



NVIDIA MLNX-OS User Manual v3.10.5000

Table of contents

Overview	3
Getting Started	8
User Interfaces	28
System Management	29
Network Management Interfaces	30
Virtualization	71
Telemetry, Monitoring, and Debuggability	92
User Management, Authentication, & Security	93
InfiniBand Switching	94
Appendixes	95
Document Revision History	96

Welcome to MLNX-OS Documentation

NVIDIA® MLNX-OS® operating system, enables the management and configuration of NVIDIA's InfiniBand switch system platforms.

MLNX-OS provides a full suite of management options, including support for UFM® (Unified Fabric Manager), SNMPv1, 2, 3, and web user interface (Web UI). In addition, it incorporates a familiar industry-standard CLI, which enables administrators to easily configure and manage the system.

These pages provide information about the scope, organization, and command line interface of MLNX-OS as well as configuration examples.

Software Download

To download the latest software, log in to the following website: enterprise-support.nvidia.com/s/ .

For common questions about the Enterprise Account please see the following webpage: nvid.nvidia.com/NvidiaUtilities/#/needHelp

Technical Support

Customers who purchased NVIDIA products directly from NVIDIA are invited to contact us through the following methods:

- URL: www.nvidia.com → Support
- E-mail: enterprisesupport@nvidia.com

Customers who purchased NVIDIA M-1 Global Support Services, please see your contract for details regarding Technical Support.

Customers who purchased NVIDIA products through an NVIDIA-approved reseller should first seek assistance through their reseller.

Document Revision History

A list of the changes made to the User Manual are provided in [User Manual Revision History](#).

Overview

Intended Audience

These pages are intended for network administrators who are responsible for configuring and managing NVIDIA's switch platforms.

Related Documentation

The following table lists the documents referenced in this User Manual.

Document Name	Description
System Hardware User Manual	This document contains hardware descriptions, LED assignments, and hardware specifications, among other things
Switch Product Release Notes	Please look up the relevant switch system/series release note file

Terminology

Term	Description
AAA	Authentication, Authorization, and Accounting: <ul style="list-style-type: none">• Authentication—verifies user credentials (username and password)• Authorization—grants or refuses privileges to a user/client for accessing specific services• Accounting—tracks network resources consumption by users
ARP	Address Resolution Protocol. A protocol that translates IP addresses into MAC addresses for communication over a local area network (LAN).
CLI	Command Line Interface. A user interface in which you type commands at the prompt.

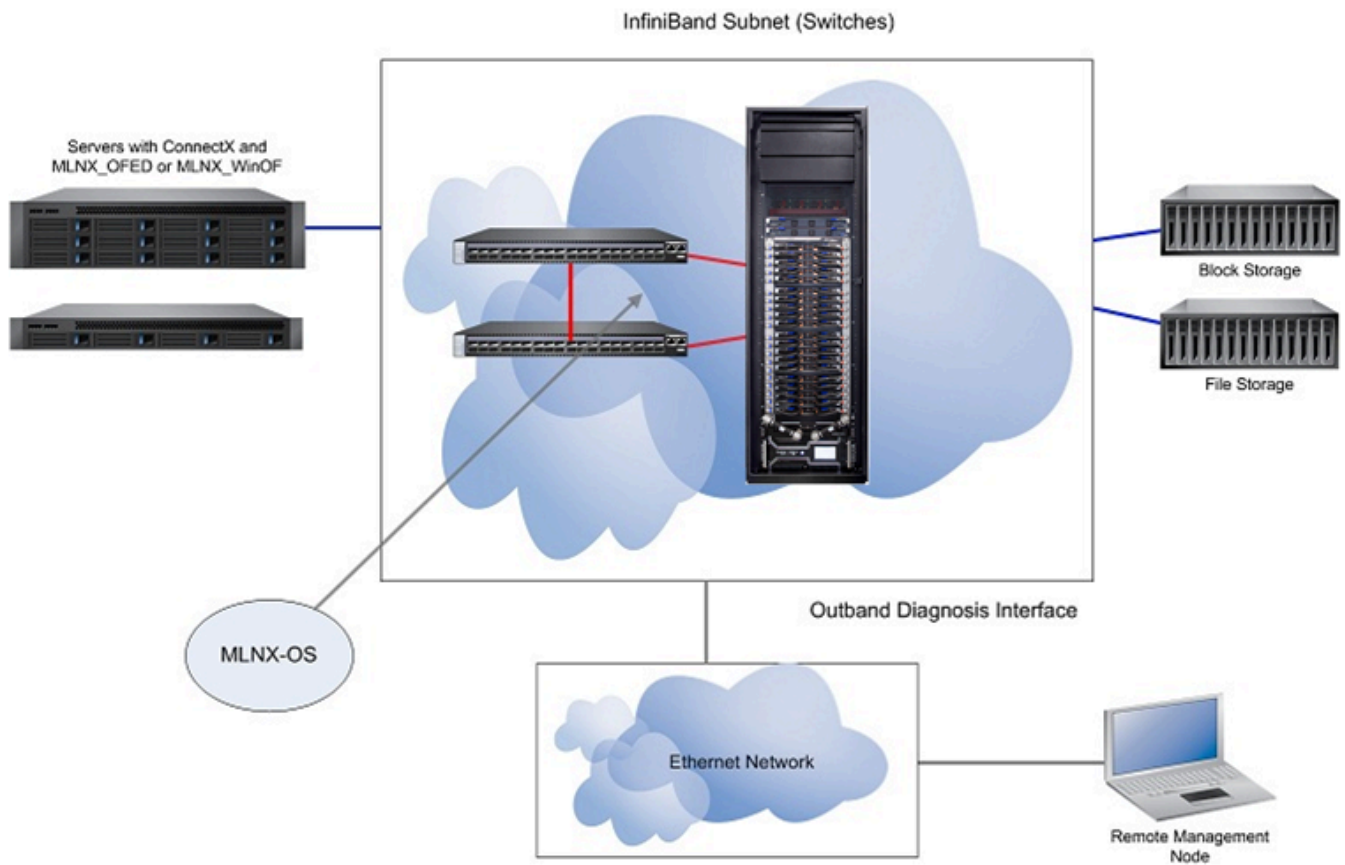
Term	Description
DCBX	Domain Name System. A hierarchical naming system for devices in a computer network.
DHCP	The Dynamic Host Configuration Protocol (DHCP) is an automatic configuration protocol used on IP networks.
Modular switch	A high density InfiniBand chassis switch system.
DNS	Domain Name System. A hierarchical naming system for devices in a computer network.
Fabric management	The use of a set of tools (APIs) to configure, discover, and manage and a group of devices organized as a connected fabric.
FTP/TFTP/sFTP	File Transfer Protocol (FTP) is a standard network protocol used to transfer files from one host to another over a TCP-based network, such as the Internet.
Gateway	A network node that interfaces with both InfiniBand and Ethernet, using different network protocols.
GID	Global Identifier. A 128-bit number used to identify a Port on a network adapter (see below), a port on a Router, or a Multicast Group.
GUID	Globally Unique Identifier. A 64-bit number that uniquely identifies a device or component in a subnet.
HA	High Availability. A system design protocol that provides redundancy of system components, thus enables overcoming single or multiple failures in minimal downtime.
Host	A computer platform executing an Operating System which may control one or more network adapters.
IB	InfiniBand
LID	Local Identifier. A 16 bit address assigned to end nodes by the subnet manager. Each LID is unique within its subnet.
LLDP	Link Layer Discovery Protocol. A vendor neutral link layer protocol used by network devices to advertise their identify, capabilities and for neighbor discovery.
MAC	A Media Access Control address (MAC address) is a unique identifier assigned to network interfaces for communications on the physical network segment. MAC addresses are used for numerous network technologies and most IEEE 802 network technologies including Ethernet.

Term	Description
MTU	Maximum Transfer Unit. The maximum size of a packet payload (not including headers) that can be sent /received from a port.
Network Adapter	A hardware device that allows for communication between computers in a network.
RADIUS	Remote Authentication Dial In User Service. A networking protocol that enables AAA centralized management for computers to connect and use a network service.
RDMA	Remote Direct Memory Access. Accessing memory in a remote side without involvement of the remote CPU.
SA	Subnet Administrator (SA) is the interface for querying and manipulating subnet management data.
SCP	Secure Copy or SCP is a means of securely transferring computer files between a local and a remote host or between two remote hosts. It is based on the Secure Shell (SSH) protocol.
SM	Subnet Manager. An entity that configures and manages the subnet, discovers the network topology, assign LIDs, determines the routing schemes and sets the routing tables. There is only one master SM and possible several slaves (Standby mode) at a given time. The SM administers switch routing tables thereby establishing paths through the fabric.
SNMP	Simple Network Management Protocol. A network protocol for the management of a network and the monitoring of network devices and their functions.
NTP	Network Time Protocol. A protocol for synchronizing computer clocks in a network.
SSH	Secure Shell. A protocol (program) for securely logging in to and running programs on remote machines across a network. The program authenticates access to the remote machine and encrypts the transferred information through the connection.
syslog	A standard for forwarding log messages in an IP network.
TACACS+	Terminal Access Controller Access-Control System Plus. A networking protocol that enables access to a network of devices via one or more centralized servers. TACACS+ provides separate AAA services.

System Features

Feature	Detail
Software management	<ul style="list-style-type: none">• Dual software image• Software and firmware updates• Docker
File management	<ul style="list-style-type: none">• FTP• TFTP• SCP
Logging	<ul style="list-style-type: none">• Event history log• SysLog support
Management interface	<ul style="list-style-type: none">• DHCP/Zeroconf• IPv6
Chassis management	<ul style="list-style-type: none">• Monitoring environmental controls• Power management• Auto-temperature control• High availability
Network management interfaces	<ul style="list-style-type: none">• SNMP v1,v2c,v3• JSON
Security	<ul style="list-style-type: none">• SSH• Telnet• RADIUS• TACACS+
Date and time	<ul style="list-style-type: none">• NTP
Cables & transceivers	<ul style="list-style-type: none">• Transceiver info
Unbreakable links	<ul style="list-style-type: none">• LLR

InfiniBand Features



Feature	Detail
Subnet manager	<ul style="list-style-type: none"> • OpenSM • Partitions • High availability

Getting Started

The procedures described in this page assume that you have already installed and powered on your switch according to the instructions in the Hardware Installation Guide, which was shipped with the product.

Configuring the Switch for the First Time

Starting the Command Line (CLI)

1. Set up an Ethernet connection between the switch and a local network machine using a standard RJ-45 connector.
2. Start a remote secured shell (SSH) to the switch using the command “ssh -l <username> <switch ip address>”.

```
rem_mach1 > ssh -l <username> <ip address>
```

3. Log into the switch (default username is admin, password admin).
4. Read and accept the EULA when prompted.
5. Once the following prompt appears, the system is ready to use.

```
NVIDIA MLNX-OS Switch Management
```

```
Password:
```

```
Last login: <time> from <ip-address>
```

```
NVIDIA Switch
```

```
Please read and accept the End User License Agreement located  
at:
```

```
https://www.mellanox.com/related-docs/prod_management_software/MLNX-OS_EULA.pdf  
switch >
```

Starting the Web User Interface (WebUI)

To start a WebUI connection to the switch platform, follow the steps below:

Warning

WebUI access is enabled by default. To disable web access, run the command “no web http enable” or “no web https enable” on the CLI.

1. Set up an Ethernet connection between the switch and a local network machine using a standard RJ-45 connector.
2. Open a web browser that is Firefox, Chrome, Internet Explorer, or Safari.

Warning

Make sure the screen resolution is set to 1024*768 or higher.

Warning

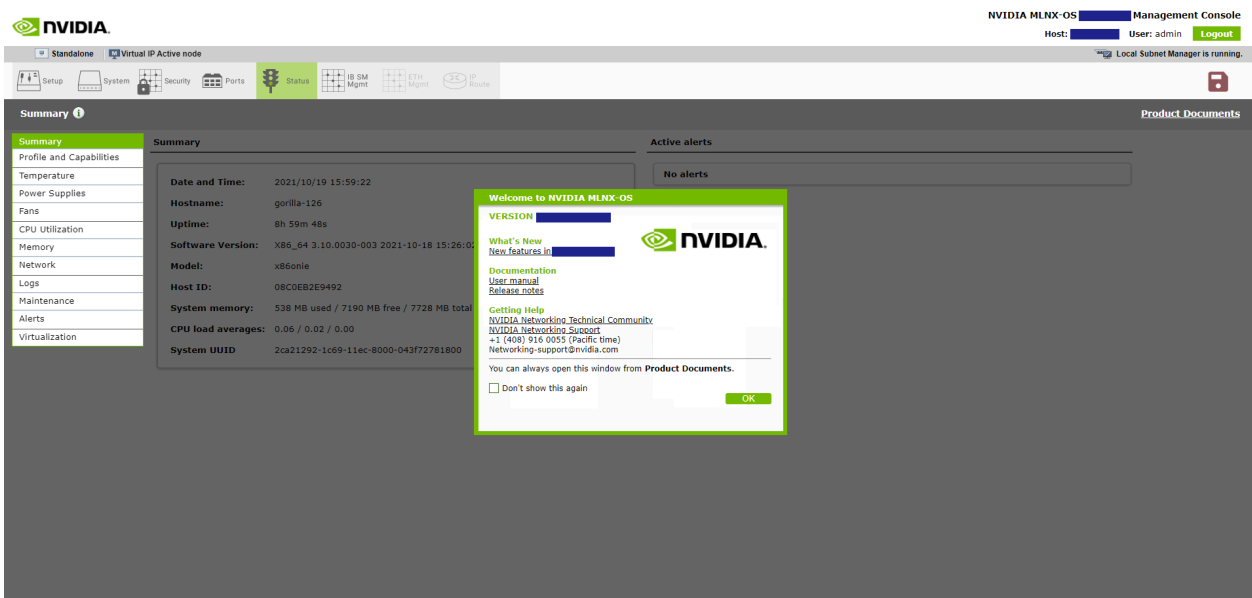
In order to access WebUI through Sarafi 5.3, enable http:

```
no web https ssl secure-cookie enable  
web http enable
```

- Type the IP address of the switch or its DNS name in the following format:
https://<switch_IP_address>.
- Log into the switch (default user name is admin, password admin).



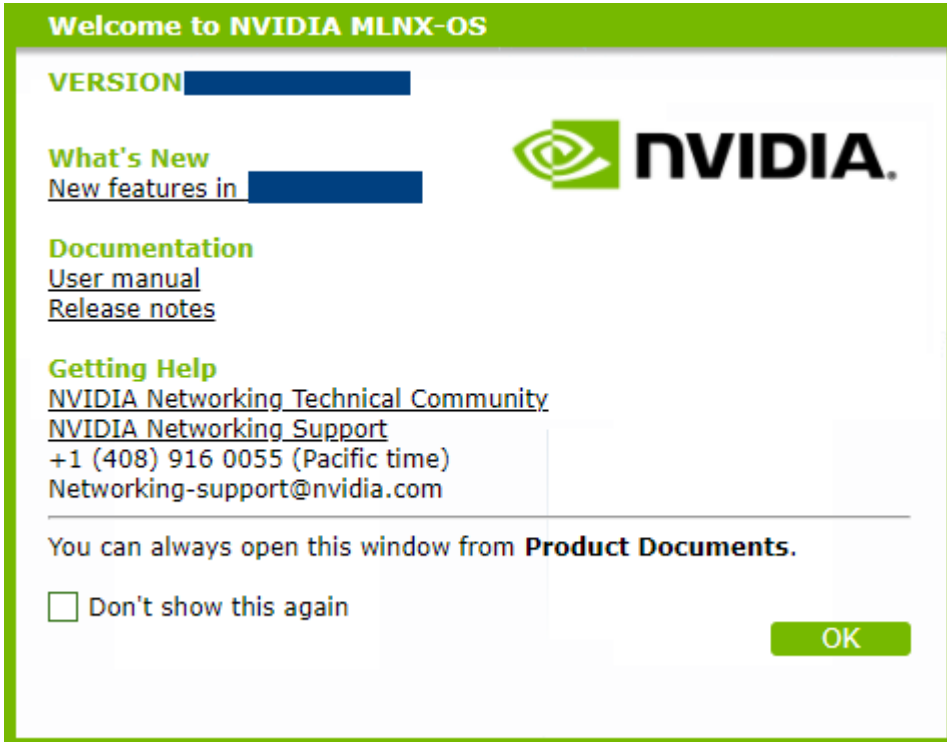
- Read and accept the EULA, if prompted. The prompt will only occur if the switch has never been accessed through the CLI before.



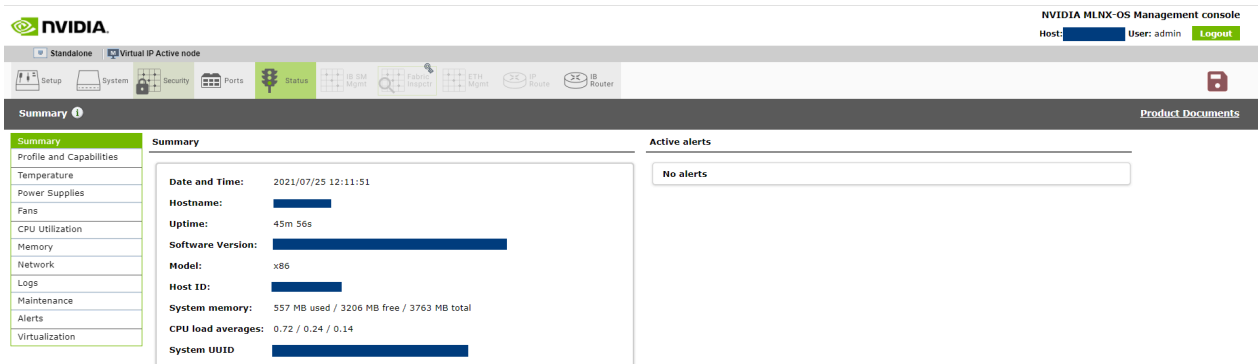
- The Welcome popup appears. After reading through the content, click OK to continue. To reach the OS documentation, click on the links under the Documentation

heading.

The link under What's New takes leads to the Changes and New Features section of the switch OS Release Notes. You may also tick the box to not show this popup again. To see this window again, click "Product Documents" on the upper right corner of the WebUI.



7. A default status summary is displayed.



Zero-touch Provisioning

Zero-Touch Provisioning (ZTP) automates initial configuration of switch systems at boot time. It helps minimize manual operation and reduce customer initial deployment cost.

ZTP allows for automatic upgrade of the switch with a specified OS image, setting up initial configuration database, and to load and run a container image file.

The initial configuration is applied using a regular text file. The user can create such a configuration file by editing the output of a “show running-config” command.

Warning

Only a textual configuration file is supported.

The user-defined docker image can be used by customers to run their own applications in a sandbox on their platform. They can therefore also be used for automating initial configuration.

Warning

Only one docker container can be launched in ZTP.

Running DHCP-ZTP

There is no explicit command to enable ZTP. It is enabled by default. Disabling it is performed by a user-initiated configuration save (using the command “configuration write”). The only way to re-enable ZTP is to run a “reset factory” command, clearing the configuration of the switch and rebooting the system.

ZTP is based on DHCP. For ZTP to work, the software enables DHCP by default on all its management interfaces. The switch OS requests option 66 (tftp-server-name) and 67 (bootfile-name) from the DHCPv4 server or option 58 (bootfile-url) from the DHCPv6 server, and waits for the DHCP responses containing file URLs. The DHCP server must be configured to send back the URLs for the software image, configuration file, and docker container image via these two options. Option 66 would contain the URL prefix to the location of the files, option 67 would contain the name of files, and option 58 would contain the complete URLs of files. The format of these two options is a string list separated by commas. The list items are placed in a fixed order:

```
<image file>, <config file>, <docker container file>
```

The item value can be empty, but the comma shall not be omitted.

To have DHCP server discern the proper files based on switch-specific information, the OS must provide identifying information for the server to classify the switches. In addition, the OS attaches option 43 (vendor-specific information) and option 60 (vendor class identifier) in DHCPv4 requests and option 17 (vendor-opts) in DHCPv6. Option 60 is set as string “Mellanox” and options 17 and 43 contain the following specific sub-options:

- System Model
- Chassis Part Number
- Chassis Serial Number
- Management MAC
- System Profile
- MLNX-OS Release Version

The corresponding subtypes respectively are defined as:

DHCP_VENDOR_ENCAPSULATED_SUBOPTION_TLV_TYPE_MODEL	1
DHCP_VENDOR_ENCAPSULATED_SUBOPTION_TLV_TYPE_PARTNUM	2
DHCP_VENDOR_ENCAPSULATED_SUBOPTION_TLV_TYPE_SERIAL	3
DHCP_VENDOR_ENCAPSULATED_SUBOPTION_TLV_TYPE_MAC	4
DHCP_VENDOR_ENCAPSULATED_SUBOPTION_TLV_TYPE_PROFILE	5
DHCP_VENDOR_ENCAPSULATED_SUBOPTION_TLV_TYPE_RELEASE	6

Upon receiving such DHCP requests from a client, the server should be able to map the switch-specific information to the target file URLs according to predefined rules.

Once the OS receives the URLs from the DHCP server, it executes ZTP as follows:

1. If the software image URL is not specified, this step is skipped. Otherwise:

1. Perform disk space cleanup if necessary and fetch the image if it does not exist locally
2. Resolve the image version:
 3. If it is already installed on active partition, proceed to step 2
 4. If it is installed on a standby partition, switch partition and reboot
 5. If it is not installed locally, install it and switch to the new image and then reboot
 6. If a reboot occurs, ZTP performs step 1 again and no image upgrade will occur
2. If configuration file URL is not specified, skip this step. Otherwise:
 1. Fetch the configuration file
 2. Apply the configuration file
3. Skip these steps if a docker image file URL is not specified. Otherwise:
 1. Fetch the docker image file
 2. Load the docker image
 3. Clean up the docker images with the same name and different tag.
 4. Start the container based on the image
 5. Remove the downloaded docker image file



Warning

While performing file transfer via HTTP, the same information as DHCP option 43 is expected to be carried in a HTTP GET request. This switch software supports the following proprietary HTTP headers:

- MlnxSysProfile
- MlnxMgmtMac
- MlnxSerialNumber

- MlnxModelName
- MlnxPartNumber
- MlnxReleaseVersion

If some sort of failure occurs, the switch waits a random number of seconds between 1 and 20 and reattempts the operation. The switch attempts this up to 10 times. ZTP progress is printed to terminals including console and active SSH sessions.

ZTP on Modular Switches

For modular switch systems, the two management nodes start ZTP individually. Status synchronization is then performed between the two nodes:

- Target software image version needs to be the same, otherwise ZTP fails
- Both nodes must install the software image successfully, otherwise ZTP fails
- ZTP failure for one node leads to failure for both
- ZTP disable on one node leads to ZTP disable for both
- ZTP abort on one node leads to ZTP abort for both

In ZTP configuration files, commands between #<CHASSIS_MASTER> and #</CHASSIS_MASTER> pair are only executed on the master.

```
#<CHASSIS_MASTER>  
  chassis ha bip 10.7.146.34 /24  
#</CHASSIS_MASTER>
```

Node reboot caused by ZTP is also synchronized:

1. Master node asks slave to reboot.
2. Slave node switches to next boot location and acknowledges the reboot request.

3. Master node reboots slave node via hardware.
4. Master node reboots itself.

ZTP and OS Upgrade

Software upgrade from non-ZTP versions to ZTP versions and vice versa is supported. When upgrading from a non-ZTP version, ZTP is disabled because ZTP is always assumed to start with an empty configuration, otherwise the final configuration becomes a mixture of the existing configuration from the stored database and new configuration from the server and hence not deterministic.

DHCPv4 Configuration Example

The following is a URL configuration example for ISC DHCPv4 server:

```
host master {
    hardware ethernet E4:1D:2D:5B:72:80;
    fixed-address 3.1.2.13;
    option tftp-server-name "scp://<user>:<password>@3.1.3.100/ztp/scp://
                                <user>:
<password>@3.1.3.100/ztp/, scp://
                                <user>:
<password>@3.1.3.100/ztp/";
    option bootfile-name "image-X86_64-3.6.4612.img, switch-1.conf,
ubuntu.img.gz";
}
```

DHCPv4 request is made out of the following components:

- Option 43 (vendor-encapsulated-options) and option 60 (vendor-class-identifier) are added in the DHCPv4 request packet

- Option 66 (tftp-server-name) and option 67 (bootfile-name) are added in the parameter request list of DHCPv4 request packet

DHCPv6 Configuration Example

The following is a DHCPv6 configuration example:

```

host master {
    .....
    option dhcp6.bootfile-url "scp://<user>:
<password>@[2000::1]/ztp/image-X86_64-
                                3.6.4612.img, scp://<user>:
<password>@[2000::1]/ztp/
                                switch.conf, scp://<user>:
<password>@[2000::1]/ztp/
                                ubuntu.img.gz";
}

```

```

host master {
    .....
    option dhcp6.bootfile-url "scp://<user>:
<password>@[2000::1]/ztp/image-X86_64-
                                23.01.0100.img, scp://<user>:
<password>@[2000::1]/ztp/
                                switch.conf, scp://<user>:
<password>@[2000::1]/ztp/
                                ubuntu.img.gz";
}

```

DHCPv6 request is made out of the following components:

- Option 17 (vendor-opts) is added in the DHCPv6 request packet

- Option 59 (bootfile-url) is added in the parameter request list of DHCPv6 request packet

ZTP Commands

no zero-touch suppress-write

	no zero-touch suppress-write Disables suppression of configuration write.
Syntax Description	N/A
Default	Enabled
Configuration Mode	config
History	3.6.5000 3.9.2400: Added note
Example	switch (config) # no zero-touch suppress-write
Related Commands	show zero-touch
Notes	<ul style="list-style-type: none"> • When ZTP is active, “configuration write” is suppressed because it may interfere with ZTP operation. Therefore, after running “no zero-touch suppress-write” if “configuration write” is performed, then ZTP is disabled as a consequence of the database save. • To automatically save the configuration at the end of applying a configuration via ZTP, append the following two commands to the end of the config files. The first command will turn off the ZTP suppress-write, then the configuration write command should work. <ul style="list-style-type: none"> ◦ no zero-touch suppress-write ◦ configuration write

zero-touch abort

	zero-touch abort
--	------------------

	Aborts on-going zero-touch process.
Syntax Description	N/A
Default	Enabled
Configuration Mode	config
History	3.6.5000
Example	switch (config) # zero-touch abort Zero-touch failed [Zero-touch is aborted by operator] Zero-touch provisioning will be aborted
Related Commands	show zero-touch
Notes	

show zero-touch

	show zero-touch Displays zero-touch status.
Syntax Description	N/A
Default	N/A
Configuration Mode	Any command mode
History	3.6.5000
Example	switch (config) # show zero-touch Zero-Touch status: Active: yes Status: Waiting for zero-touch start Suppress-write: no Configured by zero-touch: no Configuration changed after zero-touch: no
Related Commands	zero-touch abort zero-touch suppress-write
Notes	

Licenses

The software package can be extended with premium features. Installing a license allows you to access the specified premium features

Warning

This section is relevant only to switch systems with an internal management capability.

Installing OS License via CLI

To install a license via CLI:

1. Before applying a license, please make sure your system's time is configured correctly by manually setting it using the CLI command "clock set", or by using NTP using the command "ntp".
2. Login as admin and change to Config mode.

```
switch > enable  
switch # config terminal
```

3. Install the license using the key. Run:

```
switch (config) # license install <license key>
```

4. Display the installed license(s) using the following command. Run:

```
switch (config) # show licenses
License 1: <license key>
Feature: EFM_SX
Valid: yes
Active: yes
```

Make sure that the “Valid” and “Active” fields both indicate “yes”.

5. Save the configuration to complete the license installation. Run:

```
switch (config) # configuration write
```

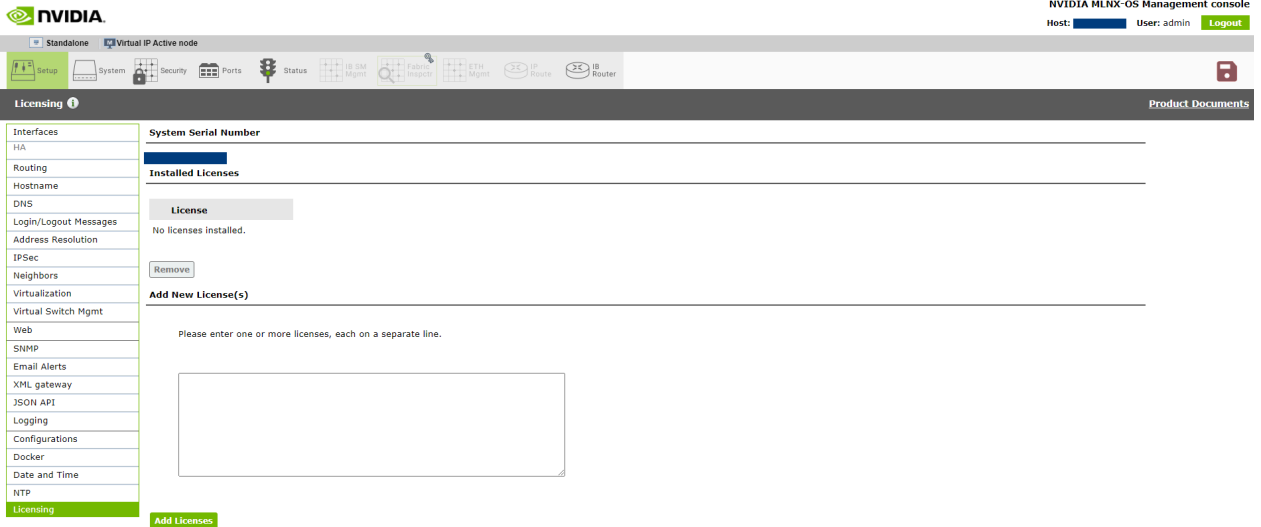
Warning

If you do not save the installation session, you will lose the license at the next system start up.

Installing OS License via Web

To install a license via WebUI:

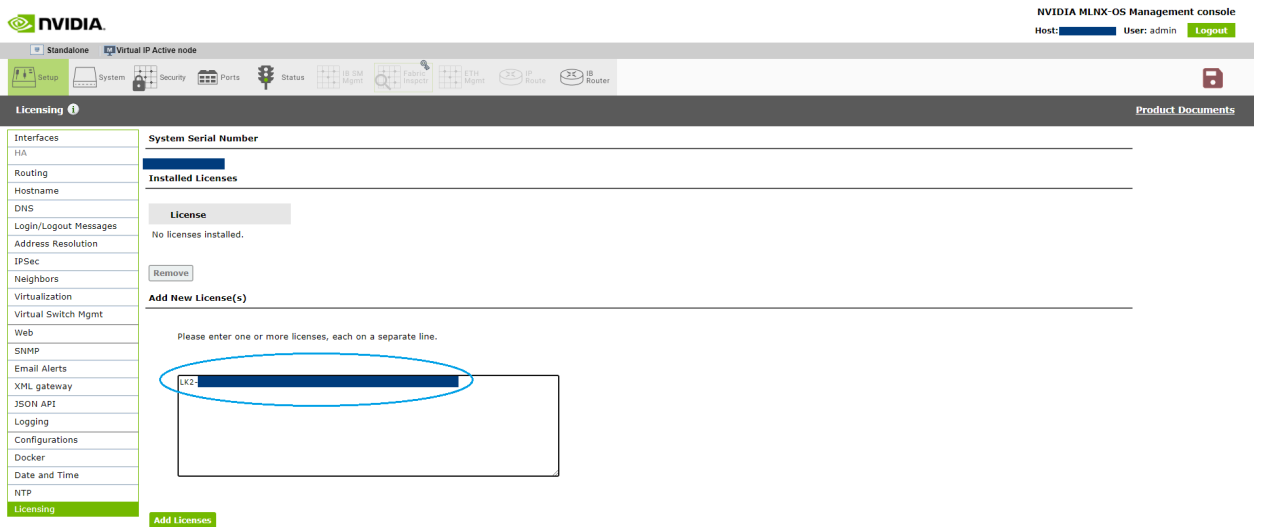
1. Login as *admin*.
2. Click the Setup tab and then Licensing on the left side navigation pane.



3. Enter your license key(s) in the text box. If you have more than one license, please enter each license in a separate line. Click “Add Licenses” after entering the last license key to install them.

Warning

If you wish to add another license key in the future, you can simply enter it in the text box and click “Add Licenses” to install it.



4. All installed licenses should now be displayed.

The screenshot shows the NVIDIA MLNX-OS Management console interface. At the top, it displays 'NVIDIA' and 'NVIDIA MLNX-OS Management console' with user information 'Host: [redacted] User: admin Logout'. The main navigation bar includes 'Virtual IP Active node' and various system status icons. The 'Licensing' page is active, showing a sidebar menu with options like 'Interfaces', 'Routing', 'DNS', etc. The main content area is divided into sections: 'System Serial Number', 'Installed Licenses', and 'Add New License(s)'. The 'Installed Licenses' section contains a table with one license entry:

License	
<input type="checkbox"/> Key	LK2 [redacted]
Feature	RESTRICTED_CMDS_GEN2
Description	Access to restricted system functionality
Valid	yes
Tied to MAC addr	7C:FE:90:65:DE:30 (ok)
Active	yes

Below the table is a 'Remove' button. The 'Add New License(s)' section has a text input field with the placeholder 'Please enter one or more licenses, each on a separate line.' and an 'Add Licenses' button.

5. Save the configuration to complete the license installation.

Warning

If you do not save the installation session, you will lose the installed licenses at the next system boot.

Retrieving a Lost License Key

In case of a lost license key, contact your authorized NVIDIA reseller and provide the switch's chassis serial number.

To obtain the switch's chassis serial number:

1. Log in to the switch.
2. Retrieve the switch's chassis serial number using the command "show inventory".

```
switch (config) # show inventory
```

Module Rev.	HW Rev.	Part Number	Serial Number	Asic
CHASSIS A1		MSB7800-ES2F	MT1602X17464	N/A
MGMT A1		MSB7800-ES2F	MT1602X17464	0
FAN1 A3		MTEF-FANF-A	MT1602X16943	N/A
FAN2 A3		MTEF-FANF-A	MT1602X16944	N/A
FAN3 A3		MTEF-FANF-A	MT1602X16956	N/A
FAN4 A3		MTEF-FANF-A	MT1602X16957	N/A
PS1 A3		MTEF-PSF-AC-A	MT1601X09908	N/A

3. Provide your authorized NVIDIA reseller with the chassis serial number for your system.
4. Once you receive the license key, you can install the license as described in the previous pages.

Additional Reading and Use Cases

For more information about getting started with NVIDIA Switches, please refer to the following Community post:

- [How To Get Started with NVIDIA Switches](#)

License Commands

- [Configuring the Switch for the First Time](#)
- - [Configuring the Switch with ZTP](#)
 - [Rerunning the Wizard](#)
- [Starting the Command Line \(CLI\)](#)
- [Starting the Web User Interface \(WebUI\)](#)
- [Zero-touch Provisioning](#)
 - [Running DHCP-ZTP](#)
 - [ZTP on Modular Switches](#)
 - [ZTP and OS Upgrade](#)
 - [DHCPv4 Configuration Example](#)
 - [DHCPv6 Configuration Example](#)
 - [ZTP Commands](#)
 - [no zero-touch suppress-write](#)
 - [zero-touch abort](#)
 - [show zero-touch](#)
- [Licenses](#)
 - [Installing OS License via CLI](#)
 - [Installing OS License via Web](#)
 - [Retrieving a Lost License Key](#)
 - [Additional Reading and Use Cases](#)
 - [License Commands](#)

- [license delete](#)
- [license install](#)
- [show licenses](#)

license delete

	license delete <license-number> Removes license keys by ID.
Syntax Description	N/A
Default	N/A
Configuration Mode	config
History	3.4.1 100
Example	switch (config) # license delete <license-number>
Related Commands	license install show licenses
Notes	Before deleting a license from a switch which is configured to a system profile other than its default, the user must first disable all interfaces and then return the switch to its default system profile.

license install

	license install<license-number> Installs a new license key.
Syntax Description	N/A
Default	N/A
Configuration Mode	config
History	3.4.1 100

Example	switch (config) # licenses install <license-key>
Related Commands	license delete show licenses
Notes	

show licenses

	show licenses Displays a list of all installed licenses.
Syntax Description	N/A
Default	N/A
Configuration Mode	config
History	3.4.1 100
Example	switch (config) # show licenses License 1: <license key> Feature: SX_CONFIG Valid: yes Active: yes
Related Commands	license delete license install
Notes	For each license, the following is displayed: <ul style="list-style-type: none"> • A unique ID which is a small integer • The text of the license key as it was added • Whether or not it is valid and active • Which feature(s) it is activating • A list of all licensable features specifying whether or not it is currently activated by a license

User Interfaces

The following pages provide information on the interfaces available for to manage and validate the status of the system.

- [LED Indicators](#)
- [Command Line Interface \(CLI\)](#)
- [Secure Shell \(SSH\)](#)
- [Web Interface Overview](#)
- [UI Commands](#)

System Management

The following pages provide information on configuring general management features on the system.

- [Management Interfaces](#)
- [Chassis Management](#)
- [UNBREAKABLE-LINK® Adapter and Switch Technology](#)
- [Upgrade/Downgrade Process](#)
- [Configuration Management](#)
- [mDNS](#)

Network Management Interfaces

SNMP

Simple Network Management Protocol (SNMP), is a network protocol for the management of a network and the monitoring of network devices and their functions. SNMP supports asynchronous event (trap) notifications and queries. MLNX-OS supports:

- SNMP versions v1, v2c and v3
- SNMP trap notifications
- Standard MIBs
- Private MIBs

Standard MIBs

The following table presents the supported textual conventions and conformance MIBs:

MIB	Standard
INET-ADDRESS-MIB	RFC-4001
SNMPV2-CONF	
SNMPV2-TC	RFC 2579
SNMPV2-TM	RFC 3417
SNMP-USM-AES-MIB	RFC 3826
IANA-LANGUAGE-MIB	RFC 2591
IANA-RTPROTO-MIB	RFC 2932
IANAifType-MIB	
IANA-ADDRESS-FAMILY-NUMBERS-MIB	

MIB	Standard
IGMP-STD-MIB	RFC2933 (See IGMP-STD-MIB Information section)

The following table presents the supported chassis and switch MIBs:

MIB	Standard	Comments
RFC1213-MIB	RFC 1213	
IF-MIB	RFC 2863	ifXTable only supported.
ENTITY-MIB	RFC 4133	
ENTITY-STATE-MIB	RFC 4268	Fan and temperature states
ENTITY-SENSOR-MIB	RFC 3433	<ul style="list-style-type: none"> • Port module transmit/receiver power sensors (for 1U systems only) • Fan and temperature sensors

Private MIBs

MIB	Description
MELLANOX-SMI-MIB	Private MIB main structure (no objects)
MELLANOX-PRODUCTS-MIB	List of OID - per managed system (sysObjID)
MELLANOX-IF-VPI-MIB	IfTable extensions
MELLANOX-EFM-MIB	Partially deprecated MIB (based on Mellanox-MIB) Traps definitions and test trap set scalar are supported.
MELLANOX-ENTITY-MIB	Enhances the standard ENTITY-MIB (contains GUID and ASIC revision).

MIB	Description
MELLANOX-POWER-CYCLE	Allows rebooting the switch system
MELLANOX-SW-UPDATE-MIB	Allows viewing what SW images are installed, uploading and installing new SW images
MELLANOX-CONFIG-DB	Allows loading, uploading, or deleting configuration files
MELLANOX-ENTITY-STATE-MIB	Extension to support state change traps Note: Currently supported for power supply insertion and extraction only
MELLANOX-XSTP-MIB	Extension to support STP information
MELLANOX-DCB-TRAPS	Extension traps for ETC and PFC
MELLANOX-QOS	Proprietary QoS MIBs
MELLANOX-WJH-MIB	Defines what-just-happened traps

Private MIBs can be downloaded from the [support](#) website.

Proprietary Traps

The following private traps are supported by the MLNX-OS

MELLANOX-EFM-MIB:

Trap	Action Required
asicChipDown	Reboot the system.
asicOverTempReset	Check fans and environmental temperature.
asicOverTemp	Check fans and environmental temperature.
lowPower	Add/connect power supplies.
internalBusError	N/A
procCrash	Generate SysDump and contact support.
cpuUtilHigh	N/A

Trap	Action Required
procUnexpectedExit	Generate SysDump and contact support.
diskSpaceLow	Clean images and sysDump files using the commands “image delete” and “file debug-dump delete”.
systemHealthStatus	Refer to Health Status table.
lowPowerRecover	N/A
insufficientFans	Check Fans and environmental conditions.
insufficientFansRecover	N/A
insufficientPower	Add/connect power supplies, or change power mode using the command “power redundancy mode”.
insufficientPowerRecover	N/A

For additional information refer to MELLANOX-EFM-MIB.

Warning

For event-to-MIB mapping, please refer to [“Supported Event Notifications and MIB Mapping”](#).

The only MELLANOX-POWER-CYCLE trap supported is `mellanoxPowerCyclePlannedReload`.

Configuring SNMP

Activate the SNMP server on your switch by running:

```
switch (config) # snmp-server enable
switch (config) # snmp-server enable notify
switch (config) # snmp-server community public ro
```

```
switch (config) # snmp-server contact "contact name"
switch (config) # snmp-server host <host IP address> traps version
2c public
switch (config) # snmp-server location "location name"
switch (config) # snmp-server user admin v3 enable
switch (config) # snmp-server user admin v3 prompt auth md5 priv
des
```

Warning

Community strings are case sensitive.

Warning

Modular switches require SNMP timeout configuration on the agent of 60 seconds.

Resetting SNMPv3 Engine ID

Warning

Resetting SNMP engine ID is not supported on modular switch systems.

Switch systems shipped with an OS versions older than 3.6.6102 have all had the exact same SNMPv3 engine ID. Going forward, however, all switch systems will ship with a

system-specific engine ID.

Upgrading the OS version to 3.6.6102 or higher does not automatically change the current engine ID. That can be done through one of the following methods after performing the software upgrade:

- Changing a switch system's profile
- Running "reset factory"
- Using the command "snmp-server engineID reset" (for more details, please see the procedure below)

To reset SNMP engine ID using "snmp-server engineID reset":

Prerequisites:

If any of the following SNMP configurations exist, please delete/disable them and re-enable/reconfigure them only after SNMP engine ID reset is performed:

1. Make sure SNMP is disabled. Run:

```
switch (config) # no snmp-server enable
```

2. Make sure no SNMP trap host is configured. Run:

```
switch (config) # no snmp-server host <ip-address>
```

3. Make sure no SNMP users are configured. Run:

```
switch (config) # no snmp-server user <username> v3
```

Procedure:

1. Check existing engine ID:

```
switch (config) # show snmp engineID
Local SNMP engineID: <current_key>
```

2. Reset existing engine ID:

```
switch (config) # snmp-server engineID reset
```

3. Verify new engine ID:

```
switch (config) # show snmp engineID
Local SNMP engineID: <new_key>
```

Configuring an SNMPv3 User

To configure an SNMPv3 user:

1. Configure the user using the command:

```
switch (config) # snmp-server user [role] v3 prompt auth <hash
type> priv <privacy type>
```

Where:

- user role—admin
- auth type—md5 or sha or sha224 or sha256 or sha384 or sha512
- priv type—des or aes-128 or 3des or aes-192 or aes-256 or aes-192-cfb or aes-256-cfb

2. Enter authentication password and its confirmation.

3. Enter privacy password and its confirmation:

```
switch (config) # snmp-server user admin v3 prompt auth md5
priv des
Auth password: ****
Confirm: ****
Privacy password: ****
Confirm: ****
```

To retrieve the system table, run the following SNMP command:

```
snmpwalk -v3 -l authPriv -a MD5 -u admin -A "<Authentication
password>" -x DES -X "<privacy password>" <system ip> SNMPv2-
MIB::system
```

Configuring SNMP Notifications (Traps or Informs)

1. Make sure SNMP and SNMP notification are enable. Run:

```
switch (config) # snmp-server enable
switch (config) # snmp-server enable notify
```

2. Configure SNMP host with the desired arguments (IP Address, SNMP version, authentication methods). More than one host can be configured. Each host may have different attributes. Run:

```
switch (config) # snmp-server host 10.134.47.3 traps version 3
user my-username auth sha my-password
```

3. Verify the SNMP host configuration. Run:

```
switch (config) # show snmp host
Notifications enabled:          yes
Default notification community: public
Default notification port:      162

Notification sinks:

  10.134.47.3
    Enabled:                    yes
    Port:                       162 (default)
    Notification type:          SNMP v3 trap
    Username:                   my-username
    Authentication type:        sha
    Privacy type:               aes-128
    Authentication password:    (set)
    Privacy password:           (set)
```

4. Configure the desired event to be sent via SNMP. Run:

```
switch (config) # snmp-server notify event interface-up
```

Warning

This particular event is used as an example only.

5. Verify the list of traps and informs being sent to out of the system. Run:

```
switch (config) # show snmp events
Events for which traps will be sent:
```

```
asic-chip-down: ASIC (Chip) Down
cpu-util-high: CPU utilization has risen too high
disk-space-low: Filesystem free space has fallen too low
health-module-status: Health module Status
insufficient-fans: Insufficient amount of fans in system
insufficient-fans-recover: Insufficient amount of fans in
system recovered
insufficient-power: Insufficient power supply
interface-down: An interface's link state has changed to down
interface-up: An interface's link state has changed to up
internal-bus-error: Internal bus (I2C) Error
liveness-failure: A process in the system was detected as
hung
low-power: Low power supply
low-power-recover: Low power supply Recover
new_root: local bridge became a root bridge
paging-high: Paging activity has risen too high
power-redundancy-mismatch: Power redundancy mismatch
process-crash: A process in the system has crashed
process-exit: A process in the system unexpectedly exited
snmp-authtrap: An SNMP v3 request has failed authentication
topology_change: local bridge triggered a topology change
unexpected-shutdown: Unexpected system shutdown
```

Warning

To print event notifications to the terminal (SSH or CONSOLE) refer to [“Monitor”](#).

Warning

For the SNMPv1 traps or informs, by default, the "agent address" field is set to the IP address of the "mgmt0" interface. In the case that "source interface" is configured to the same VRF which is used for SNMPv1 traps or informs, the IP address of the source interface is used for "agent address" field. In other cases (e.g., if source interface might be configured in some other VRF), "127.0.0.1" is used for the "agent address".

SNMP SET Operations

The OS allows the user to use SET operations via SNMP interface. This is needed to configure a user/community supporting SET operations.

Enabling SNMP SET

To allow SNMP SET operations using SNMPv1/v2:

1. Enable SNMP communities. Run:

```
switch (config) # snmp-server enable communities
```

2. Configure a read-write community. Run:

```
switch (config) # snmp-server community my-community-name rw
```

3. Make sure SNMP communities are enabled (they are enabled by default). Make sure "(DISABLED)" does not appear beside "Read-only communities" / "Read-write communities". Run:

```
switch (config) # show snmp
```

```
SNMP enabled    : yes
SNMP port      : 161
System contact :
System location:

Read-only communities:
  public

Read-write communities:
  my-community-name

Interface listen enabled: yes

Listen Interfaces:
  Interface: mgmt0

switch (config) # show snmp
No Listen Interfaces.
```

4. Configure this RW community in your MIB browser.

To allow SNMP SET operations using SNMPv3:

1. Create an SNMPv3 user. Run:

```
switch (config) # snmp-server user myuser v3 auth sha
<password1> priv aes-128 <password2>
```

 **Warning**

It is possible to use other configuration options not specified in the example above. Please refer to the command [“snmp-server user”](#) for more information.

2. Make sure the username is enabled for SET access and has admin capability level.
Run:

```
switch (config) # show snmp user
User name: myuser
  Enabled overall:          yes
  Authentication type:     sha
  Privacy type:            aes-128
  Authentication password: (set)
  Privacy password:        (set)
  Require privacy:         yes
  SET access:
    Enabled:               yes
    Capability level:      admin
```

The OS supports the OIDs for SET operation listed in the following table which are expanded upon in the following subsections.

	OID Name	OID
MELLANOX-EFM-MIB	sendTestTrapSet	1.3.6.1.4.1.33049.2.1.1.1.6.0
SNMPv2-MIB	sysName	1.3.6.1.2.1.1.5.0
MELLANOX-CONFIG-DB	mellanoxConfigDBCcmdExecute	1.3.6.1.4.1.33049.12.1.1.2.3.0
	mellanoxConfigDBCcmdFilename	1.3.6.1.4.1.33049.12.1.1.2.2.0
	mellanoxConfigDBCcmdStatus	1.3.6.1.4.1.33049.12.1.1.2.4.0
	mellanoxConfigDBCcmdStatusString	1.3.6.1.4.1.33049.12.1.1.2.5.0
	mellanoxConfigDBCcmdUri	1.3.6.1.4.1.33049.12.1.1.2.1.0
MELLANOX-POWER-CYCLE	mellanoxPowerCycleCmdExecute	1.3.6.1.4.1.33049.10.1.1.2.1.0
	mellanoxPowerCycleCmdStatus	1.3.6.1.4.1.33049.10.1.1.2.2.0
	mellanoxPowerCycleCmdStatusString	1.3.6.1.4.1.33049.10.1.1.2.3.0

	OID Name	OID
MELLANOX-SW-UPDATE	mellanoxSWUpdateCmdSetNext	1.3.6.1.4.1.33049.11.1.1.2.1.0
	mellanoxSWUpdateCmdUri	1.3.6.1.4.1.33049.11.1.1.2.2.0
	mellanoxSWUpdateCmdExecute	1.3.6.1.4.1.33049.11.1.1.2.3.0
	mellanoxSWUpdateCmdStatus	1.3.6.1.4.1.33049.11.1.1.2.4.0
	mellanoxSWUpdateCmdStatusString	1.3.6.1.4.1.33049.11.1.1.2.5.0
	mellanoxSWActivePartition	1.3.6.1.4.1.33049.11.1.1.3.0.0
	mellanoxSWNextBootPartition	1.3.6.1.4.1.33049.11.1.1.4.0.0

Sending a Test Trap SET Request

The OS allows the user to use test the notification mechanism via SNMP SET. Sending a SET request with the designated OID triggers a test trap.

Prerequisites:

1. Enable SET operations by following the instructions in [“Enabling SNMP SET”](#).
2. Configure host to which to send SNMP notifications.
3. Set a trap receiver in the MIB browser.

Procedure:

1. Send a SET request to the switch IP with the OID 1.3.6.1.4.1.33049.2.1.1.1.6.0.
2. Make sure the test trap is received by the aforementioned trap receiver (OID: 1.3.6.1.4.1.33049.2.1.2.13).

Setting Hostname with SNMP

The OS supports setting system hostname using an SNMP SET request as described in SNMPv2-MIB (sysName, OID: 1.3.6.1.2.1.1.5.0).

The restrictions on setting a hostname via CLI also apply to setting a hostname through SNMP. Refer to the command “hostname” for more information.

Power Cycle with SNMP

The OS supports power cycling its systems using an SNMP SET request as described in MELLANOX-POWER-CYCLE MIB.

Power cycle command is issued via the OID `mellanoxPowerCycleCmdExecute`. The following options are available:

- Reload—saves any unsaved configuration and reloads the switch
- Reload discard—reboots the system and discards of any unsaved changes
- Reload force—forces an expedited reload on the system even if it is busy without saving unsaved configuration (equals the CLI command `reload force`)
- Reload slave—reloads the slave management on dual management systems (must be executed from the master management module)

Warning

On modular switch systems, it is advised to connect via the BIP to make sure commands are executed from the master management.

Changing Configuration with SNMP

The OS supports making configuration changes on its systems using SNMP SET requests. Configuration requests are performed by setting several values (arguments) and then executing a command by setting the value for the relevant operation.

It is possible to set the parameters and execute the commands on the same SNMP request or separate them to several SET operations. Upon executing a command, the values of its arguments remain and can be read using GET commands.

Once a command is executed there may be two types of errors:

- Immediate: This error results in a failure of the SNMP request. This means a critical error in the SNMP request has occurred or that a previous SET request is being

executed

- Delayed: The SET request has been accepted by the switch but an error occurred during its execution.

For example, when performing a fetch (download) operation, an immediate error can occur when the given URL is invalid. A delayed error can occur if the download process fails due to network connectivity issues.

The following parameters are arguments are supported:

- Command URI—URI to fetch the configuration file from or upload the file to (for supported URI format please refer to the CLI command “configuration fetch” for more details)
- Config file name—filename to save the configuration file to or to upload to remote location

The following commands are supported:

- BinarySwitchTo—replaces the configuration file with a new binary configuration file. This option fetches the configuration file from the URI provided in the `mellanoxConfigDBCcmdUri` and switches to that configuration file. This command should be preceded by a reload command in order for the new configuration to apply.
- TextApply—fetches a configuration file in human-readable format and applies its configuration upon the current configuration.
- BinaryUpload—uploads a binary format configuration file of the current running configuration or an existing configuration file on the switch to the URI in the `mellanoxConfigDBCcmdUri` command. The filename parameter indicates what configuration file on the switch to upload.
- TextUpload—uploads a human-readable configuration file of the current running configuration or an existing configuration file on the switch to the URI in the `mellanoxConfigDBCcmdUri` command. The filename parameter indicates what configuration file on the switch to upload (same as the CLI command `configuration text generate file <filename> upload`).
- ConfigWrite—saves active configuration to a filename on the switch as given in the filename parameter. In case filename is “active”, active configuration is saved to the current saved configuration (same as the CLI command `configuration write`).
- BinaryDelete—deletes a binary based configuration file

- TextDelete—deletes a text based configuration file

Upgrading OS Software with SNMP

The OS supports upgrading its software using an SNMP SET request as described in MELLANOX-SW-UPDATE MIB.

The software upgrade command is issued via the OID `mellanoxSWUpdateCmdExecute`. The following options are available:

- Update—fetches the image from a specified URI (equivalent to the command “image fetch” followed by “image install”)
The image to update from is defined by the OID `mellanoxSWUpdateCmdUri`. The restrictions on the URI are identical to what is supported in the CLI command “[image fetch](#)”.
- Set-Next—changes the image for the next boot equivalent to the CLI command “image boot”)
The partition from which to boot is defined by the OID `mellanoxSWUpdateCmdSetNext`. The parameters for this OID are as follows:
 - 0—no change
 - 1—partition 1
 - 2—partition 2
 - 3—next partition (default)

Using the OIDs `mellanoxSWUpdateCmdStatus` and `mellanoxSWUpdateCmdStatusString`, you may view the status of the latest operation performed from the aforementioned in either integer values, or human-readable forms, respectively. The integer values presented may be as follows:

- 0—no operation
- 1-100—progress in percentage
- 101—success
- 200—failure

IF-MIB and Interface Information

The OS supports displaying information of switch ports, LAG ports, MLAG ports and VLAN interfaces on all systems via SNMP interface. This feature is enabled by default. The interface information is available in the ifTables, ifXTable and mellanoxIfVPITable.

Additionally, traps for interface up/down, and internal link suboptimal speed are enabled. It is possible to enable one or both of these traps.

Interface up/down traps are sent whenever there is a change in the interface's operational state. These traps are suppressed for internal links when the internal link's speed does not match the configured speed of the link (mismatch condition).

Additional Readings and Use Cases

For more information about this feature and its potential applications, please refer to the following community posts:

- [Getting Started with SNMP MIBs](#)
- [HowTo Use SNMP SET](#)

JSON API

JavaScript Object Notation (JSON) is a machine-to-machine data-interchange format which is supported in MLNX-OS CLI.

The JSON API allows executing CLI commands and receiving outputs in JSON format which can be easily parsed by the calling software.

Authentication

The JSON API protocol runs over HTTP/HTTPS and uses the existing web authentication mechanism.

In order to access the system via HTTP/HTTPS, an HTTP/HTTPS client is needed to send POST requests to the system.

Warning

HTTPS access to the web-based management console needs to be enabled using the command “web https enable” to allow POST requests.

The HTTPS client must first be authenticated by sending a POST request to the following URL:

```
https://<ip-address>/admin/launch?script=rh&template=json-  
request&action=json-login
```

The POST request content should contain the following data (may also be saved as a file) in a JSON format:

```
{  
  "username": "<user name>",  
  "password": "<user password>"  
}
```

After a successful login, a session ID (cookie) is returned to be used for other HTTPS requests in the system.

Authentication Example

Before sending JSON HTTPS request, the user must first authenticate.

Create a JSON format file that contains the relevant login credentials. For example, add this content to a file called "post.json":

```
{  
  "username": "admin",
```

```
"password": "admin"
}
```

Run the following from your server's shell to create a login session ID in the file: cookiejar.

```
curl -L -X POST -d @post.json -c cookiejar "http://<ip-  
address>/admin/launch?script=rh&template=json-  
request&action=json-login"
```

Upon a successful login, you will receive a reply similar to the following:

```
{  
  "status": "OK",  
  "status_message": "Successfully logged-in"  
}
```

The session ID can now be used in all other JSON HTTPS requests to the system.

If authentication fails, the following message is received:

```
{  
  "status": "ERROR",  
  "status_message": "<Invalid username or password | Please provide username and password>"  
}
```

You may also log in and execute commands in the same JSON request. In this case, the JSON file must be in the following format:

```
{  
  "username": "<user name>",
```

```
"password" : "<user password>",  
"commands | cmd" : [ "<cli command 1>", "<cli command 2>" ] | "<cli command>",  
"execution_type" : "sync | async"  
}
```

For example:

```
{  
  "username" : "admin",  
  "password" : "admin",  
  "cmd" : "show fan"  
}
```

If login is successful, the JSON API response appears. Otherwise, login failure response is presented.

Changing Initial Password Through JSON API

This section provides support for changing the default password through JSON API.

Expected Input

- To change the initial password, the payload will be as follows:

```
{  
  "username" : "admin",  
  "password" : "admin",  
  "initial_admin_password" : "admin",  
  "initial_monitor_password" : "monitor"  
}
```

Expected Outputs

- Admin and Monitor passwords cannot be changed because they have already been changed:

```
{
  "status": "ERROR",
  "status_message": "'admin' password was already set & 'monitor' password was already set"
}
```

- Admin and Monitor passwords were changed successfully:

```
{
  "status_message": "'admin' password was updated successfully & 'monitor' password was updated successfully"
}
```

- Admin and Monitor passwords were not updated:

```
{
  "status": "OK",
  "status_message": "'admin' password was updated successfully & 'monitor' password was updated successfully"
}
```

- One of the passwords of either Admin or Monitor was changed, while the other remained the same:

```
{
  "status": "<ERROR|OK>",
  "status_message": "'admin' password was already set | 'admin' password was updated successfully"
}
```

```
}
```

- When the payload does not have initial passwords, check change-password nodes to see if there is no updated password return in this JSON payload:

```
{  
  "status": "ERROR",  
  "status_message": "Please set the default password for 'admin' account  
  by using initial password parameters"  
}
```

When there is no issue with the login, flow will proceed without needing this step.

JSON API Logout

To logout, do the following:

1. Performs a POST operation on URL (the request should contain the session cookie):

```
[switch_ip]/script=rh&template=json-request&action=json-  
logout
```

2. The switch will remove the session and return the following JSON in the response text (in case of error, content will be relevant to the error):

```
{  
  "status": "OK",  
  "status_message": "Successfully logged-out"  
}
```

3. Make sure there is no cookie. A request with an invalid cookie will respond that the cookie is invalid.

Logout Example

To logout, use the “curl” tool.

```
curl -b cookiejar "http://[switch-ip]/admin/launch?script=rh&template=json-  
request&action=json-logout"
```

Sending the Request

After successful authentication, the HTTPS client can start sending JSON requests. All requests (POST and GET) should be sent to the following URL:

After the request is handled in the system the HTTPS client receives a JSON response with an indication of the request execution result. If there is data resulting from the request, it is returned as part of the response.

See [“JSON Request Format”](#) for the CLI request format.

See [“JSON Response Format”](#) for the reply format. JSON requests may also be sent using the WebUI. For more information on using the WebUI with JSON, please refer to [“JSON Request Using WebUI”](#).

JSON Request Format

JSON Execution Requests

JSON execution requests are HTTPS POST requests that contain CLI commands to be executed in the system.

Execution request can contain a single command or multiple commands to be executed.

Single command execution request format:

```
{
  "cmd" : "<CLI command to execute>"
}
```

Example:

```
{
  "cmd" : "show interfaces ethernet 1/1"
}
```

Multiple command execution request format:

```
{
  "commands" : [ "<CLI cmd 1>", "<CLI cmd 2>", ... , <CLI cmd n>" ]
}
```

Example:

```
{
  "commands" :
  [
    "show interfaces ethernet 1/1",
    "show interfaces ethernet 1/2"
  ]
}
```

In case of a multiple command request, the execution of the commands is done in the order they appear in the execution list. Note that the execution of a multiple command request will be stopped upon first failure. That is, in case the execution of one of the commands fails, none of the remaining commands will be executed.

Execution Types

Execution requests can be either synchronous (default) or asynchronous.

Synchronous requests will wait for a JSON response from the system. The synchronous request has a defined wait time after which the user will receive a timeout response. The timeout for a synchronous request is configurable by the user and is 30 seconds by default (see the CLI command `“json-gw synchronous-request-timeout”`).

Asynchronous requests will return immediately after sending the request with a reply containing a “job_id” key. The user can use the given job ID to later query for request status and execution results. Queries for asynchronous request results are guaranteed to be accessible up to 60 seconds after the request has been completed. After the result has been successfully queried it will be deleted and will no longer be accessible (even if the result is not 60 seconds old).

To specify the execution type, the user needs to add the following key to the JSON execution request:

```
"execution_type" : "<async|sync>"
```

Example:

```
{
  "execution_type" : "async",
  "cmd" : "show interfaces ethernet 1/1"
}
```

JSON Query Requests

JSON Query requests are HTTPS GET requests that contain a job ID parameter. Using a query request, the user can get information on the current execution state of an ongoing request or the execution results of a completed request. To send a query request, the user should add the following parameters to the JSON URL:

```
job_id=<job number>
```

Example:

```
https://<switch-ip-address>/admin/launch?script=json&job_id=<job number>
```

See [“JSON Examples”](#) for more examples.

JSON Response Format

Warning

Set commands normally do not return any data or output. If a set command does return an output, it will be displayed in the “status_message” field.

Single Command Response Format

The HTTPS POST response format structure is a JSON object consisting of 4 name-value pairs as follows:

```
{
  "executed_command" : "<CLI command that was executed>",
  "status" = "<OK|ERROR>",
  "status_message" = "<information on the status received>",
  "data" = {the information that was asked for in the request}
```

```
}
```

- `executed_command`—the CLI command that was executed in the request
- `status`—the result of the request execution:
 - “OK” if the execution is successful
 - “ERROR” in case of a problem with the execution
- The value type of this key is “string”.
- `data`—a JSON object containing the information requested. Returns an empty string if there is no data.
- `status message`—additional information on the received status. May be empty. The value type of this key is “string”.

Example:

```
{  
  "executed_command": "show interfaces ethernet 1/1",  
  "status": "OK",  
  "status_message": "",  
  "data":  
  {  
    "speed": "40GbE",  
    "admin_state": "up"  
  }  
}
```

See [“JSON Examples”](#) for more examples.

Multiple Command Response Format

The HTTPS response format structure is a JSON object consisting of a list of JSON results. Each JSON structure in the list is structured the same as in the single command execution response (see the [previous section](#)).

However, the status field can contain in this case an additional value, "ABORTED", in case a previous command failed. This status value indicates that the command has not been executed at all in the system.

```
{
  "results": [
    {
      "executed_command": "<...>",
      "status": "<OK|ERROR|ABORTED>",
      "status_message": "<...>",
      "data": {...}
    },
    {
      "executed_command": "<...>",
      "status": "<OK|ERROR|ABORTED>",
      "status_message": "<...>",
      "data": {...}
    },
    ...
    {
      "executed_command": "<...>",
      "status": "<OK|ERROR|ABORTED>",
      "status_message": "<...>",
      "data": {...}
    }
  ]
}
```

Example:

```
{
```

```

"results": [
  {
    "executed_command": "show interfaces ethernet 1/1",
    "status": "OK",
    "status_message": "",
    "data": {"speed": "40GbE", "admin_state": "up"}
  },
  {
    "executed_command": "show interfaces ethernet 1/100",
    "status": "ERROR",
    "status_message": "wrong interfaces name",
    "data": ""
  },
  {
    "executed_command": "show interfaces ethernet 1/2",
    "status": "ABORTED",
    "status_message": "",
    "data": ""
  }
]
}

```

See [“JSON Examples”](#) for more examples.

Query Response Format

Response to a query request can be of two types. In case the request completes its execution, the response will be similar to the single/multiple command response format, depending on the format of the request, and will display the execution results.

In case the execution is not complete yet, the response format will be similar to the single command response format. However, the status field will contain in this case the value “PENDING” to indicate that the request is still in progress. In addition, the “executed_command” field will contain the current request command being handled by the system.

Example:

```
{
  "executed_command": "show interfaces ethernet 1/1",
  "status": "PENDING",
  "status_message": "",
  "data": ""
}
```

Asynchronous Response Format

Response to an asynchronous request is similar to the HTTPS response format of the single command response. However, an additional unique field will be added, “job_id”, containing the job id number for querying the request later. The value of the job_id key is of type string.

Another difference is that the “executed_command” field will be empty.

Example:

```
{
  "executed_command": ""
  "status": "OK"
  "status_message": ""
  "data": ""
  "job_id": "2754930426"
}
```

Supported Commands

- Show commands
- Set commands— all non-interactive CLI set commands are supported

Warning

Interactive commands are commands which require user interaction to complete (e.g. type “yes” to confirm). These commands are not supported by the JSON API.

JSON Examples

The following examples use curl (a common tool in Linux systems) to send HTTPS POST requests to the system.

Synchronous Execution Request Example

Single Command

This example sends a request to query the system profile.

Request (save it to a file named req.json):

```
{"cmd" : "show system profile" }
```

Send the request:

```
curl -b /tmp/cookie -X POST -d @req.json "https://10.10.10.10/admin/launch?script=json"
```

When the system finishes processing the request, the user will receive a response similar to the following:

```
{  
  "status" : "OK",
```

```
"executed_command" : "show system profile",
"status_message" : "",
"data" : {
    "Profile" : "ib",
    "Adaptive Routing" : "yes",
    "Number of SWIDs" : "1"
}
}
```

Multiple Commands

This example sends a request to change an interface description and then queries for its status.

Request (save it to a file named req.json):

```
{"commands" : [ "interfaces ib 1/1 description test description",
"show interfaces ib 1/1 status" ] }
```

Send the request:

```
curl -b /tmp/cookie -X POST -d @req.json "https://10.10.10.10/admin/launch?
script=json"
```

When the system finishes processing the request, the user will receive a response similar to the following:

```
{
    "results" : [
        {
            "status" : "OK",
```

```

    "executed_command" : "interfaces ib 1/1 description test description",
    "status_message" : "",
    "data" : ""
  },
  {
    "status" : "OK",
    "executed_command" : "show interfaces ib 1/1 status",
    "status_message" : "",
    "data" : {
      "IB1/1" : [
        {
          "Description" : "test description",
          "Speed" : "56.0 Gbps",
          "Logical port state" : "Initialize",
          "Physical port state" : "LinkUp",
          "Current line rate" : "56.0 Gbps",
          "IB Subnet" : "infiniband-default"
        }
      ]
    }
  }
]
}

```

Asynchronous Execution Request Example

This example sends an asynchronous request to change an interface description and then queries for its status.

Request (save it to a file named req.json):

```

{"execution_type" : "async",
 "commands" : ["interfaces ib 1/1 description test description",

```

```
"show interfaces ib 1/1 status" ] }
```

Send the request:

```
curl -b /tmp/cookie -X POST -d @req.json "https://10.10.10.10/admin/launch?script=json"
```

The system immediately returns a response similar to the following:

```
{
    "executed_command": "",
    "status": "OK",
    "status_message": "",
    "data": "",
    "job_id": "91329386"
}
```

Query Request Example

This example sends a request to query for a job ID received from a previous execution request.

The request is an HTTPS GET operation to the JSON URL with the "job_id" parameter.

Send the request:

```
curl -b /tmp/cookie -X GET "https://10.10.10.10/admin/launch?script=json&job_id=91329386"
```

If the system is still processing the request, the user receives a response similar to the following:

```

{
    "executed_command": " interfaces ib 1/1 description test description ",
    "status": "PENDING",
    "status_message": "",
    "data": ""
}

```

If the system is done processing the request, the user receives a response similar to the following:

```

{
    "results": [
        {
            "status": "OK",
            "executed_command": "interfaces ib 1/1 description test description",
            "status_message": "",
            "data": ""
        },
        {
            "status": "OK",
            "executed_command": "show interfaces ib 1/1 status",
            "status_message": "",
            "data": {
                "IB1/1": [
                    {
                        "Description": "test description",
                        "Speed": "fdr",
                        "Logical port state": "Initialize",
                        "Physical port state": "LinkUp",
                        "Current line rate": "56.0 Gbps",
                        "IB Subnet": "infiniband-default"
                    }
                ]
            }
        }
    ]
}

```

```
}
}
]
}
```

Error Response Example

General Error

This example sends a request with an illegal JSON structure.

Request—without closing bracket “]” (save it to a file named req.json):

```
{"commands" : [ "interface ib 1/1 description test description" ,
"show interfaces ib 1/1 status" }
```

Send the request:

```
curl -b /tmp/cookie -X POST -d @req.json "https://10.10.10.10/admin/launch?
script=json"
```

Error response:

```
{
  "status" : "ERROR",
  "executed_command" : "",
  "status_message" : "Handle request failed. Reason:\nIllegal JSON structure found in given JSON
data.\nExpecting , delimiter: line 1 column 95 (char 94)",
  "data" : ""
}
```

Multiple Command Request Failure

This example sends a multiple command request where one command fails.

Request—with a non-existing interface (1/200) (save it to a file named req.json):

```
{
  "execution_type": "sync",
  "commands": [
    "interfaces ib 1/1 speed sdr",
    "interfaces ib 1/200 speed sdr",
    "interfaces ib 1/3 speed sdr" ]
}
```

Send the request:

```
curl -b /tmp/cookie -X POST -d @req.json "https://10.10.10.10/admin/launch?
script=json"
```

Error response:

```
{
  "results": [
    {
      "status": "OK",
      "executed_command": "interfaces ib 1/1 speed sdr",
      "status_message": "",
      "data": ""
    },
    {
      "status": "ERROR",
      "executed_command": "interfaces ib 1/200 speed sdr",
```

```
        "status_message" : "% 1st interface does not exist",
        "data" : ""
    },
    {
        "status" : "ABORTED",
        "executed_command" : "interfacesib 1/3 speed sdr",
        "status_message" : "",
        "data" : ""
    }
]
}
```

JSON Request Using WebUI

The
MLNX-OS

WebUI also allows users to send JSON HTTPS POST and GET requests.

Log into the WebUI, go to the “Setup” tab, and select “JSON API” from the left side menu.

Warning

This section is displayed only if JSON API is enabled using the command “json-gw enable”.

To Execute a JSON Request

1. Choose “Execute JSON command”.
2. Choose the “execution_type” from the drop down list.

3. In the “commands” field, type the CLI command(s) to execute.
Use the “+” and “-” buttons to add or remove additional commands to the request.
4. Click “Submit”.

The JSON response is then shown in the “JSON Response” box below.

The HTTPS method (HTTPS POST in this instance) and the URL used to send the request will be displayed next to the “HTTPS Method” and “URL” field respectively.

The screenshot shows the 'JSON API' configuration page. On the left is a navigation menu with 'JSON API' selected. The main area is divided into three sections:

- JSON Configuration:** 'Enable JSON API' is checked. There are 'Apply' and 'Cancel' buttons.
- JSON Commands:** 'Execute JSON command' is selected. A text area contains a JSON request:


```
{
  "execution_type": "sync",
  "commands": [
    "show system profile"
  ]
}
```

 There are 'Submit' and 'Cancel' buttons.
- JSON Response:** Shows the HTTP Method as 'POST' and the URL as 'http://.../admin/launch?script=json'. Below is a text box containing the JSON response:


```
{
  "results": [
    {
      "status": "OK",
      "executed_command": "show system profile",
      "status_message": "",
      "data": {
        "Profile": "ib",
        "Adaptive Routing": "yes",
        "Number of SWIDs": "1"
      }
    }
  ]
}
```

To Query an Asynchronous JSON Request

1. Choose “Query asynchronous job status”.
2. Type the job ID in the “Job ID” text box.
3. Press “Query Status”.

The JSON response is then shown in the “JSON Response” box below. The HTTPS method (HTTPS GET in this instance) and the URL used to send the request will be displayed next to the “HTTPS Method” and “URL” field respectively.

The screenshot shows the 'JSON API' configuration page. On the left is a navigation menu with 'JSON API' selected. The main content area is divided into three sections: 'JSON Configuration', 'JSON Commands', and 'JSON Response'. In the 'JSON Configuration' section, 'Enable JSON API' is checked, with 'Apply' and 'Cancel' buttons. The 'JSON Commands' section has two radio buttons: 'Execute JSON command' (unselected) and 'Query asynchronous job status' (selected). Below this, the 'Job ID' is entered as '3747623153', with 'Query Status' and 'Cancel' buttons. The 'JSON Response' section shows the 'HTTP Method' as 'GET' and the 'URL' as 'http://[redacted]/admin/launch?script=json&job_id=3747623153'. A large text box displays the following JSON response:

```
{
  "results": [
    {
      "status": "OK",
      "executed_command": "show system profile",
      "status_message": "",
      "data": {
        "Profile": "vpi-single-switch"
      }
    }
  ]
}
```

Additional Reading and Use Cases

For more information about this feature and its potential applications, please refer to the following community post:

- [Getting Started With JSON API](#)

Virtualization

MLNX-OS allows the user to run their own applications on a Linux docker image embedded in the switch software. The container is a pure application sandbox with resource isolation of both memory and compute from the system code/NOS.

Docker container implementation in the OS enhances its VM support to provide a new set of capabilities:

- Network traffic access
Docker containers are implemented in the OS in the same name-space as the network devices allowing the software to send and receive packets from the switch ports by opening a standard Linux socket over the network devices and using an IP address assigned to the device via the legacy management interface (e.g., JSON over HTTP).

Warning

It is recommended to assign a unique port number to the Linux socket to prevent ambiguity of applications between the container and the OS.

- Calling the SDK interfaces
Applications running in the docker container are able to implement a set of tools pertaining only to the container such as telemetry features within the network devices. By calling the switch SDK APIs, it can also read data that is not exposed in the OS user interface, or register to receive events that occur in the system (e.g., port up/down).

Warning

The container implementation does not limit the container developer from calling the SDK to set parameters. However this is strongly discouraged as it may cause unexpected system

behavior where the OS and the container application manage the same resources.

- Query the Linux tables provisioned by OS such as neighbor cache, routing tables, L3 interfaces attributes etc.

Limiting the Container's Resources

It is possible to configure multiple containers in dockers, however, they would compete for the same memory and compute resources allocated by the switch software (varies for different systems). To ensure system stability and that no random process is killed to free up memory, it is strongly recommended that all resource configurations done in the container utilize OS user interfaces such as JSON/SNMP and take advantage of the internal loopback interface.

Memory Resources Allocation Protocol

The Linux docker supports a hard limit to control memory resource allocation which limits the container to a given amount of user/system memory.

To set the amount of memory allocated to the container, run the following command:

```
switch (config) # docker start imagename latestver containername  
init memory 25 label newlabel privileged sdk network docker usb-  
mount
```

CPU Resource Allocation Protocol

Containers have unrestricted access to the host machine's CPU cycles but it is possible to set a number of constraints to limit the containers' access.

To set up limitations or regulate the containers access to CPU resources, run the following command:

```
docker start imagename latestver containername init cpus 0.2 label
```

```
new_label privileged sdk network
```

Upgrade Ramifications

Changing Docker Storage Driver

As a result of the upgrade, the docker's storage driver changes, which may cause a few additional changes:

- The containers and docker images become inaccessible to the user (the docker process will not run)
- The user can reach their old containers after a rollback procedure
- The “no docker” command erases all containers and images, including those that were reachable after rollback. Rolling back after running the “no docker” command may result in failure to create configured containers from unknown images.
- The user is advised to execute the “no docker” command at some point in order to clear unused disk space
- It is possible to reload the Docker images after upgrade with the command: `docker load <image_name>_<image_version>.img.gz`
- The images are presented with tab-tab after “docker load “ (in cli)
- It is also possible to load the images after rollback after "no docker" was execute. That means that containers can be restarted after upgrade/rollback if their images are loaded (with “docker load”).

It is possible to move containers from the current version to the updated one by executing the following steps:

Before upgrade:

1. **Save the container as an image**—run the command: “`docker commit <container_name> <new_image_name> <new_image_version>`”. For example: `docker commit my_name my_image my_version`. You can see the new image by running: “`show docker images`”.

2. **Save the image**—run the command: “docker save <image_name> <image_version> <file_name-optional>”. For example: docker save my_image my_version.
3. **Upload the image**—save the image to a local repository by running: “image upload <image_file_name> <destination_path>”. For example: image upload my_image_my_version.img.gz scp://username:password@fit150/auto/my_dir. The <image_file_name> is presented after clicking tab-tab.

After upgrade:

1. **Start docker**—run the “no docker shutdown” command.
2. **Fetch the restored image**—run the “image fetch <file_name>” command. For example: image fetch scp://username:password@fit150/auto/my_dir/my_image_my_version.img.gz
3. **Load the image**—run the “docker load <image_file_name>” command. For example: docker load my_image_my_version.img.gz
4. **Start a container with the defined image**—now that the image with all the content from the container is available in the new environment, start a container with this image. Run the command: “docker start <image_name> <image version> <docker_name> <starting_point>| privileged | label | memory | cpus | usb-mount”. For example: docker start my_image my_version new_container now

Warning

After an upgrade operation there is a need to rerun copy-sdk command (in case in use).

Docker Containers Commands

docker

	docker [logging-level <log-level>] no docker
--	---

	Enables dockers then enters docker configuration context. The no form of the command disables dockers, removes configuration, and deletes all containers and docker images.
Syntax Description	N/A <ul style="list-style-type: none"> log-level—logging-level for docker. Possible levels: debug error, fatal info, warn
Default	N/A
Configuration Mode	config
History	3.6.2940 3.9.2300—Added log-level option
Example	switch (config) # docker
Related Commands	
Notes	

docker login

	docker login <username> <cleartext password> [server <server address>] Logs in to remote docker repositories.	
Syntax Description	username	Username
	cleartext password	There are 2 options to enter password using the above command: <ol style="list-style-type: none"> In command—cleartext Using interactive shell—entering all needed input except the password will prompt the user to provide a password which will not be visible while typing. (masked by *)

	server	The "server" field is not mandatory. In case it is not present, the docker will try to login into docker hub repository.
Default	N/A	
Configuration Mode	config	
History	3.9.1600	
Example	switch (config) # docker login abcd 1234	
Related Commands	show docker login	
Notes		

docker logout

	docker logout [server <server address>] Logs out from remote server.	
Syntax Description	N/A	
Default	N/A	
Configuration Mode	config	
History	3.9.1600	
Example	switch (config) # docker logout	
Related Commands		
Notes	<ul style="list-style-type: none"> There is no need to provide username as only a single user can be connected to a specific server in any given time 	

commit

	<code>commit <container-name> <image-name> <image-version></code> Creates a new image from a running container.	
Syntax Description	container-name	Name of the running container to commit (limited to 180 characters)
	image-name	Name of the new image to be created
	image-version	Version of the new image to be created
Default	N/A	
Configuration Mode	config docker	
History	3.6.2940 3.6.8008: Added new character limitation for container-name	
Example	switch (config docker) # commit mycontainer test latest	
Related Commands		
Notes		

copy-sdk

	<code>copy-sdk</code> The command provides access to the switch SDK APIs giving applications running on docker access to the switch hardware.	
Syntax Description	N/A	
Default	N/A	
Configuration Mode	config docker	
History	3.6.4110 3.8.1000: Updated notes 3.8.2100: Updated notes	
Example	switch (config docker) # copy-sdk	

Related Commands	
Notes	<ul style="list-style-type: none"> • Copying SDK files to a USB mounted folder is not allowed • After an upgrade operation there is a need to rerun copy-sdk command (in case in use).

remove image

	remove image <image-name> <image-version> Removes an image from the Linux docker service.	
Syntax Description	image-name	Name of the new image to be deleted
	image-version	Version of the new image to be deleted
Default	N/A	
Configuration Mode	config docker	
History	3.6.3520 3.6.2940	
Example	switch (config docker) # remove image test latest	
Related Commands	docker	
Notes		

exec

	exec <container-name> <program-executable> Executes a program within a running container.	
Syntax Description	container-name	Name of the running container to commit (limited to 180 characters)

	program-executable	Linux command
Default	N/A	
Configuration Mode	config docker	
History	3.6.3520 3.6.2940	
Example	switch (config docker) # exec mycontainer "ls -la"	
Related Commands	docker	
Notes		

label

	<p>label <label name> no label <label name> Creates a label which can be used as a shared storage between containers. The no form of the command removes the label.</p>
Syntax Description	N/A
Default	N/A
Configuration Mode	config docker
History	3.6.4110
Example	switch (config docker) # label new_label
Related Commands	
Notes	

load

	load <image-name> Loads an image from a TAR archive.	
Syntax Description	image-name	Name of the TAR image to be loaded
Default	N/A	
Configuration Mode	config docker	
History	3.6.2940	
Example	switch (config docker) # load test	
Related Commands	docker	
Notes		

pull

	pull <image-name>[:<version>] Pulls a docker image from a docker repository.	
Syntax Description	image-name	Image name Format: Name:Version If only "Name" is provided, "version" defaults to latest
Default	N/A	
Configuration Mode	config docker	
History	3.6.2940	
Example	<pre>switch (config docker) # pull test Using default tag: latest latest: Pulling from library/test 45a2e645736c: Pull complete Digest: sha256:c577af3197aacedf79c5a204cd7f493c8e07ffbce7f88f7600bf19c688c38799 Status: Downloaded newer image for test:latest</pre>	

Related Commands	docker
Notes	

save

	save <image-name> <image-version> <filename> Saves an image to a TAR archive.	
Syntax Description	image-name	Image name
	image-version	Image version
	filename	Name of the file in which to save the image
Default	N/A	
Configuration Mode	config docker	
History	3.6.2940 3.6.8008: Updated command syntax	
Example	switch (config docker) # save busybox latest my_image Saving and compressing image: busybox version: latest this could take a while... switch (config docker) #	
Related Commands	docker docker load	
Notes	After the file is created, the filename gets appended a *.gz suffix.	

shutdown

	shutdown no shutdown Stops all docker containers, and deletes all non-auto containers. The no form of the command enables the docker Linux service and runs all configured auto-start containers
Syntax Description	N/A
Default	N/A
Configuration Mode	config docker
History	3.6.2940
Example	switch (config docker) # no shutdown
Related Commands	docker
Notes	

start

	start <image-name> <image-version> <container-name> <starting-point> [privileged {network sdk}] [cpus <max-cpu-resources>] [memory <max-memory>] [usb-mount] [host-trust [user <username>]] [logging-facility <logging-facility-level>] [user-env <env-string>] no start <container-name> Starts a new container from an image. The no form of the command stops a running docker container.	
Syntax Description	image-name	Name of the new image to start.
	image-version	Version of the image to start.
	container-name	Name of the running container to commit (limited to 180 characters).
	privileged	<ul style="list-style-type: none"> network—adds network privileges to the container (--privilege flag)

	<ul style="list-style-type: none"> • sdk—adds required mounts to use the switch SDK from the container
starting-point	<ul style="list-style-type: none"> • init—persistent, start the container after boot, when system initialization is done • data-path-ready—persistent, start the container after boot, when data-path is ready to be configured • now—start the container now, this is not persistent • now-and-data-path-ready—starts the container now and after boot, when data-path is ready to be configured • now-and-init—starts the container now and after boot, when system configuration is done
cpus	Sets how much of the available CPU resources a container can use (e.g., “cpus 1.5” guarantees at most one and a half of the available CPUs for the container).
memory	Sets the maximum amount of memory the container can use in MB. The minimum amount of memory to configure is 4MB.
usb-mount	Enables USB mount to the docker container.
host-trust	Allows SSH operation from within the container to localhost without the need to supply password.
logging-facility-level	Available Parameters: auth, authpriv daemon, ftp, kern, local0, local1, local2, local3, local4, local5, local6, local7, lpr, mail, news, syslog, user, uucp
env-string	Up to 16 user-defined environment variables. User-defined environment variable are separated by a comma (e.g., key1=value1,key2=value2)
Default	N/A
Configuration Mode	config docker
History	3.6.2940 3.6.3520: Added “privileged” parameter 3.6.8008: Added the options “now-and-data-path-ready” and “now-

	<p>and-init”, new character limitation for container-name, and updated the description of the parameter “memory”</p> <p>3.8.1000; Updated syntax description</p> <p>3.9.2000: Added host-trust option which adds support for SSH operation from within the container to localhost without the need to supply password (when activating host-trust without supplying user, user admin will be used).</p> <p>3.9.2300: Added logging-facility and user-env options</p>
Example	<pre>switch (config docker) # start centos latest test now</pre> <p>Starting docker container. Please wait (this can take a minute)...</p> <pre>switch (config) # docker start imagename latestver containername init cpus 0.2 memory 25</pre>
Related Commands	docker
Notes	<ul style="list-style-type: none"> • The no form of the command removes the container if it is not persistent. • If trust is set and username is not given, user admin will be chosen by default.

image upload

	<pre>image upload <filename> <upload_url></pre> <p>Uploads an image file to a remote host.</p>	
Syntax Description	filename	Name of file
	upload_url	FTP, TFTP, SCP and SFTP are supported (e.g., scp://username[:password]@hostname-or-ip/path/filename)
Default	N/A	
Configuration Mode	config	
History	3.6.2940	
Example	<pre>switch (config) # image upload centos.img.gz scp://username:password@192.168.10.125/var/www/html/<image_name></pre>	

Related Commands	
Notes	

file image upload

	file image upload <filename> <upload_url> Uploads a file to a remote host.	
Syntax Description	filename	Name of file
	upload_url	FTP, TFTP, SCP and SFTP are supported (e.g., scp://username[:password]@hostname/path/filename)
Default	N/A	
Configuration Mode	config	
History	3.6.2940	
Example	switch (config) # file image upload centos.img.gz scp://username:password@192.168.10.125/var/www/html/<image_name>	
Related Commands		
Notes		

show docker

	show docker Displays docker running state.	
Syntax Description	N/A	
Default	N/A	
Configuration Mode	Any command mode	

History	3.9.2000
Example	switch (config) # show docker Dockers state: started Docker log-level: warn
Related Commands	
Notes	

show docker containers

	show docker containers <container_name> Displays set parameters on containers already running, and containers planned to run in the future.
Syntax Description	N/A
Default	N/A
Configuration Mode	Any command mode
History	3.6.8008 3.8.1000: Updated example 3.9.2000: Updated example, adding host-trust option 3.9.2300: Updated example, adding "user-defined variables" and "log-facility" fields
Example	switch (config) # show docker containers cont_example: image : busybox version : latest status : running start point : data-path-ready cpu limit : 0.2 memory limit: 10m labels : - privileges : network, sdk usb mount : enabled host trust : admin log-facility: kern user-defined variables: name1: value1

name2: value2

another_container:

image : busybox

version : latest

status : -

start point : init

cpu limit : 0.2

memory limit: 10m

labels : my_label

privileges : network, sdk

usb mount : disabled

host trust : admin

log-facility: kern

user-defined variables:

name1: value1

name2: value2

OS_SYSTEM_TYPE : MSB7700

OS_VERSION : 3.9.2300

3OS_DOCKERD_VRF_CONTEXT : vrf-default

OS_DOCKERD_LINUX_VRF_CONTEXT: vrf_vrf-default

switch (config) # show docker containers cont_example

cont_example:

image : busybox

version : latest

status : running

start point : data-path-ready

cpu limit : 0.2

memory limit: 10m

labels : -

privileges : network, sdk

usb mount : enabled

host trust : admin

log-facility: kern

user-defined variables:

name1: value1

name2: value2

OS_SYSTEM_TYPE : MSB7700

OS_VERSION : 3.9.2300

OS_DOCKERD_VRF_CONTEXT : vrf-default

OS_DOCKERD_LINUX_VRF_CONTEXT: vrf_vrf-default

Related Commands	
Notes	<ul style="list-style-type: none"> • If a container is already started, the status field displays its current status • If a container is configured to run on the next boot, the start point field displays when it will start • If there is a mismatch between the configuration of a running container and its next-boot configuration, two entries for the container are shown with both of the configurations • For running containers, environment variables that are automatically passed to docker container are revealed (i.e., OS_SYSTEM_TYPE, OS_VERSION, OS_DOCKERD_VRF_CONTEXT, OS_DOCKERD_LINUX_VRF_CONTEXT) • If no user-defined variables were configured, "user-defined variables" field is hidden

show docker images

	show docker images Display docker images.
Syntax Description	N/A
Default	N/A
Configuration Mode	Any command mode
History	3.6.3520 3.6.2940: Updated example
Example	
<pre>switch (config) # show docker images ----- Image Version Created Size ----- ubuntu latest Less than a secon 117MB d ago ubuntu-sdk v1 41 seconds ago 215MB</pre>	
Related Commands	

Notes	
-------	--

show docker ps

	show docker ps Display docker containers.
Syntax Description	N/A
Default	N/A
Configuration Mode	Any command mode
History	3.6.3520 3.6.2940: Updated example
Example	
switch (config) # show docker ps	

Container Image:Version Created Status	

my_ubuntu_app ubuntu:latest 56 seconds ago Up 50 seconds	
Related Commands	
Notes	This command is available only after Linux dockers are enabled (“no dockers shutdown”)

show docker labels

	show docker labels Displays docker labels.
Syntax Description	N/A
Default	N/A
Configuration Mode	Any command mode

History	3.6.4110
Example	<pre>switch (config) # show docker labels Storage label : label_name1 configured containers list : cont_name2 active containers list : cont_name1 Storage label : label_name2</pre>
Related Commands	
Notes	

show docker login

	<pre>show docker login Displays docker login.</pre>
Syntax Description	N/A
Default	N/A
Configuration Mode	Any command mode
History	3.9.1600
Example	<pre>switch (config) # show docker login Servers: https://index.docker.io/v1/ nvcr.io</pre>
Related Commands	docker login
Notes	

show docker stats

	show docker stats [<name>] Displays Linux docker statistics.	
Syntax Description	name	Docker whose stats to display
Default	N/A	
Configuration Mode	Any command mode	
History	3.6.8008 2.9.2300: Added example	
Example	<pre> switch (config) # show docker stats ----- Container CPU % Memory Memory Memory % Block Block Pids Usage Limit IN OUT ----- container1 0.00% 952K 1000M 0.09% 0B 0B 1 </pre>	
Related Commands		
Notes	This command is available only after Linux dockers are enabled (“no dockers shutdown”)	

Telemetry, Monitoring, and Debuggability

- [Logging](#)
- [Link Diagnostic Per Port](#)
- [Signal Degradation Monitoring](#)
- [Event Notifications](#)
- [Buffer Histograms Monitoring](#)
- [Statistics and Alarms](#)
- [Management Information Bases \(MIBs\)](#)

User Management, Authentication, & Security

- [User Management & Security](#)
- [Cryptographic \(X.509, IPSec\) and Encryption](#)

InfiniBand Switching

The following pages provide information on configuring InfiniBand protocols and features.

- [Node Name](#)
- [Fabric](#)
- [IB Router](#)
- [InfiniBand Interface](#)
- [Subnet Manager](#)
- [Subnet Manager High Availability](#)

Appendixes

The document contains the following appendixes:

- [Appendix: Enhancing System Security According to NIST SP 800-131A](#)
- [Appendix: Splunk Integration with NVIDIA Products](#)
- [Appendix: Show Commands Not Supported By JSON API](#)

Document Revision History

Rev 7.2-3.10.50xx, January 2023

There are no changes to this version of the user manual. For further information on bug fixes and improvements, see the release notes of this software version.

Rev 7.2-3.10.41xx, November 2022

- Added note in the section "[Getting Started](#)"

Rev 7.1-3.10.40xx, October 2022

Added:

- The `ar-updn` option to the command "[ib sm routing-engines](#)"

Removed:

- The command "`ip l3`" command
- Puppet Agent section

Rev 7.1-3.10.31xx, August 2022

Updated:

- The command "[module-type](#)"

Rev 7.0-3.10.30xx, July 2022

Added:

- The command "[ip filter reset-to-default-rules](#)"

Rev 7.0-3.10.22xx, May 2022

There are no changes to this version of the user manual. For further information on bug fixes and improvements, see the release notes of this software version.

Rev 7.0-3.10.21xx, April 2022

There are no changes to this version of the user manual. For further information on bug fixes and improvements, see the release notes of this software version.

Rev 6.9-3.10.20xx, March 2022

Added:

- The command "[ldap nested-group-search](#)"
- The command "[ldap nested-group-depth](#)"
- The command "[ldap nested-group-count](#)"
- Note in the command "[system secure-mode enable](#)"

Updated:

- The command "[show ldap](#)"

Rev 6.9-3.10.12xx, January 2022

There are no changes to this version of the user manual. For further information on bug fixes and improvements, see the release notes of this software version.

Rev 6.9-3.10.11xx, December 2021

Updated:

- Output of the command "[show cpld](#)"

Rev 6.8-3.10.10xx, November/December 2021

Added:

- The command "[no power redundancy-mode](#)"
- The command "[show secure-boot-status](#)"
- The command "[interface ib port-type split-2](#)"
- Description of Quantum-2 interface in "[Break-Out Cables](#)" section

Removed:

- The section "Fabric Inspector"

- The command "fabric zero-counters"
- The command "show fabric"

Updated:

- The options of the command "[slogin](#)"
- Output of the command "[show cpld](#)"
- Updated default value of "[ib sm max-op-vls](#)" from 15 to 4

Rev 6.7-3.9.33xx, August 2021

Updated:

- WebUI look to reflect NVIDIA design
- Example of the command "[show banner](#)"
- Output to reflect NVIDIA

Added:

- The command "[no interface ib mtu](#)"
- The command "[file open-source-licenses upload](#)"
- [ibdiagnet](#) commands
- Note about [SNMP traps and informs](#)
- Note about IP filter

Updated:

- Updated the "[terminal sysrq enable](#)" command to be disabled by default
- Moved "file help-docs upload" and "file eula upload" command to the [Configuration Management Commands](#) section
- The options in the command "[crypto ipsec peer local](#)"
- List of possible output messages in [Link Diagnostic Per Port](#) section

- MAC addresses to all be lowercase
- The section [Automated Periodic Backup](#)
- Field in the command "[show interfaces ib](#)" from 'VL capabilities' was changed to 'VL admin capabilities'
- Field of the command "[show interfaces ib internal](#)" from 'VL capabilities' was changed to 'VL admin capabilities'
- Output of "[show ib smnodes](#)"
- Output of "[show ip interface](#)"
- Output of "[show ip interface port-channel](#)"
- [ACL](#) section to reflect the addition of ACL-based mirroring.

Rev 6.6-3.9.24xx, March 2021

Added:

- Note to the command "[no zero-touch suppress-write](#)"

Rev 6.6-3.9.23xx, February 2021

- Output of the command "[show docker stats](#)"
- "log-level" option to the command "[docker](#)"
- Output example of the command "[show docker](#)"
- Output example of the command "[show docker containers](#)"
- "logging-facility-level" and "env-string" options to the command "[docker start](#)"

Rev 6.6-3.9.21xx, January 2021

There are no changes to this version. The software version was changed due to bug fixes and improvements. For further information, see Release Notes.

Rev 6.5-3.9.20xx, November 2020

- The command "[password](#)"

- The command "[show password hardening](#)"

Updated:

- The section "[Management Information Bases \(MIBs\)](#)"
- Note in the command "[system profile](#)"

Rev 6.5-3.9.19xx, October 2020

There are no changes to this version. The software version was changed due to bug fixes and improvements. For further information, see Release Notes.

Rev 6.5-3.9.16xx, September 2020

Added:

- The command "[docker login](#)"
- The command "[docker logout](#)"
- The command "[show docker login](#)"

Rev 6.5-3.9.13xx, August 2020

Updated:

- Output "[show interfaces ib](#)"

Rev 6.4-3.9.10xx, July 2020

Added:

- Note in [SSH section](#)

Updated:

- Output of "[show ib ha](#)"
- Example of "[show ib fabric port](#)"

Rev 6.4-3.9.0900, May/June 2020

Added:

- The command "[no logging debug-files rotation criteria](#)"
- The command "[no logging files rotation criteria](#)"
- The command "[logging mac masking](#)"
- Notes in "[ssh server login attempts](#)" command
- Note to "[username](#)" command

Updated:

- The command "[show logging](#)"

Rev 6.4-3.9.0600, April 2020

There are no changes to this version. The software version was changed due to bug fixes and improvements. For further information, see Release Notes.

Rev 6.4-3.9.0450, March 2020

Added:

- The command "[show configuration auto-upload](#)"
- The command "[configuration auto-upload](#)"
- Description of [Automated Periodic Configuration File Backup](#)
- Notes to the command "[logging source-interface](#)"

Updated:

- The command "[configuration upload](#)"
- Changed the "[SSH server login record-period](#)" default from 30 days to 1 day

Rev 6.3-3.9.0300, February 2020

Added:

- The command "[show ssh server login record-period](#)"
- SSH server login record-period to the command "[show ssh server](#)"

- Ability to apply reboot to the command "[configuration text file](#)"
- [Splitting capability](#) on modular systems. For more information see "[Break-Out Cables](#)" section.

Updated:

- [LDAP](#) description

Removed:

- The command "[show running-config interface mgmt0](#)"
- The command "[show running-config interface mgmt1](#)"
- The command "power-management width"
- RSA v1 from the command "[ssh server host-key](#)"
- RSA v1 from notes on the command "[ssh server security strict](#)"
- RSA v1 from the example in the command "[show ssh server](#)"
- RSA v1 from the example in the command "[show ssh server host-keys](#)"
- The command "web proxy auth host"

Rev 6.3 December 2019 No changes to this version. The software version was changed due to bug fixes and improvements in cables and speeds. For further information, see Release Notes.

Rev 6.3 November 2019

Added:

- ca-valid option in the "[crypto certificate name](#)" command
- ca-valid option in the "[crypto certificate generation](#)" command
- New command "[ntp server-role disable](#)"
- New ca-valid option to the "[crypto certificate system-self-signed regenerate](#)" command

- The command ["logging_protocol"](#)
- "all-cas," "all-routers," "all-switches," and "all-vcas" parameters to the ["ib_partition member"](#) command

Updated:

- Output example of the ["qos_map_pcp_dei"](#) command
- Output example of the ["show_what_just_happened"](#) command
- Output example of the ["show_crypto_certificate"](#) command

Removed:

- "prefix-modes show-config" option because it is no longer available in the ["cli session"](#) command
- Terminal type vt320 from the ["cli session"](#) command
- "dcb ets enable" command is deprecated

Rev 6.2 September 2019

Added:

- The command ["ib sm calculate-missing-routes"](#)
- The command ["show ib sm calculate-missing-routes"](#)
- [HDR speed on Quantum systems](#)
- Instructions on how to [change initial password through JSON API](#)
- Instruction on [logging out through JSON API](#)
- The section ["Changing Default Password"](#) in order to conform to new law: [California's Senate Bill No. 327, Chapter 886](#)
- The command ["logging"](#)
- The command ["logging filter include"](#)
- The command ["logging filter exclude"](#)

- The command "[no logging filter](#)"

Updated:

- Description of the no form of the "[neighbor ebgp-multihop](#)" command
- Output example of "[show traffic pool interface ethernet](#)" command
- Output example of "[show interfaces ethernet description](#)" command
- Output example of "[show interfaces counters discard](#)" command
- Output example of "[show qos mapping ingress interface egress interface](#)"
- Output example of the "[show what-just-happened](#)" command
- Output example of the "[qos rewrite pcp](#)" command
- Output example of the "[qos rewrite dscp](#)" command
- Output example of the "[qos rewrite map switch-priority pcp dei](#)" command
- Moved JSON API Authentication Example from "JSON Examples" section to JSON API "[Authentication](#)" section
- BGP "[neighbor weight](#)" range

Removed

- The XML API is deprecated as of release 3.8.2000.
- "xml-gw enable" due to XML API depreciation
- The command "show xml-gw" due to XML API depreciation

Rev 6.1 August 2019

No changes to this version. The software version was changed due to bug fixes. For further information, see Release Notes.

Rev 6.0 June 2019

Added:

- "[Appendix: Show command NOT supported by JSON API](#)"

- The command "[chassis ha bipv6](#)"

Updated:

- The output for "show chassis ha" in the following sections: Malfunctioned CPU Behavior, Chassis High Availability Nodes Roles, Takeover Functionally, Chassis Management Commands

Rev 5.9 June 2019 No changes to this version. The software version was changed due to bug fixes. For further information, see Release Notes.

Rev 5.9 May 2019

Added:

- Note on port splitting on externally managed switches under section "[Break-out Cables](#)"

Updated:

- "[Web Interface Overview](#)" with note on the maximum allowed number of WebUI sessions
- The command "[system profile](#)"
- JSON "[Authentication](#)" section
- Section "[Authentication Example](#)"

Rev 5.8 April 2019

Added:

- "Additional Reading and Use Cases" sections referring to various community posts providing more information about a given subject matter
- The command "[show running-config interface](#)"
- The command "[file stats telemetry delete latest](#)"
- The command "[file stats telemetry delete all](#)"
- The command "[file stats telemetry upload latest](#)"
- The command "[file stats telemetry upload all](#)"

- Section "[Upgrade Ramifications](#)" on page "[Linux Dockers](#)"
- The command "[what just happened auto-export](#)"
- The command "[show snmp source interface](#)"
- The command "[snmp server source interface](#)"
- The command "[system manage inband-ib](#)"
- The command "[show system manage inband-ib](#)"

Updated:

- The command "[show stats sample data](#)"
- The command "[snmp-server user](#)"
- The command "[monitor session](#)"
- The command "[ib fabric import](#)"
- The command "[radius-server host](#)"
- The command "[show radius](#)"
- The command "[show ip bgp neighbors received](#)"
- Section "[Destination Interface](#)" on page "[Port Mirroring](#)"
- Section "[Configuring an SNMPv3 User](#)" on page "[Network Management Interfaces](#)"
- Page "[Important Pre-OS Upgrade Notes](#)"
- Page "[Linux Dockers](#)"
- The command "[show json-gw](#)"
- Section "[Router ID](#)" on page "[OSPF](#)"
- Section "[Memory Resources Allocation Protocol](#)" on page "[Linux Dockers](#)"
- The command "[show running-config](#)"
- The command "[start](#)"

- The command ["show docker containers"](#)
- The command ["copy-sdk"](#)
- The command ["cli session"](#)
- The command ["show hosts"](#)
- The command ["web enable"](#)
- The command ["web https"](#)
- Section ["Execution Types"](#) on page ["Network Management Interfaces"](#)
- Section ["Configuring Signal Degradation Monitoring"](#)
- The command ["port-channel load-balance ethernet"](#)
- Section ["Restoring Subnet Manager Configuration"](#)
- Page ["What Just Happened"](#)
- The command ["what just happened"](#)
- The command ["clear what just happened"](#)
- The command ["show what just happened"](#)
- The command ["ip default-gateway"](#)
- Section ["System Configuration"](#)
- The command ["logging trap"](#)
- The command ["logging port"](#)
- The command ["show logging port"](#)
- Page ["Management Source IP Address"](#)

Rev 5.7 December 2018

Added:

- The command "show stats sample data"

Updated:

Rev 5.6 December 2018No changes made since last revision.**Rev 5.5 December 2018**

Added:

- The command “email autosupport mailhub”
- The command “email autosupport recipient”
- The command “show email”
- The command “snmp-server cache enable”
- Section “Break-Out Cables”

Updated:

- Section “IB SM Mgmt”
- Section “Supported Events”
- The command “aaa authorization”
- The command “show aaa”
- Section “System File Encryption”
- The command “show memory”
- Section “Configuring an SNMPv3 User”
- The command “snmp-server user”
- The command “show snmp auto-refresh”
- The command “show puppet-agent”
- Section “Configuring IB Router”
- The command “ib sm m-key”
- The command “show ib sm m-key”

Rev 5.4 November 2018

No changes made since last revision

Rev 5.3 August 2018

Added:

- The command “web proxy auth authtype”
- The command “web proxy auth basic”
- The command “web proxy auth host”

Updated:

- The command “{ip | ipv6} route”
- The command “image install”
- The command “image options”
- Section “Authentication, Authorization and Accounting (AAA)”
- The command “aaa authorization”
- The command “show virtual-machine install”
- The command “show telemetry”
- The command “show telemetry threshold record”
- The command “show system profile”
- The command “show ib fabric messages”
- Section ["Additional Reading and Use Cases"](#) on page ["Licenses"](#)

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 2023, NVIDIA. PDF Generated on 09/02/2025