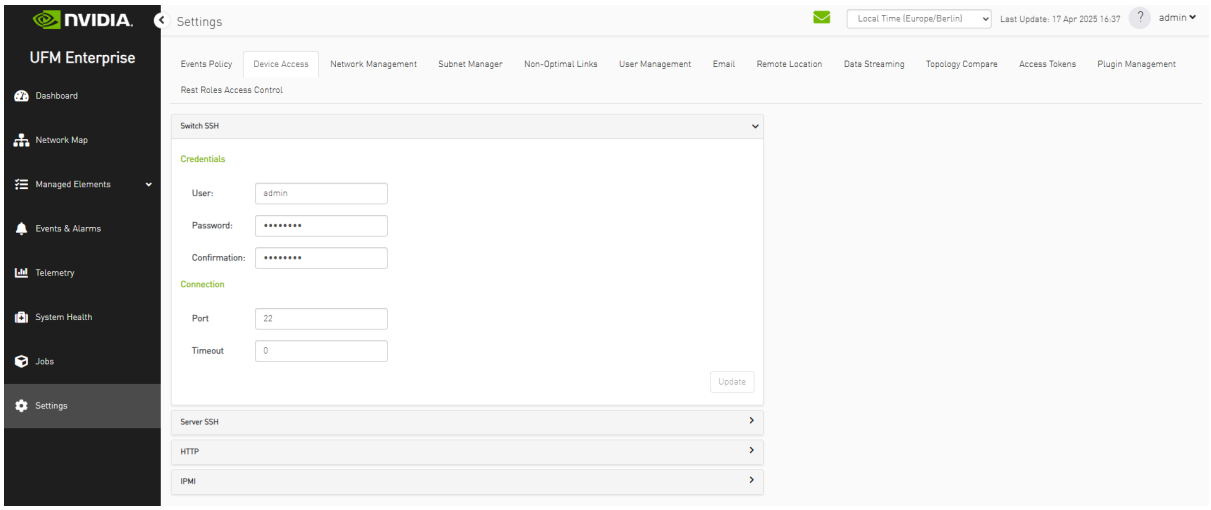
By default, upon initialization, the GNMI NVOS Events plugin captures events from all managed NVOS switches within the fabric.

It is important to ensure that the switch is visible to UFM and has a valid IP address. As illustrated in the following example, switch events will only be received from "r-ufm-sw61".

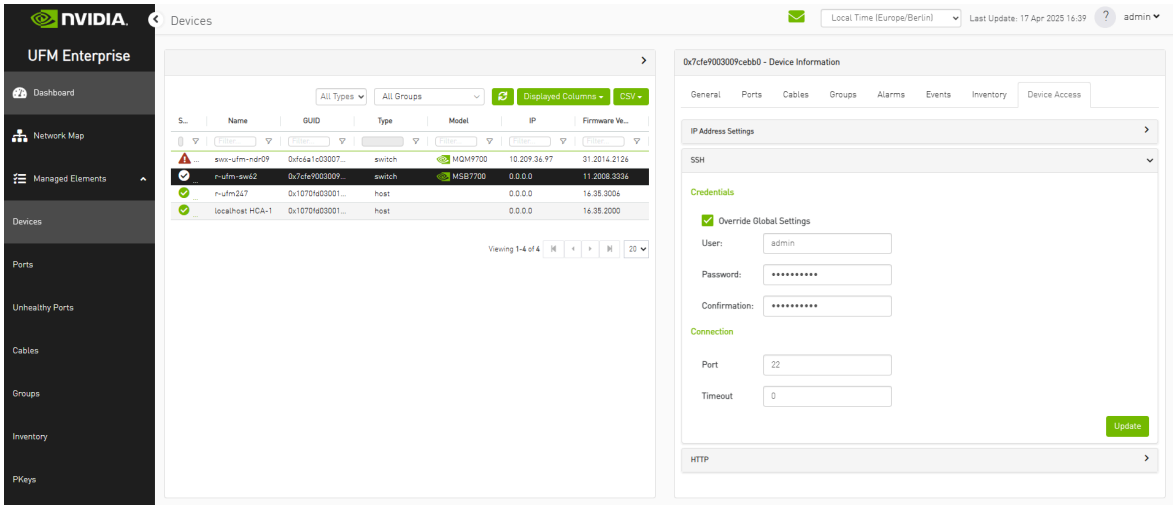


An additional requirement is to have the correct credentials to the switch. The credentials could be set globally for the whole fabric or locally for each switch.

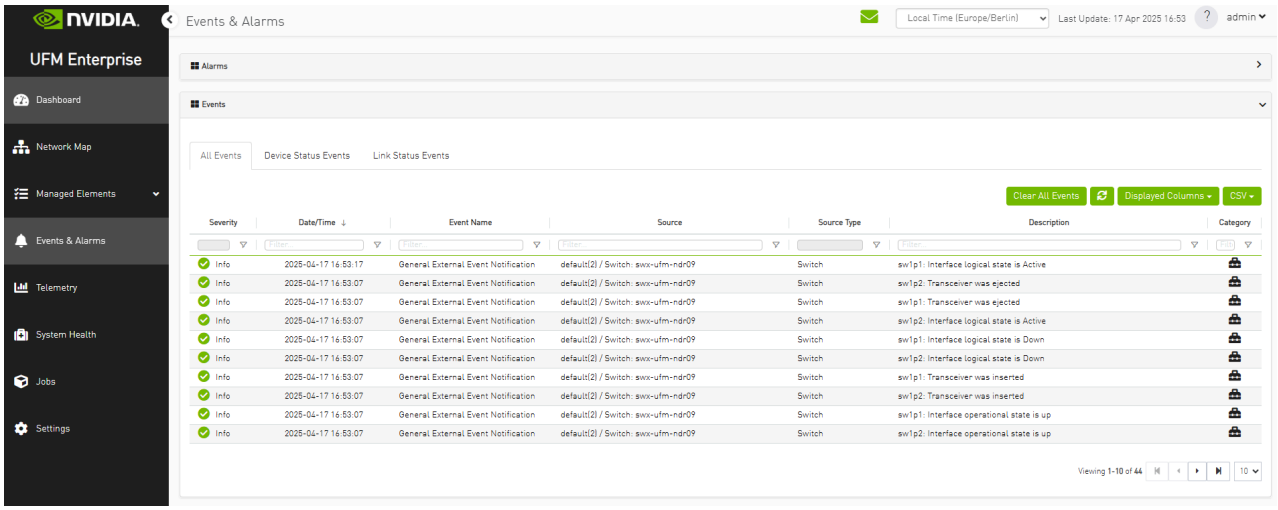
Globally:



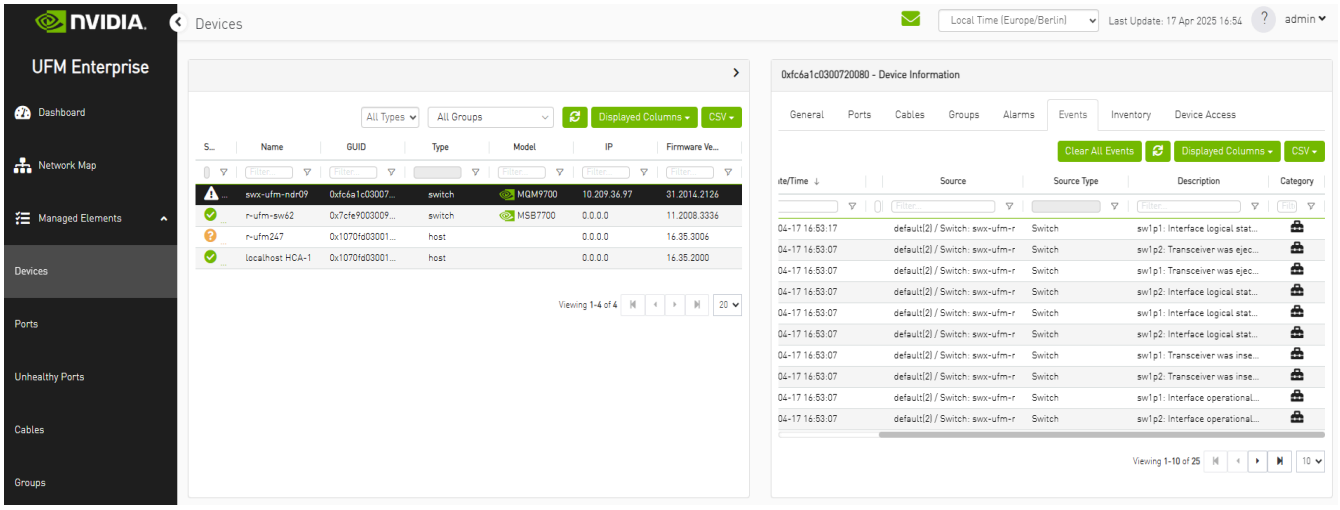
Locally:



The following is an example of events received by the GNMI NVOS Events plugin and displayed as UFM events:



Additionally, there is an option to verify events/alarms for a particular switch:



The GNMI NVOS Events plugin performs a periodic check of the fabric every 600 seconds, allowing for prompt receipt of events from new switches or updated IP addresses of existing switches in under 600 seconds. This interval may be adjusted via the "`ufm_switches_update_interval`" option. The initial update will be performed in 180 seconds after the plugin startup. This grace period gives UFM time to initialize the fabric and also gives user to set up switches credentials. This interval may be adjusted via the "`ufm_first_update_interval`" option.

Here is the list of available events:

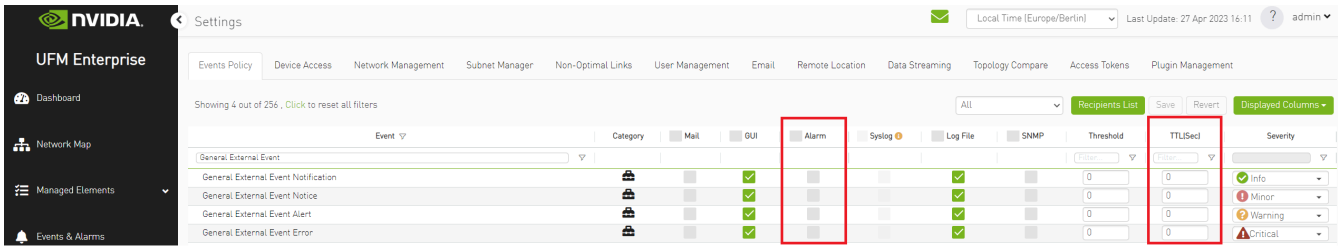
```
    Event ID  Severity       Component  Description
Timestamp
    --------  -------------  ---------  -------------------------
----------  -------------------
    313       INFORMATIONAL  sw1p1      Interface logical state
is Active    2025-04-17 14:53:09
    312       INFORMATIONAL  sw1p2      Interface logical state
is Active    2025-04-17 14:53:06
    311       INFORMATIONAL  sw1p2      Interface operational
state is up    2025-04-17 14:53:04
    310       INFORMATIONAL  sw1p1      Interface operational
state is up    2025-04-17 14:53:04
    309       INFORMATIONAL  sw1p1      Transceiver was inserted
2025-04-17 14:53:02
    308       INFORMATIONAL  sw1p2      Transceiver was inserted
2025-04-17 14:53:02
    307       INFORMATIONAL  sw1p2      Interface logical state
is Down      2025-04-17 14:52:58
    306       INFORMATIONAL  sw1p1      Interface logical state
is Down      2025-04-17 14:52:58
    305       INFORMATIONAL  sw1p1      Transceiver was ejected
2025-04-17 14:52:57
    304       INFORMATIONAL  sw1p2      Transceiver was ejected
2025-04-17 14:52:57
    303       INFORMATIONAL  sw1p2      Interface operational
state is down  2025-04-17 14:52:57
    302       INFORMATIONAL  sw1p1      Interface operational
state is down  2025-04-17 14:52:57
    301       INFORMATIONAL  System     Health status is ok
2025-04-17 10:46:07
    300       INFORMATIONAL  System     Cleared: Health status is
not ok     2025-04-17 10:46:07
    299       INFORMATIONAL  PSU2/FAN   HW component goes back to
normal     2025-04-17 10:46:07
```

```
    298        INFORMATIONAL  PSU2/FAN   Cleared: PSU2/FAN is not
working      2025-04-17 10:46:07
    297        WARNING        System     Health status is not ok
2025-04-17 10:46:01
    296        WARNING        PSU2/FAN   PSU2/FAN is not working
2025-04-17 10:46:01
    295        INFORMATIONAL  System     Health status is ok
2025-04-17 10:45:58
    294        INFORMATIONAL  System     Cleared: Health status is
not ok       2025-04-17 10:45:58
    293        INFORMATIONAL  PSU2/FAN   HW component goes back to
normal       2025-04-17 10:45:58
    292        INFORMATIONAL  PSU2/FAN   Cleared: PSU2/FAN is not
working      2025-04-17 10:45:58
    291        WARNING        System     Health status is not ok
2025-04-17 10:45:34
    290        WARNING        PSU2/FAN   PSU2/FAN is not working
2025-04-17 10:45:34
    289        INFORMATIONAL  System     Health status is ok
2025-04-17 10:45:31
    288        INFORMATIONAL  System     Cleared: Health status is
not ok       2025-04-17 10:45:31
    287        INFORMATIONAL  PSU2/FAN   HW component goes back to
normal       2025-04-17 10:45:31
    286        INFORMATIONAL  PSU2/FAN   Cleared: PSU2/FAN is not
working      2025-04-17 10:45:31
    285        WARNING        System     Health status is not ok
2025-04-17 10:45:13
    284        WARNING        PSU2/FAN   PSU2/FAN is not working
2025-04-17 10:45:13
    283        INFORMATIONAL  System     Health status is ok
2025-04-17 10:44:58
    282        INFORMATIONAL  System     Cleared: Health status is
not ok       2025-04-17 10:44:58
    281        INFORMATIONAL  PSU2/FAN   HW component goes back to
normal       2025-04-17 10:44:58
```

```
    280        INFORMATIONAL  PSU2/FAN   Cleared: PSU2/FAN is not
working      2025-04-17 10:44:58
    279        WARNING        PSU2/FAN   PSU2/FAN is not working
2025-04-17 10:44:52
    278        WARNING        PSU2/FAN   PSU2/FAN speed is out of
range        2025-04-17 10:44:16
    277        WARNING        System     Health status is not ok
2025-04-17 10:43:46
    276        WARNING        PSU2/FAN   PSU2/FAN is not working
2025-04-17 10:43:46
    275        INFORMATIONAL  System     Health status is ok
2025-04-17 10:43:43
    274        INFORMATIONAL  System     Cleared: Health status is
not ok       2025-04-17 10:43:43
    273        INFORMATIONAL  PSU2/FAN   HW component goes back to
normal       2025-04-17 10:43:43
    272        INFORMATIONAL  PSU2/FAN   Cleared: PSU2/FAN is not
working      2025-04-17 10:43:43
    271        WARNING        System     Health status is not ok
2025-04-17 10:42:46
    270        WARNING        PSU2/FAN   PSU2/FAN is not working
2025-04-17 10:42:46
    269        INFORMATIONAL  System     Health status is ok
2025-04-17 10:42:34
    268        INFORMATIONAL  System     Cleared: Health status is
not ok       2025-04-17 10:42:34
    267        INFORMATIONAL  PSU2/FAN   HW component goes back to
normal       2025-04-17 10:42:34
    266        INFORMATIONAL  PSU2/FAN   Cleared: PSU2/FAN is not
working      2025-04-17 10:42:34
    265        WARNING        System     Health status is not ok
2025-04-17 10:42:16
    264        WARNING        PSU2/FAN   PSU2/FAN is not working
2025-04-17 10:42:16
```

To ensure the uninterrupted reception of traps from switches within a large fabric, changes must be made to the UFM configuration in the [`/opt/ufm/conf/gv.cfg`] file's [Events] section. Specifically, the "`max_events`" option should be raised from 100 to 1000, while "`medium_rate_threshold`" and "`high_rate_threshold`" should both be set to 500. To implement configuration adjustments, disable and then enable the plugin.

In case of an event storm, it is necessary to adjust the Event Policy settings such that General Events are non-alarmable and the TTL is set to zero, as illustrated in the following screenshot:



Other

Additional configurations are located in "`/opt/ufm/conf/plugins/gnmi_nvos_events_plugin/gnmi_nvos_events.conf`". To implement configuration adjustments, disable and then enable the plugin. For instructions on modifying the appliance, please refer to the UFM-SDN Appliance Command Reference Guide .

Logs for the SNMP plugin are stored in "/opt/ufm/logs/gnmi_nvos_events.log". For guidance on accessing logs on the appliance, please refer to the UFM-SDN Appliance Command Reference Guide .

NVLink Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature	Description
1.2.2-0	Partitions Management	Added the ability to manage the partitions by creating, updating and removing partitions. For more information, refer to Partitions View.
	Compute and Switch Nodes Views	Added the ability to view the available compute and switch nodes. For more information, refer to Switch Nodes View and Compute Nodes View.

Bug Fixes

Plugin Version	Bug Fix
1.2.2-0	N/A

Overview

The NVLink plugin enables centralized monitoring and management of multiple NVLink domains through both the UFM UI and REST APIs. At its core is the NMX Aggregator (NMXAGGR), which connects to multiple NVLink domains, gathers data from their NMX Controllers (NMX-C), and consolidates information about monitored components. By default, the plugin includes a built-in NMXAGGR, but it can also be configured to connect to an external NMXAGGR instance—either on the same host or a different system. Communication with NVLink domains is performed via the NMX-C using a gRPC-based API.

Deployment

1. **Download the Plugin Image**

   Run the following command to download the NVLink plugin image:

   ```
   docker pull mellanox/ufm-plugin-nvlink
   ```

2. **Load the Plugin into UFM**

   After downloading, you can load the plugin into UFM using one of the following methods:

   - **Via UFM UI:**

     Navigate to **Settings → Plugins Management** in the UFM web interface.

   - **Via Command Line:**

     Execute the following command on the UFM server terminal:

     ```
     /opt/ufm/scripts/manage_ufm_plugins.sh add -p nvlink
     ```

Usage

Container Volume Mapping

The UFM plugin management system creates the following mappings between the plugin docker container file system and the host machine one:

Container Directory	Host Directory
`/config`	`/opt/ufm/files/conf/plugins/nvlink`
`/log`	`/opt/ufm/files/log/plugins/nvlink`

Any file system path mentioned in this document refers to the **container's file system**, unless stated otherwise.

NVLink Domains Connection Security

The plugin, specifically its NMXAGGR component, interacts with NVLink domains over a gRPC connection. In this setup, the domain controller (NMX-C) acts as the server, while NMXAGGR functions as the client.

NMXAGGR supports three modes of gRPC communication:

1. **Insecure** – No encryption is used. This is the default mode.
2. **Server-side TLS** – Communication is encrypted. Only the server needs to present a certificate to the client. This mode is enabled by setting the `cacert` option (refer to the Configuration section).
3. **Mutual TLS (mTLS)** – Communication is encrypted, and both the client and server must authenticate each other using certificates. This mode requires setting the `cacert`, `cert`, and `key` options (refer to the Configuration section).

Managed Domains List

All NVLink domains that are managed or monitored by NMXAGGR are recorded in a list stored in the file `<data_dir>/domains.csv` (see the *Configuration* section for the definition of `data_dir`).

This file serves as an alternative method—alongside the Web UI and REST API—for adding or removing managed domains. Each line in the file contains three comma separated values:

- host - a hostname or an IP address, both of which can incorporate numerical ranges to define multiple hosts in one line; required
- controller port - a port number of the gRPC endpoint of a domain controller (**NMX-C**); not required; default value is **9370**
- telemetry port - a port number of the management gRPC endpoint of a telemetry (**NMX-T**); not required; default value is **9351**

Examples:

- `10.222.16.333,9370,`
- `nv-dmn-01,,6666`
- `10.222.[16,17,20-28].[330-350],,`
- `nv-dmn-[01-8],9371,9355`

Note

For any changes made directly to the file to take effect, the plugin must be restarted.

Note

When NMXAGGR writes the file (as a result of changes to the managed domains list performed via UI or REST API), it expands addresses containing ranges and writes one address per line.

Configuration

The plugin can be configured by editing the config file `/config/nvlink_plugin.conf.`

There are two sections in the config file:

nmxaggr

Option	Description	Default
`use_internal`	If `true` an internal NMXAGGR will be used, otherwise an external one will be used.	`true`
`api_address`	The address of the NMXAGGR REST API server.	`http://127.0.0.1:8966`

Option	Description	Default
`cacert`	The path to a file containing trusted root certificates for verifying NMX-C servers. If not set, insecure gRPC connections will be used.	`-`
`cert`	The path to a file containing client certificate to present to NMX-C servers. Must be used with `cacert` and `key` options.	`-`
`key`	The path to a file containing client private key to present to NMX-C servers. Must be used with `cacert` and `cert` options.	`-`
`data_dir`	The path to a directory where the internal NXAGGR will store its persistent data.	/config
`periodic_fetch_delay`	In the case the plugin fails to subscribe to domain change notifications, the periodic data fetches from a domain will be performed. This option specifies the delay between those periodic fetches in a duration string format[1].	`30s`

Option	Description	Default
`supplementary_fetch_delay`	Normally, after the initial data fetch, data will be fetched from a domain only upon receiving a change notification from a domain controller. Additionally, supplementary fetch will be initiated if there is a long delay since the last fetch. This option specifies the delay in a duration string format[1].	`1m`

[1] A duration string is a sequence of decimal numbers, each with optional fraction and a unit suffix, such as `300ms`, `1.5h` or `2h45m` . Valid time units are `ns`, `us`, `ms`, `s`, `m`, `h` .

logging

Option	Description	Default
`nvlink_log`	The path to the plugin log file.	`/log/nvlink.log`
`nmxaggr_log`	The path to the internal NMXAGGR log file.	`/log/nmxaggr.log`
`log_level`	The log level. Possible values: `DEBUG`, `INFO`, `WARNING`, `ERROR`, `CRITICAL`.	`INFO`
`log_file_max_size`	The maximal size of a log file after which the file is rotated.	`10485760` [1]
`log_file_retain_count`	The number of rotated log files to retain.	`5`

[1] 10 MB

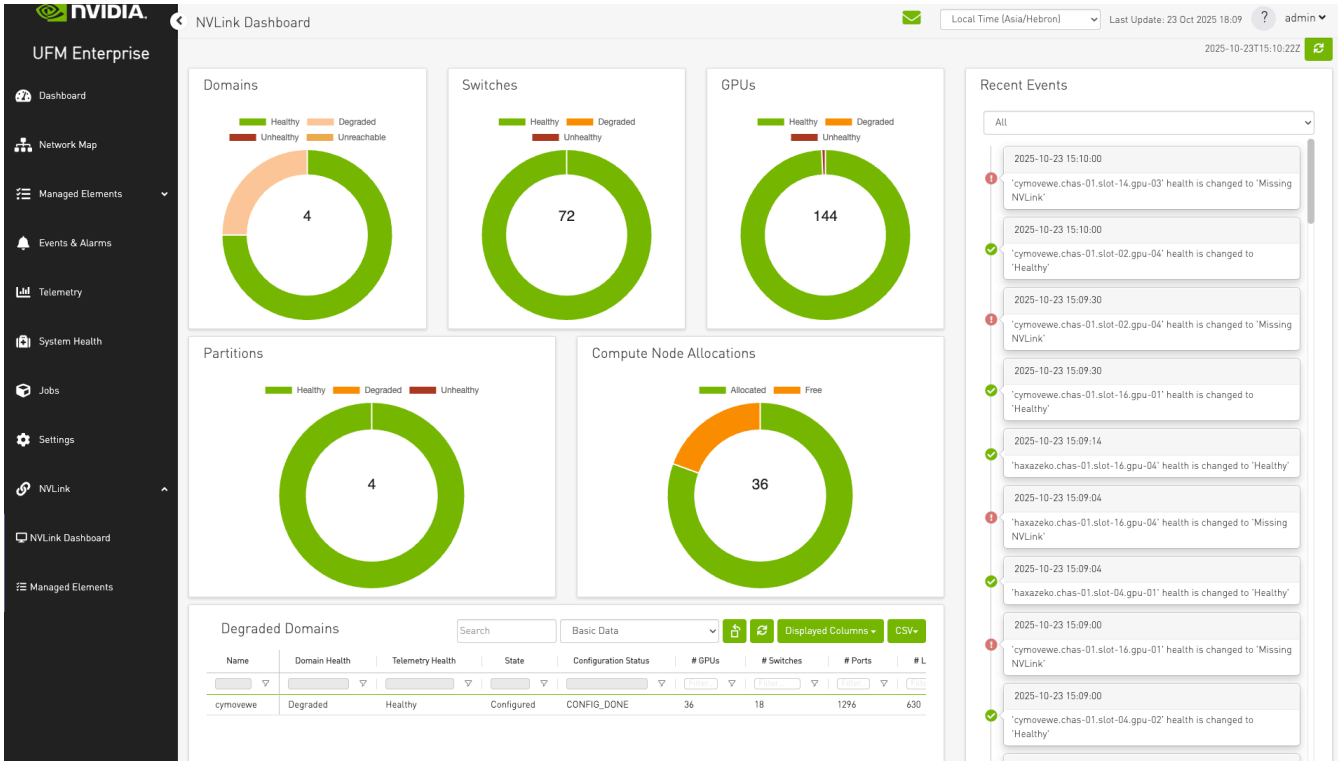
UI Views

After the plugin is activated, an **"NVLink"** section becomes available in the dashboards.

NVLink Dashboard View

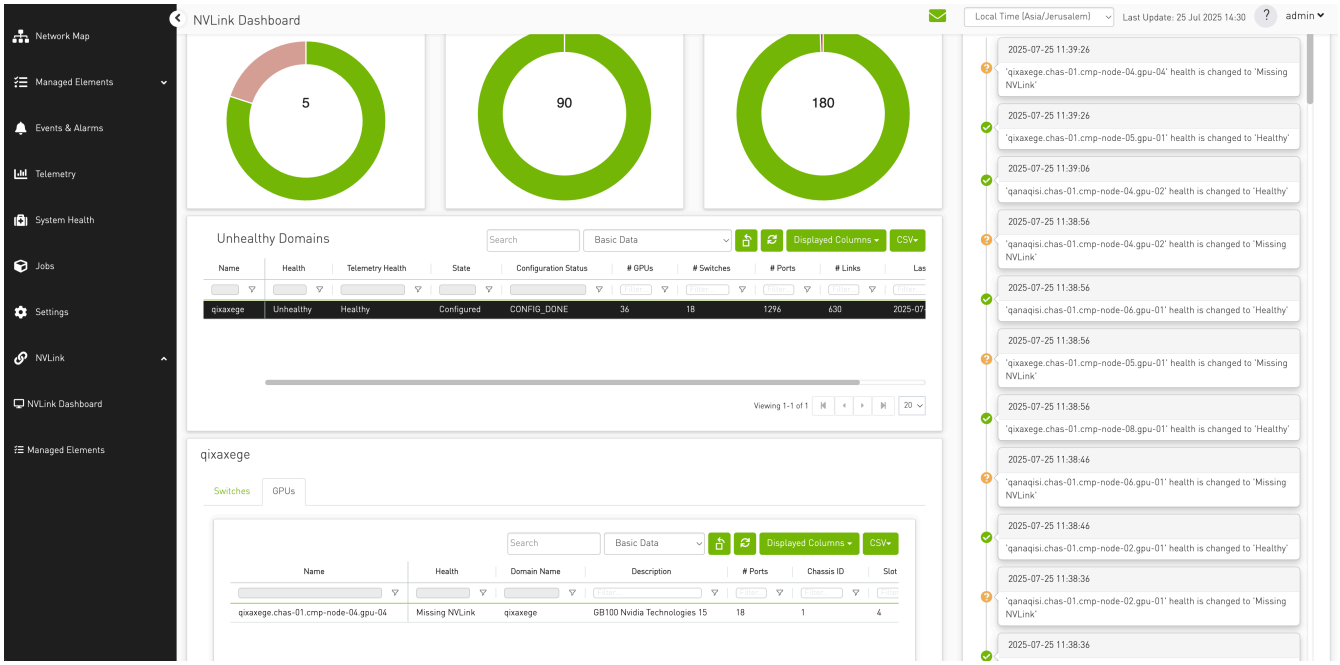
This view presents an overview of inventory elements—such as domains, switches, GPUs, Partitions and Compute Nodes Allocations—along with a filter for their health status.

Users can drill down from overall status indicators to specific elements, and further into the individual ports or links associated with each selected element.

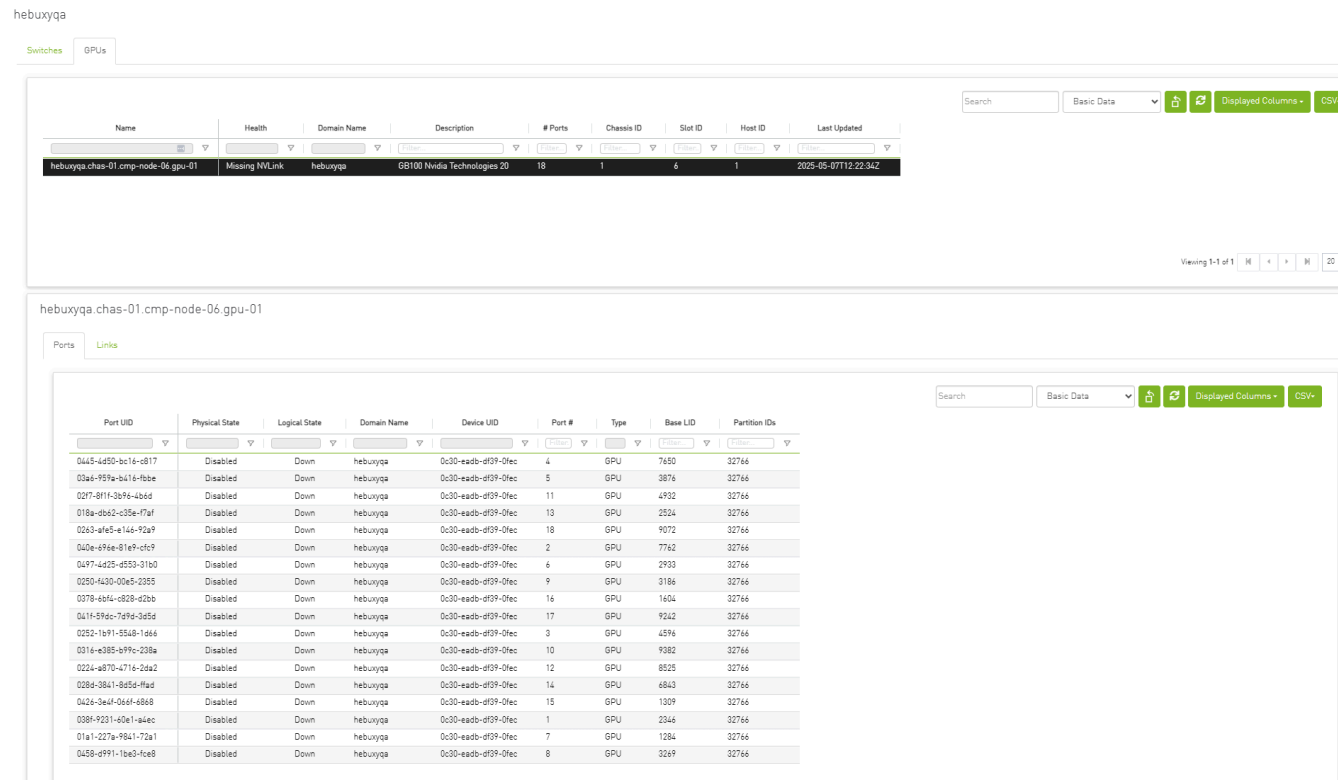


The user can select a specific domain, upon which a list of associated switches and GPUs will be displayed, as illustrated in the example below.

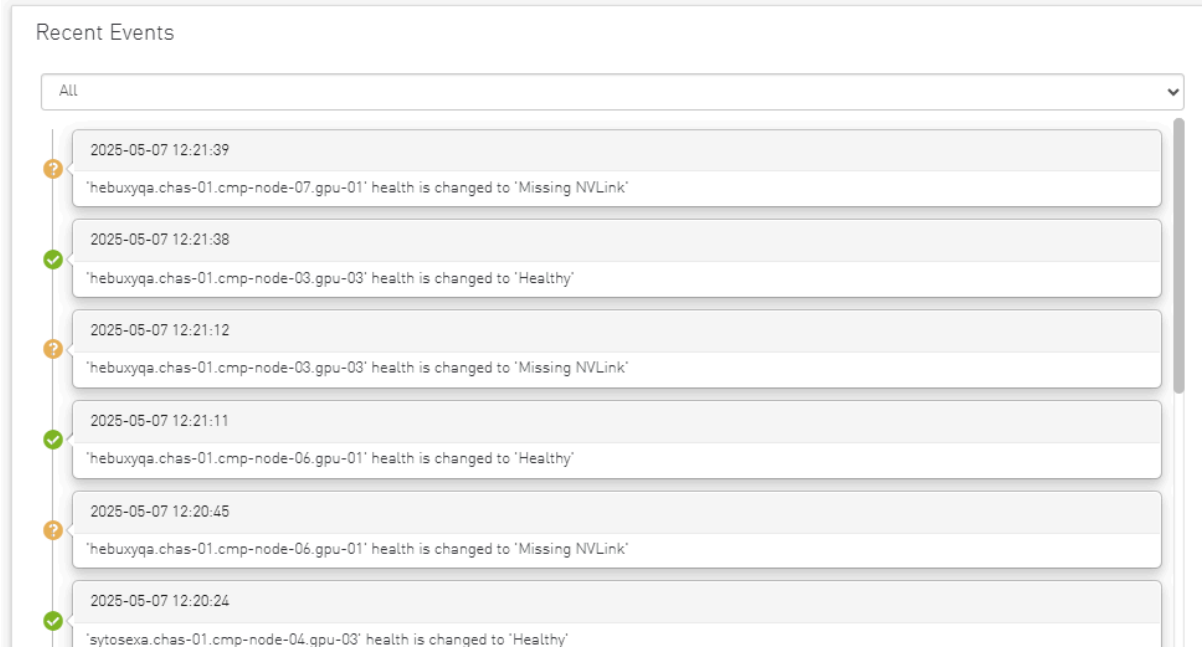
If the selected domain has any health issues, a detailed breakdown of the affected devices will also be presented.



When an unhealthy device is selected, a list of all its ports and links will be displayed.



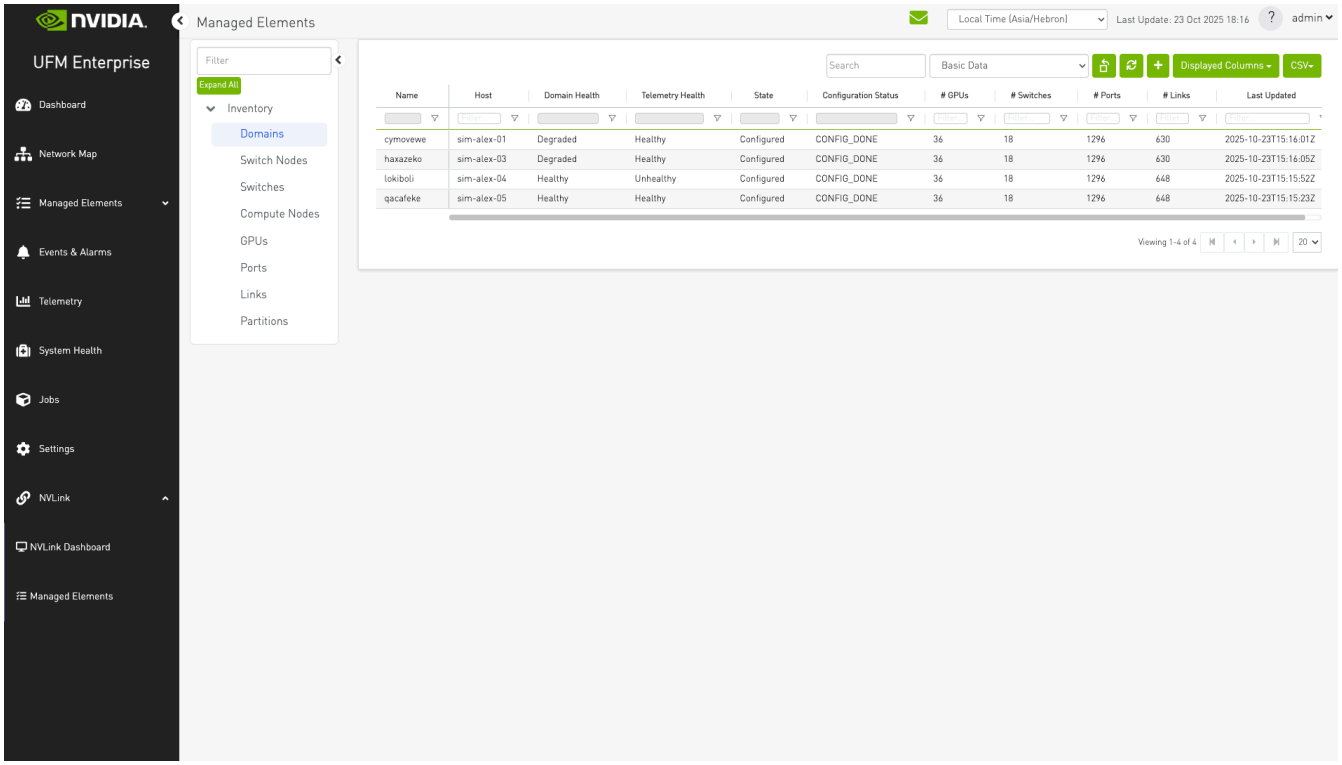
Additionally, the "Recent Events" notification panel on the right side of the screen is updated with the most recent health status changes of the devices.



Managed Elements View

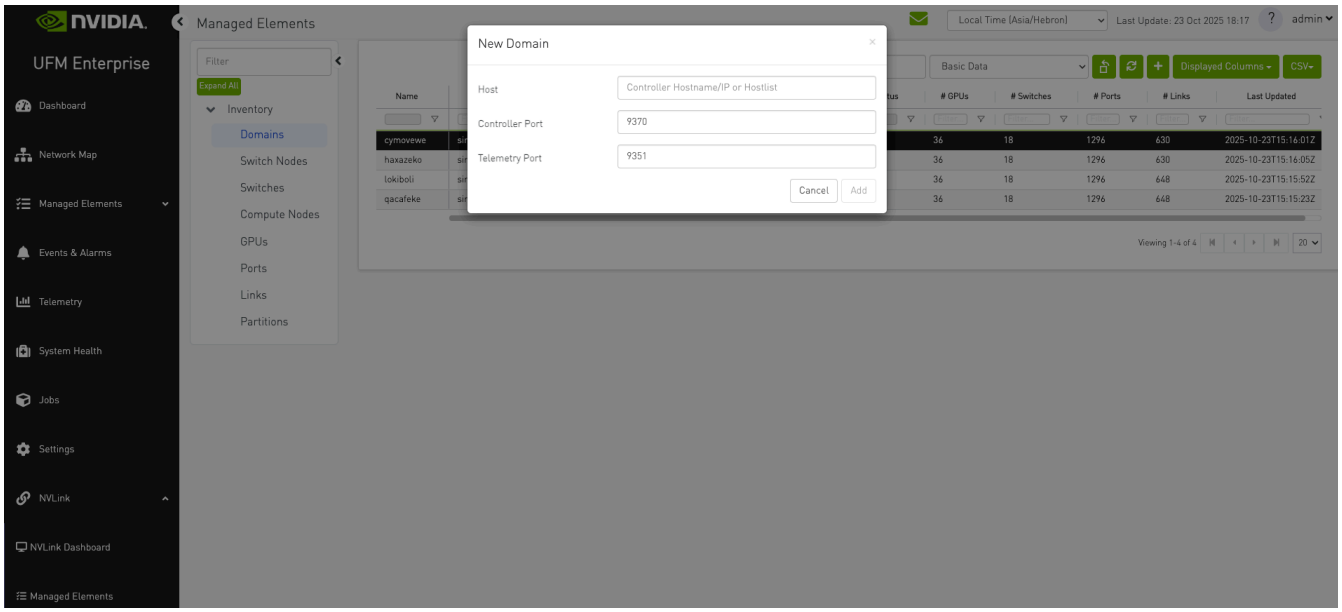
The **Managed Elements** view is a tree-tabular display that shows all inventory elements, allowing users to browse through them. It also provides the option to add or remove domains.

Domains View



Add New Domain Model

Click the **+** icon in the upper dashboard to add a new domain.



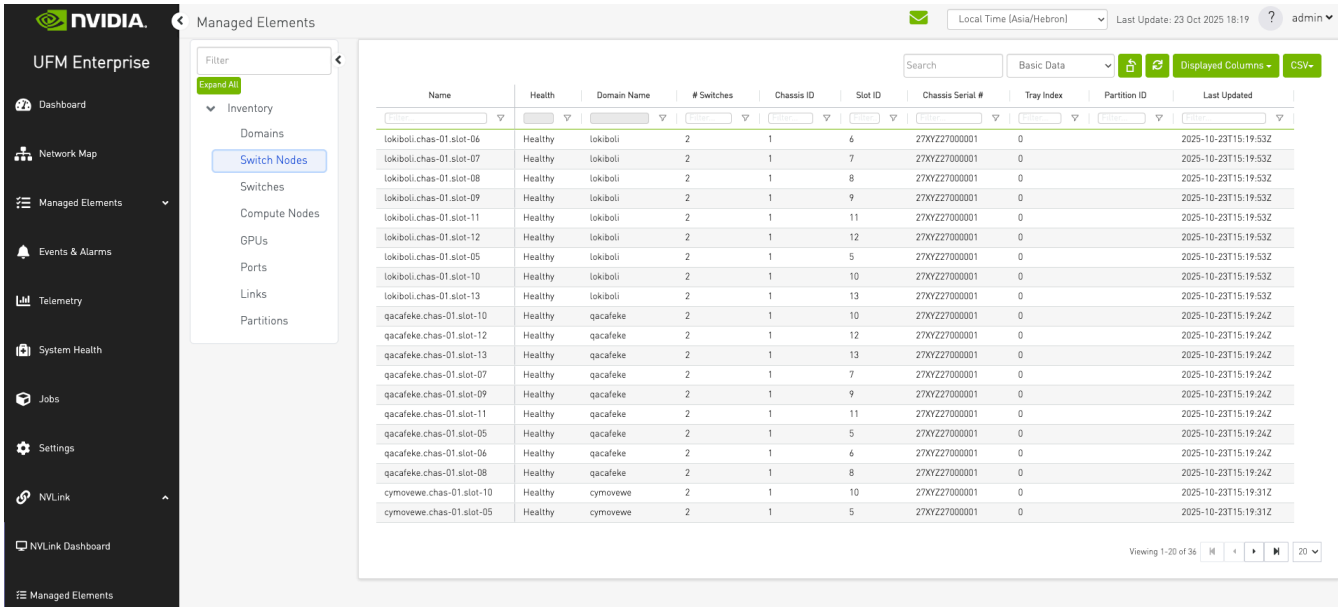
Available Actions for the Selected Domain

The following actions are available when you right-click on the selected domain's row.

Action	Description
Remove	Removes the selected domain and its elements from the inventory.
Go To Switches	Redirects you to the switches of the selected domain.
Go To GPUs	Redirects you to the GPUs of the selected domain.
Go To Ports	Redirects you to all Ports of the selected domain.
Go To Links	Redirects you to all Links of the selected domain.

Switch Nodes View

This screen presents a table listing all the switch nodes, including key details.



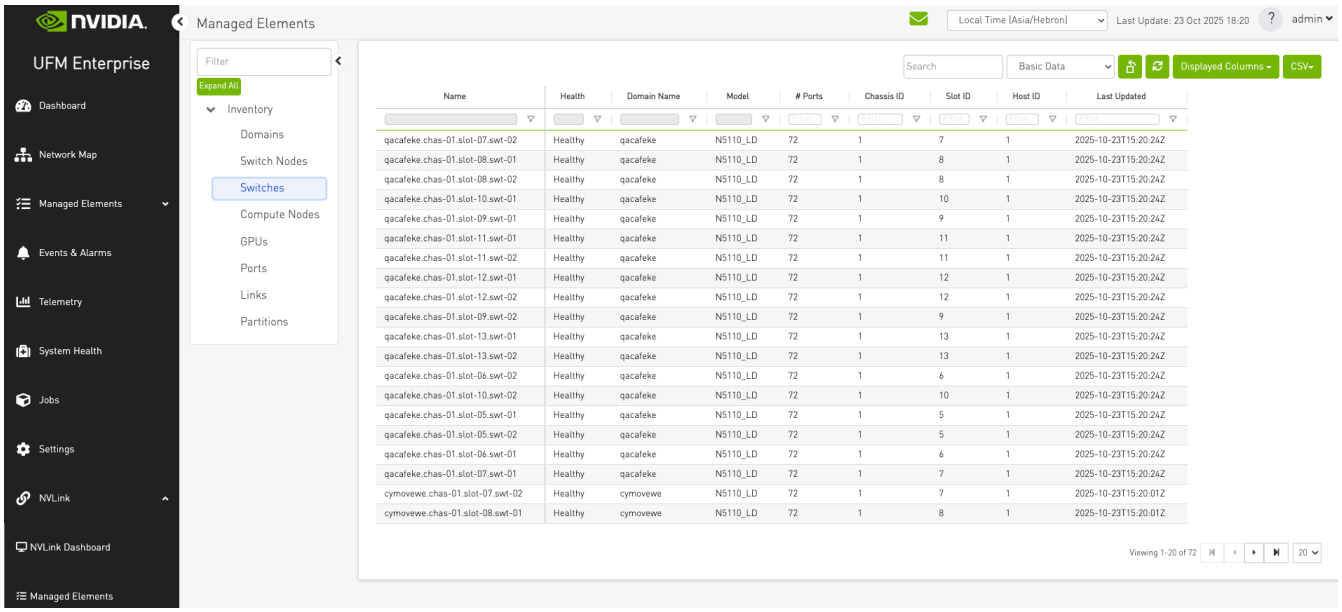
Available Actions for the Selected Switch Node

The following actions are available when you right-click on the selected switch node's row.

Action	Description
Go To Domain	It redirects you to the parent domain of the selected switch node.
Go To Switches	It redirects you to the Switches of the selected node.

Switches View

This screen presents a table listing all the switches, including key details.



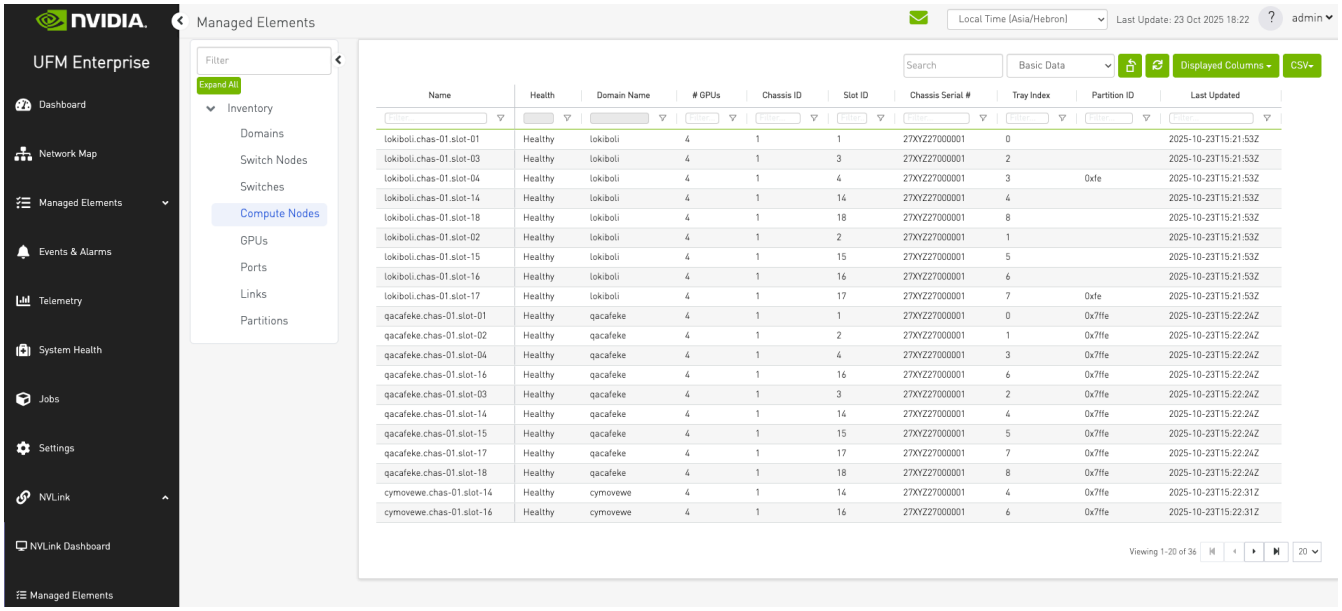
Available Actions for the Selected Switch

The following actions are available when you right-click on the selected switch's row.

Action	Description
Go To Domain	It redirects you to the parent domain of the selected switch.
Go To Ports	It redirects you to the Ports of the selected switch.

Compute Nodes View

This screen presents a table listing all the compute nodes, including key details.



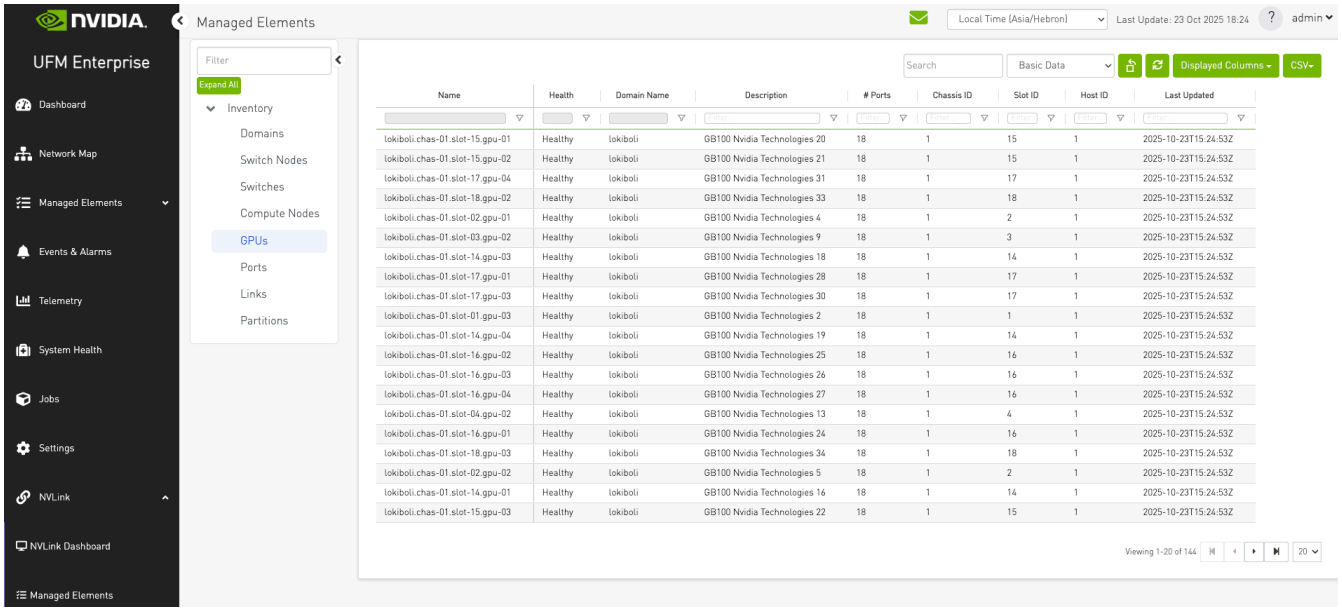
Available Actions for the Selected Compute Node

The following actions are available when you right-click on the selected compute node's row.

Action	Description
Go To Domain	It redirects you to the parent domain of the selected switch node.
Go To GPUs	It redirects you to the GPUs of the selected node.
Go To Partitions	It redirects you to the assigned partition of the selected node.

GPUs View

This screen presents a table listing all the GPUs, including key details.



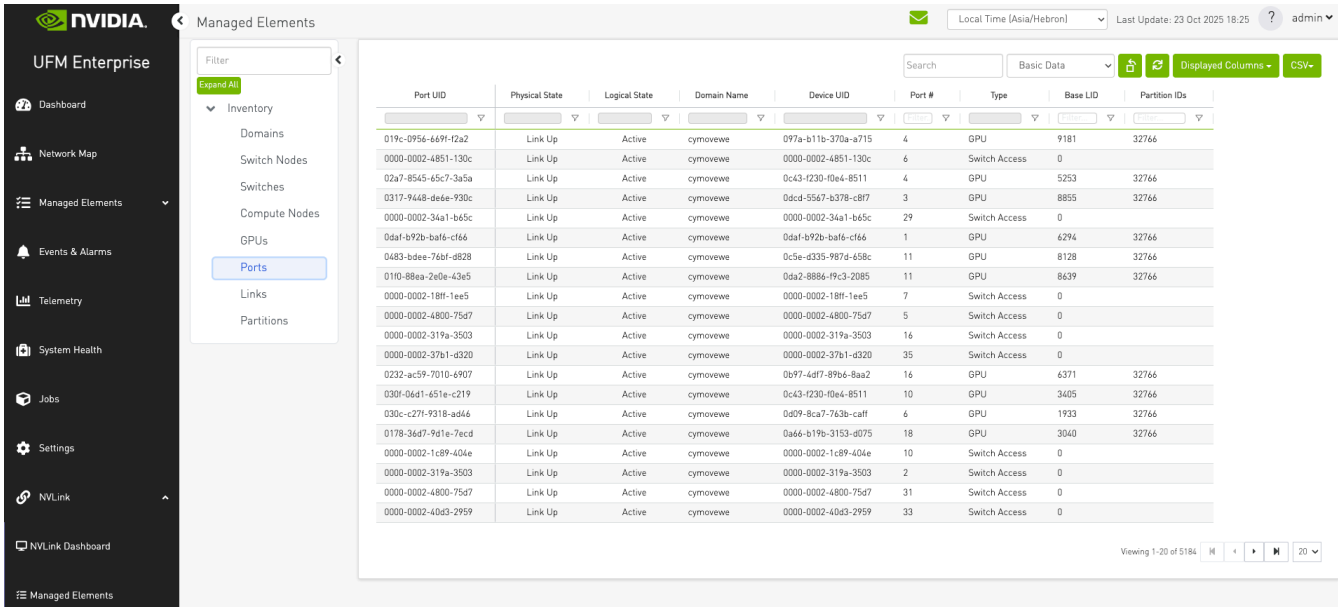
Available Actions for the Selected GPU

The following actions are available when you right-click on the selected GPU's row.

Action	Description
Go To Domain	It redirects you to the parent domain of the selected GPU.
Go To Ports	It redirects you to the Ports of the selected GPU.

Ports View

This screen presents a table listing all the ports, including key details.



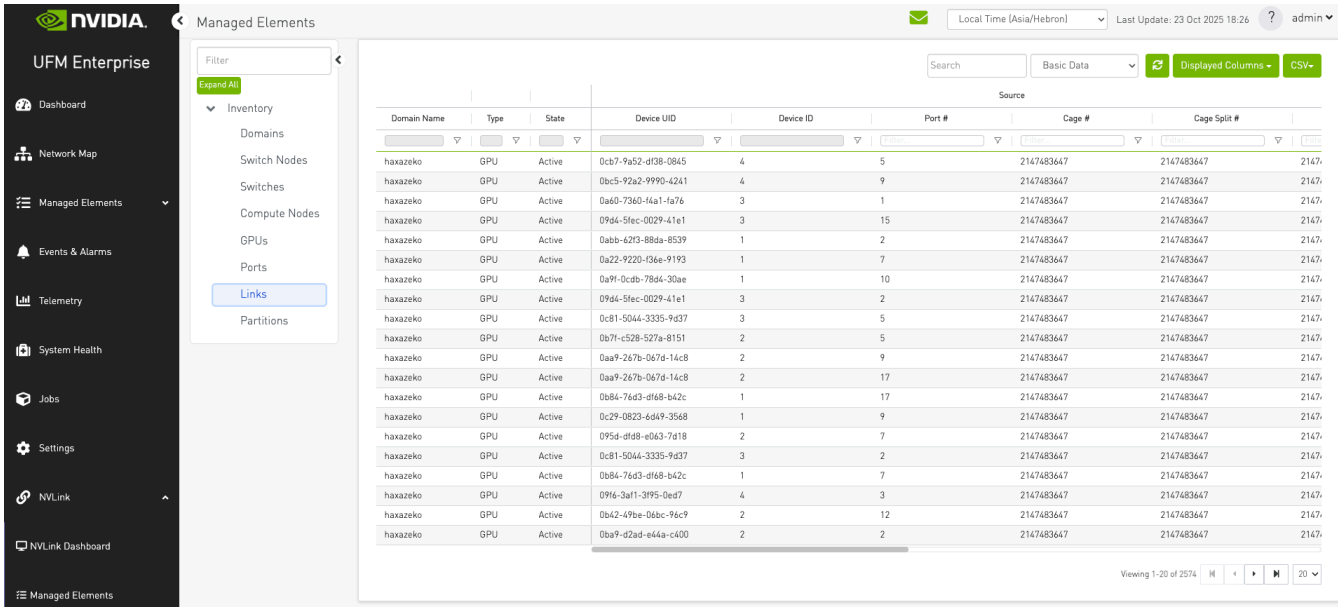
Available Actions for the Selected Port

The following actions are available when you right-click on the selected port's row.

Action	Description
Go To Domain	It redirects you to the parent domain of the selected Port.

Links View

This screen presents a table listing all the links, including key details.



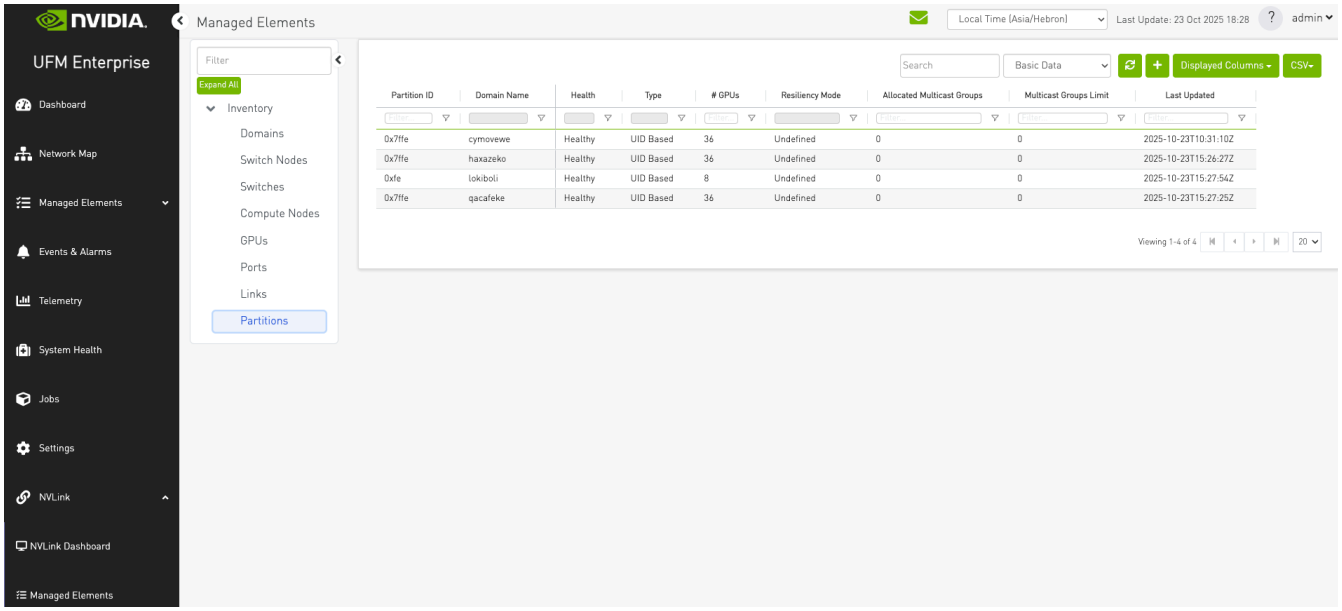
Available Actions for the Selected Link

The following actions are available when you right-click on the selected link's row.

Action	Description
Go To Domain	It redirects you to the parent domain of the selected Link.

Partitions View

This screen presents a table listing and managing all the available partitions, including key details.

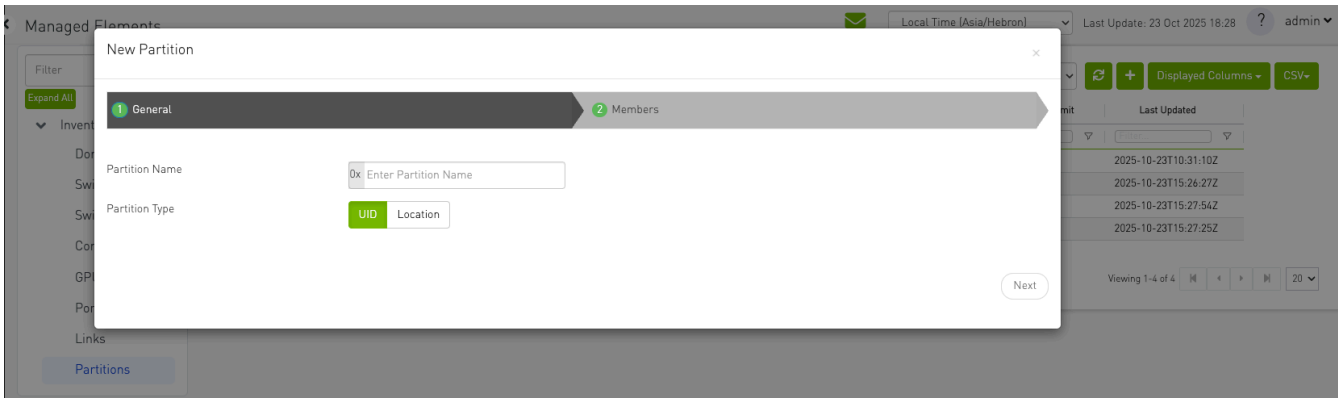


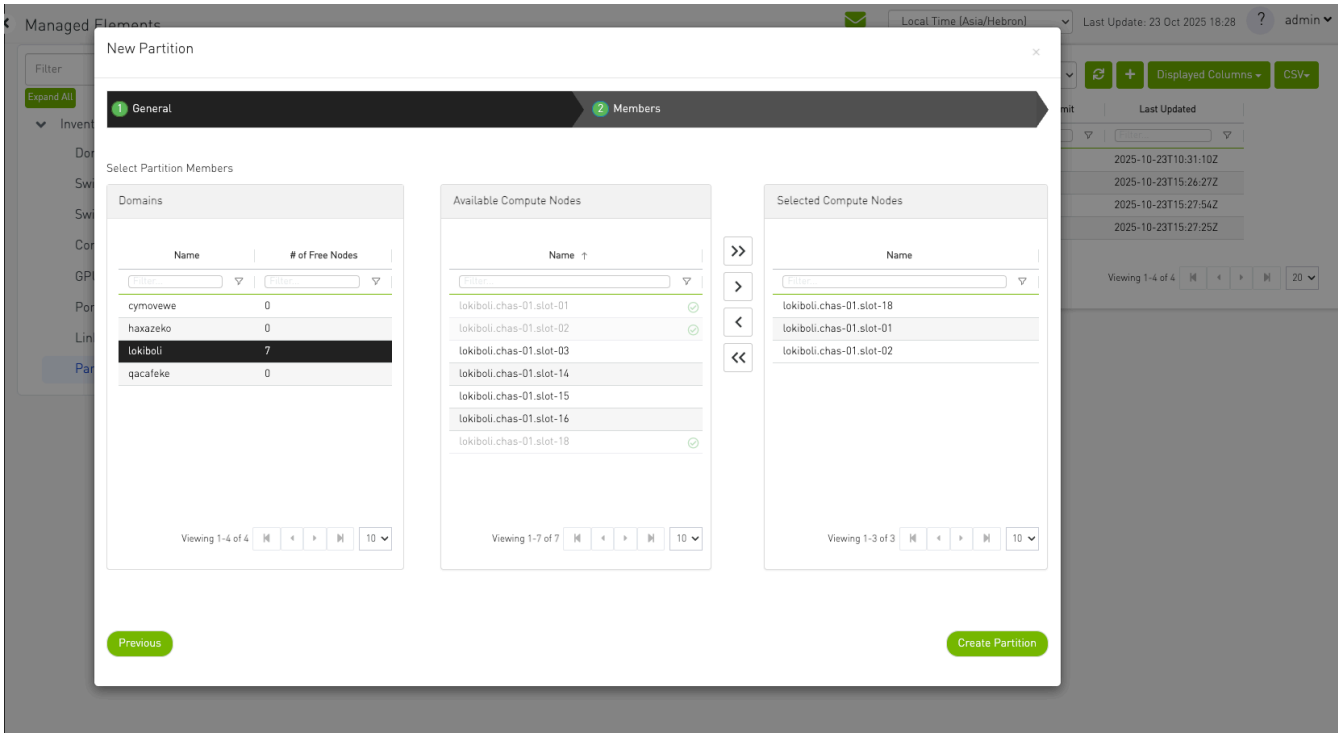
Add New Partition

Click the **+** icon in the upper dashboard to add a new partition and assign compute nodes to that partition.

To Create a new partition, a wizard with two steps will be shown to fill below fields:

1. Specify the partition ID in hex.
2. Specify the partition type (UID Or Location) Based.
3. Specify to which domain the partition should be assigned.
4. Select the compute nodes members of the new partition
5. You can not assign the same compute nodes for multiple partitions.





Available Actions for the Selected Domain

The following actions are available when you right-click on the selected domain's row.

Action	Description
Remove	Removes the selected partition.
Edit	Edit the compute nodes members of the selected partition
Go To Domain	Redirects you to the assigned domain of the selected partition.
Go To Compute Nodes	Redirects you to all Compute Nodes Members of the selected partition.

REST API

The REST API documentation is available separately (see NVLink REST API).

In addition, the API specification in the OpenAPI format can be accessed at `/ufmRestV2/plugin/nmxaggr/v1/app/swagger` endpoint of the running plugin.

Info

When operating with a standalone NMXAGGR instance, the `/ufmRestV2/plugin/nmxaggr` prefix is not required. In contrast, when operating with the NVLink plugin, the prefix must be used.

Therefore, depending on your deployment scenario—plugin mode or standalone—you should adjust the API endpoint URLs accordingly to ensure proper communication.

Unmanaged Switch Dump (USD) Plugin

Plugin Release Notes

Changes and New Features

Plugin Version	Feature
1.1.0-2	Start a background task which collects the abnormal System Dump of all/specified unmanaged switches automatically

Bug Fixes

Plugin Version	Bug Fix
1.1.0-2	N/A

Overview

The NVIDIA Unmanaged Switch Dump (USD) plugin operates as a server that uses the "`resourcedump`" command to retrieve system collections from unmanaged NVIDIA switches, compiles the data into files, and transfers these files to the same remote destination used for data collection from managed switches.

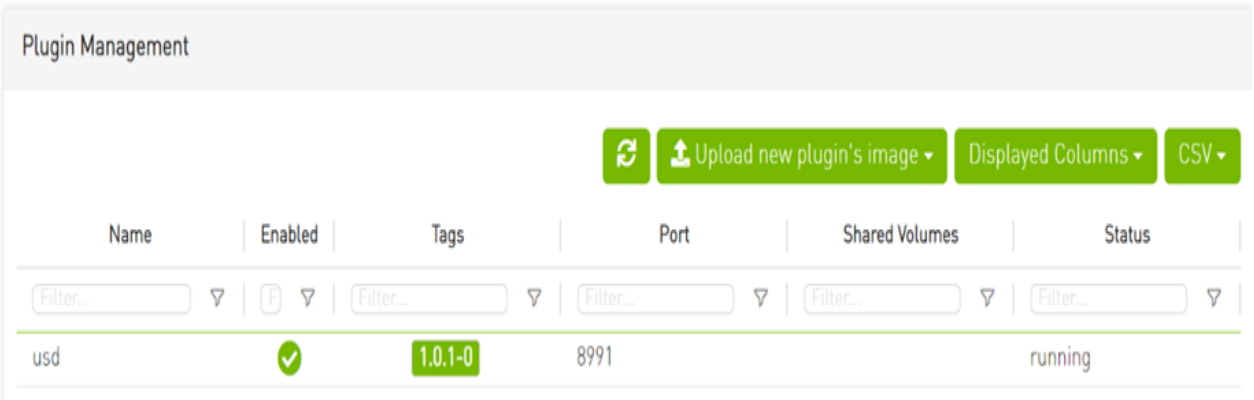
Prerequisites

- The unmanaged switch must be running firmware version 31.2014.2100 (production) or later.
- UFM version must be LTS 6.19.1.1 or higher.

Deployment

Deploying the Plugin with UFM (SA or HA)

1. Install the latest version of UFM.
2. Start UFM using the command: `/etc/init.d/ufmd start`.
3. Pull the plugin image from the specified repository.
4. Enable the plugin by running `/opt/ufm/scripts/manage_ufm_plugins.py` with the options `-p usd -t <version>`, or alternatively, enable it through the UFM UI by navigating to **Settings → Plugin Management**, right-clicking on **usd**, selecting **Add**, choosing the desired version, and clicking **Add**.
5. Verify the plugin is running by using `docker ps` or by checking the UI under **Settings → Plugin Management** to confirm it has been added and enabled.



6. If the default port for **usd** is in use, update the `usd_httpd_proxy.conf` configuration file and restart the plugin.

Deploying the Plugin on UFM-SDN Appliance

1. **Log in** as the admin user and enter configuration mode:

```
enable config terminal
```

2. **Verify that UFM is running** by executing:

```
show ufm status
```

   If UFM is not running, start it with:

```
ufm start
```

3. **Ensure Docker is active** by disabling any shutdown state:

```
no docker shutdown
```

4. **Load the latest USD plugin Docker image**:
   - If using an HA setup, repeat this step on the standby node.
   - If the appliance is connected to the internet:

```
docker pull mellanox/ufm-plugin-usd
```

- If the appliance is **offline**, use another internet-connected machine to save the image:

  ```
  docker save mellanox/ufm-plugin-usd:latest | gzip > ufm-plugin-usd.tar.gz
  ```

  Transfer the file to an SCP-accessible location, then on the appliance:

  ```
  image fetch scp://user@hostname/path-to-file/ufm-plugin-usd.tar.gz docker load ufm-plugin-usd.tar.gz
  ```

5. **Enable and start the plugin**:

   ```
   ufm plugin usd add
   ```

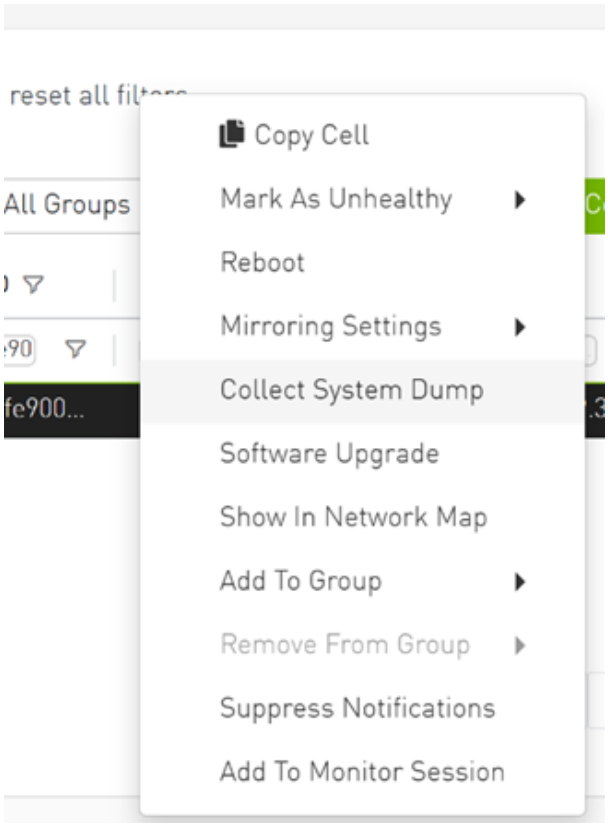
6. **Verify the plugin is running**:

   ```
   show ufm plugin
   ```

Collecting a System Dump via the UI

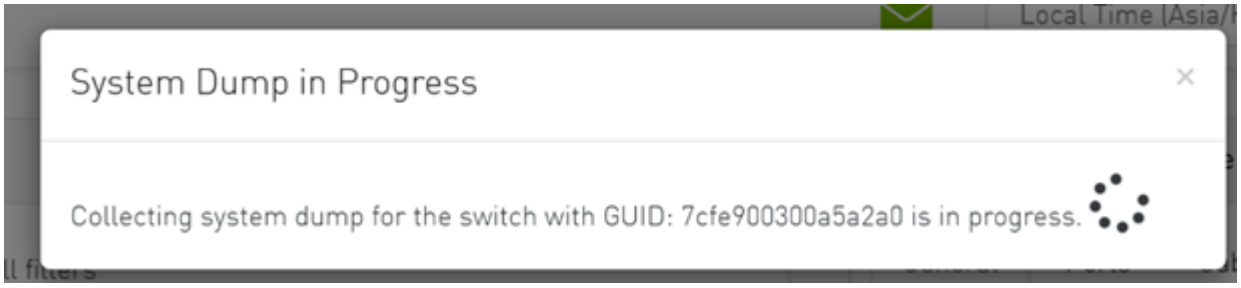
To collect a system dump from an unmanaged switch using the UI:

1. Right-click on the target unmanaged switch.



2. Select **Collect System Dump** from the context menu.

While the system dump is being generated, a progress message will be displayed.



Once the process completes successfully, a confirmation message will appear, including the dump file path in the format:

`<protocol>://<server>/<path>`

Collecting System Dump Automatically

Besides of manually starting the collection, the plugin will regularly check for unmanaged switches and collect their system dumps in the background. This helps ensure important

diagnostic data is gathered automatically, improving user experience and reducing manual work. The collection interval and supported switch models can be configured in the **poller-config** section of plugin settings.

Configurations

Logs Configurations

Parameter	Description	Default Value
`logs_file_name`	Full path of the log file .	/log/usd.log
`logs_level`	Sets the logging level for the plugin (e.g., INFO, DEBUG).	INFO
`log_file_max_size`	Maximum size of the log file before it rotates, measured in bytes	10485760
`log_file_backup_count`	Number of backup log files to retain after rotation.	5

Poller Configurations

Parameter	Description	Default Value
`polling_first_delay_secs`	Delay before the first polling cycle starts (in seconds). This allows the USD plugin and its web service to initialize properly.	5
`polling_interval_secs`	Interval between consecutive polling cycles (in seconds). A value of zero or negative disables the polling task.	300
`polling_duration_secs`	Total duration for which polling runs (in seconds). Zero means polling continues indefinitely until the plugin stops.	0
`polling_matched_models`	Comma-separated list of unmanaged switch models to poll. Leave blank to poll all unmanaged switch models.	MQM9790

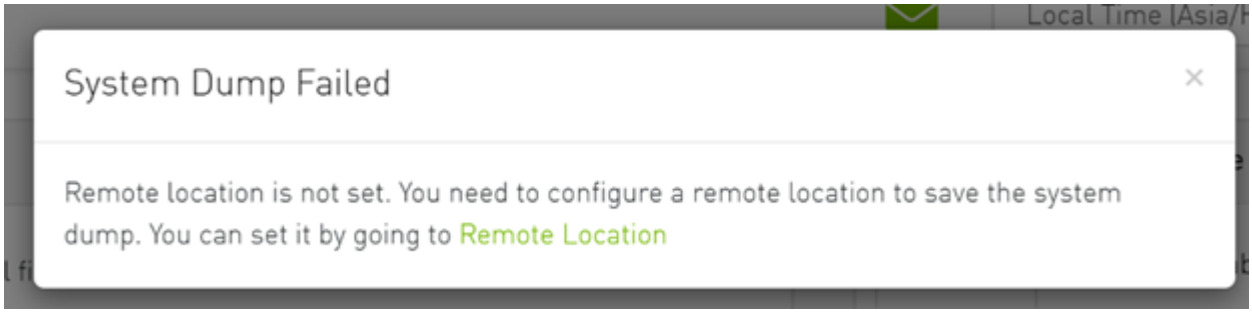
Dump Command Configurations

Parameter	Description	Default Value
`dump_cmd_timeout_secs`	The timeout duration for the MFT (Mellanox Firmware Tools) command, measured in seconds. The maximum value is 900 seconds. If the value is left blank, set to 0, or assigned a negative number, the timeout will automatically default to 300 seconds.	300

Troubleshooting

If the operation fails, an error message will be shown.

In cases where the failure is related to the remote storage location, a specific error message indicating this issue will be displayed.



UFM Infra

Overview

The **UFM Infra** feature introduces a structured architecture where services are divided into two categories, each deployed differently based on functionality:

- UFM Infra: A set of persistent infrastructure services that run on all nodes. These services support system-level operations and ensure distributed availability.
- UFM Enterprise: Services that run exclusively on the master node, responsible for management, orchestration, and user-facing functionality.

Key Benefits

- Faster API Availability after Failover : By limiting service transitions during node failures, recovery times are significantly reduced.
- Improved Modularity: Separating core infrastructure from enterprise logic simplifies maintenance and troubleshooting.
- Enhanced Scalability: Services can be scaled and managed independently across nodes.

Users can enable or disable the UFM Infra feature without requiring a reinstallation of the UFM system. For more information, refer to Enabling or Disabling UFM Infra.

Installation instructions are available at Installing UFM Infra Using Rootless with Podman.

As part of the updated architecture, a FAST-API plugin is deployed and a Redis server is required for inter-service communication. Redis can be configured in two ways:

- As an internal service (installed with UFM)
- As an external Redis instance, depending on deployment needs.

  For more information, refer to Redis-Related Configuration.

Communication Flow: Fast API, Redis, SM/SHARP Components

The following sequence describes how communication is handled between Fast API, Redis, and SM/SHARP components:

1. **Request Submission via Fast API**

   Users send REST API requests (e.g., for PKey creation or SHARP reservation actions) to the Fast API. These requests are placed into Redis queues, and a Transaction ID (TID) is returned to the user for tracking purposes.

2. **Processing by Communicators**

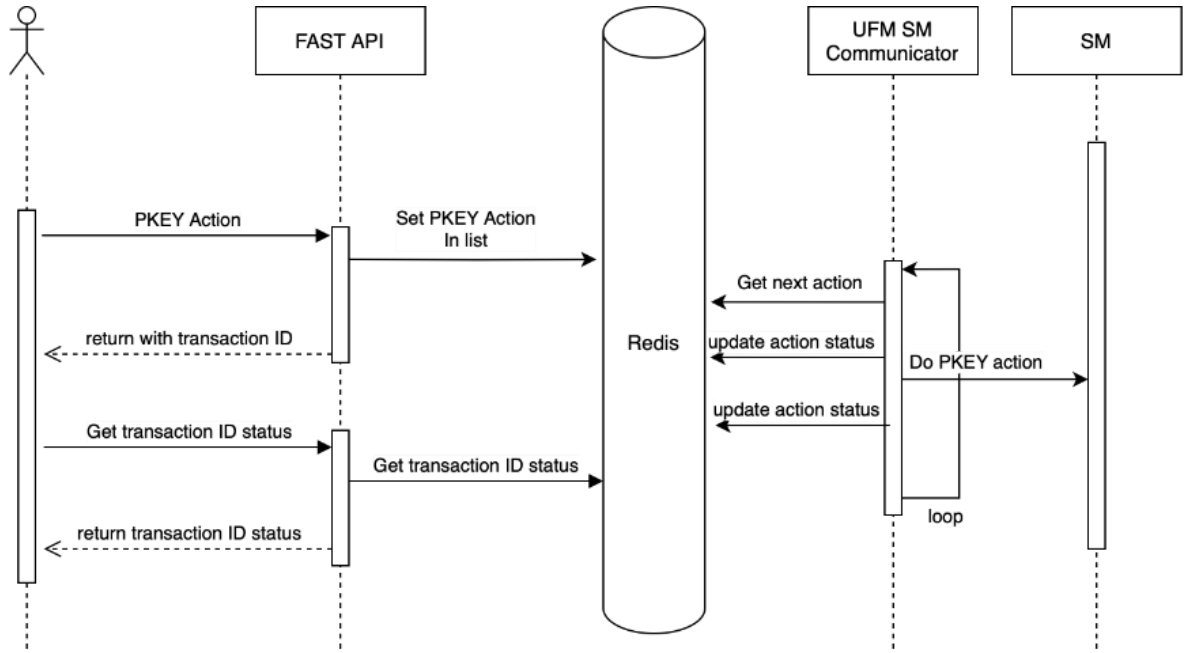
   - The SM Communicator or SHARP Communicator monitors Redis queues for new requests.
   - Upon receiving a request, the communicator forwards it to the relevant component (SM or SHARP) for execution.
   - After processing, the communicator captures the response and status.

3. **Status Updates**

   The communicators update the status of each request back into Redis. Users can query the status of their transaction using the TID provided during request submission.

4. **Configuration Storage and Retrieval**

   - Communicators store the configuration in Redis.
   - This allows the Fast API to retrieve and expose configuration data via REST APIs, giving users access to the configuration via REST APIs to understand cluster-level settings.



Configuring UFM Infra

Redis-Related Configuration

Redis configuration parameters can be modified within the `UFMInfra` section of the `gv.cfg` file. This allows for customization of Redis behavior to better suit UFM infrastructure requirements.

```
[UFMInfra]
...
# What is the host where the Redis server is running
redis_host = localhost

# What is the Redis port
redis_port = 6379

# Redis timeout in seconds
redis_socket_timeout = 5

# Flag that shows if we use external Redis database
is_external_redis = False

# Flag that shows if we use TLS connection to Redis database
is_tls_redis = False
```

Fast-API configuration

The following parameters can be modified within the Fast API configuration file:

Section	Default Value	Description
`smCommunicator`	`600`	Default Time-to-live (TTL) for SM-related transactions before expiration (in seconds)
`sharpCommunicator`	`600`	Default Time-to-live (TTL) for SHARP-related transactions before expiration (in seconds)

Enabling or Disabling UFM Infra

Prerequisites

Before enabling or disabling the UFM Infra feature, ensure the following conditions are met:

- The UFM Docker image has been installed using the `deploy_rootless_ufm` script. Refer to Installing UFM Infra Using Rootless with Podman.
- UFM High Availability (HA) is deployed using the **Enterprise Multinode** setup.
- The control script for managing the feature is available on the host at: `/opt/ufm/files/scripts/ufm_infra_feature_flag.py`
- Example:

```
ufm_infra_feature_flag.py -h
usage: ufm_infra_feature_flag.py [-h] (-e | -d) [--log-level
{DEBUG,INFO,WARNING,ERROR,CRITICAL}] [--timeout-seconds
TIMEOUT_SECONDS] [--ufm-user UFM_USER]

Control UFM Infra feature flags

This script turns on/off the UFM Infra (multi node) feature.
It
manages the UFM Infrastructure feature by controlling both
the configuration
and HA cluster resources. The script follows these flows:

Prerequisites check:
  1. Verifies Python version is 3.6 or higher
  2. Verifies script is run with root privileges
  3. Verifies ufm_user user exists (default is ufmadm but can be
overridden with --ufm-user)
  4. Validates HA configuration and UFM Infra installation

Enable flow:
  1. Stops the HA cluster and waits for all UFM containers to
stop
  2. Updates the UFM configuration to enable the Infra
feature
  3. Updates the Redis trigger file to enable topology
publishing
  4. Enables the HA resources
  5. Starts the HA cluster (only if previous steps succeeded)

Disable flow:
  1. Stops the HA cluster and waits for all UFM containers to
stop
  2. Updates the UFM configuration to disable the Infra
feature
```

```
  3. Updates the Redis trigger file to disable topology
publishing
  4. Disables the HA resources
  5. Starts the HA cluster (only if previous steps succeeded)

Note: This script requires root privileges to modify the UFM
configuration.
If any step fails, the script will exit without starting the
HA cluster.
In case of failure, manual intervention will be required to
restore the system
to a working state. The HA cluster may need to be started
manually using
'ufm_ha_cluster start' command.

optional arguments:
  -h, --help            show this help message and exit
  -e, --enable          Enable the Infra feature (mutually
exclusive with -d)
  -d, --disable         Disable the Infra feature (mutually
exclusive with -e)
  --log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL}
                        Set the logging level (default: INFO)
  --timeout-seconds TIMEOUT_SECONDS
                        Timeout for waiting for containers to
go down (default: 120 seconds)
  --ufm-user UFM_USER   The user to run the command as
(default: ufmadm)
```

Important Notes

When deploying a plugin with `ufm_infra` is installed, users can choose one of the following methods:

- **Via the UI**: Use the UFM user interface to deploy the plugin. For instructions, refer to Plugin Management.
- **Via REST API**: Deploy the plugin through UFM's REST API. For more information, refer to NVIDIA UFM Enterprise REST API Guide.
- **Using the Plugin Management Script**: Run the `manage_ufm_plugins` script **inside the UFM container** (not the `ufm_infra` container). For more information, refer to UFM Plugins Management.

Installing UFM Infra Using Docker Root

UFM Docker Root Installation Steps with Infra Mode and NFS/Shared Storage

This section provides installation instructions for UFM Enterprise running in Docker root mode with UFM infra support and NFS or shared storage. The configuration uses versioned directories and soft links to enable seamless upgrades.

Prerequisites

- Docker installed and running
- NFS or shared storage must be mounted at `/opt/ufm/shared_files`
- UFM and FAST-API tar files available
- Root or sudo privileges
- rsync
- Redis - if using an external Redis

Architecture Overview

- **NFS Mount Point**: `/opt/ufm/shared_files` - Main shared storage location
- **Version Directory**: `/opt/ufm/shared_files/ufm-<version>` - Version-specific directory

- **Soft Link**: `/opt/ufm/files` → `/opt/ufm/shared_files/ufm-<version>` - Points to current version
- **UFM Mode**: Docker root + UFM infra (creates two services: ufm-enterprise.service and ufm-infra.service)

Installation Steps

Step 1: Create UFM Base Directory

```
mkdir -p /opt/ufm
mkdir -p /opt/ufm/ufm_plugins_data
```

Step 2: Load UFM Image and Extract Version

```
# Define the UFM image file path
UFM_IMAGE_FILE="<path to ufm*-docker.img.gz>"

# Extract UFM version from filename (e.g., ufm_6.22.0-7.ubuntu24.x86_64-docker.img.gz -> 6_22_0_7)
UFM_VERSION=$(basename "$UFM_IMAGE_FILE" | sed 's/ufm_\([0-9][^.]*\.[^.]*\.[^.]*-[^.]*\)\.ubuntu.*/\1/' | tr '.-' '_')
echo "UFM Version: $UFM_VERSION"

# Load the UFM image
docker load -i "$UFM_IMAGE_FILE"
```

Step 3: Create Version-Specific Directory and Soft Link

```
# Create version-specific directory in shared storage
mkdir -p /opt/ufm/shared_files/ufm-${UFM_VERSION}

# Remove existing files link if it exists
rm -f /opt/ufm/files

# Create soft link to version-specific directory
ln -s /opt/ufm/shared_files/ufm-${UFM_VERSION} /opt/ufm/files

# Verify the soft link
ls -la /opt/ufm/files
```

Step 4: Run UFM Installer

```
docker run --name=ufm_installer --rm \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v /etc/systemd/system/:/etc/systemd_files/ \
    -v /opt/ufm/files/:/installation/ufm_files/ \
    -v [LICENSE_DIRECTORY]:/installation/ufm_licenses/ \
    mellanox/ufm-enterprise:latest \
    --install \
    --fabric-interface ib0 \
    --ufm-infra
```

Note

- Replace `ib0` with your actual InfiniBand interface name, if it is not the default ib0.
- All other UFM install flags are supported and can be added to the command.

Step 5: Load Redis Image

Load the given Redis image (in case you are not using external Redis) docker load -i ""

Step 6: Load Fast API Plugin Image

```
# Define the Fast API plugin image file path
FAST_API_IMAGE="<PATH TO FAST API.img.gz file>"

# Load the Fast API plugin image
docker load -i "$FAST_API_IMAGE"

# Extract Fast API version from Docker image tag (exclude latest tag)
FAST_API_VERSION=$(docker images --format "{{.Repository}}:{{.Tag}}" | grep "mellanox/ufm-plugin-fast_api" | grep -v ":latest" | head -1 | cut -d':' -f2)
echo "Fast API Version: $FAST_API_VERSION"
```

Step 7: Install Fast API Plugin

```
docker run --hostname $HOSTNAME --rm --name=ufm_plugin_mgmt --
entrypoint="" \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v /opt/ufm/files:/opt/ufm/shared_config_files \
    -v /dev/log:/dev/log \
    -v /sys/fs/cgroup:/sys/fs/cgroup:ro \
    -v /lib/modules:/lib/modules:ro \
    -v /opt/ufm/ufm_plugins_data:/opt/ufm/ufm_plugins_data \
    -e UFM_CONTEXT=ufm-infra \
    mellanox/ufm-enterprise:latest \
    /opt/ufm/scripts/manage_ufm_plugins.sh add -p fast_api -t
${FAST_API_VERSION} -c ufm-infra
```

Step 8: Reload systemd and Start Services (if not running HA)

```
# Reload systemd daemon
systemctl daemon-reload

# Start UFM infra service (starts first)
systemctl start ufm-infra
systemctl start ufm-enterprise
```

Installing UFM Infra Using Rootless with Podman

UFM Infra Rootless Deployment with Podman (Restricted to Oracle Linux Only)

Prerequisites

1. Download the UFM and plugins bundle tar file to `/tmp`.
2. Extract the contents using the command:

```
tar -xvf <bundle tar>
```

This archive (tar file) includes the following components:

- Relevant UFM container image
- Relevant FAST-API container image
- Relevant Infra container image (for internal Redis usage). Refer to Redis-Related Configuration for more information.
- Default plugin bundle for UFM
- UFM-HA package

HA Installation Requirements

To enable the UFM Infra feature, UFM HA must be installed in a new mode (`external-storage`), using a new product (`enterprise-multinode`).

Additionally, NFS must be configured as follows:

NFS Setup Prerequisites

- Select a dedicated NFS server to host the shared directories.
- Create a shared directory on the NFS server for UFM configuration and logs.

- Install the NFS client on each UFM node if not already present.

Enable HA ports in Firewall

If you have firewall rules that blocks non-standard ports, we need to open these ports so high availability services could communicate with each other on the HA nodes. To do so, run these commands:

```
firewall-cmd --permanent --add-service=high-availability
# or
firewall-cmd --add-service=high-availability
# and then reload the rules
firewall-cmd --reload
```

Create and Mount the UFM Directory

> **Note**
>
> At this stage, apply point #2 (Mount the UFM directory) only on the master machine.
>
> Other nodes will be visited for mount later.

1. Create the UFM directory:

   ```
   mkdir -p /opt/ufm/files/
   ```

2. Mount the UFM directory:

- If using NFS 4.2:

```
mount -t nfs4 -o context="system_u:object_r:container_file_t:s0"
<server>:/shared_folder /opt/ufm/shared_files
```

- If using NFS 3:

```
mount -t nfs -o vers=3,context="system_u:object_r:container_file_t:s0"
<server>:/shared_folder /opt/ufm/shared_files
```

3. Ensure the NFS version and mount options are compatible with the NFS server.
4. Verify that the following HA packages are installed: `pcs`, `pacemaker`, and `corosync`. Install them if they are missing.
5. Follow the HA installation steps in Run the HA Installation.

Run the HA Installation

Follow the HA installation instructions at UFM High-Availability Installation and Configuration.

When running the HA installation script, use the following command:

```
./install.sh -p enterprise-multinode -l /opt/ufm/shared_files
```

- The `-l` flag must always point to the shared directory path: `/opt/ufm/shared_files`
- No need to provide the DRBD disk argument to the installation script.

Installation Instructions

1. Check firewall status:

```
systemctl status firewalld
```

2. Configure Firewall (if active):

```
# check if firewalld is running
   systemctl status firewalld
# Permanently add port 8443 to firewalld
   firewall-cmd --permanent --add-port=8443/tcp
# reload firewalld config
   firewall-cmd --reload
```

3. Create UFM directory:

```
mkdir -p /opt/ufm
```

4. Create UFM group:

```
groupadd ufmadm -g 733
```

5. Create a UFM user:

```
useradd -d /opt/ufm -m -u 733 -g ufmadm ufmadm
```

6. Set directory ownership:

```
chown -R ufmadm:ufmadm /opt/ufm
chown -R ufmadm:ufmadm /opt/ufm/shared_files
```

7. Configure SubUID and SubGID:

```
echo "ufmadm:100000:65536" >> /etc/subuid
echo "ufmadm:100000:65536" >> /etc/subgid
```

8. Enable Login Linger for UFM ser:

```
loginctl enable-linger ufmadm
```

9. Configure Rootless Podam storage

```
sudo -u ufmadm mkdir -p /opt/ufm/.config/containers
cat <<EOF | sudo -u ufmadm tee
/opt/ufm/.config/containers/storage.conf > /dev/null
[storage]
driver = "overlay"
runroot = "/run/user/733"
EOF
```

10. Create Podman UFM socket:

```
cat <<EOF > /usr/lib/systemd/system/podman-ufm.socket
[Unit]
Description=Podman API Socket For Nvidia UFM

[Socket]
SocketUser=ufmadm
SocketGroup=ufmadm
ListenStream=%t/podman-ufm/podman-ufm.sock
SocketMode=0660

[Install]
WantedBy=sockets.target
EOF
```

11. Create Podman UFM service

```
cat <<EOF > /usr/lib/systemd/system/podman-ufm.service
[Unit]
Description=Podman API Service for Nvidia UFM
Requires=podman-ufm.socket
After=podman-ufm.socket
StartLimitIntervalSec=0

[Service]
Delegate=true
Type=exec
User=ufmadm
Group=ufmadm
KillMode=process
Environment=LOGGING="--log-level=info"
ExecStart=/usr/bin/podman \$LOGGING system service
LimitMEMLOCK=infinity

[Install]
WantedBy=default.target
EOF
```

12. Create Podman cleanup service:

```
cat <<EOF > /usr/lib/systemd/system/podman-ufm-
cleanup.service
[Unit]
Description=podman-ufm-cleanup - clean stuck rootless
containers at boot
After=podman-ufm.service
Before=ufm-enterprise.service

[Service]
Type=oneshot
User=ufmadm
Group=ufmadm
ExecStart=/usr/bin/podman system migrate

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

13. Enable and start Podman services:

```
systemctl daemon-reload
systemctl enable --now podman-ufm.socket
systemctl enable --now podman-ufm.service
systemctl enable --now podman-ufm-cleanup.service
```

14. Create Udev Rules for InfiniBand Devices

```
cat <<EOF > /etc/udev/rules.d/70-umad.rules
KERNEL=="umad*", SUBSYSTEM=="infiniband_mad", MODE="0600",
OWNER="ufmadm", GROUP="ufmadm"
KERNEL=="issm*", SUBSYSTEM=="infiniband_mad", MODE="0600",
OWNER="ufmadm", GROUP="ufmadm"
EOF

udevadm control --reload-rules
udevadm trigger
```

15. Clean and create UFM directories

```
rm -rf /opt/ufm/systemd
sudo -u ufmadm mkdir -p /opt/ufm/ufm_plugins_data
sudo -u ufmadm mkdir -p /opt/ufm/systemd
sudo -u ufmadm mkdir -p /opt/ufm/etc/apache2
```

16. Load UFM image and extract version:

```
# Extract UFM version from filename (e.g., ufm_6.22.0-
7.ubuntu24.x86_64-docker.img.gz -> 6_22_0_7)
UFM_VERSION=$(basename "$UFM_IMAGE_FILE" | sed 's/ufm_\([0-9][^.]*\.[^.]*\.[^.]*-
[^.]*\)\.ubuntu.*/\1/' | tr '.-' '_')
echo "UFM Version: $UFM_VERSION"

# Load the UFM image
sudo -u ufmadm podman load -i "$UFM_IMAGE_FILE"
```

17. Create version-specific directory and soft link:

```
# Create version-specific directory in shared storage
sudo -u ufmadm mkdir -p
/opt/ufm/shared_files/ufm-${UFM_VERSION}

# Remove existing files link if it exists
rm -f /opt/ufm/files

# Create soft link to version-specific directory
sudo -u ufmadm ln -s /opt/ufm/shared_files/ufm-${UFM_VERSION}
/opt/ufm/files

# Verify the soft link
ls -la /opt/ufm/files
```

18. Run UFM installer:

```
sudo -u ufmadm podman run -it --rm --name=ufm_installer \
                          -v /run/podman-ufm/podman-
ufm.sock:/var/run/docker.sock \
                          -v
/opt/ufm/:/installation/ufm_files/ \
                          -v
/opt/ufm/files:/installation/ufm_files/files \
                          -v
/opt/ufm/systemd:/etc/systemd_files/ \
                          mellanox/ufm-enterprise:latest \
                          --install \
                          --fabric-interface ib0 \
                          --rootless \
                          --plugin-path
/opt/ufm/ufm_plugins_data \
                          --ufm-user ufmadm \
                          --ufm-group ufmadm \
                          --ufm-infra
```

Note

Note: Replace `ib0` with your actual InfiniBand interface name, if it is not the default ib0.

Note: - All other UFM install flags are supported and can be added to the command.

19. Load Redis Image (if not using external Redis):

```
Load the given Redis image (in case you are not using external
Redis)
sudo - u ufmadm load -i "<PATH TO GIVEN REDIS IMAGE>"
```

20. Load Fast API Plugin image:

```
sudo - u ufmadm run --hostname $HOSTNAME --rm --
name=ufm_plugin_mgmt --entrypoint="" \
    -v /run/podman-ufm/podman-ufm.sock:/var/run/docker.sock \
    -v /opt/ufm/files:/opt/ufm/shared_config_files \
    -v /dev/log:/dev/log \
    -v /sys/fs/cgroup:/sys/fs/cgroup:ro \
    -v /lib/modules:/lib/modules:ro \
    -v /opt/ufm/ufm_plugins_data:/opt/ufm/ufm_plugins_data \
    -e UFM_CONTEXT=ufm-infra \
    mellanox/ufm-enterprise:latest \
    /opt/ufm/scripts/manage_ufm_plugins.sh add -p fast_api -t
${FAST_API_VERSION} -c ufm-infra
```

21. Install service files:

```
mv /opt/ufm/systemd/ufm-enterprise.service
/etc/systemd/system/ufm-enterprise.service
mv /opt/ufm/systemd/ufm-infra.service
/etc/systemd/system/ufm-infra.service
systemctl daemon-reload
```

To **start UFM as a standalone instance**, run:

```
systemctl daemon-reload
systemctl start ufm-infra
systemctl start ufm-enterprise
```

Running in HA Mode

> **Note**
>
> Do not manually start any services.

1. Ensure UFM and UFM-HA are installed on all nodes as described in the above sections.
2. Mount /opt/ufm/files on all standby nodes as described point #2 (Mount the UFM directory)
3. On one node, edit the HA configuration file:

   ```
   /etc/ufm_ha/ha_nodes.cfg
   ```

   Fill each node parameters

```
[Node.1]
# valid role options: master/standby
role = master
# Mandatory
primary_ip =
# Mandatory if dual_link = true
secondary_ip =

[Node.2]
role = standby
primary_ip =
secondary_ip =

[Node.3]
role = standby
primary_ip =
secondary_ip =
```

4. Ensure the file sync mode is set to `external-storage`, and that the shared file system is mounted prior to HA configuration.

```
[FileSync]
# valid options are: drbd/external-storage
# in case of external-storage the user MUST mount the files
system PRIOR to ha configuration
mode = external-storage
```

5. Copy the edited file to all nodes at the same path.

6. Configure the cluster, starting from standby nodes and ending with the master node:

```
ufm_ha_cluster config -p <password>
```

> **Note**
>
> Use the same password on all nodes.

7. After finishing the configuration on all nodes, run:

```
ufm_ha_cluster status
```

8. Start the cluster:

```
ufm_ha_cluster start
```

9. Check cluster status again to ensure all services have started successfully.

Troubleshooting

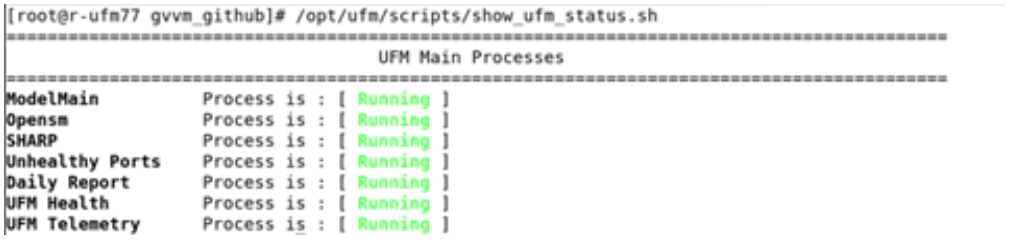
UFM General Troubleshooting

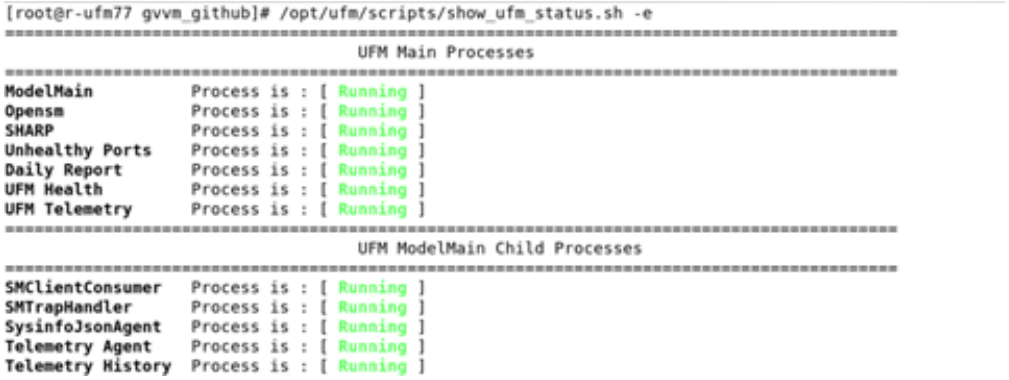
Showing UFM Processes Status

This functionality allows users to view the current status of main processes handled by the UFM.

- To view the main UFM processes, run the script `show_ufm_status.sh` under the `/opt/ufm/scripts` `.` Example: `/opt/ufm/scripts/show_ufm_status.sh`
- To view the UFM main and child processes, run the script `show_ufm_status.sh` with `-e` (extended_processes).

  Example: `/opt/ufm/scripts/show_ufm_status.sh` `-e`





UFM Startup Logging

UFM provides comprehensive startup logging capabilities that track the initialization progress of all UFM services in real-time. When enabled via the `startup_logging_enabled` configuration flag, the system creates detailed logs of each startup phase and stage, including timestamps, service names, and status updates.

This option generates several key files: `/opt/ufm/files/log/ufm_startup.log` (detailed startup logs), `/opt/ufm/files/log/ufm_startup_progress_ufm-ha-cluster.json` (only relevant in UFM HA mode), `/opt/ufm/files/log/ufm_startup_progress_ufm-enterprise.json` and `/opt/ufm/files/log/ufm_startup_progress_ufm-modelmain.json` for persistent progress tracking across all services.

This logging system provides administrators with complete visibility into the UFM startup process, making it easier to diagnose startup issues, monitor service initialization progress, and ensure all components are starting correctly.

The logging level can be configured using the `startup_logging_log_level` parameter to control the verbosity of startup messages.

For more information about the configuration, refer to Configuring UFM Logging.

Example for the content of `/opt/ufm/files/log/ufm_startup.log`:

```
2025-08-28 19:18:01 [ufm-ha-cluster] [INFO] ufm-ha-cluster Startup
Log. Host: c-141-137-180-181, IP Address: 10.141.137.181
2025-08-28 19:18:01 [ufm-ha-cluster] [INFO] [Service Startup] ufm-
ha-cluster process started [START]
2025-08-28 19:18:01 [ufm-ha-cluster] [INFO] [System Preparation]
System Preparation started [START]
2025-08-28 19:18:01 [ufm-ha-cluster] [INFO] [System Preparation -
Check HA configurations] Check HA configurations completed,
duration: 4ms [SUCCESS]
2025-08-28 19:18:01 [ufm-ha-cluster] [INFO] [System Preparation -
Validate DRBD Storage state & connectivity] Validate DRBD Storage
state & connectivity completed, duration: 51ms [SUCCESS]
2025-08-28 19:18:01 [ufm-ha-cluster] [INFO] [System Preparation -
PCS Resources Cleanup] PCS Resources Cleanup completed, duration:
635ms [SUCCESS]
2025-08-28 19:18:01 [ufm-ha-cluster] [INFO] [System Preparation -
Refresh File System (DRBD Only)] Refresh File System (DRBD Only)
completed, duration: 25ms [SUCCESS]
2025-08-28 19:18:01 [ufm-ha-cluster] [INFO] [System Preparation]
System Preparation completed (4/4 stages), duration: 727ms
[COMPLETED]
2025-08-28 19:18:02 [ufm-ha-cluster] [INFO] [Pcs Resource
Activation] Pcs Resource Activation started [START]
2025-08-28 19:18:04 [ufm-ha-cluster] [INFO] [Pcs Resource Activation
- Enable PCS resources] Enable PCS resources completed - Enabled
resources ufm-ha-watcher ufm-enterprise, duration: 529ms
[SUCCESS]
2025-08-28 19:18:04 [ufm-ha-cluster] [INFO] [Pcs Resource
Activation] Pcs Resource Activation completed (1/1 stages),
duration: 530ms [COMPLETED]
2025-08-28 19:18:04 [ufm-ha-cluster] [INFO] [Service Startup] ufm-
ha-cluster startup process completed successfully [COMPLETED]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] ufm-enterprise Startup
Log. Host: c-141-137-180-181, IP Address: 10.141.137.181
```

```
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Service Startup] ufm-enterprise process started [START]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Pre Initialization Checks - Check if UFM is already running] Check if UFM is already running completed - UFM is not running, duration: 0ms [SUCCESS]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Pre Initialization Checks] Pre Initialization Checks started [START]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Pre Initialization Checks - Verify log directories exist] Verify log directories exist completed, duration: 1ms [SUCCESS]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Pre Initialization Checks - Validate UFM license] Validate UFM license completed, duration: 9ms [SUCCESS]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Pre Initialization Checks - Verify log directories permissions] Verify log directories permissions completed, duration: 50ms [SUCCESS]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Pre Initialization Checks - Validate UFM configuration files] Validate UFM configuration files completed, duration: 275ms [SUCCESS]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Pre Initialization Checks] Pre Initialization Checks completed (5/5 stages), duration: 345ms [COMPLETED]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Environment Setup] Environment Setup started [START]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Environment Setup - Check for other running Subnet Managers] Check for other running Subnet Managers completed, duration: 78ms [SUCCESS]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Environment Setup - Check IB interface status] Check IB interface status completed, duration: 168ms [SUCCESS]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Environment Setup - Check disk space on UFM partitions] Disk space check completed successfully on /opt/ufm [SUCCESS]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Environment Setup - Check disk space on UFM partitions] Disk space check completed successfully on /opt/ufm/files [SUCCESS]
```

```
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Environment Setup - Check disk space on UFM partitions] Disk space check completed successfully on /opt/ufm/tmp [SUCCESS]
2025-08-28 19:18:10 [ufm-enterprise] [INFO] [Environment Setup - Check disk space on UFM partitions] Check disk space on UFM partitions completed, duration: 93ms [SUCCESS]
2025-08-28 19:18:11 [ufm-enterprise] [INFO] [Environment Setup - Update logrotate configuration] Update logrotate configuration completed, duration: 258ms [SUCCESS]
2025-08-28 19:18:11 [ufm-enterprise] [INFO] [Environment Setup - Sync UFM web client files] Sync UFM web client files completed, duration: 41ms [SUCCESS]
2025-08-28 19:18:11 [ufm-enterprise] [INFO] [Environment Setup - Check multisubnet mode] Check multisubnet mode completed - Multisubnet mode is disabled, continuing with normal startup, duration: 0ms [SUCCESS]
2025-08-28 19:18:11 [ufm-enterprise] [INFO] [Environment Setup] Environment Setup completed (6/6 stages), duration: 661ms [COMPLETED]
2025-08-28 19:18:11 [ufm-enterprise] [INFO] [Core Services Startup] Core Services Startup started [START]
2025-08-28 19:18:11 [ufm-enterprise] [INFO] [Core Services Startup - Start MAD Limiter] Start MAD Limiter skipped - Madlimiter is disabled, duration: 0ms [SKIPPED]
2025-08-28 19:18:11 [ufm-enterprise] [INFO] [Core Services Startup - Start OpenSM] Start OpenSM completed, duration: 375ms [SUCCESS]
2025-08-28 19:18:11 [ufm-enterprise] [INFO] [Core Services Startup - Start SM Communicators Manager] Start SM Communicators Manager skipped - SM Communicators Manager should be enabled in UFM Infra, duration: 0ms [SKIPPED]
2025-08-28 19:18:11 [ufm-enterprise] [INFO] [Core Services Startup - Start SHArP Aggregation Manager] Start SHArP Aggregation Manager skipped - Sharp is disabled, duration: 0ms [SKIPPED]
2025-08-28 19:18:13 [ufm-enterprise] [INFO] [Core Services Startup - Start UFM Primary Telemetry] Start UFM Primary Telemetry completed, duration: 972ms [SUCCESS]
```

```
2025-08-28 19:18:15 [ufm-enterprise] [INFO] [Core Services Startup - Start UFM Secondary Telemetry] Start UFM Secondary Telemetry completed, duration: 140ms [SUCCESS]
2025-08-28 19:18:15 [ufm-enterprise] [INFO] [Core Services Startup] Core Services Startup completed (6/6 stages), duration: 22ms [COMPLETED]
2025-08-28 19:18:15 [ufm-enterprise] [INFO] [Web Infra Services Startup] Web Infra Services Startup started [START]
2025-08-28 19:18:15 [ufm-enterprise] [INFO] [Web Infra Services Startup - Configure SSL certificates] Configure SSL certificates completed, duration: 14ms [SUCCESS]
2025-08-28 19:18:15 [ufm-enterprise] [INFO] [Web Infra Services Startup - Generate UFM web configuration] Generate UFM web configuration completed, duration: 483ms [SUCCESS]
2025-08-28 19:18:15 [ufm-enterprise] [INFO] [Web Infra Services Startup - Restart web server] Restart web server completed, duration: 174ms [SUCCESS]
2025-08-28 19:18:15 [ufm-enterprise] [INFO] [Web Infra Services Startup] Web Infra Services Startup completed (3/3 stages), duration: 679ms [COMPLETED]
2025-08-28 19:18:15 [ufm-enterprise] [INFO] [Ufm Services Startup] Ufm Services Startup started [START]
2025-08-28 19:18:18 [ufm-enterprise] [INFO] [Ufm Services Startup - Start Authentication service] Start Authentication service completed, duration: 140ms [SUCCESS]
2025-08-28 19:18:18 [ufm-enterprise] [INFO] [Ufm Services Startup - Start UFM Model Main] Start UFM Model Main completed, duration: 242ms [SUCCESS]
2025-08-28 19:18:20 [ufm-modelmain] [INFO] ufm-modelmain Startup Log. Host: c-141-137-180-181, IP Address: 10.141.137.181
2025-08-28 19:18:20 [ufm-modelmain] [INFO] [Service Startup] ufm-modelmain process started [START]
2025-08-28 19:18:20 [ufm-modelmain] [INFO] [Startup] Startup started [START]
2025-08-28 19:18:20 [ufm-modelmain] [INFO] [Startup - SM client consumer] SM client consumer completed, duration: 3ms [SUCCESS]
```

```
2025-08-28 19:18:20 [ufm-modelmain] [INFO] [Startup - Sysinfo JSON agent] Sysinfo JSON agent completed, duration: 6ms [SUCCESS]
2025-08-28 19:18:20 [ufm-modelmain] [INFO] [Startup - Check network management IB interface] Check network management IB interface completed - Network management interface ib0 is running and has IP address, duration: 1ms [SUCCESS]
2025-08-28 19:18:20 [ufm-modelmain] [INFO] [Startup - Initialize core infrastructure] Initialize core infrastructure completed, duration: 70ms [SUCCESS]
2025-08-28 19:18:20 [ufm-modelmain] [INFO] [Startup - Initialize ModelMain license checker] Initialize ModelMain license checker completed, duration: 3ms [SUCCESS]
2025-08-28 19:18:20 [ufm-modelmain] [INFO] [Startup - Load data from database before discovery] Load data from database before discovery completed, duration: 27ms [SUCCESS]
2025-08-28 19:18:20 [ufm-enterprise] [INFO] [Ufm Services Startup - Start Daily Reports] Start Daily Reports skipped - Daily reports are disabled, duration: 0ms [SKIPPED]
2025-08-28 19:18:21 [ufm-enterprise] [INFO] [Ufm Services Startup - Start Unhealthy Ports] Start Unhealthy Ports completed, duration: 97ms [SUCCESS]
2025-08-28 19:18:21 [ufm-enterprise] [INFO] [Ufm Services Startup - Start Telemetry Sampling] Start Telemetry Sampling completed, duration: 111ms [SUCCESS]
2025-08-28 19:18:21 [ufm-enterprise] [INFO] [Ufm Services Startup - Check UFM hardware state] Check UFM hardware state completed, duration: 0ms [SUCCESS]
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Startup - Initialize fabric site] Initialize fabric site skipped - Default site already exists, duration: 0ms [SKIPPED]
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Startup - SM trap handler] SM trap handler completed, duration: 11ms [SUCCESS]
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Startup - Telemetry agent manager] Telemetry agent manager completed, duration: 1ms [SUCCESS]
```

```
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Startup - Network
infiniband discovery] Network infiniband discovery completed,
duration: 41ms [SUCCESS]
2025-08-28 19:18:22 [ufm-enterprise] [INFO] [Ufm Services Startup -
Start UFM plugins] Start UFM plugins completed, duration: 766ms
[SUCCESS]
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Startup - REST API
server] REST API server completed - UFM Main API started,
duration: 379ms [SUCCESS]
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Startup - Fabric model
construction] Fabric model construction completed, duration: 5ms
[SUCCESS]
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Startup - Load model
from database] Load model from database completed, duration: 59ms
[SUCCESS]
2025-08-28 19:18:22 [ufm-enterprise] [INFO] [Ufm Services Startup -
Start UFM health] Start UFM health completed, duration: 282ms
[SUCCESS]
2025-08-28 19:18:22 [ufm-enterprise] [INFO] [Ufm Services Startup]
Ufm Services Startup completed (8/8 stages), duration: 718ms
[COMPLETED]
2025-08-28 19:18:22 [ufm-enterprise] [INFO] [Service Startup] ufm-
enterprise startup process completed successfully [COMPLETED]
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Startup - Load data from
database after discovery] Load data from database after discovery
completed, duration: 19ms [SUCCESS]
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Startup - Construct API
resources] Construct API resources completed, duration: 4ms
[SUCCESS]
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Startup] Startup
completed (15/15 stages), duration: 758ms [COMPLETED]
2025-08-28 19:18:22 [ufm-modelmain] [INFO] [Service Startup] ufm-
modelmain startup process completed successfully [COMPLETED]
```

Example for the content of `/opt/ufm/files/log/ufm_startup_progress_ufm-enterprise.json`

```
{
  "service_name": "ufm-enterprise",
  "start_time": "2025-08-20T14:00:44.284780",
  "end_time": "2025-08-20T14:01:10.200929",
  "status": "COMPLETE",
  "phases": {
    "pre_initialization_checks": {
      "status": "COMPLETE",
      "start_time": "2025-08-20T14:00:44.353450",
      "end_time": "2025-08-20T14:00:45.552568",
      "stages": {
        "check_already_running": {
          "status": "SUCCESS",
          "start_time": "2025-08-20T14:00:44.329303",
          "end_time": "2025-08-20T14:00:44.329303",
          "description": "Check if UFM is already running",
          "message": "UFM is not running",
          "duration_ms": 0
        },
        "check_log_directories_existence": {
          "status": "SUCCESS",
          "start_time": "2025-08-20T14:00:44.354393",
          "end_time": "2025-08-20T14:00:44.399704",
          "description": "Verify log directories exist",
          "message": "",
          "duration_ms": 45
        },
        "check_log_directories_permissions": {
          "status": "SUCCESS",
          "start_time": "2025-08-20T14:00:44.422848",
          "end_time": "2025-08-20T14:00:45.105769",
          "description": "Verify log directories permissions",
          "message": "",
          "duration_ms": 682
        },
```

```
            "validate_license": {
                "status": "SUCCESS",
                "start_time": "2025-08-20T14:00:45.130041",
                "end_time": "2025-08-20T14:00:45.165553",
                "description": "Validate UFM license",
                "message": "",
                "duration_ms": 35
            },
            "validate_config_files": {
                "status": "SUCCESS",
                "start_time": "2025-08-20T14:00:45.188146",
                "end_time": "2025-08-20T14:00:45.551215",
                "description": "Validate UFM configuration files",
                "message": "",
                "duration_ms": 363
            }
        },
        "completed_stages": 5,
        "handled_stages": 5,
        "total_stages": 5,
        "duration_ms": 199
    },
    "environment_setup": {
        "status": "COMPLETE",
        "start_time": "2025-08-20T14:00:45.612348",
        "end_time": "2025-08-20T14:00:46.657803",
        "stages": {
            "check_other_sm": {
                "status": "SUCCESS",
                "start_time": "2025-08-20T14:00:45.613700",
                "end_time": "2025-08-20T14:00:45.745571",
                "description": "Check for other running Subnet Managers",
                "message": "",
                "duration_ms": 131
            },
            "check_ib_interface": {
```

```
        "status": "SUCCESS",
        "start_time": "2025-08-20T14:00:45.768596",
        "end_time": "2025-08-20T14:00:45.974154",
        "description": "Check IB interface status",
        "message": "",
        "duration_ms": 205
    },
    "check_disk_space": {
        "status": "SUCCESS",
        "start_time": "2025-08-20T14:00:45.997901",
        "end_time": "2025-08-20T14:00:46.204390",
        "description": "Check disk space on UFM partitions",
        "message": "",
        "duration_ms": 206
    },
    "logrotate_config": {
        "status": "SUCCESS",
        "start_time": "2025-08-20T14:00:46.236403",
        "end_time": "2025-08-20T14:00:46.534769",
        "description": "Update logrotate configuration",
        "message": "",
        "duration_ms": 298
    },
    "sync_web_client_files": {
        "status": "SUCCESS",
        "start_time": "2025-08-20T14:00:46.565332",
        "end_time": "2025-08-20T14:00:46.610217",
        "description": "Sync UFM web client files",
        "message": "",
        "duration_ms": 44
    },
    "check_multisubnet_mode": {
        "status": "SUCCESS",
        "start_time": "2025-08-20T14:00:46.656620",
        "end_time": "2025-08-20T14:00:46.656620",
        "description": "Check multisubnet mode",
```

```
            "message": "Multisubnet mode is disabled, continuing with normal startup",
            "duration_ms": 0
        }
    },
    "completed_stages": 6,
    "handled_stages": 6,
    "total_stages": 6,
    "duration_ms": 45
},
"core_services_startup": {
    "status": "COMPLETE",
    "start_time": "2025-08-20T14:00:46.725547",
    "end_time": "2025-08-20T14:00:55.173574",
    "stages": {
        "mad_limiter": {
            "status": "SKIPPED",
            "start_time": "2025-08-20T14:00:46.702630",
            "end_time": "2025-08-20T14:00:46.702630",
            "description": "Start MAD Limiter",
            "message": "Madlimiter is disabled",
            "duration_ms": 0
        },
        "opensm": {
            "status": "SUCCESS",
            "start_time": "2025-08-20T14:00:46.827668",
            "end_time": "2025-08-20T14:00:47.428092",
            "description": "Start OpenSM",
            "message": "",
            "duration_ms": 600
        },
        "sharp": {
            "status": "SKIPPED",
            "start_time": "2025-08-20T14:00:47.469382",
            "end_time": "2025-08-20T14:00:47.469382",
            "description": "Start SHArP Aggregation Manager",
            "message": "Sharp is disabled",
```

```
            "duration_ms": 0
        },
        "communicators_mgr": {
            "status": "SUCCESS",
            "start_time": "2025-08-20T14:00:47.494769",
            "end_time": "2025-08-20T14:00:47.534151",
            "description": "Start Communicators Manager",
            "message": "",
            "duration_ms": 39
        },
        "primary_telemetry": {
            "status": "SUCCESS",
            "start_time": "2025-08-20T14:00:48.985180",
            "end_time": "2025-08-20T14:00:51.849616",
            "description": "Start UFM Primary Telemetry",
            "message": "",
            "duration_ms": 864
        },
        "secondary_telemetry": {
            "status": "SUCCESS",
            "start_time": "2025-08-20T14:00:52.355037",
            "end_time": "2025-08-20T14:00:55.172100",
            "description": "Start UFM Secondary Telemetry",
            "message": "",
            "duration_ms": 817
        }
    },
    "completed_stages": 6,
    "handled_stages": 6,
    "total_stages": 6,
    "duration_ms": 448
},
"web_infra_services_startup": {
    "status": "COMPLETE",
    "start_time": "2025-08-20T14:00:55.202439",
    "end_time": "2025-08-20T14:00:58.032880",
```

```
    "stages": {
      "ssl_configuration": {
        "status": "SUCCESS",
        "start_time": "2025-08-20T14:00:55.203511",
        "end_time": "2025-08-20T14:00:55.258061",
        "description": "Configure SSL certificates",
        "message": "",
        "duration_ms": 54
      },
      "web_config_generation": {
        "status": "SUCCESS",
        "start_time": "2025-08-20T14:00:55.282592",
        "end_time": "2025-08-20T14:00:56.690531",
        "description": "Generate UFM web configuration",
        "message": "",
        "duration_ms": 407
      },
      "web_server_restart": {
        "status": "SUCCESS",
        "start_time": "2025-08-20T14:00:56.715527",
        "end_time": "2025-08-20T14:00:58.031300",
        "description": "Restart web server",
        "message": "",
        "duration_ms": 315
      }
    },
    "completed_stages": 3,
    "handled_stages": 3,
    "total_stages": 3,
    "duration_ms": 830
  },
  "ufm_services_startup": {
    "status": "COMPLETE",
    "start_time": "2025-08-20T14:00:58.095138",
    "end_time": "2025-08-20T14:01:10.140721",
    "stages": {
```

```
"auth_service": {
    "status": "SUCCESS",
    "start_time": "2025-08-20T14:00:58.114734",
    "end_time": "2025-08-20T14:01:00.306014",
    "description": "Start Authentication service",
    "message": "",
    "duration_ms": 191
},
"ufm_main": {
    "status": "SUCCESS",
    "start_time": "2025-08-20T14:01:00.330510",
    "end_time": "2025-08-20T14:01:01.187987",
    "description": "Start UFM Model Main",
    "message": "",
    "duration_ms": 857
},
"daily_reports": {
    "status": "SUCCESS",
    "start_time": "2025-08-20T14:01:03.231535",
    "end_time": "2025-08-20T14:01:03.279463",
    "description": "Start Daily Reports",
    "message": "",
    "duration_ms": 47
},
"unhealthy_ports": {
    "status": "SUCCESS",
    "start_time": "2025-08-20T14:01:03.304244",
    "end_time": "2025-08-20T14:01:04.478741",
    "description": "Start Unhealthy Ports",
    "message": "",
    "duration_ms": 174
},
"telemetry_sampling": {
    "status": "SUCCESS",
    "start_time": "2025-08-20T14:01:04.513890",
    "end_time": "2025-08-20T14:01:04.627906",
```

```
            "description": "Start Telemetry Sampling",
            "message": "",
            "duration_ms": 114
        },
        "check_ufm_hardware_state": {
            "status": "SUCCESS",
            "start_time": "2025-08-20T14:01:04.654889",
            "end_time": "2025-08-20T14:01:04.691246",
            "description": "Check UFM hardware state",
            "message": "",
            "duration_ms": 36
        },
        "plugins": {
            "status": "SUCCESS",
            "start_time": "2025-08-20T14:01:04.714797",
            "end_time": "2025-08-20T14:01:09.110861",
            "description": "Start UFM plugins",
            "message": "",
            "duration_ms": 396
        },
        "ufm_health": {
            "status": "SUCCESS",
            "start_time": "2025-08-20T14:01:09.390525",
            "end_time": "2025-08-20T14:01:10.114399",
            "description": "Start UFM health",
            "message": "",
            "duration_ms": 723
        }
    },
    "completed_stages": 8,
    "handled_stages": 8,
    "total_stages": 8,
    "duration_ms": 45
  }
},
"current_phase": null,
```

```
    "current_stage" :  null
}
```

Collecting System Dump

This script collects detailed system dumps from multiple UFM components, including:

- **UFM Enterprise** (running on Docker or bare-metal)
- **Host system information**
- **Cyber-AI system data** (if installed)
- **UFM High Availability (HA)** information (if configured)

All collected data is packaged into a `.tar.gz` archive and saved in the specified output directory or the default backup location.

This archive should be shared with the UFM Support team for troubleshooting any UFM-related issues.

Usage

```
ufm_sysdump [OPTIONS]
Example: /opt/ufm/files/scripts/ufm_sysdump.sh
```

Options

Option	Description	Default / Notes
`--output`, `-O`, `-o <DIR>`	Specify the output folder for the sysdump	**Default:** `/tmp/`
`--time`, `-t <TIME>`	Collect data starting from the specified time	**Format:** `YYYY-MM-DD_HH:MM:SS`

Option	Description	Default / Notes
`--archives`, `-a <NUM>`	Limit the number of archive files to keep	**Default:** `3`
`--container`	Allow running inside a container	By default, the script exits if run in a container
`--help`, `-h`	Show this help message and exit	—

Examples

```
/opt/ufm/files/scripts/ufm_sysdump.sh
Collect sysdump with default prameters.
    /opt/ufm/files/scripts/ufm_sysdump.sh  --output
/opt/backups/             # Collect sysdump to specific folder
    /opt/ufm/files/scripts/ufm_sysdump.sh  --time 2025-01-
01_00:00:00         # Collect data from specific time
    /opt/ufm/files/scripts/ufm_sysdump.sh  --archives
5                       # Keep up to 5 archive files
    /opt/ufm/files/scripts/ufm_sysdump.sh  --
container                          # Allow running inside container
    /opt/ufm/files/scripts/ufm_sysdump.sh  --
help                               # Show this help message
```

Note

One UFM Bare-Metal instance needs to run this script from the host. On UFM-HA, this script must be run from the host that is currently the master.

Note

By default, the maximum in `--archives` is 15.

Restoring UFM Data Upon In-Service Upgrade Failure

Note

These instructions apply in high availability scenario only.

In the event of an in-service upgrade failure, the previous version of UFM's data will be safeguarded as a backup in the "`/opt/ufm/BACKUP`" directory, formatted as "`ufm_upgrade_backup_<prev_version>-<new_version<_<date>.zip`".

To restore the data on the un-upgraded node, follow these steps:

1. Copy the backup file from the upgraded node to the un-upgraded node using the following command:

```
scp /opt/ufm/BACKUP/ufm_upgrade_backup_<prev_version>-
<new_version<_<date>.zip
root@<unupgraded_node_ip>:/opt/ufm/BACKUP/
```

2. Perform a failover of UFM to the master node, which is mandatory for data mount migration (including '/opt/ufm/files') to the master node: On the Master node, execute:

```
ufm_ha_cluster takeover
```

3. Stop UFM on the un-upgraded node:

```
 ufm_ha_cluster stop
```

4. Restore UFM configuration files from the backup:

```
/opt/ufm/scripts/ufm_restore.sh -f
/opt/ufm/BACKUP/ufm_upgrade_backup_<prev_version>-
<new_version<_<date>.zip
```

5. Start UFM on the un-upgraded node (Note: Only the upgraded node can function until the upgrade issue is resolved, and failovers will not work).

Now, the issue that caused the upgrade failure can be addressed. If the problem is resolved, you can attempt the in-service upgrade again by failing UFM over to the upgraded node.

Alternatively, if needed, you can revert the changes made by reinstalling the old UFM version on the upgraded node.

UFM High-Availability Troubleshooting

Refer to the NVIDIA UFM High-Availability User Guide for HA monitoring and troubleshooting.

Split-Brain Recovery in HA Installation

The split-brain problem is a DRBD synchronization issue (HA status shows `DUnknown` in the DRBD disk state), which occurs when both HA nodes are rebooted. For example, in cases of electricity shut-down. To recover, please follow the below steps:

- **Step 1:** Manually choose a node where data modifications will be discarded.

It is called the split-brain victim. Choose wisely; all modifications will be lost! When in doubt, run a backup of the victim's data before you continue.

When running a Pacemaker cluster, you can enable maintenance mode. If the split-brain victim is in the Primary role, bring down all applications using this resource. Now switch the victim to the Secondary role:

```
victim# drbdadm secondary ha_data
```

- **Step 2:** Disconnect the resource if it's in connection state **WFConnection**:

```
victim# drbdadm disconnect ha_data
```

- **Step 3:** Force discard of all modifications on the split-brain victim:

```
victim# drbdadm -- --discard-my-data connect ha_data
```

For DRBD 8.4.x:

```
victim# drbdadm connect --discard-my-data ha_data
```

- **Step 4:** Resync starts automatically if the survivor is in a WFConnection network state. If the split-brain survivor is still in a Standalone connection state, reconnect it:

```
survivor# drbdadm connect ha_data
```

Now the resynchronization from the survivor (SyncSource) to the victim (SyncTarget) starts immediately. There is no full sync initiated, but all modifications on the victim will be overwritten by the survivor's data, and modifications on the survivor will be applied to the victim.

Performing Failover on Non-Master Node

The `ufm_ha_cluster` failover action fails with the following error: "Cannot perform failover on non-master node". To fix, follow the below action:

- **Step 1**: Verify that `/etc/hosts` file on both the master and standby UFM hosts contains the correct host names and IP addresses mapping.
- **Step 2**: If necessary, fix the mapping and retry the failover command.

Appendixes

- SM Configurations
- Appendix – Diagnostic Utilities
- Appendix - Supported Port Counters and Events
- Appendix – Used Ports
- Appendix – Configuration Files Auditing
- Appendix – IB Router
- Appendix – NVIDIA SHARP Integration
- Appendix - UFM SLURM Integration
- Appendix - Switch Grouping
- Appendix – Device Management Feature Support
- Appendix - MAD Limiter
- Appendix - UFM Versions Manager

SM Configurations

- Appendix – SM Default Files
- Appendix – UFM Subnet Manager Default Properties
- Appendix – Partitioning
- Appendix – Enhanced Quality of Service
- Appendix – Routing Chains
- Appendix – Adaptive Routing

- Appendix – Security Features
- Appendix – SM Activity Report
- Appendix – OpenSM Configuration Files for Congestion Control
- Appendix – UFM AI Mode

Appendix – SM Default Files

The SM default files are located under the following paths:

- Default SM configuration file – /opt/ufm/files/conf/opensm/opensm.conf
- Default node name map file – /opt/ufm/files/conf/opensm/ib-node-name-map
- Default partition configuration file – /opt/ufm/files/conf/opensm/partitions.conf
- Default QOS policy configuration file – /opt/ufm/files/conf/opensm/qos-policy.conf
- Default prefix routes file – /opt/ufm/files/conf/opensm/prefix-routes.conf

Appendix – UFM Subnet Manager Default Properties

The following table provides a comprehensive list of UFM SM default properties.

Category	Property	Config File Attribute	Default
Generic	Subnet Prefix	subnet_prefix	0xfe80000000(
	LMC	lmc	0
	SM LID	master_sm_lid	0

Category	Property	Config File Attribute	Default
Security Keys	M_Key	m_key	0x0000000000
		m_key_per_port	FALSE
		m_key_lease_period	60
		m_key_lookup	TRUE
	SM_Key	sm_key	0x0000000000
	SA_Key	sa_key	0x0000000000
	Key manager	key_mgr_seed	0x0000000000
	Congestion control security keys	cc_key_lease_period	60
		cc_key_protect_bit	1

Category	Property	Config File Attribute	Default
	Vendor specific MADs security keys	vs_key_enable	0
		vs_key_lease_period	60
		vs_key_ci_protect_bits	1
	Partition enforcement	part_enforce	both (default- o enforcement er

Category	Property	Config File Attribute	Default
Limits	Packet Life Time	packet_life_time	0x12
	VL Stall Count	vl_stall_count	0x07
	Leaf VL Stall Count	leaf_vl_stall_count	0x07
	Head Of Queue Life time	head_of_queue_lifetime	0x12
	Leaf Head Of Queue Life time	leaf_head_of_queue_lifetime	0x10
	Local PHY Error Threshold	local_phy_errors_threshold	0x08
	Overrun Errors Threshold	overrun_errors_threshold	0x08
	Subnet Timeout	subnet_timeout	18 (1 second)

Category	Property	Config File Attribute	Default
	VS MADs on the wire	vs_max_outstanding_mads	500
Link Speed	Force Link Speed	force_link_speed	15 (Do NOT change
	Force Link Speed	force_link_speed_ext	31 (Do NOT change
	Force Link Speed	force_link_speed_ext2	7 (Do NOT change

Category	Property	Config File Attribute	Default
SM Threading	SMP MAD/Trap processing	smp_threads	0
	SA/GMP MAD processing	gmp_threads	0
	SA/GMP Trap processing	gmp_traps_threads_num	1
	Routing Threads	routing_threads_num	0
	Routing Threads Per Core	max_threads_per_core	0

Category	Property	Config File Attribute	Default
Sweep	Sweep Interval	sweep_interval	10
	Reassign Lids	reassign_lids	FALSE (disablec
	Force Heavy Sweep	force_heavy_sweep_window	-1
	Sweep On trap	sweep_on_trap	TRUE (enabled)
	Alternative Route Calculation	max_alt_dr_path_retries	0
	Fabric Rediscovery	max_seq_redisc	2
	Offsweep Rebalancing Enable	offsweep_balancing_enabled	FALSE
	Offsweep Rebalancing Window	offsweep_balancing_window	180

Category	Property	Config File Attribute	Default
Handover	SM Priority	sm_priority	15
	Ignore Other SMs	ignore_other_sm	FALSE (disablec
	Polling Timeout	sminfo_polling_timeout	10
	Polling Retries	polling_retry_number	4
	Honor GUID-to-LID File	honor_guid2lid_file	FALSE (disabled)
	Allowed SM GUID list	allowed_sm_guids	(null) (disabled)

Category	Property	Config File Attribute	Default
MAD handling	Max Wire SMPs	max_wire_smps	32
		max_wire_smps2	32
		max_wire_smps_per_device	2
	Transaction Timeout	transaction_timeout	200
	Transaction Timeout	long_transaction_timeout	1000
	Transaction Retries	transaction_retries	3
	Max Message FIFO Timeout	max_msg_fifo_timeout	10000
	Max Message FIFO Length	max_msg_fifo_len	20000

Category	Property	Config File Attribute	Default
Logging	Log File	log_file	/opt/ufm/files/l
	Log Flags	log_flags	Error and Info 0x03
	Force Log Flush	force_log_flush	FALSE (disabled)
	Log Max Size	log_max_size	4096
	Accumulate Log File	accum_log_file	TRUE (enabled)
	Dump Files Directory	dump_files_dir	/opt/ufm/files/l
	Syslog log	syslog_log	0x0
	Dump tables	dump_ar	FALSE

Category	Property	Config File Attribute	Default
Misc	Node Names Map File	node_name_map_name	(null)
	SA database File	sa_db_file	(null)
	Client Reregistration	no_clients_rereg	FALSE (disabled)
		client_rereg_mode	2
	Exit On Fatal Event	exit_on_fatal	TRUE (enabled)
	Enable NVIDIA SHARP support	sharp_enabled	2

Category	Property	Config File Attribute	Default
Multicast	Disable Multicast	disable_multicast	FALSE (disabled)
	Multicast Group Parameters	default_mcg_mtu	0
	Multicast Group Parameters	default_mcg_rate	0
	MC root file	mc_roots_file	(null)
	Incremental Multicast Routing (IMR)	enable_inc_mc_routing	TRUE

Category	Property	Config File Attribute	Default
QoS	Settings	qos	TRUE
	Settings	# QoS default options qos_max_vls 4qos_high_limit 0qos_vlarb_high 0:0qos_vlarb_low 0:160,1:112qos_sl2vl 0,1,1,1,1,1,1,1,15,15,15,15,15,15,15,15 # QoS CA options qos_ca_max_vls 0qos_ca_high_limit -1qos_ca_vlarb_high (null)qos_ca_vlarb_low (null)qos_ca_sl2vl (null) # QoS Switch Port 0 options qos_sw0_max_vls 0qos_sw0_high_limit -1qos_sw0_vlarb_high (null)qos_sw0_vlarb_low (null)qos_sw0_sl2vl (null) # QoS Switch external ports options qos_swe_max_vls 0qos_swe_high_limit -1qos_swe_vlarb_high (null)qos_swe_vlarb_low (null)qos_swe_sl2vl (null) # QoS Switch-to-switch external port options qos_sw2sw_max_vls 0qos_sw2sw_high_limit -1qos_sw2sw_vlarb_high (null)qos_sw2sw_vlarb_low (null)qos_sw2sw_sl2vl (null) # QoS Router ports options qos_rtr_max_vls 0qos_rtr_high_limit -1qos_rtr_vlarb_high (null)qos_rtr_vlarb_low (null)qos_rtr_sl2vl (null)	
	Maximal Operational VL	max_op_vls	2

Category	Property	Config File Attribute	Default
	QoS Policy	qos_policy_file	/opt/ufm/files/c policy.conf
Unhealthy Ports	Enabling Unhealthy Ports	hm_unhealthy_ports_checks	TRUE
	Configuration file	hm_ports_health_policy_file	(null)
	Unhealthy actions	hm_sw_manual_action	no_discover
	MADs validation	validate_smp	TRUE

Category	Property	Config File Attribute	Default
Routing	Unicast Routing Engine	routing_engine	ar_updn
	Root GUIDs file	root_guid_file	/opt/ufm/files/c
	Unicast Routing Caching	use_ucast_cache	TRUE

Category	Property	Config File Attribute	Default
	Adaptive routing	ar_sl_mask	0xFFFF
		enable_ar_by_device_cap	TRUE
		enable_ar_group_copy	TRUE
		ar_mode	3
		ar_transport_mask	0x000A
		cache_ar_group_id	TRUE

Category	Property	Config File Attribute	Default
	Adaptive Routing in Asymmetric Tree topologies	ar_tree_asymmetric_flow	1
		ar_tree_asymmetric_flow_threshold	0
		ar_tree_asymmetric_flow_threshold_limit	2
		routing_flags	0x0

Category	Property	Config File Attribute	Default
	SHIELD/PFRN	shield_mode	3
		pfrn_sl	4
		pfrn_mask_clear_timeout	180
		pfrn_mask_force_clear_timeout	720
		pfrn_over_router_enabled	2

Category	Property	Config File Attribute	Default
	Held back switches	held_back_sw_file	(null)
	GUID Ordering During Routing	guid_routing_order_file	(null)
	Torus Routing	torus_config	/opt/ufm/files/c 2QoS.con
	Routing Chains	pgrp_policy_file	(null)
		topo_policy_file	(null)
		rch_policy_file	(null)
		max_topologies_per_sw	4

Category	Property	Config File Attribute	Default
	Hash based Forwarding	hbf_sl_mask	0xFFFF
		hbf_hash_type	0
		hbf_seed_typ	0
		hbf_seed	0xFFFFFFFF
		hbf_hash_fields	0x0000000040

Category	Property	Config File Attribute	Default
		hbf_weights	auto
	Randomization	scatter_ports	8
		guid_routing_order_no_scatter	TRUE
		use_scatter_for_switch_lid	FALSE
	updn lid tracking mode	updn_lid_tracking_mode	FALSE
Events	Event Subscription Handling	drop_subscr_on_report_fail	TRUE
		drop_event_subscriptions	TRUE
		drop_unreachable_event_subscriptions	TRUE

Category	Property	Config File Attribute	Default
Virtualization	Virtualization enabled	virt_enabled	2
	Maximum ports in virtualization process	virt_max_ports_in_process	64
Router	Router aguid enable	rtr_aguid_enable	0 (Disabled)
	Router path record flow label	rtr_pr_flow_label	0
	Router path record tclass	rtr_pr_tclass	0
	Router path record sl	rtr_pr_sl	0
	Router path record MTU	rtr_pr_mtu	4 (IB_MTU_LEN
	Router path record rate	rtr_pr_rate	16 (IB_PATH_RE

Category	Property	Config File Attribute	Default
SA Security	SA Tnhanced Trust Model (SAETM)	sa_enhanced_trust_model	FALSE
	Untrusted GuidInfo records	sa_etm_allow_untrusted_guidinfo_rec	FALSE
	Guidinfo record requests by VF	sa_etm_allow_guidinfo_rec_by_vf	FALSE
	Untrusted proxy requests	sa_etm_allow_untrusted_proxy_requests	FALSE
	Max number of multicast groups	sa_etm_max_num_mcgs	128
	Max number of service records	sa_etm_max_num_srvcs	32
	Max number of event subscriptions	sa_etm_max_num_event_subs	32
	SGID spoofing	sa_check_sgid_spoofing	TRUE
	Topo config	topo_config_file	(null)
		topo_config_enabled	FALSE
	Tenant Manager	tenants_policy_enabled	FALSE
		tenants_policy_file	/opt/ufm/files/c policy.conf

Category	Property	Config File Attribute	Default
Application performance	Fabric profile	fabric_mode_profile	none
		fabric_mode_policy_file	/opt/ufm/files/c policy.conf
Congestion Control	Congestion Control	mlnx_congestion_control	0
		congestion_control_policy_file	/opt/ufm/files/c
		ppcc_algo_dir	/opt/ufm/files/c
		cc_max_outstanding_mads	500

Category	Property	Config File Attribute	Default
Statistics/ Performance Logging	SM Statistics	osm_stats_interval	60
		osm_stats_dump_limit	20
		osm_stats_dump_per_sm_port	TRUE
	SM Performance Logging	osm_perflog_dump_limit	20
		enable_performance_logging	TRUE

Note

Modifying the following parameters does not require a system manager (SM) restart and will not toggle fabric ports:

- Qos_high_limit (for HCA only)
- Qos_vlarb_high (for HCA only)
- Qos_vlarb_low
- Qos_sl2vl

However, incorrect VLARB or SL2VL settings can impact application performance or lead to traffic failure.

Configuring UFM for SR-IOV

Single-root I/O virtualization (SR-IOV) enables a PCI Express (PCIe) device to appear to be multiple separate physical PCIe devices.

UFM is ready to work with SR-IOV devices by default. You can fine-tune the configuration using the SM configuration.

The following arguments are available for ConnectX-5 and later devices:

Argument	Value	Description
virt_enabled	• 0 – no virtualization support • 1 – disable virtualization on all virtualization supporting ports • 2 – enable virtualization on all virtualization supporting ports (default)	Virtualization support
virt_max_ports_in_process	Possible values: 0-65535; where 0 processes all pending ports Default: 64	Maximum number of ports to be processed simultaneously by the virtualization manager
virt_default_hop_limit	Possible values: 0-255 Default: 2	Default value for hop limit to be returned in path records where either the source or destination are virtual ports

Isolating Switch From Routing

UFM can isolate particular switches from routing in order to perform maintenance of the switches with minimal interruption to the existing traffic in the fabric.

Isolating a switch from routing is done via UFM Subnet Manager as follows:

1. Create a file that includes either the node GUIDs or system GUID of the switches under maintenance. For example:

```
0x1234566
0x1234567
```

2. Set the filename of the parameter held_back_sw_file in the /conf/opensm.conf file (the same as the file created in Step 1).

3. Run:

```
kill -s HUP 'pidof opensm'
```

Once SM completes rerouting, the traffic does not go through the ports of isolated switches.

To attach the switch to the routing:

1. Remove the GUID of the switch from the list of isolated switches defined in Step 1 of the isolation process.

2. Run:

```
kill -s HUP 'pidof opensm'
```

Once SM completes rerouting, traffic will go through the switch.

Appendix – Partitioning

Partitioning enforces isolation of the fabric. The default partition is created on all managed devices. Devices that are running an SM, all switches, routers, and gateways are added to the default partition with full membership. By default, all the HCA ports are also added to the default partition with FULL membership.

Partitioning is provisioned to the Subnet Manager via the partitions.conf configuration file, which cannot be removed or manually modified.

Note

For those who use NVIDIA gateway systems, for proper system functionality, disable the automatic partitioning by changing the attribute gateway_port_partitioning = none in the /opt/ufm/files/conf/gv.cfg configuration. Restart UFM for the change to take effect.

If required, you can add an extension to the *partitions.conf* file that is generated by UFM. You can edit the file, */opt/ufm/files/conf/partitions.conf.user_ext,* and the content of this extension file will be added to the *partitions.conf* file. Files synchronization is done by UFM on every logical model change. However, it can also be triggered manually by running the */opt/ufm/scripts/sync_partitions_conf.sh* script. The script validates and merges the */opt/ufm/files/conf/partitions.conf.user_ext* file into the */opt/ufm/files/conf/opensm/partitions.conf* file and starts the heavy sweep on the Subnet Manager.

Note

The maximum length of the line in the partitions.conf file is 4096 characters. However, to enable long PKeys, it is possible to split the pkey membership to multiple lines:

IOPartition=0x4, ipoib, sl=0, defmember=full : <port-guid1> , <port-guid2> ;

IOPartition=0x4, ipoib, sl=0, defmember=full : <port-guid3> , <port-guid4> ;

The *partitions.conf.user_ext* uses the same format as the *partitions.conf* file. See SM Partitions.conf File Format for the format of the *partitions.conf* file.

For example, to add server ports to PKey 4:

```
IOPartition=0x4, ipoib, sl=0, defmember=full : 0x8f10001072a41;
```

SM Partitions.conf File Format

This appendix presents the content and format of the SM *partitions.conf* file.

```
OpenSM Partition configuration
==============================

The default partition will be created by OpenSM unconditionally
even
when partition configuration file does not exist or cannot be
accessed.

The default partition has P_Key value 0x7fff. OpenSM's port will
always
have full membership in default partition. All other end ports
will have
full membership if the partition configuration file is not found
or cannot
be accessed, or limited membership if the file exists and can be
accessed
but there is no rule for the Default partition.

Effectively, this amounts to the same as if one of the following
rules
below appear in the partition configuration file:
In the case of no rule for the Default partition:
Default=0x7fff : ALL=limited, SELF=full ;
In the case of no partition configuration file or file cannot be
accessed:
Default=0x7fff : ALL=full ;


File Format
===========

Comments:

Line content followed after \'#\' character is comment and
ignored by
```

```
parser.

General file format:

<Partition Definition>:[<newline>]<Partition Properties>;

     Partition Definition:
       [PartitionName][=PKey][,ipoib_bc_flags]
[,defmember=full|limited]

        PartitionName  - string, will be used with logging. When
omitted
                         empty string will be used.
        PKey           - P_Key value for this partition. Only low
15 bits will
                         be used. When omitted will be
autogenerated.
        ipoib_bc_flags - used to indicate/specify IPoIB
capability of this partition.

        defmember=full|limited - specifies default membership for
port guid
                        list. Default is limited.

     ipoib_bc_flags:
        ipoib_flag|[mgroup_flag]*

        ipoib_flag - indicates that this partition may be used
for IPoIB, as
                     a result the IPoIB broadcast group will be
created with
                     the flags given, if any.

     Partition Properties:
       [<Port list>|<MCast Group>]* | <Port list>
```

```
     Port list:
        <Port Specifier>[,<Port Specifier>]

     Port Specifier:
        <PortGUID>[=[full|limited]]

        PortGUID         - GUID of partition member EndPort.
Hexadecimal
                           numbers should start from 0x, decimal
numbers
                           are accepted too.
        full or limited  - indicates full or limited membership
for this
                           port.  When omitted (or unrecognized)
limited
                           membership is assumed.

     MCast Group:
        mgid=gid[,mgroup_flag]*<newline>

                    - gid specified is verified to be a Multicast
address
                      IP groups are verified to match the rate
and mtu of the
                      broadcast group.  The P_Key bits of the
mgid for IP
                      groups are verified to either match the
P_Key specified
                      in by "Partition Definition" or if they are
0x0000 the
                      P_Key will be copied into those bits.

     mgroup_flag:
        rate=<val>  - specifies rate for this MC group
                      (default is 3 (10GBps))
        mtu=<val>   - specifies MTU for this MC group
```

```
                        (default is 4 (2048))
        sl=<val>    - specifies SL for this MC group
                        (default is 0)
        scope=<val> - specifies scope for this MC group
                        (default is 2 (link local)).  Multiple
scope settings
                        are permitted for a partition.
                        NOTE: This overwrites the scope nibble of
the specified
                              mgid.  Furthermore specifying
multiple scope
                              settings will result in multiple MC
groups
                              being created.
        qkey=<val>      - specifies the Q_Key for this MC group
                          (default: 0x0b1b for IP groups, 0 for
other groups)
                          WARNING: changing this for the
broadcast group may
                                   break IPoIB on client nodes!!!
        tclass=<val>    - specifies tclass for this MC group
                          (default is 0)
        FlowLabel=<val> - specifies FlowLabel for this MC group
                          (default is 0)

     newline: '\n'


Note that values for rate, mtu, and scope, for both partitions
and multicast
groups, should be specified as defined in the IBTA specification
(for example,
mtu=4 for 2048).

There are several useful keywords for PortGUID definition:
```

```
 - 'ALL' means all end ports in this subnet.
 - 'ALL_CAS' means all Channel Adapter end ports in this subnet.
 - 'ALL_SWITCHES' means all Switch end ports in this subnet.
 - 'ALL_ROUTERS' means all Router end ports in this subnet.
 - 'SELF' means subnet manager's port.

Empty list means no ports in this partition.



Notes:
-----

White space is permitted between delimiters ('=', ',',':',';').

PartitionName does not need to be unique, PKey does need to be
unique.
If PKey is repeated then those partition configurations will be
merged
and first PartitionName will be used (see also next note).

It is possible to split partition configuration in more than one
definition, but then PKey should be explicitly specified
(otherwise
different PKey values will be generated for those definitions).



Examples:
--------

 Default=0x7fff : ALL, SELF=full ;
 Default=0x7fff : ALL, ALL_SWITCHES=full, SELF=full ;

 NewPartition , ipoib : 0x123456=full, 0x3456789034=limited,
0x2134af2306 ;
```

```
 YetAnotherOne = 0x300 : SELF=full ;
 YetAnotherOne = 0x300 : ALL=limited ;

 ShareIO = 0x80 , defmember=full : 0x123451, 0x123452;
 # 0x123453, 0x123454 will be limited
 ShareIO = 0x80 : 0x123453, 0x123454, 0x123455=full;
 # 0x123456, 0x123457 will be limited
 ShareIO = 0x80 : defmember=limited : 0x123456, 0x123457,
0x123458=full;
 ShareIO = 0x80 , defmember=full : 0x123459, 0x12345a;
 ShareIO = 0x80 , defmember=full : 0x12345b, 0x12345c=limited,
0x12345d;

 # multicast groups added to default
 Default=0x7fff,ipoib:
        mgid=ff12:401b::0707,sl=1 # random IPv4 group
        mgid=ff12:601b::16    # MLDv2-capable routers
        mgid=ff12:401b::16    # IGMP
        mgid=ff12:601b::2     # All routers
        mgid=ff12::1,sl=1,Q_Key=0xDEADBEEF,rate=3,mtu=2 # random
group
        ALL=full;


Note:
----

The following rule is equivalent to how OpenSM used to run prior
to the
partition manager:

Default=0x7fff,ipoib:ALL=full;
```

Appendix – Enhanced Quality of Service

Enhanced QoS provides a higher resolution of QoS at the service level (SL). Users can configure rate limit values per SL for physical ports, virtual ports, and port groups, using enhanced_qos_policy_file configuration parameter.

Valid values of this parameter:

- Full path to the policy file through which Enhanced QoS Manager is configured
- "null" – to disable the Enhanced QoS Manager (default value)

> **Note**
>
> To enable Enhanced QoS Manager, QoS must be enabled in SM configuration file.

Enhanced QoS Policy File

The policy file is comprised of two sections:

- **BW_NAMES**: Used to define bandwidth setting and name (currently, rate limit is the only setting). Bandwidth names are defined using the syntax:

```
<name> = <rate limit in 1Mbps units>
```

Example:

```
My_bandwidth = 50
```

- **BW_RULES**: Used to define the rules that map the bandwidth setting to a specific SL of a specific GUID. Bandwidth rules are defined using the syntax:

```
<guid>|<port group name> = <sl id>:<bandwidth name>, <sl id>:
<bandwidth name>…
```

Examples:

```
0x2c90000000025 = 5:My_bandwidth, 7:My_bandwidth
Port_grp1 = 3:My_bandwidth, 9:My_bandwidth
```

Notes

- **Rate limit = 0** represents unlimited rate limit.
- Any unspecified SL in a rule will be set to 0 (unlimited) rate limit automatically.
- “default” is a well-known name which can be used to define a default rule used for any GUID with no defined rule (If no default rule is defined, any GUID without a specific rule will be configured with unlimited rate limit for all SLs).
- Failure to complete policy file parsing leads to an undefined behavior. User must confirm no relevant error messages in SM log in order to ensure Enhanced QoS Manager is configured properly.
- An empty file with only ‘BW_NAMES’ and ‘BW_RULES’ keywords configures the network with an unlimited rate limit.
- The VPORT_BW_RULES section is optional and includes virtual port GUIDs only (including the vport0 GUID). Physical port GUIDs added to this section are treated as vport0 GUIDs.

Policy File Example

The below is an example of configuring all ports in the fabric with rate limit of 50Mbps on SL1, except for GUID 0x2c90000000025, which is configured with rate limit of 100Mbps on SL1. In this example, all SLs (other than SL1) are unlimited.

```
------------------------------------------------------------------
------
BW_NAMES
bw1 = 50
bw2 = 100
BW_RULES
default: 1:bw1
0x2c90000000025: 1:bw2
------------------------------------------------------------------
-------
VPORT_BW_RULES
default = all:DEF_BW_2
```

Appendix – Routing Chains

The routing chains feature is offering a solution that enables one to configure different parts of the fabric and define a different routing engine to route each of them. The routings are done in a sequence (hence the name "chains") and any node in the fabric that is configured in more than one part is left with the last routing engine updated for it.

Configuring Routing Chains

The configuration for the routing chains feature consists of the following steps:

1. Define the port groups.
2. Define topologies based on previously defined port groups.
3. Define configuration files for each routing engine.
4. Define routing engine chains over defined topologies.

Defining Port Groups

The basic idea behind the port groups is the ability to divide the fabric into sub-groups and give each group an identifier that can be used to relate to all nodes in this group. The port groups are used to define the participants in each of the routing algorithms.

Defining Port Group Policy File

In order to define a port group policy file, set the parameter 'pgrp_policy_file' in the opensm configuration file, as follows:

```
/opt/ufm/files/conf/opensm/port_groups_policy_file.conf
```

Configuring Port Group Policy

The port groups policy file details the port groups in the fabric. The policy file should be composed of one or more paragraphs that define a group. Each paragraph should begin with the line 'port-group' and end with the line 'end-port-group'.

For example:

```
port-group
…port group qualifiers…
end-port-group
```

Port Group Qualifiers

Note

Unlike the port group's begining and ending which do not require a colon, all qualifiers must end with a colon (':'). Also – a colon is a predefined mark that must not be used inside qualifier values. An inclusion of a colon in the name or the use of a port group, will result in the policy's failure.

Table 62: Port Group Qualifiers

Parameter	Description	Example
name	Each group must have a name. Without a name qualifier, the policy fails.	name: grp1
use	'use' is an optional qualifier that one can define in order to describe the usage of this port group (if undefined, an empty string is used as a default).	use: first port group

Rule Qualifiers

There are several qualifiers used to describe a rule that determines which ports will be added to the group. Each port group may contain one or more rules of the rule qualifiers in Table 63 (at least one rule shall be defined for each port group).

Table 63: Rule Qualifiers

Parameter	Description	Example
guid list	Comma separated list of guids to include in the group. If no specific physical ports were configured, all physical ports of the guid are chosen. However, for each guid, one can detail specific physical ports to be included in the group. This can be done using the following syntax: • Specify a specific port in a guid to be chosen port-guid: 0x283@3 • Specify a specific list of ports in a guid to be chosen port-guid: 0x286@1/5/7 • Specify a specific range of ports in a guid to be chosen port-guid: 0x289@2-5 • Specify a list of specific ports and ports ranges in a guid to be chosen port-guid: 0x289@2-5/7/9-13/18 • Complex rule port-guid: 0x283@5-8/12/14, 0x286, 0x289/6/8/12	port-guid: 0x283, 0x286, 0x289
port guid range	It is possible to configure a range of guids to be chosen to the group. However, while using the range qualifier, it is impossible to detail specific physical ports. Note: A list of ranges cannot be specified. The below example is invalid and will cause the policy to fail: port-guid-range: 0x283-0x289, 0x290-0x295	port-guid-range: 0x283-0x289

Parameter	Description	Example
port name	One can configure a list of hostnames as a rule. Hosts with a node description that is built out of these hostnames will be chosen. Since the node description contains the network card index as well, one might also specify a network card index and a physical port to be chosen. For example, the given configuration will cause only physical port 2 of a host with the node description 'kuku HCA-1' to be chosen. port and hca_idx parameters are optional. If the port is unspecified, all physical ports are chosen. If hca_idx is unspecified, all card numbers are chosen. Specifying a hostname is mandatory. One can configure a list of hostname/port/hca_idx sets in the same qualifier as follows: port-name: hostname=kuku; port=2; hca_idx=1 , hostname=host1; port=3, hostname=host2 Note: port-name qualifier is not relevant for switches, but for HCA's only.	port-name: hostname=kuku; port=2; hca_idx=1
port regexp	One can define a regular expression so that only nodes with a matching node description will be chosen to the group	port-regexp: SW.*
	It is possible to specify one physical port to be chosen for matching nodes (there is no option to define a list or a range of ports). The given example will cause only nodes that match physical port 3 to be added to the group.	port-regexp: SW.*:3
union rule	It is possible to define a rule that unites two different port groups. This means that all ports from both groups will be included in the united group.	union-rule: grp1, grp2
subtract rule	One can define a rule that subtracts one port group from another. The given rule, for example, will cause all the ports which are a part of grp1, but not included in grp2, to be chosen. In subtraction (unlike union), the order does matter, since the purpose is to subtract the second group from the first one. There is no option to define more than two groups for union/subtraction. However, one can unite/subtract groups which are a union or a subtraction themselves, as shown in the port groups policy file example.	subtract-rule: grp1, grp2

Predefined Port Groups

There are 3 predefined port groups that are available for use, yet cannot be defined in the policy file (if a group in the policy is configured with the name of one of these predefined groups, the policy fails) –

- ALL – a group that includes all nodes in the fabric
- ALL_SWITCHES – a group that includes all switches in the fabric.
- ALL_CAS – a group that includes all HCA's in the fabric.

Port Groups Policy Examples

```
port-group
name: grp3
use: Subtract of groups grp1 and grp2
subtract-rule:  grp1, grp2
end-port-group

port-group
name: grp1
port-guid: 0x281, 0x282, 0x283
end-port-group

port-group
name: grp2
port-guid-range: 0x282-0x286
port-name: hostname=server1 port=1
end-port-group

port-group
name: grp4
port-name: hostname=kika port=1 hca_idx=1
end-port-group

port-group
name: grp3
union-rule: grp3, grp4
end-port-group
```

Defining Topologies Policy File

In order to define a port group policy file, set the parameter ‘topo_policy_file’ in the opensm configuration file.

```
/opt/ufm/files/conf/opensm/topo_policy_file.conf
```

Configuring Topology Policy

The topologies policy file details a list of topologies. The policy file should be composed of one or more paragraphs which define a topology. Each paragraph should begin with the line 'topology' and end with the line 'end-topology'.

For example:

```
topology
…topology qualifiers…
end-topology
```

Topology Qualifiers

> **Note**
>
> Unlike topology and end-topology which do not require a colon, all qualifiers must end with a colon (':'). Also – a colon is a predefined mark that must not be used inside qualifier values. An inclusion of a column in the qualifier values will result in the policy's failure.

All topology qualifiers are mandatory. Absence of any of the below qualifiers will cause the policy parsing to fail.

Parameter	Description	Example
id	Topology ID. Legal Values – any positive value. Must be unique.	id: 1

Parameter	Description	Example
sw-grp	Name of the port group that includes all switches and switch ports to be used in this topology.	sw-grp: some_switches
hca-grp	Name of the port group that includes all HCA's to be used in this topology.	hca-grp: some_hosts

Configuration File per Routing Engine

Each engine in the routing chain can be provided by its own configuration file. Routing engine configuration file is the fraction of parameters defined in the main opensm configuration file.

Some rules should be applied when defining a particular configuration file for a routing engine:

- Parameters that are not specified in specific routing engine configuration file are inherited from the main opensm configuration file.
- The following configuration parameters are taking effect only in the main opensm configuration file:
- qos and qos_* settings like (vl_arb, sl2vl, etc.)
- lmc
- routing_engine

Defining Routing Chain Policy File

In order to define a port group policy file, set the parameter 'rch_policy_file' in the opensm configuration file, as follows:

```
/opt/ufm/files/conf/opensm/routing_chains_policy.conf
```

First Routing Engine in Chain

The first unicast engine in a routing chain must include all switches and HCA's in the fabric (topology id must be 0). The path-bit parameter value is path-bit 0 and it cannot be changed.

Configuring Routing Chains Policy

The routing chains policy file details the routing engines (and their fallback engines) used for the fabric's routing. The policy file should be composed of one or more paragraphs which defines an engine (or a fallback engine). Each paragraph should begin with the line 'unicast-step' and end with the line 'end-unicast-step'.

For example:

```
unicast-step
…routing engine qualifiers…
end-unicast-step
```

Routing Engine Qualifiers

> **Note**
>
> Unlike unicast-step and end-unicast-step which do not require a colon, all qualifiers must end with a colon (':'). Also – a colon is a predefined mark that must not be used inside qualifier values. An inclusion of a colon in the qualifier values will result in the policy's failure.

Parameter	Description	Example
id	'id' is mandatory. Without an id qualifier for each engine, the policy fails. • Legal values – size_t value (0 is illegal). • The engines in the policy chain are set according to an ascending id order, so it is highly crucial to verify that the id that is given to the engines match the order in which you would like the engines to be set.	is: 1
engine	This is a mandatory qualifier that describes the routing algorithm used within this unicast step. Currently, on the first phase of routing chains, legal values are minhop/ftree/updn.	engine: minhop
use	This is an optional qualifier that enables one to describe the usage of this unicast step. If undefined, an empty string is used as a default.	use: ftree routing for cluster 1
config	This is an optional qualifier that enables one to define a separate opensm config file for a specific unicast step. If undefined, all parameters are taken from main opensm configuration file.	config: /etc/config/opensm2.cfg
topology	Define the topology that this engine uses. • Legal value – id of an existing topology that is defined in topologies policy (or zero that represents the entire fabric and not a specific topology). • Default value – If unspecified, a routing engine will relate to the entire fabric (as if topology zero was defined). • Notice: The first routing engine (the engine with the lowest id) MUST be configured with topology: 0 (entire fabric) or else, the routing chain algorithm will fail.	topology: 1

Parameter	Description	Example
fallback-to	This is an optional qualifier that enables one to define the current unicast step as a fallback to another unicast step. This can be done by defining the id of the unicast step that this step is a fallback to. • If undefined, the current unicast step is not a fallback. • If the value of this qualifier is a non-existent engine id, this step will be ignored. • A fallback step is meaningless if the step it is a fallback to did not fail. • It is impossible to define a fallback to a fallback step (such definition will be ignored)	-
path-bit	This is an optional qualifier that enables one to define a specific lid offset to be used by the current unicast step. Setting lmc > 0 in main opensm configuration file is a prerequisite for assigning specific path-bit for the routing engine. Default value is 0 (if path-bit is not specified)	Path-bit: 1

Dump Files per Routing Engine

Each routing engine on the chain will dump its own data files if the appropriate log_flags is set (for instance 0x43).

- The files that are dumped by each engine are:
    - opensm-lid-matrix.dump
    - opensm-lfts.dump
    - opensm.fdbs
    - opensm-subnet.lst

These files should contain the relevant data for each engine topology.

Note

sl2vl and mcfdbs files are dumped only once for the entire fabric and NOT by every routing engine.

- Each engine concatenates its ID and routing algorithm name in its dump files names, as follows:
  - opensm-lid-matrix.2.minhop.dump
  - opensm.fdbs.3.ftree
  - opensm-subnet.4.updn.lst
- If a fallback routing engine is used, both the routing engine that failed and the fallback engine that replaces it, dump their data.

If, for example, engine 2 runs ftree and it has a fallback engine with 3 as its id that runs minhop, one should expect to find 2 sets of dump files, one for each engine:

- opensm-lid-matrix.2.ftree.dump
- opensm-lid-matrix.3.minhop.dump
- opensm.fdbs.2.ftree
- opensm.fdbs.3.munhop

Appendix – Adaptive Routing

> **Note**
>
> As of UFM v6.4, Adaptive Routing plugin is no longer required for Adaptive Routing and SHIELD configuration. AR is now part of the core Subnet Manager implementation. However, upgrading UFM to v6.4 from an earlier version using the AR plugin will remain possible.

Important Notes:

- For XDR topologies, only the `ar_updn` routing mode is supported.
- The `ar_tree_asymmetric_flow` parameter accepts the following values: `0`, `1`, or `3`.
- The `root_guid` file must be provided by the customer.
- The `ar_ftree` routing mode is no longer supported for XDR topologies.

For further information on how to set up AR and SHIELD, please refer to Adaptive routing, HBF and SHIELD.

Appendix – Security Features

SA Key

SA Key is used to identify trusted SA requests.

Configuration Parameters

Parameter	Description
sa_key	Key for trusted SA requests as defined in IB specifications. Must be non-zero.

SA Enhanced Trust Model (SAETM)

Standard SA has a concept of trust-based requests on the SA_Key that is part of each SA MAD. A **trusted request** is when the SA_Key value is not equal to zero but equals the SA configured value, while an **untrusted request** is when the SA_Key value equals zero in the request. If a request has a non-zero SA_Key value that is different from the configured SA key, it will be dropped and reported.

When SAETM is enabled, the SA limits the set of untrusted requests allowed. Untrusted requests that are not allowed according to SAETM will be silently dropped (for the set of untrusted requests allowed, see the following section below).

SAETM feature is disabled by default. To enable it, set the sa_enhanced_trust_model parameter to TRUE.

Additional SAETM Configuration Parameters

Parameter	Description
sa_etm_allow_untrusted_guidinfo_rec	Defines whether to allow GUIDInfoRecord as part of the SAETM set of untrusted requests allowed (see section below)
sa_etm_allow_guidinfo_rec_by_vf	Defines whether to drop GUIDInfoRecord from non-physical ports (see section below)
sa_etm_allow_untrusted_proxy_requests	Defines the behavior for proxy requests (see section below)
sa_etm_max_num_mcgs/ sa_etm_max_num_srvcs/ sa_etm_max_num_event_subs	Defines the registration limits in SAETM (see section below)
sa_check_sgid_spoofing	Enables SGID spoofing checks.

List of Untrusted SA Requests

The following table lists the untrusted requests allowed when SAETM is enabled:

Request	Request Type
`MCMemberRecord`	Get/Set/Delete
`PathRecord`	Get
`PathRecord`	GetTable (only if both destination and source are specified (e,g. only point to point))

Request	Request Type
`ServiceRecord`	Get/Set/Delete
`ClassPortInfo`	Get
`InformInfo`	Set (for non-SM security traps)
`GUIDInfoRecord`	Set/Delete – this request can only be part of this set depending on the values of sa_etm_allow_untrusted_guidinfo_rec and sa_etm_allow_guidinfo_rec_by_vf – see elaboration below.

Notes:

- When `sa_etm_allow_untrusted_guidinfo_rec` is set to FALSE (and SAETM is enabled), the SA will drop GUIDInfoRecord Set/Delete untrusted requests.

- When `sa_etm_allow_guidinfo_rec_by_vf` is set to FALSE (and SAETM is enabled), the SA will drop GUIDInfoRecord Set/Delete requests from non-physical ports.

- In case `sa_etm_allow_untrusted_guidinfo_rec` is FALSE, GUIDInfoRecord Set/Delete requests will become part of the SAETM set of untrusted requests allowed.

- When `sa_etm_allow_guidinfo_rec_by_vf` is FALSE, the requests will only be allowed from physical ports.

Proxy SA Requests

SA modification request (SET/DELETE) is identified as a proxy operation when the port corresponding with the requester source address (SLID from LRH / SGID from GRH) is diffident than the port for which the request applies:

- For `MCMemberRecord`, when the `MCMemberRecord.PortGID` field does not match the requester address.

  For `ServiceRecord`, when the `ServiceRecord.ServiceGID` field does not match requester address.

  For the `GUIDInfoRecord`, when the LID field in the RID of the record does not match the requester address.

When `sa_etm_allow_untrusted_proxy_requests` is set to FALSE and SAETM is enabled, untrusted proxy requests will be dropped.

Registration Limits

When any of `sa_etm_max_num_mcgs`, `sa_etm_max_num_srvcs` or `sa_etm_max_num_event_subs` parameters is set to 0, the number of this parameter's registrations can be unlimited. When the parameter's value is different than 0, attempting to exceed the maximum number of registrations will result in the request being silently dropped. Consequently, the requester and request info will be logged, and an event will be generated for the Activity Manager.

The following parameters control the maximum number of registrations:

Parameter	Description
`sa_etm_max_num_mcgs`	Maximum number of multicast groups per port/vport that can be registered.
`sa_etm_max_num_srvcs`	Maximum number of service records per port/vport that can be registered.
`sa_etm_max_num_event_subs`	Maximum number of event subscriptions (InformInfo) per port/vport that can be registered.

SAETM Logging

When requesting an operation that is not part of the SAETM set of untrusted requests, it will be silently dropped and eventually written to the SM log.

The logging of the dropped MADs is repressed to not overload the OpenSM log. If the request that needs to be dropped was received from the same requester many times consecutively, OpenSM logs it only if the request number is part of the following sequence:

0, 1, 2, 5, 10, 20, 50, 100, 200... (similar to the trap log repression).

SGID Spoofing

SA can validate requester addresses by comparing the SLID and SGID of the incoming request. SA determines the requester port by the SLID and SGID field of the request. SGID spoofing is when the SGID and SLID do not match.

When `sa_check_sgid_spoofing` parameter is enabled, SA checks for SGID spoofing in every request that includes GRH, unless the SLID belongs to a router port in that same request. In case the request SGID does not match its SLID, the request will be dropped. The default value of this parameter is TRUE.

Configuration File Parameters - Value Example

```
```
sa_enhanced_trust_model FALSE
sa_etm_allow_untrusted_proxy_requests FALSE
sa_etm_allow_untrusted_guidinfo_rec FALSE
sa_etm_allow_guidinfo_rec_by_vf FALSE
sa_etm_max_num_mcgs 128
sa_etm_max_num_srvcs 32
sa_etm_max_num_event_subs 32
sa_rate_threshold 0
sa_check_sgid_spoofing TRUE
```
```

Service Key

The `ServiceKey` feature may be used to allow authentication of the creation, replacement and deletion of ServiceRecords with selected ServiceNames, according to IB Specification.

SM Support mapping service name to service key using a file.

The path to service name to service key map file is set using *service_name2key_map_file* parameter in SM configuration file.

Each line in the file contains mapping from service name to service key (in IPv6 format).

For example:

```
SHArP.AggregationManager 1111:2222:3333:4444:5555:6666:7777:8888
```

M_Key Authentication

M_Key Per Port

This feature increases protection on the fabric as a unique M_Key is generated and set for each HCA, router, or switch port.

OpenSM calculates an M_Key per port by performing a hash function on the port GUID of the device and the M_Key configured in opensm.conf.

To enable M_Key per port, set the parameter below in addition to the parameters listed in the previous section:

```
m_key_per_port TRUE
```

Class Management Keys

This section outlines the Subnet Manager (SM) feature for configuring class management keys.

The supported management classes include:

1. **M_Key** (Subnet Manager classes 0x01 and 0x81)
2. **CC_Key** (Congestion Control class 0x21)
3. **VS_Key** (Vendor Specific class 0x0A)
4. **N2N_Key** (Node-to-Node class 0x0C)

MADs are used to initialize and configure network devices in the subnet, and therefore, are considered privilege operation.

There is a mechanism to authorize management packets based on:

- A key specified in the MAD headers.
- A key stored locally on each HCA/Router port or switch port0 in the subnet.

Authentication is performed by the management entity at the destination port and is achieved by comparing the key contained in the MAD header with the key residing at the destination port.

As such, when the feature is enabled, MADs sent to network devices by diagnostics and monitoring tools should be sent with the correct key.

Note

- When the `M_Key` feature is enabled with `m_key_protection_level` set to 2, both GET and SET MAD operation are required, including the destination port key, meaning the diagnostic and monitoring tools that uses SMP MADs will be unable to operate without the correct destination port key.
- When `m_key_protection_level` is set to 1, only SET MAD operations require including the destination port key, meaning the diagnostic and monitoring tools that uses SMP MADs will be able to operate even without specifying the correct destination port key.
- To disable the mechanism after it was enabled, need to explicitly disable the feature in SM configuration. Once the feature is disabled in SM configuration, SM will disable the feature on the network devices.

M_Key

Subnet Manager Class Parameters:

Parameter	Description	Values
`m_key`	The base M_Key or seed used for generation.	- 0: Disables the feature. - 0xFFFFFFFFFFFFFFFF: Enables use of a random seed to generate M_Keys.- Values from 0x1 to 0xFFFFFFFFFFFFFFFE: Specify an explicit M_Key or seed for generating M_Keys.
`m_key_per_port`	Determines whether each device port receives a unique M_Key.	- TRUE: Enabled. SM uses the m_key value as a seed to generate a distinct M_Key for each port. - FALSE: Disabled. SM uses the m_key value as a uniform M_Key for all devices.
`m_key_lease_period`	Duration (in seconds) of the M_Key lease, as defined by the InfiniBand specification.	Default: 60 seconds.
`m_key_protection_level`	Defines the protection level according to the InfiniBand specification.	No specific values provided (likely a configurable parameter according to InfiniBand specifications).

Notes:

- If `m_key_per_port` is enabled and `m_key_protection_level` is set to `0`, the SM will override it to `2`.
- If `m_key_per_port` is enabled and `m_key` is `0`, the SM will automatically set it to `0xFFFFFFFFFFFFFFFF`.
- If `m_key_per_port` is enabled and `m_key_lease_period` is `0`, the SM will default it to `60` seconds.

Note

Please note that some configuration values in `opensm.conf` may be overwritten by UFM when UFM starts executing.

Specifically, `m_key_per_port` and `m_key` may be overwritten. So, in order to avoid misconfigurations due to potential overwrites, please make sure that these parameters are configured correctly in **gv.cfg** (UFM configuration file).

Set the following configuration in gv.cfg, in [SubnetManager] section:

m_key_per_port = true

global_m_key_seed = 0xFFFFFFFFFFFFFFFF

Warning

If `m_key_per_port` is set to **true**, it is crucial to revert this setting to **false** before uninstalling UFM.

Failing to do so may prevent both the Subnet Manager (SM) and any InfiniBand (IB) utilities from sending MADs over the IB network after a fresh UFM installation.

Common Parameters for CC_Key / VS_Key / N2N_Key

For all management classes *other than* the Subnet Manager class, the `key_mgr_seed` parameter controls the seed used to generate unique keys per device for each class.

Configuration Parameters

Configuration Parameter	Description	Values
`key_mgr_seed`	Seed value for generating class management keys.	- 0xFFFFFFFFFFFFFFFF: Enables use of a random seed for key generation. - Values from 0x1 to 0xFFFFFFFFFFFFFFFE: Specify a fixed seed for key generation.
`gmp_traps_threads_num`	Number of threads dedicated to handling GMP key violation traps.	No specific values provided (likely a configurable integer).

Example Configuration

```
key_mgr_seed 0x0000000000000001
gmp_traps_threads_num 1
```

CC Key

The **CC Key** is a security mechanism for the Congestion Control (CC) Class `0x21`. When enabled, the Subnet Manager assigns a unique CC Key to each port in the subnet that supports both the CC Class and CC Key feature. The extent of protection is controlled by configuration parameters in the SM configuration file.

Once enabled, if a device receives a Management Datagram (MAD) within the protected scope using an incorrect CC Key, it notifies the Subnet Manager of a key violation. (Refer to section **3.1.2:** `cc_key_lease_period` for more details.)

Note: When the CC Key feature is enabled, all CC configuration MADs must carry the correct key. Therefore, Congestion Control can only be configured on a port *after* its CC Key has been set.

Support for CC Key requires Congestion Control to be either disabled or enabled (`mlnx_congestion_control` must be set to `1` or `2`).

Configuration Parameters

Parameter	Description
`cc_key_enable`	Controls whether CC Keys are configured and how: • `0` – Ignore: No key configuration MADs are sent.• `1` – Disable: Ports use `0` for CC Key, lease period, and protection bits (no protection).• `2` – Enable: Each port gets a unique CC Key. Lease period and protection level are applied based on other parameters.
`cc_key_lease_period`	Lease period in seconds (per IB specifications). • On receiving a MAD with an invalid key, the device notifies the Subnet Manager and waits this duration.• If no response, protection expires and a new CC Manager can reconfigure keys.• If response received (TrapRepress + KeyInfo Set), protection continues.• `0` means an infinite lease period.
`cc_key_protect_bit`	Defines the protection level: • `0` – Key is protected, but can be read via KeyInfo GET.• `1` – Full protection; all CC MADs must have the correct key.
`cc_max_outstanding_mads`	Maximum number of CC class MADs allowed in the network at any given time.

Important: These configurations are valid only when non-zero keys are applied to the ports (i.e., `cc_key_enable = 2`). If ports are configured with a zero key, no protection is enforced.

Example

```
```
cc_key_enable 2
cc_key_lease_period 10
cc_key_protect_bit 1
cc_max_outstanding_mads 500
```
```

Vendor Specific (VS) Key

The **VS Key** is a security feature for the Vendor Specific (VS) Class `0xA`. The Subnet Manager assigns a unique VS Key to each port in the subnet that supports the VS Class and the VS Key feature. The scope of protection is defined by parameters in the Subnet Manager's configuration file.

Once the feature is enabled, if a device receives a Management Datagram (MAD) within the protected scope using an incorrect key, it notifies the Subnet Manager of a key violation. (See table below `vs_key_lease_period` for details.)

Configuration Parameters

Parameter	Description
`vs_key_enable`	Controls whether VS Keys are configured: • `0` – Ignore: No key configuration MADs are sent.• `1` – Disable: Ports are set with `0` values for VS Key, lease period, and protection bits (no protection).• `2` – Enable: Each port is configured with a VS Key, lease period, and protection level as defined by related parameters.
`vs_key_lease_period`	Lease period in seconds. When a device receives a MAD with an incorrect key:• It notifies the Subnet Manager and waits for a response for the duration of the lease period.• If no response, key protection expires and a new VS Manager can reconfigure the keys.• If response received (TrapRepress + VS KeyInfo Set), key protection is maintained.• A value of `0` means infinite lease period.
`vs_key_ci_protect_bits`	Defines the VS Key protection level using both the VS protect bit and the protection scope: • `0` – Key is protected, but readable via KeyInfo GET.• `1` – Full protection; all VS MADs must have the correct key.

Note that the above settings are only effective when non-zero keys are configured on the subnet ports (i.e., `vs_key_enable` is set to `2`). Ports with a zero key are not protected.

Example Configuration

```
vs_key_enable 2
vs_key_lease_period 60
vs_key_ci_protect_bits 1
vs_max_outstanding_mads 500
```

Node-to-Node (N2N) Key

The N2N Key is a security feature for the Node-to-Node (N2N) Class `0xC`. The Subnet Manager assigns a unique N2N Key to each port in the subnet that supports both the N2N Class and the N2N Key feature. The protection scope is defined through parameters in the Subnet Manager's configuration file.

Once the feature is enabled, if a device receives a Management Datagram (MAD) within the protected scope that contains an incorrect key, it notifies the Subnet Manager of the key violation.

> **Note**
>
> When the N2N Key feature is enabled, all Node-to-Node configuration MADs must include the correct N2N Key.

Configuration Parameters

Parameter	Description	Values
n2n_key_enable	Determines if N2N key configuration is enabled.	- 0: Ignore – no key configuration MADs are sent. - 1: Disable – configure subnet ports to hold 0 values for N2N key, lease period, and protection bits, meaning no protection is provided.- 2: Enable – configure key for each port. Set lease period and protection level as defined by n2n_key_lease_period, n2n_key_protect_bit, and n2n_key_enable attributes.
n2n_key_lease_period	Lease period in seconds according to IB specifications. Defines how long the device waits for a response after receiving a MAD with an incorrect key.	- If 0: Lease period is infinite. - If a response is received before the period ends, key protection remains.- If no response is received before the period ends, key protection is lost, and new N2N Manager can reconfigure the N2N keys.
n2n_key_protect_bit	Protection level for N2N Keys.	- 0: Protection is provided, but N2N managers can read the key by KeyInfo GET. - 1: Protection is provided for all N2N MADs.
n2n_max_outstanding_mads	Maximum number of N2N class MADs in the network at once.	No specific values provided (likely a configurable parameter).

Please note that the above information is valid only if non-zero keys have been configured for the subnet ports (n2n_key_enable is set to 2). Ports with a zero key do not provide protection.

Example Configuration

```
n2n_key_enable 2
n2n_key_lease_period 10
n2n_key_protect_bit 1
n2n_max_outstanding_mads 500
```

Periodic Key Updates

The Subnet Manager (SM) supports periodic updates of the class management keys for network devices:

- The periodic key update feature applies to all management class keys supported by SM.
- This feature will take effect for management classes that are enabled and only on devices that support dual keys.
- Devices that do not support dual keys will not undergo periodic key updates.
- During each update cycle, SM will generate and assign new keys to all nodes based on a new random seed.
- After each key update, SM will update the relevant files (**guid2mkey, guid2cckey, guid2vskey, guid2_m2n_key, guid2_n2n_key**) with the new keys.
- The generated keys are unique to each port, derived from the SHA512 of:
    - /dev/urandom
    - Port GUID
    - Key type
- Periodic update will take effect per management class, if that class key is enabled. The sections above explain how to enable each key type (N2N, VS, CC, M_Keys).

In order to avoid potentially interfering with currently executing diagnostic tools, after startup, SM waits one interval before changing keys for devices.

Configuration Parameter

Configuration Parameter	Description	Values
periodic_key_update	Time interval in minutes for periodic key updates.	- 0: Disable the feature. - Non-zero: Interval in minutes to regenerate keys.

Example Configuration

```
# Perform a periodic update every 1440 minutes (24 hours), for each
supporting class
periodic_key_update 1440

# Enable

# Use random seed for M_Key
m_key 0xffffffffffffffff
m_key_per_port TRUE
m_key_lease_period 60
m_key_protection_level 2

# Use random seed for VS_Key, CC_Key and N2N_Key
key_mgr_seed 0xffffffffffffffff

# Enable VS_Key
vs_key_enable 2
vs_key_lease_period 60
vs_key_ci_protect_bits 1

# Enable N2N_Key
n2n_key_enable 2
n2n_key_lease_period 60
n2n_key_protect_bit 1

# Congestion control disabled, but CC_Key is enabled
mlnx_congestion_control 1
cc_key_enable 2
cc_key_lease_period 60
cc_key_protect_bit 1
```

Key Violation Trap Logging

A key violation occurs when a port receives a MAD with an incorrect key (e.g., a Class 0xA MAD with an incorrect VS Key in the MAD header). In this case, the port reports the violation to the Subnet Manager (SM), which then sends a TrapRepress and KeyInfo Set (for the relevant key class) to the affected port.

Additionally, the SM logs the key violation event. However, if the same violation is reported consecutively by the same requester, the SM will only log it if the request number falls within the following sequence:

0, 1, 2, 5, 10, 20, 50, 100, 200...

Appendix – SM Activity Report

SM can produce an activity report in a form of a dump file that details the different activities done in the SM. Activities are divided into subjects. The table below specifies the different activities currently supported in the SM activity report.

Reporting of each subject can be enabled individually using the configuration parameter activity_report_subjects:

- Valid values:

  Comma-separated list of subjects to dump. The current supported subjects are:

-
  - "mc" – activity IDs 1, 2 and 8
  - "prtn" – activity IDs 3, 4, and 5
  - "virt" – activity IDs 6 and 7
  - "routing" – activity IDs 8-12

  Two predefined values can be configured as well:

  - "all" – dump all subjects
  - "none" – disable the feature by dumping none of the subjects

- Default value: "none"

SM Supported Activities

Activity ID	Activity Name	Additional Fields	Comments	Description
1	mcm_member	- MLid - MGid - Port Guid - Join State	Join state: 1 - Join -1 - Leave	Member joined/left MC group
2	mcg_change	- MLid - MGid - Change	Change: 0 - Create 1 - Delete	MC group created/deleted
3	prtn_guid_add	- Port Guid - PKey - Block index - Pkey Index		Guid added to partition
4	prtn_create	-PKey - Prtn Name		Partition created
5	prtn_delete	- PKey - Delete Reason	Delete Reason: 0 - empty prtn 1 - duplicate prtn 2 - sm shutdown	Partition deleted
6	port_virt_discover	- Port Guid - Top Index		Port virtualization discovered
7	vport_state_change	- Port Guid - VPort Guid - VPort Index - VNode Guid - VPort State	VPort State: 1 - Down 2 - Init 3 - ARMED 4 - Active	Vport state changed
8	mcg_tree_calc	- mlid		MCast group tree calculated
9	routing_succeed	routing engine name		Routing done successfully

Activity ID	Activity Name	Additional Fields	Comments	Description
10	routing_failed	routing engine name		Routing failed
11	ucast_cache_invalidated			ucast cache invalidated
12	ucast_cache_routing_done			ucast cache routing done

OpenSM Configuration

Enabling Congestion Control

In order to enable congestion control, set the parameter `mlnx_congestion_control` in the OpenSM configuration file to 2. For example:

```
mlnx_congestion_control 2
```

To disable congestion control, set the parameter value to 1. For example:

```
mlnx_congestion_control 1
```

Defining a Congestion Control Policy File

To define a congestion control policy file, set the parameter `congestion_control_file` in OpenSM configuration file to point to congestion control policy file. For example:

```
congestion_control_policy_file
/opt/ufm/files/conf/opensm/conf/congestion_control_policy_file
```

The file includes a reference to an active algorithm file name. The algorithm file has to be inside the `ppcc_algo_dir`.

For Example:

```
ca_algo_import_start
    algo_start
        algo_id:1
        algo_file_name: active_algo_file_name
        # PPCC parameter by name, as defined in algo profile
        parameters: (BW_G,400),(ALPHA,3932),(MAX_DEC,63569),
(MAX_INC,69468),(AI,36),(HAI,1200)
    algo_end
ca_algo_import_end
```

The `ca_algo_import` block contains all the algo blocks that map an `algo_id` to an algorithm profile file. The `algo_id` field of the algo blocks must be unique and start from 1. This block is used to import the various PCC algorithms into the configuration and associate them with their `algo_id` values.

Defining a directory for PPCC algorithm profiles

To define a directory for the programmable congestion control algorithm profiles, set the parameter `ppcc_algo_dir` in OpenSM configuration file. For Example:

```
ppcc_algo_dir /opt/ufm/files/conf/opensm/conf/ppcc_algo_dir
```

Appendix – UFM AI Mode

Starting with UFM Enterprise v6.21.0, UFM is optimized by default to enhance AI workload performance through the SM configuration.

```
#Fabric Mode feature
#Indicates which profile from fabric mode policy file to use.
#If set to none, feature is disabled, and SM will not change
current
#device configuration
fabric_mode_profile ai_mode
```

The Fabric Mode policy file is located at:

```
# The file holding the Fabric Mode policy
fabric_mode_policy_file /opt/ufm/files/conf/opensm/fabric-mode-
policy.conf
```

To disable AI mode in UFM:

1. Set `fabric_mode_profile` to `none`.
2. Signal the SM to apply the updated configuration.

```
#Fabric Mode feature
#Indicates which profile from fabric mode policy file to use.
#If set to none, feature is disabled, and SM will not change
current
#device configuration
fabric_mode_profile none
```

Appendix – Diagnostic Utilities

Note

For UFM-SDN Appliance, all the below diagnostics commands have ib prefix.

For example, for UFM-SDN Appliance, the command `ibstat` is `ib ibstat.`

InfiniBand Diagnostics Commands

Command	Description
ibstat	Shows the host adapters status.
ibstatus	Similar to ibstat but implemented as a script.
ibnetdiscover	Scans the topology.
ibaddr	Shows the LID range and default GID of the target (default is the local port).
ibroute	Displays unicast and multicast forwarding tables of the switches.
ibtracert	Displays unicast or multicast route from source to destination.
ibping	Uses vendor MADs to validate connectivity between InfiniBand nodes. On exit, (IP) ping-like output is shown.
ibsysstat	Obtains basic information for the specific node which may be remote. This information includes: hostname, CPUs, memory utilization.
sminfo	Queries the SMInfo attribute on a node.
smpdump	A general purpose SMP utility which gets SM attributes from a specified SMA. The result is dumped in hex by default.
smpquery	Enables a basic subset of standard SMP queries including the following: node info, node description, switch info, port info. Fields are displayed in human readable format.
perfquery	Dumps (and optionally clears) the performance counters of the destination port (including error counters).
ibswitches	Scans the net or uses existing net topology file and lists all switches.

Command	Description
ibhosts	Scans the net or uses existing net topology file and lists all hosts.
ibnodes	Scans the net or uses existing net topology file and lists all nodes.
ibportstate	Gets the logical and physical port states of an InfiniBand port or disables or enables the port (only on a switch). Note: This tool can change port settings. Should be used with caution.
saquery	Issues SA queries.
ibdiagnet	ibdiagnet scans the fabric using directed route packets and extracts all the available information regarding its connectivity and devices.
ibnetsplit	Automatically groups hosts and creates scripts that can be run to split the network into sub-networks each containing one group of hosts.
Ibqueryerrors	Queries IB spec-defined errors from all fabric ports. Note: This tool can change reset port counters Should be used with caution.
smparquery	Queries adaptive-routing related settings from a particular switch. Note: This tool can change reset port counters Should be used with caution.

Diagnostic Tools

Model of operation: All utilities use direct MAD access to operate. Operations that require QP 0 mads only, may use direct routed mads, and therefore may work even in subnets that are not configured. Almost all utilities can operate without accessing the SM, unless GUID to lid translation is required.

Dependencies

Multiple port/Multiple CA support:

When no InfiniBand device or port is specified (as shown in the following example for "Local umad parameters"), the tools select the interface port to use by the following criteria:

1. The first InfiniBand ACTIVE port.
2. If not found, the first InfiniBand port that is UP (physical link up).

If a port and/or CA name is specified, the **tool** attempts to fulfill the user's request and will fail if it is not possible.

For example:

```
ibaddr          # use the 'best port'
ibaddr -C mthca1        # pick the best port from mthca1 only.
ibaddr -P 2             # use the second (active/up) port from
the first available IB device.
ibaddr -C mthca0 -P 2   # use the specified port only.
```

Common Options & Flags

Most diagnostics take the following flags. The exact list of supported flags per utility can be found in the usage message and can be shown using util_name -h syntax.

```
# Debugging flags
-d   raise the IB debugging level. May be used several times (-
ddd or -d -d -d).
-e   show umad send receive errors (timeouts and others)
-h   show the usage message
-v   increase the application verbosity level.
     May be used several times (-vv or -v -v -v)
-V   show the internal version info.
```

```
# Addressing flags
-D          use directed path address arguments.
            The path is a comma separated list of out ports.
            Examples:
            "0"   # self port
            "0,1,2,1,4"   # out via port 1, then 2, ...
-G          use GUID address arguments.
            In most cases, it is the Port GUID.
            Examples:
            "0x08f1040023"
-s <smlid>  use 'smlid' as the target lid for SA queries.
```

```
# Local umad parameters:
-C <ca_name>     use the specified ca_name.
-P <ca_port>     use the specified ca_port.
-t <timeout_ms>  override the default timeout for the
                 solicited mads.
```

CLI notation: all utilities use the POSIX style notation, meaning that all options (flags) must precede all arguments (parameters).

Utilities Descriptions

ibstatus

A script that displays basic information obtained from the local InfiniBand driver. Output includes LID, SMLID, port state, link width active, and port physical state.

Syntax

```
ibstatus [-h] [devname[:port]]
```

Examples:

```
ibstatus                      # display status of all IB ports
ibstatus mthca1               # status of mthca1 ports
ibstatus mthca1:1 mthca0:2    # show status of specified ports
```

See also: ibstat

ibstat

Similar to the ibstatus utility but implemented as a binary and not as a script. Includes options to list CAs and/or ports.

Syntax

```
ibstat [-d(ebug) -l(ist_of_cas) -p(ort_list) -s(hort)] <ca_name>
[portnum]
```

Examples:

```
ibstat            # display status of all IB ports
ibstat mthca1     # status of mthca1 ports
ibstat mthca1 2   # show status of specified ports
ibstat -p mthca0  # list the port guids of mthca0
ibstat –l         # list all CA names
```

See also: ibstatus

ibroute

Uses SMPs to display the forwarding tables (unicast (LinearForwardingTable or LFT) or multicast (MulticastForwardingTable or MFT)) for the specified switch LID and the optional lid (mlid) range. The default range is all valid entries in the range 1...FDBTop.

Syntax

```
ibroute [options] <switch_addr> [<startlid> [<endlid>]]
```

Nonstandard flags:

```
-a              show all lids in range, even invalid entries.
-n              do not try to resolve destinations.
-M              show multicast forwarding tables. In this case
the range
                parameters are specifying mlid range.
node-name-map   node name map file
```

Examples:

```
ibroute 2            # dump all valid entries of switch lid 2
ibroute 2 15         # dump entries in the range 15...FDBTop.
ibroute -a 2 10 20   # dump all entries in the range 10..20
ibroute -n 2         # simple format
ibroute -M 2         # show multicast tables
```

See also: ibtracert

ibtracert

Uses SMPs to trace the path from a source GID/LID to a destination GID/LID. Each hop along the path is displayed until the destination is reached or a hop does not respond. By using the -m option, multicast path tracing can be performed between source and destination nodes.

Syntax

```
ibtracert [options] <src-addr> <dest-addr>
```

Nonstandard flags:

```
-n              simple format; don't show additional information.
-m <mlid>       show the multicast trace of the specified mlid.
-f <force>      force
node-name-map   node name map file
```

Examples:

```
ibtracert 2 23          # show trace between lid 2 and 23
ibtracert -m 0xc000 3 5 # show multicast trace between lid 3
and 5 for mcast lid 0xc000.
```

smpquery

Enables a basic subset of standard SMP queries including the following node info, node description, switch info, port info. Fields are displayed in human readable format.

Syntax

```
smpquery [options] <op> <dest_addr> [op_params]
```

Currently supported operations and their parameters:

```
nodeinfo <addr>
nodedesc <addr>
portinfo <addr> [<portnum>]   # default port is zero
switchinfo <addr>
pkeys <addr> [<portnum>]
sl2vl <addr> [<portnum>]
vlarb <addr> [<portnum>]
GUIDInfo (GI) <addr>
MlnxExtPortInfo (MEPI) <addr> [<portnum>]
Combined (-c) : use Combined route address argument
node-name-map : node name map file
extended (-x) :  use extended speeds
```

Examples:

```
smpquery nodeinfo 2       # show nodeinfo for lid 2
smpquery portinfo 2 5     # show portinfo for lid 2 port 5
```

smpdump

A general purpose SMP utility that gets SM attributes from a specified SMA. The result is dumped in hex by default.

Syntax

```
smpdump [options] <dest_addr> <attr> [mod]
```

Nonstandard flags:

```
-s       show output as string
```

Examples:

```
smpdump -D 0,1,2 0x15 2       # port info, port 2
smpdump 3 0x15 2              # port info, lid 3 port 2
```

ibaddr

Can be used to show the LID and GID addresses of the specified port or the local port by default. This utility can be used as simple address resolver.

Syntax

```
ibaddr [options] [<dest_addr>]
```

Nonstandard flags:

```
gid_show  (-g) : show gid address only
lid_show   (-l) : show lid range only
Lid_show (-L) : show lid range (in decimal) only
```

Examples:

```
ibaddr                # show local address
ibaddr 2               # show address of the specified port lid
ibaddr -G 0x8f1040023  # show address of the specified port guid
```

sminfo

Issues and dumps the output of an sminfo query in human readable format. The target SM is the one listed in the local port info or the SM specified by the optional SM LID or by the SM direct routed path.

> **⚠ Warning**
>
> CAUTION: Using sminfo for any purpose other than a simple query might result in a malfunction of the target SM.

Syntax

```
sminfo [options] <sm_lid|sm_dr_path> [sminfo_modifier]
```

Nonstandard flags:

```
-s <state>          # use the specified state in sminfo mad
-p <priority>       # use the specified priority in sminfo mad
-a <activity>       # use the specified activity in sminfo mad
```

Examples:

```
sminfo          # show sminfo of SM listed in local portinfo
sminfo 2        # query SM on port lid 2
```

perfquery

Uses PerfMgt GMPs to obtain the PortCounters (basic performance and error counters) from the Performance Management Agent (PMA) at the node specified. Optionally show aggregated counters for all ports of node. Also, optionally, reset after read, or only reset counters.

```
perfquery [options]  [<lid|guid> [[port] [reset_mask]]]
```

Nonstandard flags:

```
-a                      Shows aggregated counters for all ports
of the destination lid.
-r                      Resets counters after read.
-R                      Resets only counters.
Extended (-x)           Shows extended port counters
Xmtsl (-X)              Shows Xmt SL port counters
Rcvsl  ,( -S)           Shows Rcv SL port counters
Xmtdisc  (-D)           Shows Xmt Discard Details
rcverr, (-E)    Shows Rcv Error Details
extended_speeds (-T)    Shows port extended speeds counters
oprcvcounters   Shows Rcv Counters per Op code
flowctlcounters Shows flow control counters
vloppackets     Shows packets received per Op code per VL
vlopdata        Shows data received per Op code per VL
vlxmitflowctlerrors     Shows flow control update errors per VL
vlxmitcounters  Shows ticks waiting to transmit counters per VL
swportvlcong    Shows sw port VL congestion
rcvcc   Shows Rcv congestion control counters
slrcvfecn       Shows SL Rcv FECN counters
slrcvbecn       Shows SL Rcv BECN counters
xmitcc  Shows Xmit congestion control counters
vlxmittimecc    Shows VL Xmit Time congestion control counters
smplctl (-c)    Shows samples control
loop_ports (-l)         Iterates through each port
```

Examples:

```
perfquery               # read local port's performance counters
perfquery 32 1         # read performance counters from lid 32,
port 1
perfquery -a 32        # read from lid 32 aggregated performance
counters
perfquery -r 32 1     # read performance counters from lid 32
port 1 and reset
perfquery -R 32 1     # reset performance counters of lid 32 port
1 only
perfquery -R -a 32    # reset performance counters of all lid 32
ports
perfquery -R 32 2 0xf000  # reset only non-error counters of lid
32 port 2
```

ibping

Uses vendor mads to validate connectivity between InfiniBand nodes. On exit, (IP) ping like output is show. ibping is run as client/server. The default is to run as client. Note also that a default ping server is implemented within the kernel.

Syntax

```
ibping [options] <dest lid|guid>
```

Nonstandard flags:

```
-c <count>      stop after count packets
-f              flood destination: send packets back to back w/o
delay
-o <oui>        use specified OUI number to multiplex vendor MADs
-S              start in server mode (do not return)
```

ibnetdiscover

Performs InfiniBand subnet discovery and outputs a human readable topology file. GUIDs, node types, and port numbers are displayed as well as port LIDs and node descriptions. All nodes (and links) are displayed (full topology). This utility can also be used to list the current connected nodes. The output is printed to the standard output unless a topology file is specified.

Syntax

```
ibnetdiscover [options] [<topology-filename>]
```

Nonstandard flags:

```
l       Lists connected nodes
H       Lists connected HCAs
S       Lists connected switches
g       Groups
full (-f)       Shows full information (ports' speed and width,
vlcap)
show (-s)       Shows more information
Router_list  (-R)       Lists connected routers
node-name-map   Nodes name map file
cache   filename to cache ibnetdiscover data to
load-cache      filename of ibnetdiscover cache to load
diff    filename of ibnetdiscover cache to diff
diffcheck       Specifies checks to execute for --diff
ports : (-p)    Obtains a ports report
max_hops (-m)   Reports max hops discovered by the library
outstanding_smps (-o)   Specifies the number of outstanding SMP's
which should be issued during the scan
```

ibhosts

Traces the InfiniBand subnet topology or uses an already saved topology file to extract the CA nodes.

Syntax

```
ibhosts [-h] [<topology-file>]
```

Dependencies: ibnetdiscover, ibnetdiscover format

ibswitches

Traces the InfiniBand subnet topology or uses an already saved topology file to extract the InfiniBand switches.

Syntax

```
ibswitches [-h] [<topology-file>]
```

Dependencies: ibnetdiscover, ibnetdiscover format

ibportstate

Enables the port state and port physical state of an InfiniBand port to be queried or a switch port to be disabled or enabled.

Syntax

```
ibportstate [-d(ebug) -e(rr_show) -v(erbose) -D(irect) -G(uid) -s
smlid -V(ersion) -C ca_name -P ca_port -t timeout_ms] <dest
dr_path|lid|guid> <portnum> [<op>]
```

Supported ops: enable, disable, query, on, off, reset, speed, espeed, fdr10, width, down, arm, active, vls, mtu, lid, smlid, lmc, mkey, mkeylease, mkeyprot

Examples:

```
ibportstate 3 1 disable                 # by lid
ibportstate -G 0x2C9000100D051 1 enable # by guid
ibportstate -D 0 1                      # by direct route
```

ibnodes

Uses the current InfiniBand subnet topology or an already saved topology file and extracts the InfiniBand nodes (CAs and switches).

Syntax

```
ibnodes [<topology-file>]
```

Dependencies: ibnetdiscover, ibnetdiscover format

ibqueryerrors

Queries or clears the PMA error counters in PortCounters by walking the InfiniBand subnet topology.

```
ibqueryerrors [options]
```

Syntax

```
Options:
  --suppress, -s <err1,err2,...>  suppress errors listed
  --suppress-common, -c   suppress some of the common counters
  --node-name-map <file>  node name map file
  --port-guid, -G <port_guid>  report the node containing the
port
                              specified by <port_guid>
  --, -S <port_guid>      Same as "-G" for backward compatibility
  --Direct, -D <dr_path>  report the node containing the port
specified
                          by <dr_path>
  --skip-sl               don't obtain SL to all destinations
  --report-port, -r       report port link information
  --threshold-file <val>  specify an alternate threshold file,
default: /etc/infiniband-diags/error_thresholds
  --GNDN, -R              (This option is obsolete and does
nothing)
  --data                  include data counters for ports with
errors
  --switch                print data for switches only
  --ca                    print data for CA's only
  --router                print data for routers only
  --details               include transmit discard details
  --counters              print data counters only
  --clear-errors, -k      Clear error counters after read
  --clear-counts, -K      Clear data counters after read
  --load-cache <file>     filename of ibnetdiscover cache to load
  --outstanding_smps, -o <val>  specify the number of outstanding
SMP's
                             which should be issued during the
scan
  --config, -z <config>   use config file, default:
/etc/infiniband-diags/ibdiag.conf
  --Ca, -C <ca>           Ca name to use
  --Port, -P <port>       Ca port number to use
```

```
--timeout, -t <ms>      timeout in ms
--m_key, -y <key>       M_Key to use in request
--errors, -e            show send and receive errors
--verbose, -v           increase verbosity level
--debug, -d             raise debug level
--help, -h              help message
--version, -V           show version
```

smparquery

Issues Adaptive routing-related queries to the fabric switch.

Syntax

```
```

```
Supported ops (and aliases, case insensitive):
    ARInfo (ARI) <addr>
    ARGroupTable (ARGT) <addr> [<plft>] [<group_table>]
[<blocknum>]
    ARLFTTable (ARLT) <addr> [<plft>] [<blocknum>]
    PLFTInfo (PLFTI) <addr>
    PLFTDef (PLFTD) <addr> [<blocknum>]
    PLFTMap (PLFTM) <addr> [<plft>] [<control_map>]
    PortSLToPLFTMap (PLFTP) <addr> [<blocknum>]
    RNSubGroupDirectionTable (DIRT) <addr> [<blocknum>]
    RNGenStringTable (GSTR) <addr> [<plft>] [<blocknum>]
    RNGenBySubGroupPriority (GSGP) <addr>
    RNRcvString (RSTR) <addr> [<blocknum>]
    RNXmitPortMask (RNXM) <addr> [<blocknum>]
    PortRNCounters (RNPC) <addr>


Options:
    Main
        -C|--Ca <ca>                    : Ca name to use
        -P|--Port <port>                : Ca port number to use
        -D|--Direct                     : use Direct address
argument
        -L|--Lid                        : use LID address argument
        -h|--help                       : help message
        -V|--version                    : show version
        -d|--debug                      : Print debug logs
```

saquery

Issues SA queries.

Syntax

```
saquery [-h -d -P -N -L -G -s -g][<name>]
```

Queries node records by default.

dPNL (-L)G (-G)S (-S)G (-g)L (-l)O (-O)m(-m)x (-x)c (-c)S (-S)l (-I)list (-D)src-to-dst (<src:dst>)sgid-to-dgid (<sgid-dgid>)node-name-map smkey <val>slid <lid>dlid <lid>mild <lid>sgid <gid>dgid <gid>gid <gid>mgid <gid>Reversible", 'r', 1, NULL"numb_path ", 'n', 1, NULL"pkey: P_Key (PathRecord, MCMemberRecord).qos_class (-Q)slmtu : (-M)rate (-R)pkt_lifetime qkey (-q) (PathRecord, MCMemberRecord).tclass (-T)flow_label : (-F) hop_limit : (-H) scopejoin_state (-J)proxy_join (-X)service_id	Enables debuggingGets PathRecord infoGets NodeRecord infoReturns just the Lid of the name specifiedReturns just the Guid of the name specifiedReturns the PortInfoRecords with isSM capability mask bit onGets multicast group infoReturns the unique Lid of the name specifiedReturns name for the Lid specifiedGets multicast member info (if multicast group specified, list member GIDs only for group specified for example 'saquery -m 0xC000')Gets LinkRecord info"Gets the SA's class port infoGets ServiceRecord infoGets InformInfoRecord (subscription) infothe node desc of the CA'sGets a PathRecord for <src:dst> where src and dst are either node names or LIDsGets a PathRecord for <sgid-dgid> where sgid and dgid are addresses in IPv6 formatSpecifies a node name map fileSA SM_Key value for the query. If non-numeric value (like 'x') is specified then saquery will prompt for a value. Default (when not specified here or in ibdiag.conf) is to use SM_Key == 0 (or \"untrusted\")Source LID (PathRecord)Destination LID (PathRecord)Multicast LID (MCMemberRecord)Source GID (IPv6 format) (PathRecord)Destination GID (IPv6 format) (PathRecord)Port GID (MCMemberRecord)Multicast GID (MCMemberRecord)Reversible path (PathRecord)Number of paths (PathRecord) QoS Class (PathRecord)Service level (PathRecord, MCMemberRecord)MTU and selector (PathRecord, MCMemberRecord)Rate and selector (PathRecord, MCMemberRecord)Packet lifetime and selector (PathRecord, MCMemberRecord)If non-numeric value (like 'x') is specified then saquery will prompt for a value.Traffic Class (PathRecord, MCMemberRecord)Flow Label (PathRecord, MCMemberRecord)Hop limit (PathRecord, MCMemberRecord)Scope (MCMemberRecord)Join state (MCMemberRecord)Proxy join (MCMemberRecord)ServiceID (PathRecord)

Dependencies: OpenSM libvendor, OpenSM libopensm, libibumad

ibsysstat

```
ibsysstat [options] <dest lid|guid> [<op>]
```

Nonstandard flags:

```
Current supported operations:
  ping - verify connectivity to server (default)
  host - obtain host information from server
  cpu  - obtain cpu information from server
  -o <oui>    use specified OUI number to multiplex vendor mads
  -S          start in server mode (do not return)
```

ibnetsplit

Automatically groups hosts and creates scripts that can be run in order to split the network into sub-networks containing one group of hosts.

Syntax

- Group:

  ```
  ibnetsplit [-v][-h][-g grp-file] -s <.lst|.net|.topo> <-r
  head-ports|-d max-dist>
  ```

- Split:

  ```
  ibnetsplit [-v][-h][-g grp-file] -s <.lst|.net|.topo>
  -o out-dir
  ```

- Combined:

```
ibnetsplit [-v][-h][-g grp-file] -s <.lst|.net|.topo> <-r
head-ports|-d max-dist> -o out-dir
```

Usage

- Grouping:

  The grouping is performed if the -r or -d options are provided.

- If the -r is provided with a file containing group head ports, the algorithm examines the hosts distance from the set of node ports provided in the head-ports file (these are expected to be the ports running standby SM's).

- If the -d is provided with a maximum distance of the hosts in each group, the algorithm partition the hosts by that distance.

> **Note**
>
> This method of analyzation may not be suitable for some topologies.

The results of the identified groups are printed into the file defined by the -g option (default ibnetsplit.groups) and can be manually edited. For groups where the head port is a switch, the group file uses the FIRST host port as the port to run the isolation script from.

- Splitting:

  - If the -o flag is included, this algorithm analyzes the MinHop table of the topology and identifies the set of links and switches that may potentially be used for routing each group ports. The cross-switch links between switches of the group to other switches are declared as split-links and the commands to turn them off using Directed Routes from the original Group Head ports are written into the out-dir provided by the -o flag.

  Both stages require a subnet definition file to be provided by the -s flag. The supported formats for subnet definition are:

- *.net - for ibnetdiscover
- *.lst - for opensm-subnet.lst or ibiagnet.lst
- *.topo - for a topology file

HEAD PORTS FILE

This file is provided by the user and defines the ports by which grouping of the other host ports is defined.

Format:

Each line should contain either the name or the GUID of a single port. For switches the port number shall be 0.

```
<node-name>/P<port-num>|<PGUID>
```

GROUPS FILE

This file is generated by the program if the head-ports file is provided to it. Alternatively it can be provided (or edited) by the user if different grouping is desired. The generated script for isolating or connecting the group should be run from the first node in each group.

Format:

Each line may be either:

```
GROUP: <group name>
<node-name>/P<port-num>|<PGUID>
```

ibdiagnet

ibdiagnet scans the fabric using directed route packets and extracts all the available information regarding its connectivity and devices.

It then produces the following files in the output directory (see below):

- "ibdiagnet2.log" - A log file with detailed information.
- "ibdiagnet2.db_csv" - A dump of the internal tool database.
- "ibdiagnet2.lst" - A list of all the nodes, ports and links in the fabric.
- "ibdiagnet2.pm" - A dump of all the nodes PM counters.
- "ibdiagnet2.mlnx_cntrs" - A dump of all the nodes Mellanox diagnostic counters.
- "ibdiagnet2.net_dump" - A dump of all the links and their features.
- "ibdiagnet2.pkey" - A list of all pkeys found in the fabric.
- "ibdiagnet2.aguid" - A list of all alias GUIDs found in the fabric.
- "ibdiagnet2.sm" - A dump of all the SM (state and priority) in the fabric.
- "ibdiagnet2.fdbs" - A dump of unicast forwarding tables of the fabric switches.
- "ibdiagnet2.mcfdbs" - A dump of multicast forwarding tables of the fabric switches.
- "ibdiagnet2.slvl" - A dump of SLVL tables of the fabric switches.
- "ibdiagnet2.nodes_info" - A dump of all the nodes vendor specific general information for nodes who supports it.
- "ibdiagnet2.plft" - A dump of Private LFT Mapping of the fabric switches.
- "ibdiagnet2.ar" - A dump of Adaptive Routing configuration of the fabric switches.
- "ibdiagnet2.vl2vl" - A dump of VL to VL configuration of the fabric switches.

Load plugins from:

/tmp/ibutils2/share/ibdiagnet2.1.1/plugins/

You can specify additional paths to be looked in with "IBDIAGNET_PLUGINS_PATH" env variable.

```
Plugin Name                                   Result     Comment
libibdiagnet_cable_diag_plugin-2.1.1     Succeeded  Plugin loaded
libibdiagnet_phy_diag_plugin-2.1.1       Succeeded  Plugin loaded
```

Syntax

```
[-i|--device <dev-name>] [-p|--port <port-num>]
[-g|--guid <GUID in hex>] [--skip <stage>]
[--skip_plugin <library name>] [--sc]
[--scr] [--pc] [-P|--counter <<PM>=<value>>]
[--pm_pause_time <seconds>] [--ber_test]
[--ber_thresh <value>] [--llr_active_cell <64|128>]
[--extended_speeds <dev-type>] [--pm_per_lane]
[--ls <2.5|5|10|14|25|FDR10|EDR20>]
[--lw <1x|4x|8x|12x>] [--screen_num_errs <num>]
[--smp_window <num>] [--gmp_window <num>]
[--max_hops <max-hops>] [--read_capability <file name>]
[--write_capability <file name>]
[--back_compat_db <version.sub_version>]
[-V|--version] [-h|--help] [-H|--deep_help]
[--virtual] [--mads_timeout <mads-timeout>]
[--mads_retries <mads-retries>] [-m|--map <map-file>]
[--vlr <file>] [-r|--routing] [--r_opt <[vs,][mcast,]>]
[--sa_dump <file>] [-u|--fat_tree]
[--scope <file.guid>] [--exclude_scope <file.guid>]
[-w|--write_topo_file <file name>]
[-t|--topo_file <file>] [--out_ibnl_dir <directory>]
[-o|--output_path <directory>]
Cable Diagnostic (Plugin)
[--get_cable_info] [--cable_info_disconnected]
Phy Diagnostic (Plugin)
[--get_phy_info] [--reset_phy_info]
```

Options

```
-i|--device <dev-name>          : Specifies the name of the device
of the port
                                 used to connect to the IB fabric
(in case
                                 of multiple devices on he local
system).
-p|--port <port-num>            : Specifies the local device's port
number
                                 used to connect to the IB fabric.
-g|--guid <GUID in hex>         : Specifies the local port GUID
value of the
                                 port used to connect to the IB
fabric. If
                                 GUID given is 0 than ibdiagnet
displays
                                 a list of possible port GUIDs and
waits
                                 for user input.
--skip <stage>                  : Skip the executions of the given
stage.
                                 Applicable skip stages
(vs_cap_smp
                                 vs_cap_gmp | links | pm |
                                 speed_width_check | all).
--skip_plugin <library name>    : Skip the load of the given
library name.
                                 Applicable skip plugins:
                                 (libibdiagnet_cable_diag_plugin-
2.1.1 |
                                  libibdiagnet_phy_diag_plugin-
2.1.1).
--sc                             : Provides a report of Mellanox
counters
--scr                            : Reset all the Mellanox
counters (if –sc
```

```
                                    option selected).
--pc                               : Reset all the fabric PM
counters.
-P|--counter <<PM>=<value>>        : If any of the provided PM is
greater then
                                    its provided value than print
it.
--pm_pause_time <seconds>          : Specifies the seconds to wait
between
                                    first counters sample and
second counters
                                    sample. If seconds given is 0
than no
                                    second counters sample will be
done.
                                    (default=1).
--ber_test                         :Provides a BER test for each
port.
                                    Calculate BER for each port
and check no
                                    BER value has exceeds the BER
threshold.
                                    (default threshold="10^-12").
--ber_thresh <value>               :Specifies the threshold value
for the
                                    BER test. The reciprocal
number of the
                                    BER should be provided.
Example: for
                                    10^-12 than value need to be
                                    1000000000000 or 0xe8d4a51000
                                    (10^12).If threshold given is
0 than all
                                    BER values for all ports will
be
                                    reported.
```

```
--llr_active_cell <64|128>          : Specifies the LLR active
cell size
                                      for BER test, when LLR is
active in the
                                      fabric.
--extended_speeds <dev-type>        : Collect and test port
extended speeds
                                      counters. dev-type: (sw |
all).
--pm_per_lane                       : List all counters per lane
(when
                                      available).
--ls <0|2.5|5|10|14|25|50|100|FDR10>    : Specifies the expected
link speed.
--lw <1x|4x|8x|12x>                 : Specifies the expected
link width.
--screen_num_errs <num>             : Specifies the threshold
for printing
                                      errors to screen.
(default=5).
--smp_window <num>                  : Max smp MADs on wire.
(default=8).
--gmp_window <num>                  : Max gmp MADs on wire.
(default=128).
--max_hops <max-hops>               : Specifies the maximum hops
for the
                                      discovery process.
(default=64).
--read_capability <file name>       : Specifies capability masks
                                      configuration file, giving
capability
                                      mask configuration for the
fabric.
                                      ibdiagnet will use this
mapping for
```

```
                                        Vendor Specific MADs
sending.
--write_capability <file name>       : Write out an example file
for
                                        capability masks
configuration,
                                        and also the default
capability
                                        masks for some devices.
--back_compat_db <version.sub_version> : Show ports section in
                                         "ibdiagnet2.db_csv"
according to
                                         given version. Default
version 2.0.
-V|--version                         : Prints the version of the
tool.
-h|--help                            : Prints help information
(without
                                         plugins help if exists).
-H|--deep_help                       : Prints deep help
information
                                         (including plugins help).
--virtual                            : Discover VPorts during
discovery
                                         stage.
--mads_timeout <mads-timeout>        : Specifies the timeout (in
                                          milliseconds) for sent
and received
                                          mads. (default=500).
--mads_retries <mads-retries>        : Specifies the number of
retreis for
                                         every timeout mad.
(default=2).
-m|--map <map-file>                  : Specifies mapping file,
that maps
                                         node guid to name
```

```
                                        (format: 0x[0-9a-fA-F]+
"name").
                                        Maping file can also be
specified by
                                        Environment variable

"IBUTILS_NODE_NAME_MAP_FILE_PATH".
--src_lid <src-lid>                   : source lid
--dest_lid <dest-lid>                 : destination lid
--dr_path <dr-path>                   : direct route path
-o|--output_path <directory>          : Specifies the directory
where the
                                        Output files will be
placed.

(default="/var/tmp/ibdiagpath/").
Cable Diagnostic (Plugin)
--get_cable_info                      : Indicates to query all
QSFP cables
                                        for cable information.
Cable
                                        information will be
stored
                                        in "ibdiagnet2.cables".
--cable_info_disconnected             : Get cable info on
disconnected
                                        ports.
Phy Diagnostic (Plugin)
--get_phy_info                        : Indicates to query all
ports for phy
                                        information.
--reset_phy_info                      : Indicates to clear all
ports phy
                                        information.
```

Cable Diagnostic (Plugin):

This plugin performs cable diagnostic. It can collect cable info (vendor, PN, OUI etc..) on each valid QSFP cable, if specified.

It produces the following files in the output directory (see below):

- "ibdiagnet2.cables" - In case specified to collect cable info, this file will contain all collected cable info.

Phy Diagnostic (Plugin)

This plugin performs phy diagnostic.

Load Plugins from**:**

```
/tmp/ibutils2/share/ibdiagnet2.1.1/plugins/
```

You can specify additional paths to be looked in with "IBDIAGNET_PLUGINS_PATH" env variableLoad plugins from:

```
Plugin Name                                   Result     Comment
libibdiagnet_cable_diag_plugin-2.1.1          Succeeded  Plugin
loaded
libibdiagnet_phy_diag_plugin-2.1.1            Succeeded  Plugin
loaded
```

Syntax

```
[-i|--device <dev-name>] [-p|--port <port-num>]
[-g|--guid <GUID in hex>] [--skip <stage>]
[--skip_plugin <library name>] [--sc]
[--scr] [--pc] [-P|--counter <<PM>=<value>>]
[--pm_pause_time <seconds>] [--ber_test]
[--ber_thresh <value>] [--llr_active_cell <64|128>]
[--extended_speeds <dev-type>] [--pm_per_lane]
[--ls <2.5|5|10|14|25|FDR10|EDR20>]
[--lw <1x|4x|8x|12x>] [--screen_num_errs <num>]
[--smp_window <num>] [--gmp_window <num>]
[--max_hops <max-hops>] [--read_capability <file name>]
[--write_capability <file name>]
[--back_compat_db <version.sub_version>]
[-V|--version] [-h|--help] [-H|--deep_help]
[--virtual] [--mads_timeout <mads-timeout>]
[--mads_retries <mads-retries>] [-m|--map <map-file>]
[--src_lid <src-lid>] [--dest_lid <dest-lid>]
[--dr_path <dr-path>] [-o|--output_path <directory>]
Cable Diagnostic (Plugin)
[--get_cable_info] [--cable_info_disconnected]
Phy Diagnostic (Plugin)
[--get_phy_info] [--reset_phy_info]
```

Options

-i|--device <dev-name>-p|--port <port-num>-g|--guid <GUID in hex>--skip <stage>--skip_plugin <library name>--sc--scr--pc-P|--counter <<PM>=<value>>--pm_pause_time <seconds>--ber_test--ber_thresh <value>--llr_active_cell <64|128>--extended_speeds <dev-type>--pm_per_lane:List all counters per lane (when available).--ls <2.5|5|10|14|25|FDR10|EDR20>--lw <1x|4x|8x|12x>--screen_num_errs <num>--smp_window <num>--gmp_window <num>--max_hops <max-hops>--read_capability <file name>--write_capability <file name>--back_compat_db <version.sub_version>-V|--version-h|--help-H|--deep_help--virtual--mads_timeout <mads-timeout>--mads_retries <mads-retries>-m|--map <map-file>--src_lid <src-lid>--dest_lid <dest-lid>--dr_path <dr-path>-o|--output_path <directory>Cable Diagnostic (Plugin)--get_cable_info--cable_info_disconnectedPhy Diagnostic (Plugin)--get_phy_info--reset_phy_info	:Specifies the name of the device of the port used to connect to the IB fabric (in case of multiple devices on the local system).:Specifies the local device's port number used to connect to the IB fabric.:Specifies the local port GUID value of the port used to connect to the IB fabric. If GUID given is 0 than ibdiagnet displays a list of possible port GUIDs and waits for user input.:Skip the executions of the given stage. Applicable skip stages: (vs_cap_smp | vs_cap_gmp | links | pm | speed_width_check | all).:Skip the load of the given library name. Applicable skip plugins:(libibdiagnet_cable_diag_plugin-2.1.1 | libibdiagnet_phy_diag_plugin-2.1.1).:Provides a report of Mellanox counters:Reset all the Mellanox counters (if -sc option selected). :Reset all the fabric PM counters. :If any of the provided PM is greater then its provided value than print it.:Specifies the seconds to wait between first counters sample and second counters sample. If seconds given is 0 than no second counters sample will be done. (default=1).:Provides a BER test for each port. Calculate BER for each port and check no BER value has exceeds the BER threshold.(default threshold="10^-12").:Specifies the threshold value for the BER test. The reciprocal number of the BER should be provided. Example: for 10^-12 than value need to be 1000000000000 or 0xe8d4a51000(10^12).If threshold given is 0 than all BER values for all ports will be reported.:Specifies the LLR active cell size for BER test, when LLR is active in the fabric.:Collect and test port extended speeds counters. dev-type: (sw | all). :Specifies the expected link speed.:Specifies the expected link width.:Specifies the threshold for printing errors to screen. (default=5).:Max smp MADs on wire. (default=8).:Max gmp MADs on wire. (default=128).:Specifies the maximum hops for the discovery process.(default=64).:Specifies capability masks configuration file, giving capability mask configuration for the fabric. ibdiagnet will use this mapping for Vendor Specific MADs sending.:Write out an example file for capability masks configuration, and also the default capability masks for some devices.:Show ports section in "ibdiagnet2.db_csv" according to given version. Default version 2.0.:Prints the version of the tool.:Prints help information (without plugins help if exists).:Prints deep help information (including plugins help).:Discover VPorts during discovery stage.:Specifies the timeout (in milliseconds) for sent and received mads.(default=500).:Specifies the number of retries for every timeout mad.(default=2).:Specifies mapping file, that maps node guid to name (format: 0x[0-9a-fA-F]+ "name"). Mapping file can also be specified by environment variable "IBUTILS_NODE_NAME_MAP_FILE_PATH".:source liddestination lid:direct route path:Specifies the directory where the output files will be placed. (default="/var/tmp/ibdiagpath/").

	:Indicates to query all QSFP cables for cable information. Cable information will be stored in "ibdiagnet2.cables".:Get cable info on disconnected ports. :Indicates to query all ports for phy information.:Indicates to clear all ports phy information.

Appendix - Supported Port Counters and Events

Port counters and events are available in the following views:

- Events and Port Counters area, at the bottom of the UFM window
- Error window (Error tab) in the Manage Devices tab
- In the New Monitoring Session window, in the Monitor tab, when clicking Create New Session
- Event Log in the Log tab (click Show Event Log)

InfiniBand Port Counters

The following tables list and describe the port counters and events currently supported:

- InfiniBand Port Counters
- Calculated Port Counters

InfiniBand Port Counters	
Counter	Description
Xmit Data (in bytes)	Total number of data octets, divided by 4, transmitted on all VLs from the port, including all octets between (and not including) the start of packet delimiter and the VCRC, and may include packets containing errors. All link packets are excluded. Results are reported as a multiple of four octets.

InfiniBand Port Counters	
Rcv Data (in bytes)	Total number of data octets, divided by 4, received on all VLs at the port. All octets between (and not including) the start of packet delimiter and the VCRC are excluded and may include packets containing errors. All link packets are excluded. When the received packet length exceeds the maximum allowed packet length specified in C7-45: the counter may include all data octets exceeding this limit. Results are reported as a multiple of four octets.
Xmit Packets	Total number of packets transmitted on all VLs from the port, including packets with errors and excluding link packets.
Rcv Packets	Total number of packets, including packets containing errors and excluding link packets, received from all VLs on the port.
Rcv Errors	Total number of packets containing errors that were received on the port including: • Local physical errors (ICRC, VCRC, LPCRC, and all physical errors that cause entry into the BAD PACKET or BAD PACKET DISCARD states of the packet receiver state machine) • Malformed data packet errors (LVer, length, VL) • Malformed link packet errors (operand, length, VL) • ackets discarded due to buffer overrun (overflow)
Xmit Discards	Total number of outbound packets discarded by the port when the port is down or congested for the following reasons: • Output port is not in the active state • Packet length has exceeded NeighborMTU • Switch Lifetime Limit exceeded • Switch HOQ Lifetime Limit exceeded, including packets discarded while in VLStalled State.
Symbol Errors	Total number of minor link errors detected on one or more physical lanes.
Link Error Recovery	Total number of times the Port Training state machine has successfully completed the link error recovery process.
Link Error Downed	Total number of times the Port Training state machine has failed the link error recovery process and downed the link.
Local Integrity Error	The number of times that the count of local physical errors exceeded the threshold specified by LocalPhyErrors

InfiniBand Port Counters	
Rcv Remote Physical Error	Total number of packets marked with the EBP delimiter received on the port.
Xmit Constraint Error	Total number of packets not transmitted from the switch physical port for the following reasons: • FilterRawOutbound is true and packet is raw • PartitionEnforcementOutbound is true and packet fails partition key check or IP version check
Rcv Constraint Error	Total number of packets received on the switch physical port that are discarded for the following reasons: • FilterRawInbound is true and packet is raw • PartitionEnforcementInbound is true and packet fails partition key check or IP version check
Excess Buffer Overrun Error	The number of times that OverrunErrors consecutive flow control update periods occurred, each having at least one overrun error
Rcv Switch Relay Error	Total number of packets received on the port that were discarded when they could not be forwarded by the switch relay for the following reasons: • DLID mapping • VL mapping • Looping (output port = input port)
VL15 Dropped	Number of incoming VL15 packets dropped because of resource limitations (e.g., lack of buffers) in the port
XmitWait	The number of ticks during which the port selected by PortSelect had data to transmit but no data was sent during the entire tick because of insufficient credits or of lack of arbitration.

InfiniBand Calculated Port Counters	
Counter	Description
Normalized XmitData	Effective port bandwidth utilization in % XmitData incremental/ Link Capacity

InfiniBand Calculated Port Counters	
Normalized Congested Bandwidth	Amount of bandwidth that was suppressed due to congestion (XmitWait incremental/ Time) * Link CapacitySeparate counters are used for Tier 4 ports and for the rest of the ports.

Supported Traps and Events

Device events are listed as VDM or CDM in the Source column of the Events table in the UFM GUI. For information about defining event policy, see Events Policy.

For the full supported events and alarms, please refer to Threshold-Crossing Events Reference.

Appendix – Used Ports

The following is the list of ports used by the UFM Server for internal and external communication:

Port	Purpose
25(tcp)	Used by smtp mail server, for sending daily reports and events summary via email
80(tcp), 443(tcp)	Used by WS clients (Apache Web Server)
8000(udp)	Used for UFM server listening for REST API requests (redirected by Apache web server)
6306(udp)	Used for Multicast requests – communication with latest UFM Agents Closed by default (as UFM Agent is deprecated and disabled by default)
8081(tcp), 8082(tcp)	Used for internal communication with Subnet Manager

Optional ports, in case working with HA nodes

Port	Purpose
2224(tcp)	Used by pacemaker for communication between the two UFM HA nodes

Port	Purpose
7788(udp)	Used by DRBD for filesystem synchronization between the two UFM HA nodes
5403(tcp), 5404(tcp), 5405(udp)	Used by corosync port

Appendix – Configuration Files Auditing

The main purpose of this feature is to allow users to track changes made to selected configuration files. When activating the feature, all the changes are reflected in specific log files which contain information about the changes and when they took place.

To activate this feature:

In *TrackConfig* section in gv.cfg, file value of *track_config* key should be set to **true** and value of *track_conf_files* key should contain a comma-separated list of defined conf files to be tracked.

By default – ALL conf-files are tracked. To activate the feature, after *track_config* key is set to true, the UFM server should be restarted.

Example:

```
[TrackConfig]
# track config files changes
track_config = true
# Could be selected options (comaseparated) UFM, SM, SHARP,
Telemetry. Or ALL for all the files.
track_conf_files = ALL
```

The below lists the configuration files that can be tracked:

Conf File Alias	Configuration Files
UFM	/opt/ufm/files/conf/gv.cfg
SM	/opt/ufm/files/conf/opensm/opensm.conf
SHARP	/opt/ufm/files/conf/sharp2/sharp_am.cfg
Telemetry	/opt/ufm/files/conf/telemetry/launch_ibdiagnet_config.ini
ALL	All the above configuration files.

Once the feature is activated and the UFM server is restarted, the UFM generates file which list the changes made in each of the tracked conf files. These files are located in */opt/ufm/files/auditing/* directory and the file naming convention is as follows: original conf file name with audit.log suffix.

Example: For gv.cfg, the name of the changes-tracking file is gv.cfg.audit.log. Changes are stored in auditing files in “linux diff”-like format.

```
cat /opt/ufm/files/auditing/gv.cfg.audit.log
=== Change occurred at 2022-07-24 07:31:48.679247 ===
---
+++
@@ -45,7 +45,7 @@
mon_mode_discovery_period = 60
check_interface_retry = 5
# The number of times to try if the InfiniBand fabric interface is
down. The duration of each retry is 1 second.
-ibport_check_retries = 90
+ibport_check_retries = 92
ws_address = UNDEFINED
ws_port = 8088
ws_protocol = https
```

Appendix – IB Router

IB router provides the ability to send traffic between two or more IB subnets thereby potentially expanding the size of the network to over 40k end-ports, enabling separation

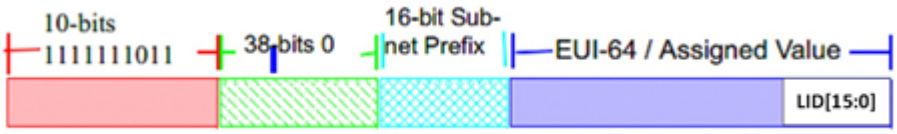
and fault resilience between islands and IB subnets, and enabling connection to different topologies used by different subnets.

The forwarding between the IB subnets is performed using GRH lookup. The IB router's basic functionality includes:

- Removal of current L2 LRH (local routing header)
- Routing table lookup – using GID from GRH
- Building new LRH according to the destination according to the routing table

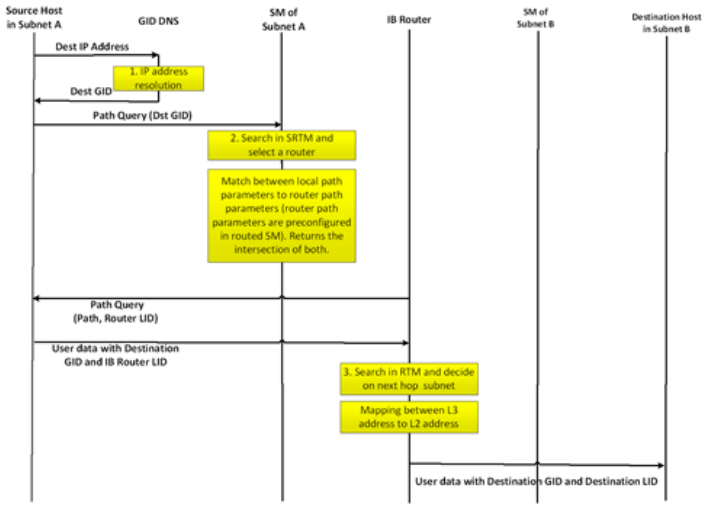
The DLID in the new LRH is built using simplified GID-to-LID mapping (where LID = 16 LSB bits of GID) thereby not requiring to send for ARP query/lookup.

Site-Local Unicast GID Format



For this to work, the SM allocates an alias GID for each host in the fabric where the alias GID = {subnet prefix[127:64], reserved[63:16], LID[15:0}. Hosts should use alias GIDs in order to transmit traffic to peers on remote subnets.

Host-to-Host IB Router Unicast Flow



IB Router Scripts

The following scripts are supplied as part of UFM installation package.

set_num_of_subnets.sh

- **Arguments**

```
/opt/ufm/scripts/ib_router/set_num_of_subnets.sh --hostname
<hostname> --username <username> --password <password> --num-
of-subnets <num-of-subnets>
```

- **Description** – Configures system profile to InfiniBand allowing multiple switch IDs
- **Syntax Description**

hostname	IB router hostname or IP address
username	IB router username
password	IB router user password
num-of-subnets	Specified number of subnets (AKA SWIDs) to be initialized by the system. Value range: 2-6

- **Example**

```
/opt/ufm/scripts/ib_router/set_num_of_subnets.sh --hostname
10.6.204.12 --username admin --password admin --num-of-
subnets 6
```

Note

As a result of running this script, reboot is performed and all configuration is removed

add_interfaces_to_subnet.sh

- **Arguments**

```
/opt/ufm/scripts/ib_router/add_interfaces_to_subnet.sh  --hostname <hostname> --username <username> --password <password> --interface <interface | interface-range> --subnet <subnet>
```

- **Description**

  Maps an interface to a subnet and enables it

- **SyntaxDescription**

hostname	IB router hostname or IP address
username	IB router username
password	IB router user password
interface | interface-range	Single IB interface or range of IB interfaces. Single IB interface: 1/<interface> Range of IB interfaces: 1/<interface>-1/<interface>
subnet	Name of IB subnet (AKA SWID): infiniband-default, infiniband-1…infiniband-5

- **Example**

```
/opt/ufm/scripts/ib_router/add_interfaces_to_subnet.sh  --hostname 10.6.204.12 --username admin --password admin --interface 1/1-1/6 --subnet infiniband-1
```

remove_interfaces_from_subnet.sh

- **Arguments**

```
/opt/ufm/scripts/ib_router/remove_interfaces_from_subnet.sh
--hostname <hostname> --username <username> --password
<password> --interface <interface | interface-range>
```

- **Description**

  Un-maps an interface from a subnet after it has been disabled

- **Syntax Description**

hostname	IB router hostname or IP address
username	IB router username
password	IB router user password
interface | interface-range	Single IB interface or range of IB interfaces. Single IB interface: 1/<interface> Range of IB interfaces: 1/<interface>-1/<interface>

- **Example**

```
/opt/ufm/scripts/ib_router/remove_interfaces_from_subnet.sh
--hostname 10.6.204.12 --username admin --password admin --
interface 1/6Example
```

add_subnet_to_router.sh

- **Arguments**

```
/opt/ufm/scripts/ib_router/add_subnet_to_router.sh --hostname
<hostname> --username <username> --password <password> --
subnet <subnet>
```

- **Description**

  Creates routing on IB subnet interface and enables routing on that interface

- **Syntax Description**

hostname	IB router hostname or IP address
username	IB router username
password	IB router user password
subnet	Name of IB subnet (AKA SWID): infiniband-default, infiniband-1... infiniband-5

- **Example**

```
/opt/ufm/scripts/ib_router/add_subnet_to_router.sh --hostname
10.6.204.12 --username admin --password admin --subnet
infiniband-3Example
```

Note

As a result of running this script, the set of commands that allow control of IB router functionality is being enabled

remove_subnet_from_router.sh

- **Arguments**

```
/opt/ufm/scripts/ib_router/remove_subnet_from_router.sh  --
hostname <hostname> --username <username> --password
<password> --subnet <subnet>
```

- **Description**

  Destroys routing on IB subnet interface after routing on that interface has been disabled

- **Syntax Description**

hostname	IB router hostname or IP address
username	IB router username
password	IB router user password
subnet	Name of IB subnet (AKA SWID): infiniband-default, infiniband-1… infiniband-5

- **Example**

```
/opt/ufm/scripts/ib_router/remove_subnet_from_router.sh  --
hostname 10.6.204.12 --username admin --password admin --
subnet infiniband-defaultExample
```

set_ufm_sm_router_support.sh

- **Arguments**

```
/opt/ufm/scripts/ib_router/set_ufm_sm_router_support.sh [-c
<subnet prefix>] [-r][-h]
```

- **Description**

  [-c <subnet prefix>]: Used for updating OpenSM configuration file with new subnet prefix and forces OpenSM to re-read configuration.

  [-r]: Used for resetting OpenSM configuration to default value and canceling IB routing.

- **Syntax Description**

-c	Configure new IB subnet prefix. Should be followed by new IB router subnet prefix value
-r	Reset to default
-h	Show help

- **Example**

```
/opt/ufm/scripts/ib_router/set_ufm_sm_router_support.sh -c
0xfec0000000001234Examples
```

```
/opt/ufm/scripts/ib_router/set_ufm_sm_router_support.sh -r
```

IB Router Configuration

Step 1: Configure multi-switch. Run:

```
/opt/ufm/scripts/set_num_of_subnets.sh --hostname 10.6.204.12 --
username admin --password admin --num-of-subnets 6
```

Step 2: Map interface to a subnet. Run:

```
/opt/ufm/scripts/add_ports_to_subnet.sh --hostname 10.6.204.12 --
username admin --password admin --interface 1/1 --subnet
infiniband-default
```

Step 3: Create routing on IB subnet interface. Run:

```
/opt/ufm/scripts/add_subnet_to_router.sh --hostname 10.6.204.12
--username admin --password admin --subnet infiniband-default
```

Appendix – NVIDIA SHARP Integration

NVIDIA Scalable Hierarchical Aggregation and Reduction Protocol (SHARP)™

NVIDIA SHARP is a technology that improves the performance of MPI operation by offloading collective operations from the CPU and dispatching to the switch network, and eliminating the need to send data multiple times between endpoints. This approach decreases the amount of data traversing the network as aggregation nodes are reached, and dramatically reduces the MPI operation time.

NVIDIA SHARP software is based on:

- Hardware capabilities in Quantum-1 and later generations
- Hierarchical communication algorithms (HCOL) library into which NVIDIA SHARP capabilities are integrated[1]

- NVIDIA SHARP Aggregation Manager, running on UFM

1. These components should be installed from HPCX or MLNX_OFED packages on compute nodes. Installation details can be found in SHARP Deployment Guide.

NVIDIA SHARP Aggregation Manager

Aggregation Manager (AM) is a system management component used for system level configuration and management of the switch-based reduction capabilities. It is used to set up the NVIDIA SHARP trees, and to manage the use of these entities.

AM is responsible for:

- NVIDIA SHARP resource discovery
- Creating topology aware NVIDIA SHARP trees
- Configuring NVIDIA SHARP switch capabilities
- Managing NVIDIA SHARP resources
- Assigning NVIDIA SHARP resource upon request
- Freeing NVIDIA SHARP resources upon job termination

AM is configured by a topology file created by Subnet Manager (SM): subnet.lst. The file includes information about switches and HCAs.

NVIDIA SHARP AM Configuration

By default, when running NVIDIA SHARP AM by UFM, there is no need to run further configuration. To modify the configuration of NVIDIA SHAPR AM, you can edit the following NVIDIA SHARP AM configuration file: `/opt/ufm/files/conf/sharp/sharp_am.cfg`.

Running NVIDIA SHARP AM in UFM



To run NVIDIA SHARP AM within UFM, do the following:

1. Enable NVIDIA SHARP AM in conf/gv.cfg UFM configuration file by running the command "ib sharp enable" or by setting the sharp_enabled parameter to true (it is true by default):

```
[Sharp]
sharp_enabled = true
```

2. (Optional) Enable NVIDIA SHARP allocation in conf/gv.cfg UFM configuration file by setting the sharp_allocation_enabled parameter to true (it is false by default):

```
[Sharp]
sharp_allocation_enabled = true
```

> **Note**
>
> For further information about SHARP allocation methods, refer to the NVIDIA SHARP Documentation.

Operating NVIDIA SHARP AM with UFM

If NVIDIA SHARP AM is enabled, running UFM will run NVIDIA SHARP AM, and stopping UFM will stop NVIDIA SHARP AM.

To

start UFM with NVIDIA SHARP AM (enabled):

```
/etc/init.d/ufmd start
```

The same command applies to HA, using /etc/init.d/ufmha.

Upon startup of UFM or SHARP Aggregation Manager, UFM will resend all existing persistent allocation to SHARP AM.

To stop UFM with NVIDIA SHARP AM (enabled):

```
/etc/init.d/ufmd stop
```

To stop only NVIDIA SHARP AM while leaving UFM running:

```
/etc/init.d/ufmd sharp_stop
```

To start only NVIDIA SHARP AM while UFM is already running:

```
/etc/init.d/ufmd sharp_start
```

Upon startup of UFM or SHARP Aggregation Manager, UFM will resend all existing persistent allocation to SHARP AM.

To restart only NVIDIA SHARP AM while UFM is running:

```
/etc/init.d/ufmd sharp_restart
```

Upon startup of UFM or SHARP Aggregation Manager, UFM will resend all existing persistent allocation to SHARP AM.

To display NVIDIA SHARP AM status while UFM is running:

```
/etc/init.d/ufmd sharp_status
```

Monitoring NVIDIA SHARP AM by UFM Health

UFMHealth monitors SHARP AM and verifies that NVIDIA SHARP AM is always running. When UFMHealth detects that NVIDIA SHARP AM is down, it will try to re-start it, and will trigger an event to the UFM to notify it that NVIDIA SHARP AM is down.

Managing NVIDIA SHARP AM by UFM High Availability (HA)

In case of a UFM HA failover or takeover, NVIDIA SHARP AM will be started on the new master node using the same configuration that was used prior to the failover/takeover.

NVIDIA SHARP AM Logs

NVIDIA SHARP AM log file (sharp_am.log) at `/opt/ufm/files/log`.

NVIDIA SHARP AM log files are rotated by UFM logrotate mechanism.

NVIDIA SHARP AM Version

NVIDIA SHARP AM version can be found at `/opt/ufm/sharp/share/doc/SHARP_VERSION`.

Appendix - UFM SLURM Integration

Simple Linux Utility for Resource Management (SLURM) is a job scheduler for Linux and Unix-like kernels.

By integrating SLURM with UFM, you can:

- Assign partition keys (PKeys) to SLRUM nodes that are assigned for specific SLURM jobs.
- Create SHARP reservations based on SLURM nodes assigned for specific SLURM jobs.

Prerequisites

- UFM 6.9.0 (or newer)
- Python 3.0 on SLURM controller

- UFM-SLURM integration files (provided independently)

Automatic Installation

A script is provided to install the UFM-SLURM integration automatically.

1. Using the SLURM controller, extract the UFM-SLURM integration tar file:

```
tar -xf ufm_slurm_integration.tar.gz
```

2. Run the installation script using root privileges.

```
sudo ./install.sh
```

Manual Installation

To install the UFM-SLURM integration manually:

1. Extract the UFM-SLURM integration tar file:

```
tar -xf ufm_slurm_integration.tar.gz
```

2. Copy the UFM-SLURM integration files to the SLURM controller folder.

3. Change the permissions of the UFM-SLURM integration files to 755.

4. Modify the SLURM configuration file on the SLURM controller, `/etc/slurm/slurm.conf`, and add/modify the following two parameters:

```
PrologSlurmctld=/etc/slurm/ufm-prolog.sh
EpilogSlurmctld=/etc/slurm/ufm-epilog.sh
```

UFM SLURM Config File

The integration process uses a configuration file located at `/etc/slurm/ufm_slurm.conf`. This file is used to configure settings and attributes for UFM-SLURM integration.

Here are the contents:

Attribute Name	Description	Optionality
`ufm_server`	IP of UFM server to connect to	Mandatory
`auth_type`	Should be `token_auth`, or `basic_auth` If you select `basic_auth`, you need to set `ufm_server_user` and `ufm_server_pass`If you select `token_auth`, you need to set `token_auth`	Mandatory
`ufm_server_user`	Username of UFM server used to connect to UFM, if you set `auth_type=basic_auth`	Mandatory, depends on the `auth_type`
`ufm_server_pass`	UFM server user password	Mandatory, depends on the `auth_type`
`token`	Generated token when you set uth_typea to `token_auth`	Mandatory, depends on the `auth_type`
`pkey_allocation`	Controls whether UFM SLURM Integration allocates a Pkey for SLURM jobs. • When set to false: No PKey is allocated; the default management PKEY (0x7fff) is used. • When set to true: A PKey is allocated based on the ' `pkey_allocation_mode` ' parameter setting.	Mandatory, default is True

Attribute Name	Description	Optionality
`pkey_allocation_mode`	The PKey allocation mode is either static or dynamic. The default value is static. • Static: uses a statically assigned PKey when creating a new PKEY. • Dynamic: uses a dynamic assignment when creating a new PKey based on the `slurm-job-id`.	Static or Dynamic, default is static
`pkey`	Hexadecimal string between "0x0001"-"0x7ffe" exclusive. • Applicable only when assigning a static PKey for all SLURM jobs. • If the `pkey_allocation_mode` is set to static, and the PKey is not set, the default management PKey (0x7fff) will be used.	Optional, default is "0x7fff" (This is the default management PKey)
`ip_over_ib`	PKey is a member in a multicast group that uses IP over InfiniBand	Hidden param, default is True
`index0`	If true, the API will store the PKey at index 0 of the PKey table of the GUID	Hidden param, default is False
`sharp_allocation`	By setting `sharp_allocation` to true, UFM SLURM Integration creates new SHARP allocation with all SLURM job IDs allocated to hosts	Mandatory, default is False
`partially_alloc`	By setting this to false, UFM will fail the SHARP allocation request if at least one node does not exist in the fabric	Optional, default is False
`app_resources_limit`	Application resources limitation	Hidden param, default is -1
`log_file_name`	Name of integration logging file	Optional

Configuring UFM for NVIDIA SHARP Allocation

To configure UFM for NVIDIA SHARP allocation/deallocation you must set `sharp_enabled` and `enable_sharp_allocation` to true in gv.cfg file.

Generate token_auth

If you set `auth_type=token_auth` in UFM SLURM's config file, you must generate a new token by logging into the UFM server and running the following `curl` command:

```
curl -H "X-Remote-User:admin" -XPOST
http://127.0.0.1:8000/app/tokens
```

Then you must copy the generated token and paste it into the config file beside the `token_auth` parameter.

Prolog and Epilog

After submitting jobs on SLURM, there are two scripts that are automatically executed:

- `ufm-prolog.sh` – the prolog script is executed when a job is submitted and before running the job itself. It creates the partition key (pkey) assignment and/or NVIDIA SHARP reservation and assigns the SLURM job hosts for them.
- `ufm-epilog.sh` – the epilog script is executed when a job is complete. It removes the partition key (PKey) assignment and/or NVIDIA SHARP reservation and free the associated SLURM job hosts.

Integration Files

The integration use scripts and configuration files to work, which should be copied to SLURM controller `/etc"/slurm`. Here is a list of these files:

File Name	Description
`ufm-prolog.sh`	Bash file which executes jobs related to UFM after the SLURM job is completed

File Name	Description
`ufm-epilog.sh`	Bash file which executes jobs related to UFM before the SLURM job is executed
`ufm_slurm.conf`	UFM-SLURM integration configuration file
`ufm_slurm_prolog.py`	Python script file which creates the partition key (pkey) assignment and/or SHARP reservation when the prolog bash script is running
`ufm_slurm_epilog.py`	Python script file which removes partition key (pkey) assignment and/or SHARP reservation based on the SLURM job hosts.
`ufm_slurm_utils.py`	Utility Python file containing functions and utilities used by the integration process

Running UFM-SLURM Integration

Using the SLURM controller, execute the following commands to run your batch job:

```
$ sbatch -N4 slurm_demo.sh
Submitted batch job 1
```

Note

N4 is the number of compute nodes used to run the jobs.
`slurm_demo.sh` is the job batch file to be run.

The output and result are stored on the working directory `slurm-{id}.out` where `{id}` is the ID of the submitted job.

In the above example, after executing `sbatch` command, you can see that the submitted job ID is 1. Therefore, the output file would be stored in slurm-1.out.

Execute the following command to see the output:

```
$cat slurm-1.out
```

On the UFM side, a PKey is created in case the `pkey_allocation` parameter is set to true in the configuration file. The PKey allocation mode specifies the PKey creation type, which can either be static or dynamic. The default value is static.

- Static: Uses a statically assigned PKey when creating a new PKEY.
- Dynamic: Uses a dynamic assignment when creating a new PKey based on the `slurm-job-id`.

In addition, the UFM-SLURM will create SHARP AM reservation in case the `sharp_allocation` parameter is set to true in the `ufm_slurm.conf` file.

After the SLURM job is completed, the UFM removes the job-related partition key (PKey) assignment and SHARP reservation, if they were created.

From the moment a job is submitted by the SLURM server until its completion, a log file named `/tmp/ufm_slurm.log` logs all of the actions and errors that occurred during the execution.

This log file can be changed by modifying the `log_file_name` parameter in `/etc/slurm /ufm_slurm.conf`.

Appendix - Switch Grouping

To facilitate the logical grouping of 1U switches into a "director-like switch" group, the UFM implements a special dedicated group of interconnected 1U switches based on a YAML configuration file. This group, which is of type "superswitch", only includes 1U switches connected to each other, with some functioning as lines and others as spines.

To access the configuration file for superswitches, users can define the path in the [SubnetManager] section of the `gv.cfg` file, using the variable name "`super_switch_config_file_path`". For instance, the path can be specified as follows:
`super_switch_config_file_path=/opt/ufm/files/conf/super_switches_conf`
.

It is important to note that the file must be located in the `/opt/ufm/files file` tree, as it should be replicated between master and slave UFM servers in a high-availability configuration.

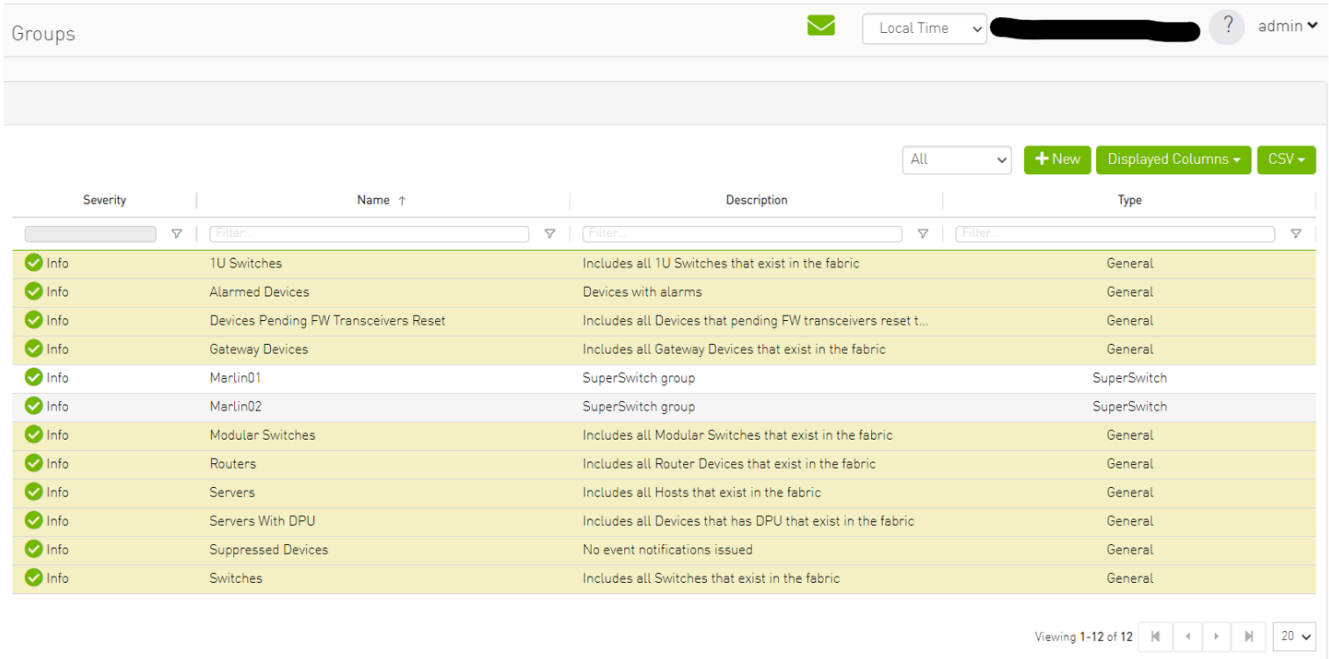
The structure of the superswitch definition should be as follows, based on the following example:

```
superswitch:
  - name: "Marlin01" # Director switch name
    description: "primary dc switch" # Free text with the customer
facing description
    location: "US, NC, DC01" # Director switch location (global
location, includes all racks/switches)
    racks: # Director switch Racks definitions
      #Rack definition
      - name: "rack A" # Director switch rack name
        location:
          dc-grid-row: "A" # formalized rack location in DC
          dc-grid-column: "1" # formalized
          comments: "left-most rack in the line" #Cutomer facing commnent on
the rack
        leafs: # List of Director switch leafs (for the rack
specified)
          - guid: "0x043f720300922a00" #required filed. Switch GUID.
            location-u: 1 # required field. Device location in
rack: "U#"
            description: "MF0;gorilla-01:MQM9700/U1" # optional field.
          - guid: "0x043f720300899cc0" #required filed. Switch GUID.
            location-u: XX # required field. Device location in
rack: "U#"
            description: "MF0;gorilla-01:MQM9700/U2" # optional field.
        spines: # List of Director switch spines (for the rack
specified)
          - guid: "0x043f720900922a00" #required filed. Switch GUID.
            location-u: 10 # required field. Device location in
rack: "U#"
            description: "MF0;gorilla-02:MQM9700/U1" # optional field.
          - guid: "0x043f720900899cc0" #required filed. Switch GUID.
            location-u: XX # required field. Device location in
rack: "U#"
            description: "MF0;gorilla-02:MQM9700/U2" # optional field.
  - name: "Marlin02" # Director switch name
```

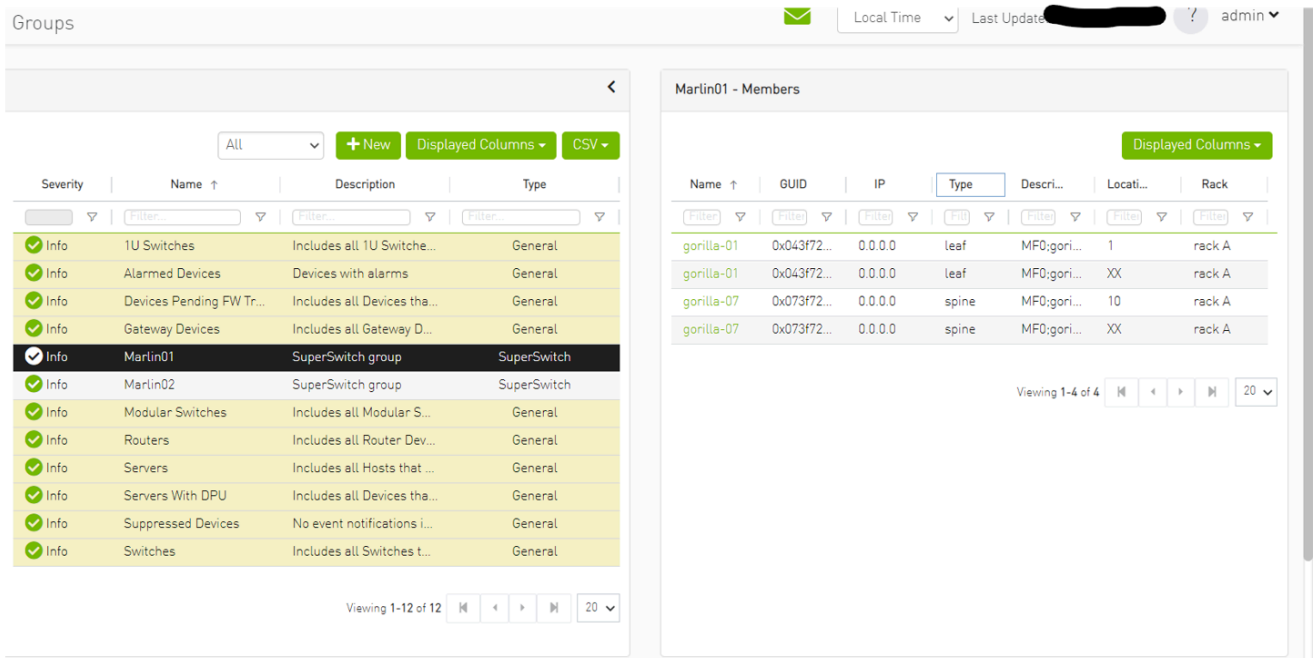
```
    description: "primary dc switch" # Free text with the customer
facing description
    location: "US, NC, DC01" # Director switch location (global
location, includes all racks/switches)
    racks: # Director switch Racks definitions
      #Rack definition
      - name: "rack B" # Director switch rack name
        location:
          dc-grid-row: "B" # formalized rack location in DC
          dc-grid-column: "1" # formalized
          comments: "left-most rack in the line" #Cutomer facing commnent on
the rack
        leafs: # List of Director switch leafs (for the rack
specified)
          - guid: "0x093f720300922a00" #required filed. Switch GUID.
            location-u: 1 # required field. Device location in
rack: "U#"
            description: "MF0;gorilla-03:MQM9700/U1" # optional field.
          - guid: "0x093f720300899cc0" #required filed. Switch GUID.
            location-u: XX # required field. Device location in
rack: "U#"
            description: "MF0;gorilla-03:MQM9700/U2" # optional field.
        spines: # List of Director switch spines (for the rack
specified)
          - guid: "0x093f720900922a00" #required filed. Switch GUID.
            location-u: 10 # required field. Device location in
rack: "U#"
            description: "MF0;gorilla-04:MQM9700/U1" # optional field.
          - guid: "0x093f720900899cc0" #required filed. Switch GUID.
            location-u: XX # required field. Device location in
rack: "U#"
            description: "MF0;gorilla-04:MQM9700/U2" # optional field
```

UI Presentation

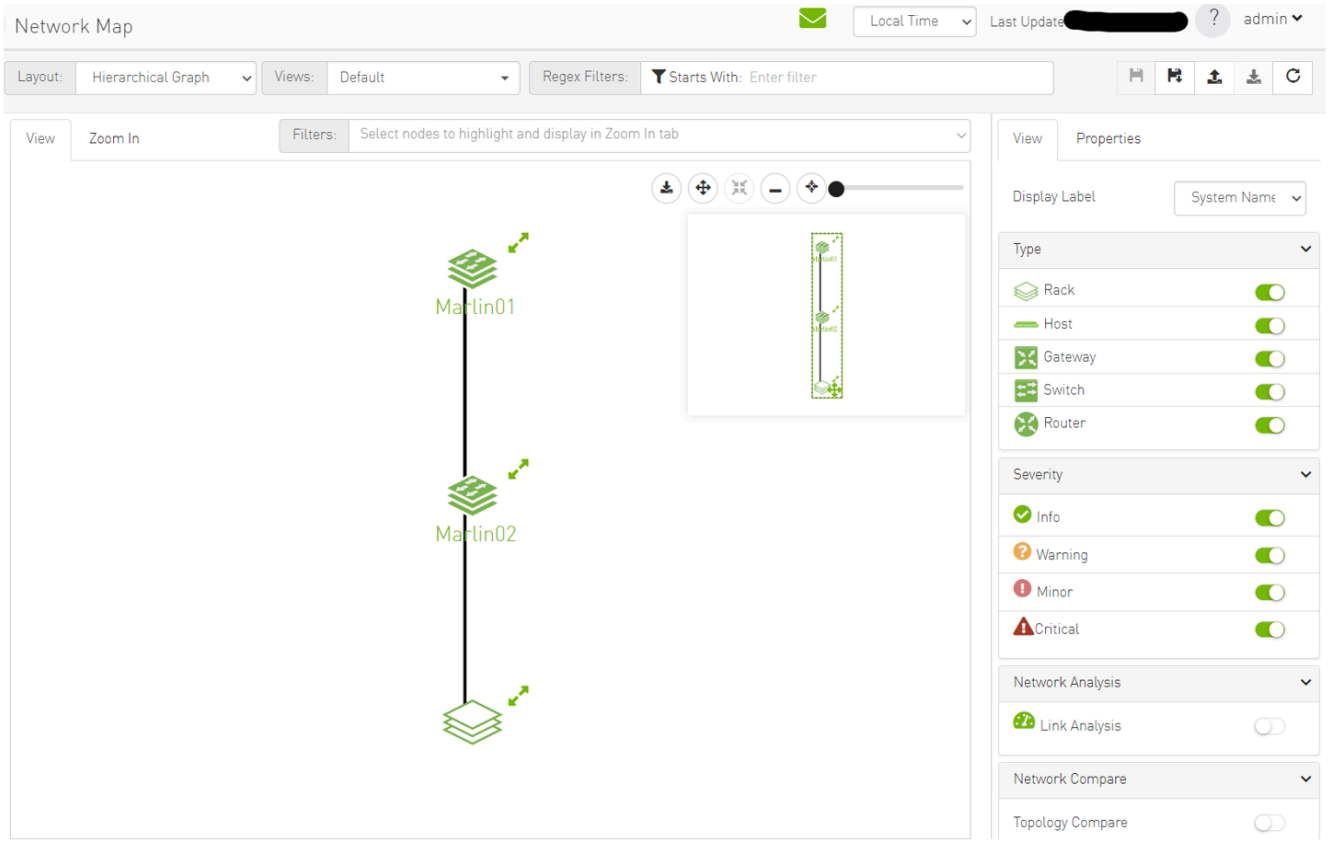
The logical grouping can be accessed under the "Groups" view, specifically listed as "SuperSwitch group" type.



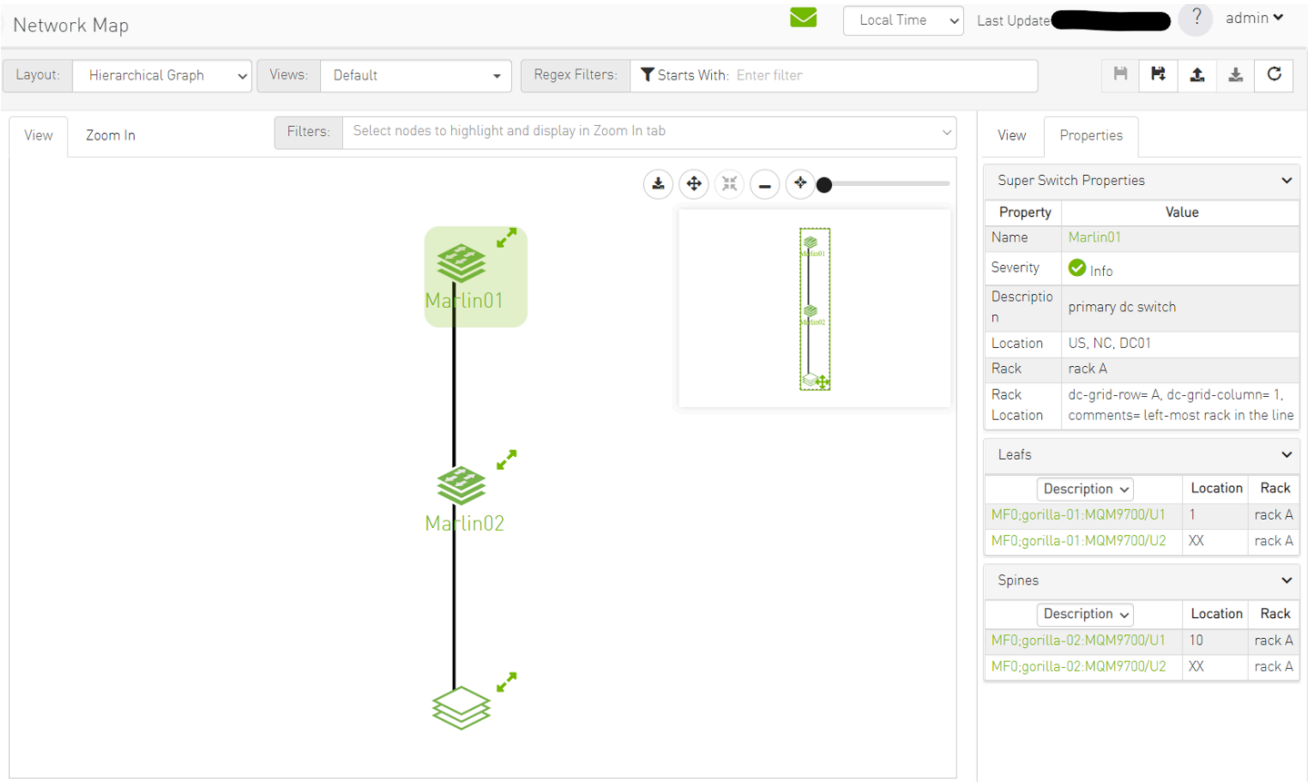
Upon selecting the group type SuperSwitch, additional columns containing information related to the SuperSwitch are added to the details view.



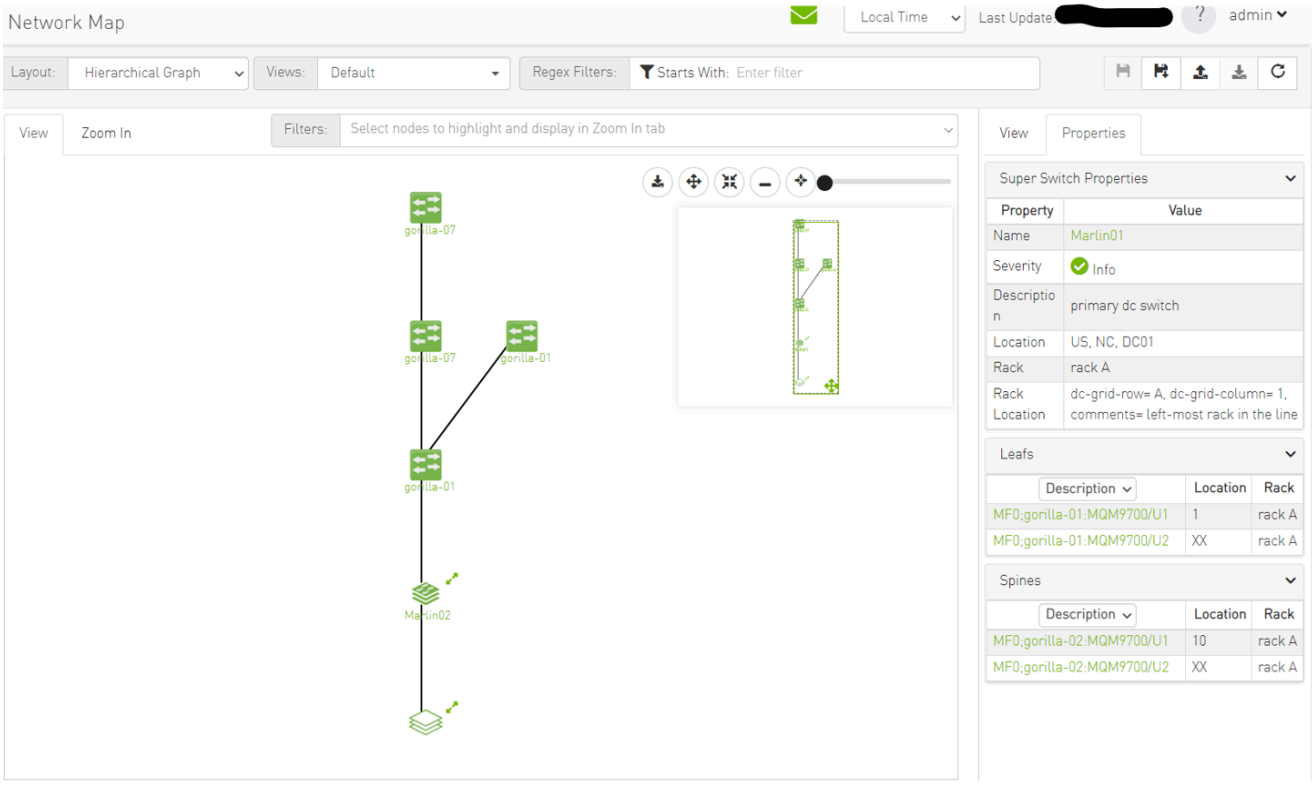
An icon for the SuperSwitch group in its collapsed view exists on the network map.

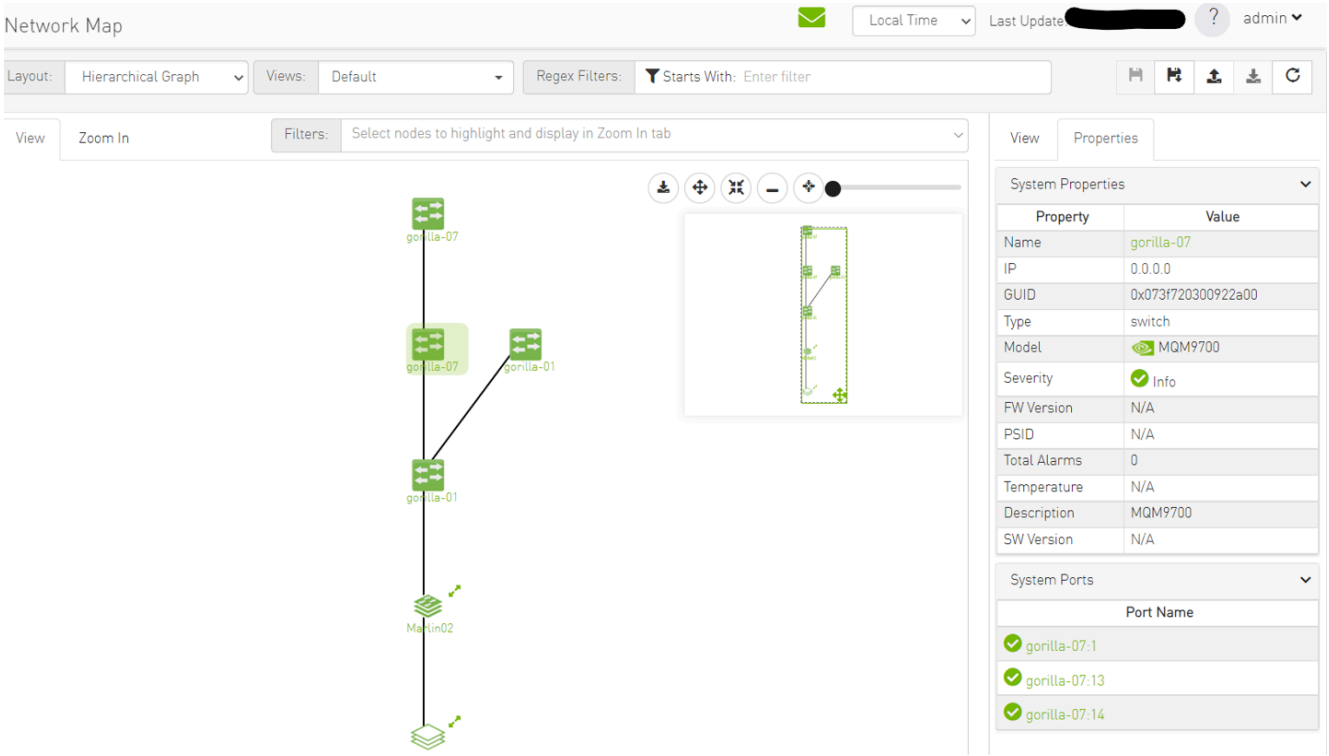


Upon selecting the SuperSwitch group, all of its properties can be viewed in the details view.

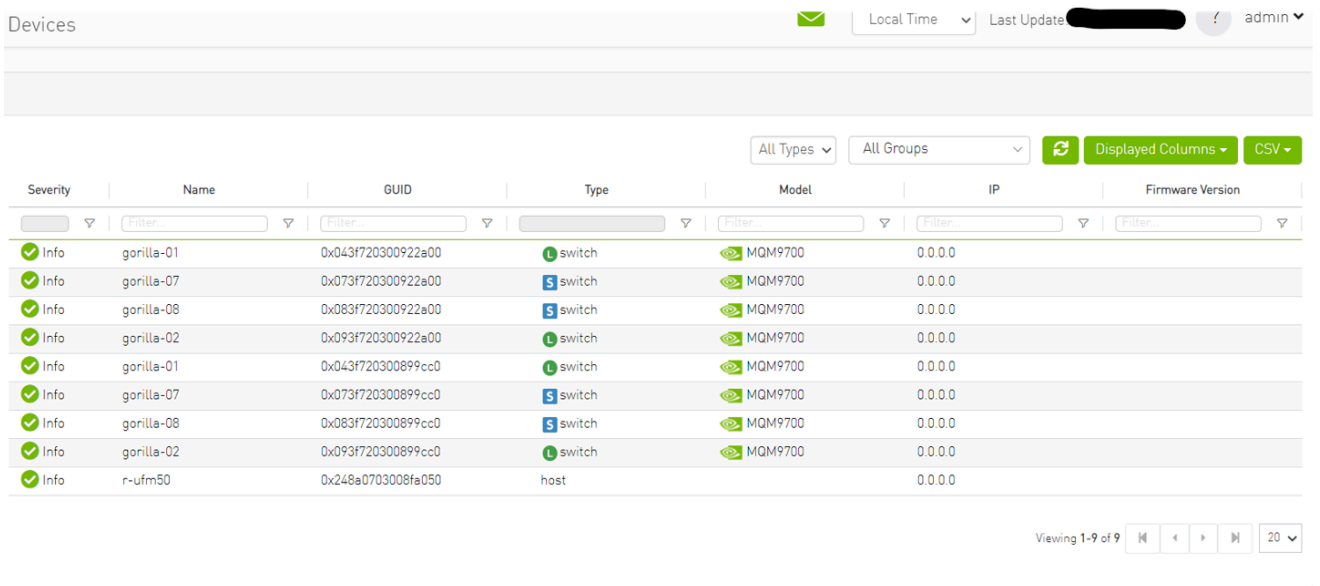


Expanding the SuperSwitch group icon displays all the switches included in the group as separate 1U switches, along with their respective properties.

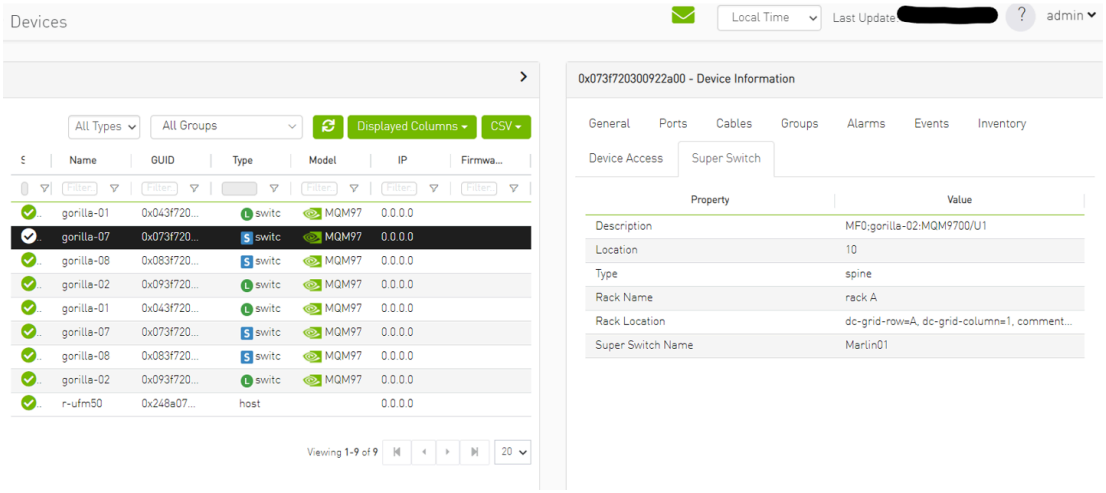




On the devices view, switches that are part of the SuperSwitch group are marked with an additional icon that indicates their role in the group. The "S" icon denotes spines, while the "L" icon denotes lines.



Selecting a switch that belongs to the SuperSwitch group in the properties view allows you to view all the switch properties related to the SuperSwitch group.



Note

Each SuperSwitch definition can include one or more racks where each embedded rack can include multiple leafs and spines switches.

Appendix – Device Management Feature Support

The following table describes the management features available on supported devices.

Feature	10 Gb Ethernet Gateway Module	Grid Director 4700/ 4200/ 4036/ 4036E v3.5	Managed IS5000 Switchesv	Managed SX6000 Switches	Externally Managed IS5000 / SX6000 Switches	Gateway BX5020	HP C-Class
Discovery							
IB L2 Discovery	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Feature	10 Gb Ethernet Gateway Module	Grid Director 4700/ 4200/ 4036/ 4036E v3.5	Managed IS5000 Switchesv	Managed SX6000 Switches	Externally Managed IS5000 / SX6000 Switches	Gateway BX5020	HP C-Class
Advanced Discovery (IP, hostname, Hosts: CPU, memory, FW version)	Yes	Yes	No	Yes	No	No	No
Ethernet access Management interface	Yes	Yes	Yes	Yes	No	No	No
Provisioning/ Configuration							
IB Partitioning (pkey)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
QoS: SL (SM configuration)	N/A	Yes	Yes	Yes	Yes	Yes	Yes
QoS: Rate Limit (SM configuration)	N/A	Yes	Yes	Yes	Yes	Yes	Yes
Interface/VIF Configuration (IP, hostname, mtu, Bonding)	N/A	N/A	N/A	N/A	N/A	No	N/A
Device Monitoring							
Device Resources: CPU, Memory, Disk	No	Yes	No	No	No	No	No

Feature	10 Gb Ethernet Gateway Module	Grid Director 4700/ 4200/ 4036/ 4036E v3.5	Managed IS5000 Switchesv	Managed SX6000 Switches	Externally Managed IS5000 / SX6000 Switches	Gateway BX5020	HP C-Class
Get device alerts (Temperature, PS, Fan) **Note:** This feature is not supported on Switch-X switches.	Yes	Yes	No	Yes	Yes	No	No
L1 (Physical Port) – Monitoring	Yes	Yes	Yes	Yes	Yes	Yes	Yes
L2-3 (Interface/VIF) – Monitoring	No	No	No	No	No	No	No
Congestion Monitoring per port (enables congestion map)	No	Yes	Yes	Yes	Yes	Yes	Yes
Congestion Monitoring per flow (Advanced Package)	No	Yes	No	No	No	No	No
Device Management							
Add/remove to/from Rack	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Add/remove to/from Logical Server	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Feature	10 Gb Ethernet Gateway Module	Grid Director 4700/ 4200/ 4036/ 4036E v3.5	Managed IS5000 Switchesv	Managed SX6000 Switches	Externally Managed IS5000 / SX6000 Switches	Gateway BX5020	HP C-Class
View/clear Alarms	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SSH terminal to device	Yes	Yes	Yes	Yes	No	No	No
Power On	No	No	No	No	No	No	No
Reboot	No	No	No	Yes (SX3606 only)	No	No	No
Shutdown	No	No	No	No	No	No	No
Port Enable/Disable	No	Yes	Yes	Yes	Yes	Yes	Yes
Firmware Upgrade (HCA & switch)	No	Yes	No	Yes (Upon SW upgrade – SX6036 only)	No	No	No
Inband Firmware Upgrade (over InfiniBand connection)	No	No	No	No	Yes	No	No
Software Upgrade (OFED & switch)	No	Yes	No	Yes (SX3606 only)	No	No	No
Protocols							

Feature	10 Gb Ethernet Gateway Module	Grid Director 4700/ 4200/ 4036/ 4036E v3.5	Managed IS5000 Switchesv	Managed SX6000 Switches	Externally Managed IS5000 / SX6000 Switches	Gateway BX5020	HP C-Class
Communication UFM Server – Device	IB/SNMP	IB/UDP /SSH	IB	IB/HTTP/ SSH	IB	IB	IB

1. QoS Rate Limit (SM configuration): On ConnectX HCAs-only, for hosts.
2. XmitWait counter monitoring requires ConnectX HCAs with firmware version 2.6 and above.
3. This feature requires that the IP address is configured.

Appendix - MAD Limiter

Overview

MAD Limiter is a security tool designed to mitigate Denial-of-Service (DoS) and Distributed Denial-of-Service (DDoS) attacks targeting the management node.

It achieves this by dropping MADs targeting QP1 based on the receive rate of each node (identified by source LID addresses) within the network.

This feature enhances network stability and security, particularly in high-traffic environments.

Key Features

- **DoS/DDoS Protection**: Automatically identifies and limits excessive packet rates from individual nodes to protect the management node.
- **Source-Based Rate Limiting**: Operates by monitoring and controlling traffic based on the source LID address of each node.

Requirements

- **Hardware Compatibility**: The MAD Limiter requires NVIDIA ConnectX-7 adapter cards for operation.
- **Container Deployment**: To run the MAD Limiter in a containerized environment, the container must be started by root.

Known Limitations

Event Logging: The current version does not generate Unified Fabric Manager (UFM) events with details of nodes exceeding the rate limits. This feature is planned for a future release.

Configuration

MAD Limiter is disabled by default. To enable this feature, please follow these instructions:

1. Open the UFM configuration file located at `/opt/ufm/files/conf/gv.cfg`.
2. Change the parameter `mad_limiter_enabled` from false to true.
3. Restart the UFM service by executing the following command:

```
systemctl restart ufm-enterprise
```

The default limits of MAD Limiter are designed to ensure that telemetry tools, such as `ibdiagnet`, operate without experiencing a slowdown. However, adjustments to these limits can be made by following the steps outlined below:

1. Access the configuration file for MAD Limiter located at `/opt/ufm/files/conf/mad_limiter/config.cfg`.
2. Modify the parameter `global_rate_limit` to set the overall rate of MADs from all nodes.
3. Adjust the parameter `node_rate_limit` to control the rate of MADs originating from a single node.

4. Alter the parameter `node_rate_shared_limit` to regulate the burst capacity of MADs from an individual node.

Appendix - UFM Versions Manager

Note

- This tool is compatible only with Oracle Linux v8.10.
- It supports UFM version 6.22.2 and later.
- The tool operates exclusively in Podman rootless mode, except for configuration backup and restore, which is supported on UFM Enterprise XDR Appliance (Gen 3.5).
- UFM Infra mode is not supported by this tool.

UFM Versions Manager is a command-line tool for managing UFM system backups, configurations, and migrations. It provides comprehensive backup and restore capabilities for UFM systems running in Podman containers, with full support for both Standalone (SA) and High Availability (HA) environments.

The tool provides four core capabilities:

- **Configuration Snapshots** - Fast config-only backups (up to 10 snapshots with rotation)
- **Full System Backup/Restore** - Complete disaster recovery with Docker images
- **Migration** - Move UFM to new hardware (SA or HA environments)
- **Upgrade** - Upgrade UFM to new versions with automatic backup and version validation

Features

- Automatic environment detection (SA vs HA)

- Rootless Podman support
- Version compatibility checking
- HA cluster support with master/standby coordination
- Dry-run mode for all operations
- Parallel processing for faster backups
- Comprehensive logging to stdout and syslog

Requirements

System Requirements

- Python 3.9 or higher (up to Python 3.12, including)
- Root access
- Podman container runtime
- UFM in Docker (only Docker/Podman UFM deployments supported)
- Sufficient disk space:
    - Config snapshots: ~10 MB per snapshot
    - Full system backups: 3-10 GB, depending on the number of plugins

HA Environment Requirements

- HA version 2.1.0 or higher (for migration operations only)
- In HA setups:
    - Upgrade operation can only run from Standby node.
    - Other commands must run from master node
    - SSH access to standby node

Supported Platforms

- UFM Enterprise XDR Appliance (Gen 3.5)
- Oracle Linux 8

Logging

Log Locations

Log	Location
stdout (Console)	• All operations logged to console output in real-time • Shows INFO, WARNING, and ERROR messages
Syslog	• WARNING and ERROR messages logged to system syslog • View with: `journalctl | grep ufm_versions_mgr`

Log Levels

- **Default:** INFO level (shows operation progress)
- **Verbose mode:** DEBUG level (use `--verbose` flag for detailed output)

Viewing Logs

```
# View syslog entries
journalctl | grep ufm_versions_mgr
```

Known Limitations

Limitation	Description
HA Configurations Not Backed Up	HA cluster configurations are not saved or restored by this tool
Full System Operations Blocked on Appliance	Full backup/restore not supported on UFM Appliance (only config snapshots allowed)
Single Full Backup	Only one full system backup maintained at a time (config snapshots: up to 10)
Strict Version Compatibility	Config snapshots can only be restored to same UFM and plugin versions
Must Run from HA Master Node	In HA environments, tool must execute from master
HA Full Backup on Local Storage	Full backups are stored on the master node's local storage (`/opt/ufm/backup/downgrade/`), NOT on DRBD shared storage. After HA failover, the new master cannot access backups created on the old master. User must manually copy the backup directory from old master to new master if restore is needed. Configuration snapshots are unaffected (stored on shared storage
Root Access Required	Tool must run as root user
UFM Docker Only	Only supports UFM running in Docker/Podman containers
HA Migration Version	Requires HA version 2.1.0 or higher
Upgrade Not Available on Appliance	The `upgrade` command is only supported on Docker/Podman UFM deployments, not on UFM Appliance 3.5
No Concurrent Operations	Only one instance can run at a time

Command Reference

Operation	Command	Downtime
Config snapshot	`ufm_versions_mgr backup --config`	No

Operation	Command	Downtime
Config restore	`ufm_versions_mgr restore --config --snapshot-number 1`	Yes
Full backup	`ufm_versions_mgr backup`	No
Full restore (SA)	`ufm_versions_mgr restore`	Yes
Full restore (HA)	`ufm_versions_mgr restore`	Yes
SA migration	`ufm_versions_mgr restore --migrate-sa --target-ip X.X.X.X`	No
HA migration	`ufm_versions_mgr restore --migrate-ha --target-ip X.X.X.X --hacluster-password PWD`	Yes
Upgrade (SA)	`ufm_versions_mgr upgrade FILE.gz`	Yes
Upgrade (HA)	`ufm_versions_mgr upgrade FILE.gz`	No

Getting Help

Description	Command
General help	`ufm_versions_mgr --help`
Backup command help	`ufm_versions_mgr backup --help`
Restore command help	`ufm_versions_mgr restore --help`
Upgrade command help	`ufm_versions_mgr upgrade --help`
Show version	`ufm_versions_mgr --version`

Note

Use `--verbose` for detailed output during operations.

Use `--dry-run` to preview any operation before executing.

Default Backup Locations

Config snapshots: `/opt/ufm/files/backup/configurations/`

Full system backups: `/opt/ufm/backup/downgrade/`

Installation and Uninstallation

UFM Versions Manager is distributed as part DEB/RPM packages or pre-installed on UFM Appliance 3.5. This page covers installation, upgrade, and uninstallation procedures.

Installation

1. Install the package. Run:

OS	Command
RHEL/Oracle Linux	`sudo rpm -ivh ufm-versions-mgr-*.x86_64.rpm`
On Ubuntu/Debian	`sudo dpkg -i ufm-versions-mgr_*_amd64.deb`

2. Verify installation. Run:

```
ufm_versions_mgr --version
```

3. Install the new version. Run:

OS	Command
RHEL/Oracle Linux	`sudo rpm -Uvh ufm-versions-mgr-NEW_VERSION.x86_64.rpm`
On Ubuntu/Debian	`sudo dpkg -i ufm-versions-mgr_NEW_VERSION_amd64.deb`

4. Verify the upgrade. Run:

```
ufm_versions_mgr --version
```

> **Note**
>
> Existing backups are preserved during upgrade.

Uninstallation

1. Remove the package. Run:

OS	Command
RHEL/Oracle Linux	`sudo rpm -e ufm-versions-mgr`
On Ubuntu/Debian	`sudo dpkg -r ufm-versions-mgr`

2. Verify the removal. Run:

```
which ufm_versions_mgr
```

> **Note**
>
> Backup data is preserved during uninstall.

Configurations Snapshot

Configuration snapshots provide fast backup and restore of UFM configurations.

The backup includes**:**

- UFM configuration files (`/opt/ufm/files/conf` all the files that are saved when calling the UFM backup.sh script)
- Plugin configurations
- UFM Install arguments
- UFM version and plugin versions

Creating Configuration Snapshot

Basic Snapshot

Create a configuration snapshot. Run:

```
ufm_versions_mgr backup --config
```

Snapshot with Label

Create a snapshot with a descriptive label. Run:

```
ufm_versions_mgr backup --config --label "My own label"
```

List Snapshots

List available snapshots. Run:

```
ufm_versions_mgr backup --config --list
```

Command Options

Option	Description
`--config`	Configuration-only backup
`--label TEXT`	Optional description for snapshot
`--list`	List existing snapshots
`--dry-run`	Preview operation without executing
`--verbose`	Enable detailed output

Restore Configuration Snapshot

Check Version Compatibility

Check if snapshot can be restored. Run:

```
ufm_versions_mgr restore --config --check-version --snapshot-
number 1
```

Note

The UFM and plugin versions must be identical for the restore process to complete successfully.

Restore Snapshot

1. Restore latest snapshot (snapshot #1). Run:

```
ufm_versions_mgr restore --config
```

2. Restore specific snapshot. Run:

```
ufm_versions_mgr restore --config --snapshot-number 2
```

Note

- A pre-restore backup is automatically created to enable rollback if needed.
- The UFM service is stopped during the restore process.

Preview Restore (Dry-Run)

Preview restore operation. Run:

```
ufm_versions_mgr restore --config --snapshot-number 1 --dry-run
```

Configuration Validation

Upon every backup operation, the tool validates the schema of some configurations.

If the files are not aligned, the operation will be blocked since the current is invalid.

Snapshot Rotation

- Maximum snapshots: 10
- Automatic rotation when creating 11th snapshot
- Oldest snapshot (snapshot-10) deleted after confirmation
- Snapshots renumbered: 9→10, 8→9, ..., 1→2, new→1

Setting Up Periodic Configuration Snapshots

Option 1: Simple Periodic Snapshot (Recommended for Most Users)

This approach uses a cron job to create a new configuration snapshot at a regular interval.

Keep in mind that UFM supports **a maximum of 10 configuration snapshots**.

Once this limit is reached, the oldest snapshot is automatically deleted when a new one is created.

Suggested Interval

Since the snapshot limit is 10:

- **Every 12 hours** → keeps ~5 days of history
- **Every 24 hours (daily)** → keeps ~10 days of history

For most environments, **daily backups** are sufficient and provide a good balance between retention and frequency.

1. Open the root user's crontab:

```
sudo crontab -e
```

2. Add one of the following lines depending on your desired frequency:

   Every 12 hours:

```
0 */12 * * * /usr/bin/ufm_versions_mgr backup --config --label "Periodic config backup"
```

   Every day at midnight:

```
0 0 * * * /usr/bin/ufm_versions_mgr backup --config --label "Daily config backup"
```

3. Save and exit. Cron will automatically schedule and run the snapshots.

Option 2: Advanced Snapshot — Only When Configuration Changes

This approach is more efficient and avoids unnecessary snapshots when UFM configuration files haven't changed.

It uses a custom shell script that:

- Computes a checksum of key UFM configuration files and plugin configurations.
- Compares it with the previous checksum.
- Creates a new snapshot **only if changes are detected**.

1. Create custom script in /usr/local/bin/

```
#!/bin/bash

# List of files to track (can be modified based on your needs and what is important)
UFM_CONF_FILES=(
        "/opt/ufm/files/conf/opensm/opensm.conf"
        "/opt/ufm/files/conf/gv.cfg"
        "/opt/ufm/files/conf/opensm/partitions.conf"
        "/opt/ufm/files/conf/opensm/opensm-health-policy.conf"
)

PLUGINS_DIR="/opt/ufm/files/conf/plugins"

CHECKSUM_FILE="/opt/ufm/ufm_conf_checksum"

# Compute combined checksum
echo  "$(date '+%F %T') - Getting checksum of tracked files..."

CURRENT=""
for f in "${UFM_CONF_FILES[@]}"; do
     [ -f "$f" ] && CURRENT="$CURRENT$(md5sum "$f" | awk '{print $1}')"
done

if [ -d "$PLUGINS_DIR" ]; then
     PLUGINS_SUM=$(find "$PLUGINS_DIR" -type f \( -name "*.conf" -o -
name "*.cfg" -o -name "*.json" -o -name "*.plugin_version" \) \
          ! -path "*/run/*" ! -path "*/lock/*" \
          -exec md5sum {} + 2>/dev/null | sort | md5sum | awk '{print
$1}')
     CURRENT="$CURRENT$PLUGINS_SUM"
fi

CURRENT_SUM=$(echo "$CURRENT" | md5sum | awk '{print $1}')
LAST_SUM=$(cat "$CHECKSUM_FILE" 2>/dev/null)

echo  "$(date '+%F %T') - Current checksum: $CURRENT_SUM, Last checksum: $LAST_SUM"
```

```
if [ "$CURRENT_SUM" != "$LAST_SUM" ]; then
        echo "$(date '+%F %T') - Configuration changed, creating snapshot..."
        /usr/bin/ufm_versions_mgr backup --config --label "Periodic backup"
        echo "$CURRENT_SUM" > "$CHECKSUM_FILE"
else
        echo "$(date '+%F %T') - No configuration change detected, skipping snapshot."
fi

echo "<============================>"
```

2. Make the Script Executable

```
chmod +x /usr/local/bin/ufm_backup_if_changed.sh
```

3. Open a crontab to edit

```
sudo crontab -e
```

4. Add:

```
0 */6 * * * /usr/local/bin/ufm_backup_if_changed.sh >> /var/log/ufm_periodic_config_backup.log
```

Script Output example

```
<============================>

2025-11-07 22:35:24 - Getting checksum of tracked files...
```

2025-11-07 22:35:24 - Current checksum: 5556f600a2513b9300e11c139c2a44d7, Last checksum: 5556f600a2513b9300e11c139c2a44d7

2025-11-07 22:35:24 - No configuration change detected, skipping snapshot.

<==========>

2025-11-07 22:36:03 - Getting checksum of tracked files...

2025-11-07 22:36:03 - Current checksum: 82f78302f08bb7a8b4ae7759f050e3f1, Last checksum: 5556f600a2513b9300e11c139c2a44d7

2025-11-07 22:36:03 - Configuration changed, creating snapshot...

[INFO] Using container runtime: podman

[INFO] HA environment detected - validating master node

[INFO] Running on HA master node - proceeding

[INFO] HA Version: 6.1.1-2

[INFO] Creating configuration snapshot...

[INFO] Label: Periodic backup

[INFO] Starting backup to /opt/ufm/files/backup/configurations/ufm-config-snapshot-temp/ufm-config-snapshot-20251107-223604.zip

[INFO] Validating UFM configuration files

[INFO] Creating temporary UFM container: ufm-config-validator

[INFO] Temporary UFM container completed successfully

[INFO] Configuration validation PASSED

[INFO] Creating temporary UFM container: ufm-restore-118159

[INFO] Temporary UFM container completed successfully

[INFO] Backup completed successfully

[INFO] Snapshot created successfully: /opt/ufm/files/backup/configurations/ufm-config-snapshot-temp/ufm-config-snapshot-20251107-223604.zip

[INFO] Creating snapshot info

[INFO] Snapshot info created successfully: /opt/ufm/files/backup/configurations/ufm-config-snapshot-temp

[INFO] Rotating snapshots

[INFO] Renaming /opt/ufm/files/backup/configurations/ufm-config-snapshot-temp to /opt/ufm/files/backup/configurations/ufm-config-snapshot-1

[INFO] Configuration snapshot created successfully!

[INFO] Snapshot location: /opt/ufm/files/backup/configurations

<============================>

Summary:

Method	Snapshot Frequency	Pros	Cons
Simple (fixed interval)	12–24h	Easy to set up; predictable history	May create redundant backups
Advanced (config change based)	On change	Efficient; avoids duplicates	Slightly more complex setup

HA Considerations

In HA environments:

- Tool automatically detects HA mode
- Must run from master node
- Backs up master configuration
- Stores in shared location

Note

- The same commands apply to both SA and HA environments.
- The tool automatically detects and adapts to the environment.

Troubleshooting

Issue	Solution
No configuration snapshots found	Create first snapshot: `sudo ufm_versions_mgr backup --config`

System Backup and Restore

Full system backup creates a complete snapshot of your UFM system including Docker images and configurations. This enables complete disaster recovery and version downgrade capability.

The backup includes:

- UFM enterprise Docker image
- All plugin Docker images
- UFM configuration files
- Plugin configurations
- Install arguments
- UFM version and plugin versions

Preserved During Restore (Not Replaced):

- PKEY configurations (`/opt/ufm/files/conf/opensm/partitions.conf`)

- Unhealthy ports configuration (`/opt/ufm/files/conf/opensm/opensm-health-policy.conf`)
- Existing backups (`/opt/ufm/files/backup` , `/opt/ufm/backup`)

> **Note**
>
> Only a single full system backup is kept at any time; creating a new backup overwrites the existing one.

Storage Location Considerations

Standalone (SA):

- Full backup stored at `/opt/ufm/backup/downgrade/1/`
- Accessible to the local system only

High Availability (HA):

- Full backup stored at `/opt/ufm/backup/downgrade/1/` on the **master node's local storage**
- **NOT stored on DRBD shared storage** (unlike configuration snapshots)
- **Important**: If HA failover occurs, the new master will NOT have access to backups created on the old master
- **User Action Required**: After failover, if you need to restore a backup created on the old master, you must manually copy `/opt/ufm/backup/downgrade/` from the old master to the new master node

Create Full System Backup

> **Note**
>
> During backup, the tool performs schema validation on some of the config files; if this step fails, the backup operation stops with a proper error.

Basic Backup

Create a full system backup. Run:

```
ufm_versions_mgr backup
```

Backup with Options

Create a backup with a label. Run:

```
ufm_versions_mgr backup --label "Before production deployment"
```

Create backup in custom location. Run:

```
ufm_versions_mgr backup --backup-dir /mnt/external/ufm-backup
```

Create backup with more workers for faster operation. Run:

```
ufm_versions_mgr backup --max-workers 10
```

Command Options

Option	Description
`--label TEXT`	Optional description for backup
`--backup-dir PATH`	Custom backup location (default: `/opt/ufm/backup/downgrade/`)
`--max-workers N`	Number of parallel workers (4-10, default: 8)
`--list`	List existing backup
`--dry-run`	Preview operation
`--verbose`	Enable detailed output

Backup List

List existing full system backup. Run:

```
ufm_versions_mgr backup --list
```

Preview Backup (Dry-Run)

Preview backup operation. Run:

```
ufm_versions_mgr backup --dry-run
```

Note

The UFM service continues to run during the backup, with no downtime.

Restore Full System

Standalone (SA) Restore

1. Preview restore. Run:

```
ufm_versions_mgr restore --dry-run
```

2. Restore full system. Run:

```
ufm_versions_mgr restor
```

Restore Process:

1. Validate backup integrity
2. Save current PKEY and health policy configurations
3. Stop UFM service
4. Uninstall current UFM
5. Load Docker images from backup
6. Install UFM with saved install arguments
7. Restore configurations

8. Restore preserved files (PKEY, health policies)
9. Start UFM service

> **Note**
>
> The UFM service is stopped for the duration of the restore operation.

High Availability (HA) Restore

Restore full system in HA environment. Run on master node:

```
ufm_versions_mgr restore
```

Restore Process

1. Validate backup integrity
2. Save current PKEY and health policy configurations
3. Stop HA cluster
4. Get standby node IP from HA configuration
5. Uninstall UFM on master
6. Load Docker images on master
7. SSH to standby: Uninstall UFM and load images
8. Install UFM on master with saved install arguments
9. Install UFM on standby with saved install arguments.
10. Restore configurations

11. Restore preserved files
12. Start HA cluster

> **Note**
>
> - The restore must be executed from the master node.
> - SSH trust with the standby node is set up automatically.
> - The entire HA cluster is offline for the duration of the restore.

Restore from Custom Location

Restore from custom backup directory. Run:

```
ufm_versions_mgr restore --backup-dir /mnt/external/ufm-backup
```

Preserved Files

During full system restore, the following files from the **current system** are preserved:

	File / Directories	Reason
PKey Configurations	`/opt/ufm/files/conf/opensm/partitions.conf`	Represents current fabric partitioning. Changing PKEYs can disrupt running workloads.

	File / Directories	Reason
Unhealthy Ports Configuration	/ opt /ufm/files/conf/opensm/opensm-health-policy.conf	Current health policies should survive version changes.
Existing Backups	`/opt/ufm/files/backup`, `/opt/ufm/backup`	Maintain ability to restore again.

Migration

UFM Versions Manager provides migration capabilities to move UFM installations to new hardware for both Standalone (SA) and High Availability (HA) environments.

Migration Types:

- **Standalone Migration** - Move UFM from one server to another (source unchanged)
- **HA Migration (RMA)** - Replace standby node in HA cluster

Standalone Migration

Requirements

Source system:

- UFM SA installation
- SSH access to target

Target system:

- Same or newer OS version
- Podman rootless is installed and configured as instructed in the podman rootless install steps.
- Sufficient disk space

Network:

- SSH connectivity between source and target

Migrate SA to Remote Host

> **Note**
>
> During the migration, UFM validates some config files' schema; in case of failure, the operation is stopped

1. Prepare the target host with OS , Docker/Podman, and SSH.
2. Preview migration. Run on source:

```
ufm_versions_mgr restore --migrate-sa --target-ip 192.168.1.30 --dry-run
```

3. Perform migration. Run on source:

```
ufm_versions_mgr restore --migrate-sa --target-ip 192.168.1.30
```

4. SSH to target and start UFM. Run:

```
ssh root@192.168.1.30
systemctl start ufm-enterprise
```

5. Verify UFM running on target. Run:

```
systemctl status ufm-enterprise
```

Command Options

Option	Description
`--migrate-sa`	Standalone migration mode
`--target-ip IP`	Target host IP address
`--dry-run`	Preview migration
`--verbose`	Enable detailed output

> **Note**
>
> - The source UFM stays running and unchanged.
> - The target UFM is installed but does not start automatically.

High Availability Migration

Requirements

- HA version 2.1.0 or higher
- Tool run from master node
- Hacluster password
- New node with OS ,Podman rootless, and SSH configured

Migrate HA Standby Node

1. Prepare new node with OS , Docker/Podman, network configuration, and SSH.
2. Preview migration. Run on master:

```
ufm_versions_mgr restore --migrate-ha \
    --target-ip 192.168.1.20 \
    --hacluster-password mypassword \
    --dry-run
```

3. Perform migration. Run on master:

```
ufm_versions_mgr restore --migrate-ha \
    --target-ip 192.168.1.20 \
    --hacluster-password mypassword
```

4. Verify cluster status. Run:

```
ufm_ha_cluster status
```

Command Options

Option	Description
`--migrate-ha`	HA migration mode
`--target-ip IP`	Target node primary IP
`--target-secondary-ip IP`	Target node secondary IP (optional)
`--hacluster-password PWD`	Hacluster user password

Option	Description
`--dry-run`	Preview migration
`--verbose`	Enable detailed output

Note

- The migration must be performed from the master node.
- The HA cluster is taken offline during the migration.
- SSH trust is configured automatically.

Dual Network Configuration

For HA with management and backup networks. Run:

```
ufm_versions_mgr restore --migrate-ha \
    --target-ip 192.168.1.20 \
    --target-secondary-ip 192.168.2.20 \
    --hacluster-password mypassword
```

Migration Process Details

SA Migration Steps

1. Create temporary backup on source
2. Establish SSH trust with target

3. Check target (cleanup if UFM exists)
4. Stream Docker images to target (parallel, compressed)
5. Transfer configuration files
6. Install UFM on target
7. Cleanup temporary backup

HA Migration Steps

1. Validate HA environment and version
2. Check/cleanup target node
3. Stop HA cluster
4. Backup images
5. Stream images to target
6. Copy service file
7. Execute HA RMA
8. Restart cluster

Upgrade

The upgrade feature allows you to upgrade UFM to a new version using the `ufm_versions_mgr upgrade` command. It provides automated upgrade with backup protection for both Standalone (SA) and High Availability (HA) environments.

Key Features

- Automatic backup before upgrade (can be skipped)

- Version validation
- SA: In-place upgrade

Requirements

General	• Valid UFM .gz image file • Not supported on UFM Appliance
HA Environment	• Must run from the standby node (not master) • Same ufm_versions_mgr version on both nodes • SSH access from standby to master

Upgrade in Standalone (SA)

SA Upgrade Process

The SA upgrade performs an in-place upgrade with the following steps:

1. Validate .gz file exists and is readable
2. Create full backup (unless --skip-backup)
3. Tag current UFM image as backup
4. Load new UFM image
5. Run container upgrade command
6. Copy service file (if rootless environment)
7. Start UFM service
8. Cleanup backup image tag

Basic SA Upgrade

Upgrade UFM on standalone system. Run:

```
ufm_versions_mgr upgrade /path/to/ufm-6.24.0.gz
```

SA Upgrade with Options

Skip automatic backup (not recommended). Run:

```
ufm_versions_mgr upgrade /path/to/ufm-6.24.0.gz --skip-backup
```

Preview upgrade without executing. Run:

```
ufm_versions_mgr upgrade /path/to/ufm-6.24.0.gz --dry-run
```

Upgrade with detailed output. Run:

```
ufm_versions_mgr upgrade /path/to/ufm-6.24.0.gz --verbose
```

Command Options

Option	Description
`UFM_GZ_FILE`	Path to new UFM .gz image file (required)
`--skip-backup`	Skip automatic backup before upgrade (not recommended)
`--dry-run`	Preview operation without executing
`--verbose`	Enable detailed output

Note

- A full backup is automatically created before the upgrade to enable rollback if needed.
- Upgrading the SA results in a short UFM downtime while the service restarts.

Upgrade in High Availability (HA)

HA Upgrade Process

The HA upgrade follows a staged approach to ensure zero downtime:

1. Detect the environment and confirm the upgrade is running on the standby node.
2. Discover the master node IP from the cluster.
3. Validate or establish SSH trust with the master node.
4. Verify that the tool is installed on the master and that the version matches.
5. Create a backup on the master via SSH (unless `--skip-backup` is specified).
6. Read backup metadata from the master.
7. Prepare the standby node:
    - Tag the current image as a backup
    - Load the new UFM image
    - Clean up the backup tag

After Step 7: Perform a manual failover, then upgrade the new standby node by loading the UFM docker image, and running the install command the tool outputed.

Downtime: None - Only when the user triggers the failover.

Basic HA Upgrade

Step 1: Upgrade the standby node. Run on standby:

```
ufm_versions_mgr upgrade /path/to/ufm-6.24.0.gz
```

Step 2: Perform manual failover. Run on master:

```
ufm_ha_cluster failover
```

Step 3: Upgrade new STANDBY (old master). Run on new standby:

```
ufm_versions_mgr upgrade /path/to/ufm-6.24.0.gz
```

HA Upgrade with Options

Skip backup on master (not recommended). Run on standby:

```
ufm_versions_mgr upgrade /path/to/ufm-6.24.0.gz --skip-backup
```

Preview upgrade. Run on standby:

```
ufm_versions_mgr upgrade /path/to/ufm-6.24.0.gz --dry-run
```

Command Options

`UFM_GZ_FILE`	Path to new UFM .gz image file (required)
`--skip-backup`	Skip backup on master node (not recommended)
`--dry-run`	Preview operation without executing
`--verbose`	Enable detailed output

Note

- The upgrade must be executed from the STANDBY node, not the master.
- The tool version must be identical on both nodes.
- The master node remains unaffected throughout the upgrade process.

Upgrade Process Details

SA Upgrade Steps (Detailed)

1. **Validate .gz file** - Check file exists, readable, valid format
2. **Create full backup** - Full system backup for rollback (unless --skip-backup)
3. **Tag current image** - Tag `ufm:latest` as `ufm:version-backup`
4. **Load new image** - Load new UFM from the .gz file
5. **Run upgrade** - Execute docker/podman run with `--upgrade` flag

6. **Copy service file** - If rootless, copy service file to systemd
7. **Start UFM** - Start ufm-enterprise service
8. **Cleanup** - Remove backup image tag

Rollback: If step 4 or 5 fails, automatic rollback re-tags backup as latest

HA Upgrade Steps (Detailed)

1. **Detect environment** - Validate running on STANDBY node (not master)
2. **Discover master IP** - Get master node IP from HA cluster
3. **Validate SSH trust** - Establish/verify SSH trust with master
4. **Validate tool on master** - Check ufm_versions_mgr installed and same version
5. **Backup on master** - SSH to master, trigger full backup (unless --skip-backup)
6. **Read master metadata** - SSH to master, read backup info for validation
7. **Prepare standby**:
    - Tag current image as backup
    - Load new UFM image from .gz
    - Remove backup tag

After Upgrade: User performs manual failover, then repeats on new standby

Documentation History

- Document Revision History
- Release Notes History

Document Revision History

Version	Date	Description
6.23.11	Dec 15, 2025	• Installation Notes - Updated the MFT version • Bug Fixes in This Release - Added bug fixes
6.23.1	Nov 10, 2025	Document Revision History in UFM Enterprise v6.23.1
6.22.2	Sep 5, 2025	Document Revision History in UFM Enterprise v6.22.2
6.22.1	Aug 8, 2025	Document Revision History in UFM Enterprise v6.22.1
6.21.2	Jun 3, 2025	Document Revision History in UFM Enterprise v6.21.2
6.21.0	May 5, 2025	Document Revision History in UFM Enterprise v6.21.0
6.20.2	Feb 23, 2025	Document Revision History in UFM Enterprise v6.20.2
6.20.1	Feb 12, 2025	Document Revision History in UFM Enterprise v6.20.1
6.20.0	Feb 10, 2025	Document Revision History in UFM Enterprise v6.20.0
6.19.5	Dec 24, 2024	Document Revision History in UFM Enterprise v6.19.5
6.19.0	Nov 7, 2024	Document Revision History in UFM Enterprise v6.19.5
6.18.2	Sep 25, 2024	Document Revision History in UFM Enterprise v6.18.2
6.18.0	Aug 14, 2024	Document Revision History in UFM Enterprise v6.18.0
6.17.0	May 7, 2024	Document Revision History in UFM Enterprise v6.17.0

Release Notes History

- Changes and New Features History

- Bug Fixes History
- Known Issues History

Changes and New Features History

Version	Link to Changes and New Features
6.22.2	Changes and New Features in v6.22.2
6.22.1	Changes and New Features in v6.22.1
6.21.2	Changes and New Features in v6.21.2
6.21.0	Changes and New Features in v6.21.0
6.20.2	Changes and New Features in v6.20.2
6.20.0	Changes and New Features in v6.20.0
6.19.6	Changes and New Features in v6.19.6
6.19.5	Changes and New Features in v6.19.5
6.18.2	Changes and New Features in v6.18.2
6.18.0	Changes and New Features in v6.18.0
6.17.0	Changes and New Features in v6.17.0

Bug Fixes History

Version	Link to Documentation
6.22.2	Bug Fixes in UFM Enterprise User Manual v6.22.2
6.22.1	Bug Fixes in UFM Enterprise User Manual v6.22.1
6.21.2	Bug Fixes in UFM Enterprise User Manual v6.21.2
6.21.0	Bug Fixes in UFM Enterprise User Manual v6.21.0
6.20.2	Bug Fixes in UFM Enterprise User Manual v6.20.2
6.20.1	Bug Fixes in UFM Enterprise User Manual v6.20.1
6.20.0	Bug Fixes in UFM Enterprise User Manual v6.20.0
6.19.6	Bug Fixes in UFM Enterprise User Manual v6.19.6

Version	Link to Documentation
6.19.5	Bug Fixes in UFM Enterprise User Manual v6.19.5
6.19.0	Bug Fixes in UFM Enterprise User Manual v6.19.0
6.18.2	Bug Fixes in UFM Enterprise User Manual v6.18.2
6.18.0	Bug Fixes in UFM Enterprise User Manual v6.18.0
6.17.0	Bug Fixes in UFM Enterprise User Manual v6.17.0

Known Issues History

Version	Link to Documentation
6.22.2	Known Issues in UFM Enterprise User Manual v6.22.2
6.22.1	Known Issues in UFM Enterprise User Manual v6.22.1
6.21.2	Known Issues in UFM Enterprise User Manual v6.21.2
6.21.0	Known Issues in UFM Enterprise User Manual v6.21.0
6.20.2	Known Issues in UFM Enterprise User Manual v6.20.2
6.20.1	Known Issues in UFM Enterprise User Manual v6.20.1
6.20.0	Known Issues in UFM Enterprise User Manual v6.20.0
6.19.6	Known Issues in UFM Enterprise User Manual v6.19.6
6.19.5	Known Issues in UFM Enterprise User Manual v6.19.5
6.19.0	Known Issues in UFM Enterprise User Manual v6.19.0
6.18.2	Known Issues in UFM Enterprise User Manual v6.18.2
6.18.0	Known Issues in UFM Enterprise User Manual v6.18.0
6.17.0	Known Issues in UFM Enterprise User Manual v6.17.0

EULA, Legal Notices and 3rd Party Licenses

Legal Notice

Third-Party Licenses

License Agreement

This license is a legal agreement (“Agreement”) between you and Mellanox Technologies, Ltd. ("NVIDIA") and governs the use of the NVIDIA UFM software and materials provided hereunder (“SOFTWARE”). If you are entering into this Agreement on behalf of a company or other legal entity, you represent that you have the legal authority to bind the entity to this Agreement, in which case “you” will mean the entity you represent.

You agree to use the SOFTWARE only for purposes that are permitted by (a) this license, and (b) any applicable law, regulation, or generally accepted practices or guidelines in the relevant jurisdictions.

1. License. Subject to the terms and conditions of this Agreement and payment of applicable subscription fee, NVIDIA MELLANOX grants you a personal, non-exclusive, non-sublicensable (except as provided in this Agreement), non-transferable, non-commercial license to install and use the Software for your internal business purposes for configuring, operating, and managing your InfiniBand network and not for further distribution.

2. Authorized Users. You may allow access and use of the Software to: (i) employees and contractors of your entity provided that the access and use of the Software is made from your secure network to perform work on your behalf and (ii) If you are an academic institution you may allow users enrolled or employed by the academic institution to access and use the Software from your secure network (“Authorized Users”). You hereby undertake to be responsible and liable for any non-compliance with the terms of this Agreement by your Authorized Users. You further agree to immediately resolve any non-compliance by your Authorized Users of which you become aware and endeavor take necessary steps to prevent any new occurrences.

3. Limitations Your license to use the SOFTWARE is restricted as follows:

3.1 The SOFTWARE is licensed for your use in systems with the registered NVIDIA Host Channel Adapter (HCA) Products or related adapter products.

3.2 Each copy of the SOFTWARE shall be limited to the number of HCAs indicated in the applicable purchase order.

3.3 You may use software back-up utilities to make one back-up copy of the Software Product. You may use the back-up copy solely for archival purposes

3.4 You may not use the SOFTWARE in conjunction with a number of managed nodes or managed devices which is beyond the allowable limit or copy the SOFTWARE on additional hardware. You shall not use any features which are not included in the scope of this Agreement as described in the accompanying documentation.

3.5 You may not reverse engineer, decompile or disassemble, or remove copyright or other proprietary notices from any portion of the SOFTWARE or copies of the SOFTWARE.

3.6 You may not disclose the results of benchmarking, competitive analysis, regression, or performance data relating to the SOFTWARE without the prior written permission from NVIDIA Mellanox.

3.7 Except as expressly provided in this license, you may not copy, sell, rent, sublicense, transfer, distribute, modify, or create derivative works of any portion of the SOFTWARE. For clarity, unless, you have an agreement with NVIDIA Mellanox for this purpose you may not distribute or sublicense the SOFTWARE as a stand-alone product.

3.8 You may not bypass, disable, or circumvent any technical limitation, encryption, security, digital rights management, or authentication mechanism in the SOFTWARE.

3.9 You may not use the Software in any manner that would cause it to become subject to an open source software license. As examples, licenses that require as a condition of use, modification, and/or distribution that the Software be: (i) disclosed or distributed in source code form; (ii) licensed for the purpose of making derivative works; or (iii) redistributable at no charge.

3.10 Unless you have an agreement with NVIDIA Mellanox for this purpose, you may not use the Software with any system or application where the use or failure of the system or application can reasonably be expected to threaten or result in personal injury, death, or catastrophic loss. Examples include use in avionics, navigation, military, medical, life support or other life critical applications. NVIDIA Mellanox does not design, test, or manufacture the Software for these critical uses and NVIDIA

Mellanox shall not be liable to you or any third party, in whole or in part, for any claims or damages arising from such uses.

3.11 You agree to defend, indemnify and hold harmless NVIDIA Mellanox and its affiliates, and their respective employees, contractors, agents, officers and directors, from and against any and all claims, damages, obligations, losses, liabilities, costs or debt, fines, restitutions and expenses (including but not limited to attorney's fees and costs incident to establishing the right of indemnification) arising out of or related to your use of the Software outside of the scope of this license, or not in compliance with its terms.

4. Updates. NVIDIA Mellanox may, at its option, make available patches, workarounds, or other updates to this Software. Unless the updates are provided with their separate governing terms, they are deemed part of the Software licensed to you as provided in this license. You agree that the form and content of the Software that NVIDIA Mellanox provides may change without prior notice to you. While NVIDIA Mellanox generally maintains compatibility between versions, NVIDIA Mellanox may in some cases make changes that introduce incompatibilities in future versions of the SOFTWARE.

5. Pre-Release Versions. Software versions identified as alpha, beta, preview, early access or otherwise as pre-release may not be fully functional, may contain errors or design flaws, and may have reduced or different security, privacy, availability, and reliability standards relative to commercial versions of NVIDIA Mellanox software and materials. You may use a pre-release Software version at your own risk, understanding that these versions are not intended for use in production or business-critical systems. NVIDIA Mellanox may choose not to make available a commercial version of any pre-release Software. NVIDIA Mellanox may also choose to abandon development and terminate the availability of a pre-release Software at any time without liability.

6. Third-Party Components. The Software may include third-party components with separate legal notices or terms as may be described in proprietary notices accompanying the Software or as provided in an Exhibit to this Agreement. If and to the extent there is a conflict between the terms in this license and the third-party license terms, the third-party terms control only to the extent necessary to resolve the conflict. For details regarding the third party components, please review Exhibit A.

7. OWNERSHIP

   7.1 NVIDIA Mellanox or its licensors reserves all rights, title, and interest in and to the Software not expressly granted to you under this license NVIDIA Mellanox and its suppliers hold all rights, title, and interest in and to the Software, including their respective intellectual property rights. The Software is copyrighted and protected by

the laws of the United States and other countries, and international treaty provisions.

7.2 Subject to the rights of NVIDIA Mellanox and its suppliers in the Software, you hold all rights, title, and interest in and to your applications and your derivative works of the sample source code delivered in the Software including their respective intellectual property rights.

8. You may, but are not obligated to, provide to NVIDIA Mellanox Feedback. “Feedback” means suggestions, fixes, modifications, feature requests or other feedback regarding the Software. Feedback, even if designated as confidential by you, shall not create any confidentiality obligation for NVIDIA Mellanox. NVIDIA Mellanox and its designees have a perpetual, non-exclusive, worldwide, irrevocable license to use, reproduce, publicly display, modify, create derivative works of, license, sublicense, and otherwise distribute and exploit Feedback as NVIDIA Mellanox sees fit without payment and without obligation or restriction of any kind on account of intellectual property rights or otherwise.

9. No Warranties. THE SOFTWARE IS PROVIDED AS-IS. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW NVIDIA MELLANOX AND ITS AFFILIATES EXPRESSLY DISCLAIM ALL WARRANTIES OF ANY KIND OR NATURE, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE. NVIDIA MELLANOX DOES NOT WARRANT THAT THE SOFTWARE WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION THEREOF WILL BE UNINTERRUPTED OR ERROR-FREE, OR THAT ALL ERRORS WILL BE CORRECTED.

10. Limitations of Liability. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW NVIDIA MELLANOX AND ITS AFFILIATES SHALL NOT BE LIABLE FOR ANY SPECIAL, INCIDENTAL, PUNITIVE OR CONSEQUENTIAL DAMAGES, OR FOR ANY LOST PROFITS, PROJECT DELAYS, LOSS OF USE, LOSS OF DATA OR LOSS OF GOODWILL, OR THE COSTS OF PROCURING SUBSTITUTE PRODUCTS, ARISING OUT OF OR IN CONNECTION WITH THIS LICENSE OR THE USE OR PERFORMANCE OF THE SOFTWARE, WHETHER SUCH LIABILITY ARISES FROM ANY CLAIM BASED UPON BREACH OF CONTRACT, BREACH OF WARRANTY, TORT (INCLUDING NEGLIGENCE), PRODUCT LIABILITY OR ANY OTHER CAUSE OF ACTION OR THEORY OF LIABILITY, EVEN IF NVIDIA MELLANOX HAS PREVIOUSLY BEEN ADVISED OF, OR COULD REASONABLY HAVE FORESEEN, THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT WILL NVIDIA MELLANOX AND ITS AFFILIATES TOTAL CUMULATIVE LIABILITY UNDER OR ARISING OUT OF THIS LICENSE EXCEED US$10.00. THE NATURE OF THE LIABILITY OR THE NUMBER OF CLAIMS OR SUITS SHALL NOT ENLARGE OR EXTEND THIS LIMIT.

11. T Your rights under this license will terminate automatically without notice from NVIDIA Mellanox (a) upon expiration of your subscription, (b) if you fail to comply with

any term and condition of this license including non-payment of applicable fees, or (c) if you commence or participate in any legal proceeding against NVIDIA Mellanox with respect to the Software. NVIDIA Mellanox may terminate this license with advance written notice to you, if NVIDIA Mellanox decides to no longer provide the Software in a country or, in NVIDIA Mellanox's sole discretion, the continued use of it is no longer commercially viable. Upon any termination of this license, you agree to promptly discontinue use of the Software and destroy all copies in your possession or control. All provisions of this license will survive termination, except for the license granted to you.

12. Product Support. Product support for the Software Product is provided by NVIDIA Mellanox or its authorized agents under the applicable subscription license, in accordance with NVIDIA Mellanox's standard support and maintenance terms and conditions. For product support, please refer to NVIDIA Mellanox support number provided in the documentation.

13. Applicable Law. This license will be governed in all respects by the laws of the United States and of the State of Delaware, without regard to the conflicts of laws principles. The United Nations Convention on Contracts for the International Sale of Goods is specifically disclaimed. You agree to all terms of this license in the English language. The state or federal courts residing in Santa Clara County, California shall have exclusive jurisdiction over any dispute or claim arising out of this license. Notwithstanding this, you agree that NVIDIA Mellanox shall still be allowed to apply for injunctive remedies or urgent legal relief in any jurisdiction.

14. No Assignment. This license and your rights and obligations thereunder may not be assigned by you by any means or operation of law without NVIDIA Mellanox's permission. Any attempted assignment not approved by NVIDIA MELLANOX in writing shall be void and of no effect. NVIDIA Mellanox may assign, delegate, or transfer this license and its rights and obligations, and if to a non-affiliate you will be notified.

15. E The Software is subject to United States export laws and regulations. You agree to comply with all applicable U.S. and international export laws, including the Export Administration Regulations (EAR) administered by the U.S. Department of Commerce and economic sanctions administered by the U.S. Department of Treasury's Office of Foreign Assets Control (OFAC). These laws include restrictions on destinations, end-users and end-use. By accepting this license, you confirm that you are not currently residing in a country or region currently embargoed by the U.S. and that you are not otherwise prohibited from receiving the Software.

16. Government Use. The Software is, and shall be treated as being, "Commercial Items" as that term is defined at 48 CFR § 2.101, consisting of "commercial computer software" and "commercial computer software documentation", respectively, as such terms are used in, respectively, 48 CFR § 12.212 and 48 CFR §§ 227.7202 & 252.227-

7014(a)(1). Use, duplication or disclosure by the U.S. Government or a U.S. Government subcontractor is subject to the restrictions in this license pursuant to 48 CFR § 12.212 or 48 CFR § 227.7202. In no event shall the US Government user acquire rights in the Software beyond those specified in 48 C.F.R. 52.227-19(b)(1)-(2).

17. Please direct your legal notices or other correspondence to NVIDIA Corporation, 2788 San Tomas Expressway, Santa Clara, CA, 95051 United States of America, Attention: Legal Department and to: NBU-Legal_Notices@exchange.nvidia.com

18. Entire Agreement. This license is the final, complete, and exclusive agreement between the parties relating to the subject matter of this license and supersedes all prior or contemporaneous understandings and agreements relating to this subject matter, whether oral or written. If any court of competent jurisdiction determines that any provision of this license is illegal, invalid, or unenforceable, the remaining provisions will remain in full force and effect. Any amendment or waiver under this license shall be in writing and signed by representatives of both parties.

(v APR. 28, 2022)

Exhibit A

SOFTWARE includes the following open source/ freeware that are subject to specific license conditions listed in the table below, which may be updated from time to time by NVIDIA Mellanox or the Open Source provider. The below table is current as of December 2021. To obtain source code for software provided under licenses that require redistribution of source code, including the GNU General Public License or for update queries contact: http://www.mellanox.com/page/gnu_code_request .This offer is valid for a period of three (3) years from the date of the distribution of this product by NVIDIA Mellanox.

Component name	Version
@candlefw/wick	0.8.12
ABSender	master-20121122
APBS	apbs-0.3.1
Amazon Kindle Source Code	6.2

Component name	Version
Amiga Research OS	20120217
Apache ActiveMQ	2.2.2
Apache HTTP Server	1.3.7, 1.3.8
Apache HTTP Server	2, 2.0.11, 2.0.23, 2.0.25, 2.0.26, 2.0.30, 2.0.33, 2.0.35, 2.0.36 2.0.39, 2.0.40, 2.0.41, 2.0.43, 2.1.0
Apache HTTP Server	2.0.59, 2.1.1, 2.1.10, 2.1.2, 2.1.3, 2.1.4, 2.1.5, 2.1.6, 2.1.7, 2.1 2.2.1, 2.2.2 2.2.12, 2.2.13, 2.2.14, 2.2.15, 2.2.16, 2.2.17, 2.2.22, 2.2.26, 2.2.3, 2.2.4, 2.2. 2.2.7, 2.2.9, 2.3.0, 2.3.1, 2.3.4
Apache HTTP Server	STRIKER_2_1_0_RC1
Apache Portable Runtime	0.9.13, 0.9.15, 1.2.0, 1.2.10, 1.2.11, 1.2.12, 1.2.7, 1.2.8, 1.2.9 1.3.1, 1.3.10, 1.3.12, 1.3.2, 1.3.3, 1.3.4, 1.3.5, 1.3.7, 1.3.8, 1.3.9, 1.4.7, 1.5.1, 1.5.2; APR_1_0_RC2; JCW_0_9_5_PRE1
Apache Portable Runtime	0.9.4 APACHE_2_0_37APACHE_2_0_40APACHE_2_0_44APACHE

Component name	Version
Apache Portable Runtime	APU_1_0_RC1
Apache Tomcat	1.1.0, 6.0.24
BIND9 (Berkeley Internet Name Domain)	9.9.11
Berkeley DB	4.5.20
Chromium (Google Chrome)	32.0.1700.102
Cinder	v0.8.0

Component name	Version
Clonezilla	1.2.10
Cron	3.0pl1
CyanogenMod - android_external_busybox	cm-10.1-M1, cm-10.1-M2
D-Bus	1.2.6
DHCP (ISC)	4.3.6
Darik's Boot and Nuke	dban-2.0.0
Debian Games	11.04.1+repack
FLAC - Free Lossless Audio Codec	flac-1.1.1-beta1-src
FarGroup/FarManager	builds/3.0.2890
FreeBSD	5.5, 6, 9.0-BETA1, release/11.2.0,12.2, 2.2.0, 2.2.6, 5.0.0cvs

Component name	Version
FreeBSD	bsd_44_lite
FreeBSD Ports	RELEASE_4_5_0 RELEASE_4_6_0
FreeNAS	0.7
GD	2.0.1beta, 2.0.32, 2.0.33, 2.0.34RC1, 2.0.35, 2.0.35RC5
GD	2.0.36_rc1
GLib	1.2.3, 2.14.6, 2.19.5
GNU Compiler Collection	4.7.0
GNU Libtool	1.4.1
GNU Parted	1.8.1, 2.4
GNU Parted	2.4
Gentoo Linux	release_1_3_17

Component name	Version
Heimdal Kerberos	heimdal-0.0n
HipHop Virtual Machine for PHP	HHVM-3.1.0
Kablink	1.1 Alpha1
Less	374
Less	429
LineageOS	cm-10.1.0-RC1
Linux Test Project	2004
Linux-Pam	0.59, 0.72, 0.74, 0.76, 0.99.1.0, 0.99.2.0, 0.99.4.0, 0.99.5.0, 0 0.99.6.2, 1.0.0
Linux-Pam	1.0.1

Component name	Version
MapServer	rel-1-0-0
Merruk-Technology	2.0-20121113
MinGW - Minimalist GNU for Windows	binutils-2.20
MythTV	v0.13
NFS	1.0.6
Net-SNMP	5.0.9, 5.4.2.1, 5.5.2.pre1, 5.7.3, END-UCD-SNMP. Ext-5-3-cvs20050331, JBPN-CBL-1, 5.0.11.1, 5.2.2
Net-SNMP	5.1.2, Ext-5-0, Ext-5-0-2, Ext-5-0-4, Ext-5-4-1-1, V4-2-patch merge2
Net-SNMP	Ext-5-0, Ext-5-0-4

Component name	Version
Net-SNMP	Ext-5-4-1-1
Net-SNMP	V4-2-patches-merge2
Net-SNMP	5.2.4 source code, 5.2.5 pre-releases, 5.3.1, 5.3.2 pre-releas pre-releases, 5.5, Ext-4-0-pre5, Ext-4-1-pre1, Ext-5-0-2-pre1,Ext-5-0-7-pre1, Ext-5-0-8-pre1, Ext-5-2-2rc 2-pre2, Ext-5-2-pre3, Ext-5-3-pre1, Ext-5-3-pre3, Ext-5-3-pre4, Ext-5-4-1-pre1, Ext-5-4-1-pre3, Ext-5-4-pre1, 4-pre1, Ext-5-4-pre4, Ext-5-5-pre1, Ext-5-5-pre2, Ext-5-5-pre3, Ext-5-5-rc1, Ext-5-5-rc3, 5.3.0.1. 5.8.1.pre1, 5
NetBSD	1.1, 1.5, 2
OpenFabrics Enterprise Distribution - OFED	1.2, 1.5, 3.3.2018
OpenFabrics Enterprise Distribution - OFED	3.1.8

Component name	Version
OpenLDAP	2.4.44
OpenSSH	5.3p1, 7.4p1,7.7, 7.7p1, 7.8, 7.8p1, 7.9, 7.9p1, 8.0p1, pre-re
OpenSSH	7.2p2, 7.6p1
OpenWrt	12.09, 14.07
PCRE	7.1, 7.4, 7.6
PCRE	4, 7.6, 7.7, 7.8
PHP	MERGE_FROM_NEW_LOOK_2001_TAG_1
PortableApps.com	WinMerge 2.10.0 , 2.6.12Source
Python programming language	v2.4a2
Qualcomm Kernel Tree for MSM/QSD family and Android 4.4	ath-201808291719
TACACS+ client library and PAM module	1.2.10, 1.2.9
Stephane-D/SGDK	V1.62

Component name	Version
TACACS+ client library and PAM module	1.3.2
Tarifa	Tarifa019.tar
Tcl/Tk	8.1.1
Tecla Library	1.2.3, 1.4.0, 1.4.1, 1.5.0, 1.6.0, 1.6.2
The GWARE Project	2.10.2
TizenRT	1.1_Public_Release
UC-7402.7408.7410.7420-LX Plus Source	20100210
WinMerge	2.11.1.7
XAMPP	1.4.5, 1.6.4
XAMPP	1.6.4
XQilla	1.1.0
YaST	broken/svn/openSUSE-9_3

Component name	Version
Zile (Zile is Lossy Emacs)	1.4, 1.5, 1.5.2, 1.5.3, 1.6, 1.6.1, 1.6.2
afwall	V2.6.0.1, v2.8.0, v2.9.0, v2.9.1, v2.9.4
alcatel	20
alcatel	4/18/2012, 20120601, 918
appweb	3.0B.0-0
asuswrt-merlin	376.48, 376.48, 380.62
asuswrt-merlin	378.51, 380.62
avahi	v0.6
awokengazebo-lfi	lfi-20080723
beefproject	beef-0.4.3.1
bitswitcher	0.2.0, 0.3.0, 0.3.3

Component name	Version
buildroot-kindle	master-20130206
busybox	1.10.0, 1.12.0, 1.2.0, 1.4.0, 1.5.0, 1.8.0, 1_11_0, 1_13_0, 1 1 1_16_0, 1_17_1 17 1, 1_17_2, 1_18_0, 1_18_2, 1_19_0, 1_19_1, 1_19_4, 1_20_2, 1_21_0, 1_24_0, 1_29_0, 1_7_0
busybox	1_14_0, 1_15_0, 1_17_0, 1_19_2, 1_19_3, 1_20_0, 1_20_1,
catboost/catboost	v0.2
curl	7.16.0
decorator-ko	26, 28
file	5.22
fluxcapacitor	0
fvpatwds : fvpat Webdev Server	fvpatwds v0.1.4
generator-minxing	1.0.2
geonkick	2.3.6
hostap-ct	If-5.1.7, If-5.3.3, If-5.3.3b, If-5.3.4, If-5.3.5

Component name	Version
hostapd	hostap_0_5_2, hostap_0_5_3, hostap_0_5_6,
howl	0.9.4, 0.9.6, 0.9.7, 0.9.9, 1.0.0,0.9.3, 0.9.1
illumos-joyent	20121101
krb5/krb5	1.0-alpha3, 1.0-beta2, 1.0-beta5
libevent - an event notification library	0.1, 1.0d, 1.0e,1.4.1-beta
libexpat	1.95.0, 1.95.1, 1.95.2, 2.0.0, v19991013
libexpat	V19991013
linux-yocto-dev	v2.6.12
littlekernel-m900-eclair	master-20110326
lmdb	0.9.18

Component name	Version
math-linux	0.0.1
mod_dup	2.5.0
ngx_pagespeed	1.9.32.4-dbg-ssl-crash
nss_ldap	253
opensm	3.3.17
pGina	Plugin Bundle 05-11-2006
pam_radius	release_2_0_0
protovis	3.3.1
root-project	5-13-04e

Component name	Version
rsyslog	sysklogd-1.4.1-import
rtems-libbsd	5.1
rtl8186 - toolchain	0.5.5_src
snake-os	0.9
ssmtp	2.61
svn://svn.tug.org/texlive/trunk	texlive-2009.0
util-linux	2.11q, 2.11w, 2.12a, 2.13-pre1
videolan/vlc	0.5.0

Component name	Version
wakame-vdc	v13.06.0
wpa_supplicant - IEEE 802.1X, WPA, WPA2, RSN, IEEE 802.11i	0.5.0, 0.5.3, 0.5.5, 0.5. 6, 0.5.8, 0.6.0, 0.6.10, 0.6.2, 0.6.3, 0.6.4, 0.6.8, 0.7.0, 0.7.1, 0. 0.7.3, 1, 2, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.7+git20190108+11ce7a1, , 2.7~git20180504+60a5737, 2.7~git20180606+b915f2c, 2.7~git20180706+420b5dd
xorp.ct	1.5, xorp-1-7
zeroconf	0.9

Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation (“NVIDIA”) makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer (“Terms of Sale”). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer’s own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer’s sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer’s product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, “MATERIALS”) ARE BEING PROVIDED “AS IS.” NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF

ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

Trademarks

NVIDIA and the NVIDIA logo are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

© Copyright 2025, NVIDIA. PDF Generated on 01/15/2026