NetVisor OS Configuration Guide

Arista Networks

www.arista.com

Arista NetVisor Version 2022.7.0.2
DOC-05946-01

Headquarters	Support	Sales
5453 Great America Parkway Santa Clara, CA 95054, USA +1-408-547-5500	+1-855-438-8638 (U.S & Canada) +1-650-289-4717 (International)	+1-408 547-5501 +1-866 497-0000
www.arista.com	**pln-pnsupport@arista.com**	**sales@arista.com**

Table of Contents

Glossary

Glossary of NetVisor UNUM and NetVisor OS Terms

To review the Glossary, refer to the online document.

About the NetVisor OS CLI

This chapter provides information about Arista Networks NetVisor OS command line interface (CLI) on a NetVisor OS switch.

Entering Commands and Getting Help

Commands, options, and arguments are entered at the CLI prompt. A command name must be typed, but included command-completion and help features contribute to the command entry process.

To display a list of commands you use within a command mode, enter a question mark (?), or use the tab key, or type help at the command prompt.

You also display keywords and arguments for each command with this context-sensitive help feature.

Use the complete commands and display keywords and arguments for each command using the tab key to assist with context-sensitive command help and completion.

Table 1, lists the command that you enter to get help specific to a command, keyword, or argument.

Table 1-1: Getting Help

`abbreviated-  command-entry?`	Displays a list of commands that begin with a specific character string. Do not leave a space between the string and question mark.
`abbreviated-  command-entry` `<tab>`	Completes a partial command name.
`?`	Lists all commands.
`command` `?`	Lists all keywords for the command. Leave a space between the command and the question mark.
`command  keyword` `?`	Lists all arguments for the keyword. Leave a space between the command and the question mark.

Where a text string is used, such as *name-string*, the following characters are allowed as part of the text string:

- a-z, A-Z, 0-9, _ (underscore), . (period), , (comma), : (colon), and - (dash).

Note: If you enter an invalid command, using the ? and tab key has no effect and does not return any changes to the CLI.

Note: The CLI has an editing ability similar to UNIX and Linux functionality using emacs keys. For example, **^p** steps backward through previous commands, **^n** moves to the next command in the history, **^a** moves to the first character in the command and **^e** moves to the end of the line, **^u** erases the current line, and **^w** erases the previous word. Also, you can use the up and down arrows on your keyboard to retrieve the last command entered at the CLI.

Finding Command Options

The syntax can consist of optional or required keywords.

To display keywords for a command, enter a question mark (?) at the command prompt or after entering part of a command followed by a space.

NetVisor OS CLI displays a list of available keywords along with a brief description of the keywords.

For example, if you want to see all of the keywords for the command **user**, enter **user ?**.

Table 2, displays examples of using the question mark (?) to assist you with entering commands.

Table 1-2: Finding Command Options

CLI network-admin@switch > ?
 	
`All commands:`
`acl-ip-create`
`acl-ip-delete`
`...`
	Displays a list of commands that begin with a specific character string. Do not leave a space between the string and question mark.

`Switch> user auth`
`User: <user>`
`Password: <password>`
	Completes a partial command name.

?	Lists all commands.
command ?	Lists all keywords for the command. Leave a space between the command and the question mark.
command option ?	Lists all arguments for the option. Leave a space between the command and the question mark.

Note: Other useful options, especially for displaying statistics, include: `sort`, `interval`, `start-time`, `end-time`, `duration`, `count`, `show-interval`, and `show-diff-interval`. The commands that display the statistics have `show-diff-interval` as a command option, while other show commands have `show-interval` as a command option.

About Alternate Command Format

NetVisor OS provides with an alternate command format, where the commands start with a verb instead of a noun. This format omits the hyphen in the command names.

For example, `connection-stats-show` can also be entered as `show  connection-stats`.

The command formats have the same features and can be used interchangeably.

Specifying IP Address Netmasks

Some commands call for the specification of an IP address netmask. The NetVisor OS supports both Classless Inter-Domain Routing (CIDR) and subnet notations.

For example, to specify the range of IP addresses from 192.168.0.0 to 192.168.0.255, you can express as:

- IP address: 192.168.0.0/24 or
- IP address: 192.168.0.0:: 255.255.255.0 (netmask)

Here is a sample of the CIDR to subnet notation mapping:

CIDR	Subnet Mask
/24	255.255.255.0
/25	255.255.255.128
/26	255.255.255.192

Specifying Measurement Units

Many commands include input and output of capacity and throughput. Network values are always in bits and storage values in bytes.

Scale factors are allowed on input and displayed in output as well as shown in the following table.

Table 1-3: Scale Numbers

Scale Indicator	Meaning (Networking)	Meaning (Storage)
K or k	Kilobits	Kilobytes
M or m	Megabits	Megabytes
G or g	Gigabits	Gigabytes
T or t	Terabits	Terabytes

Customizing Show Output Formats

The output generated by the show commands can be customized by using the optional arguments described in the following table.

Table 1-4: Show Output Formats

`format <column_name1>, <column_name2>, <column_nameX>`	Displays only the columns matching the list of column header names. **NOTE**: The list of column names is comma-separated without spaces.
`format all`	Displays all available column headers. This output is also called **verbose mode**. By default, show commands output a terse set of the most commonly useful column headers.
`layout horizontal|vertical`	Specify if you want to display show output in a horizontal or vertical format. Vertical may be more useful for verbose show outputs. For example: `CLI (network-admin@switch) > vlan-show format all layout vertical`
`parsable-delim <separator>`	Displays the output of show command by separating columns by the specified <separator> character(s). For example, parsable-delim , produces a comma-separated output (CSV). **NOTE**: If the parsable-delim option is specified, the column header names (titles) are suppressed from the output.
`pager on|off`	Displays the output as pages (pager `on`) or as a single scroll-able page (pager `off`).

Below are some examples of how the output for the show commands are displayed in NetVisor OS CLI.

To display the select columns, enter the command followed by the option `format <column_name1>, <column_name2>, <column_nameX>` as below:

```
CLI (network-admin@pn-1) switch pn-1 lldp-show format switch, local-port,
chassis-id, port-id, port-desc

switch   local-port chassis-id  port-id port-desc
-------  ---------- ----------  ------- ------------------------------------
pn-1      1          0e0000e8     1      spine-cluster cluster link
pn-1      2          1100013f     5      trunk from cluster-1 to spine cluster
pn-1      3          11000141     5      trunk from cluster-1 to spine cluster
pn-1      4          11000147     5      trunk from cluster-2 to spine cluster
pn-1      5          11000145     5      trunk from cluster-2 to spine cluster
```

To display the output on a local switch that you had logged in (the * indicates the local switch in the example below). You don't need to specify the switch-name for all the commands that you run on the local switch.

```
CLI (network-admin@switch-1) > switch-local
CLI (network-admin@switch-1*) >
```

To view the LLDP details, use the command:

```
CLI (network-admin@usspine02-new) > lldp-show

switch        local-port bezel-port chassis-id port-id port-desc                          sys-name

------------  ----------  ----------  ----------  -------  ---------------------------------  -------------
usspine02-new 37          10          09000d50     53       PN Switch Port(53),Bezel Port(50) usleaf01-new

usspine02-new 41          11          0b00214e     53       PN Switch Port(53),Bezel Port(50) usleaf02-new

usspine02-new 73          19          09000d4f     77       PN Switch Port(77),Bezel Port(20) ghspine01-new

usspine02-new 77          20          0b00214f     77       PN Switch Port(77),Bezel Port(20) ghspine02-new

usspine02-new 89          23          0b000824     61       PN Switch Port(61),Bezel Port(52) usleaf04-new

usspine02-new 97          25          0b002134     61       PN Switch Port(61),Bezel Port(52) usleaf03-new

..
usspine02-new 125         32          09000ba3     101      PN Switch Port(101),Bezel Port(32) usleaf08-new

ghleaf06-new  1           1           0b003236     1        PN Switch Port(1),Bezel Port(1)    ghleaf05-new

ghleaf06-new  5           2           09000d4f     1        PN Switch Port(1),Bezel Port(1)    ghspine01-new
```

To display all the switches in a fabric, for example, use the `switch <tab>` command:

```
CLI (network-admin@pn-sw01) > switch
pn-sw01
pn-swl04
pn-swl02
pn-sw02
leaf_clust_1       pn-swl01,pn-swl02
leaf_clust_2       pn-swl03,pn-swl04
even-cluster-nodes pn-swl02,pn-swl04,pn-swl06,pn-sw02
odd-cluster-nodes  pn-swl01,pn-swl03,pn-swl05,pn-sw01
spine_clust        pn-sw01,pn-sw02
*                  all switches in the fabric
```

To display the output in vertical format, for example, use the command:

```
CLI (network-admin@pn-swl03) > cluster-show layout vertical
switch:                     pn-swl01
name:                       pn-swl0102
state:                      online
cluster-node-1:             pn-swl01
cluster-node-2:             pn-swl02
tid:                        4034
mode:                       master
ports:                      1,125,272
remote-ports:               1,125,272
cluster-sync-timeout(ms):   2000
cluster-sync-offline-count: 3
switch:                     pn-swl02
name:                       pn-swl0102
state:                      online
cluster-node-1:             pn-swl01
cluster-node-2:             pn-swl02
tid:                        4034
mode:                       slave
ports:                      1,125,272
remote-ports:               1,125,272
cluster-sync-timeout(ms):   2000
cluster-sync-offline-count: 3
```

Specifying a Switch or Fabric for Command Scope

In NetVisor Unified Cloud Fabric, a switch is designed as the building block of a fabric and the goal of NetVisor OS design is to manage the fabric of switches as a single switch. Due to this capability, you can run the CLI commands on a local switch, a cluster of switches, other switches in the fabric, or the entire fabric. You do not have to login to each switch to run the commands because all the switches are part of the same fabric. By default, you can run the commands on the switch that you had logged-in.

For example, to disable a port (`port 5`) on the switch that you had logged in (`switch-1`), use the command,

```
CLI(network-admin@Switch-1) >  port-config-modify port 5 disable
```

To specify a different switch for a single command, use the `switch` prefix. For example, use `switch pn-switch-2 port-config-modify port 5 enable` to enables port 5 on pn-switch-2, even if the CLI is connected to a different switch in the fabric:

```
CLI(network-admin@Switch-1) >  switch pn-switch-2 port-config-modify port 5
enable
```

To specify a different switch for a series of commands, use the `switch` prefix with no command. For example,

```
CLI(network-admin@Switch-1) >  switch pn-switch-2 <return>

CLI(network-admin@pn-switch-2) >
```

The CLI prompt changes to indicate that `pn-switch-2` is the switch you are executing commands. You can run other commands on `pn-switch-2` even though the switch that you are physically connected to is switch-1.

For most CLI show commands, the command displays results from all switches in the fabric by default. For example, when the CLI command `port-show` is entered on the switch, it shows the ports of all switches in the fabric.

To specify that a CLI show command should apply to a specific switch, use the switch prefix to the CLI command. For example, to view the show output of the `port-show` command of the switch named pn-switch-1, type the command:

```
CLI(network-admin@Switch-1) > switch pn-switch-1 port-show

 switch        port      bezel-port     status
-----------    ----     ----------      --------------------
pn-switch-1     9         3             phy-up,host-disabled
pn-switch-1     10        3.2           phy-up,host-disabled
```

Displaying NIC Information and Statistics

Starting from NetVisor OS version 6.0.0, you can view the NIC information and packet statistics for management and rear-facing interfaces.

To view the switch NIC information, use the command: `switch-nic-info-show`.

```
CLI  (network-admin@Leaf1)  >  switch-nic-info-show
```

`switch-nic-info-show`	Display switch NIC information.
`name` *`switch-nic-stats-list-name`*	Specify the name of the interface.

For example, to display the NIC information for the interface eth0, use the command:

```
CLI (network-admin@Leaf1) > switch-nic-info-show name eth0

name driver version    firmware     speed rxqnum txqnum rxqsize txqsize autoneg rxpause txpause
---- ------ ------- -------------- ----- ------ ------ ------- ------- ------- ------- -------
eth0 igb    5.4.0-k 3.25,0x800005cd 1000    4      4      4096    4096    on      on      off
```

The output displays the interface name, driver, version, firmware, speed, reception queue size, and transmitter queue size, among other details.

The `switch-nic-stats-show` command displays NIC statistics including the number of input packets, input bytes, output packets, output bytes, errors, and drops.

```
CLI  (network-admin@Leaf1)  >  switch-nic-stats-show
```

`switch-nic-stats-show`	Display switch NIC statistics.
`time` *`date/time:  yyyy-mm-ddTHH:mm:ss`*	Specify the time to start statistics collection.
`start-time` *`date/time:  yyyy-mm-ddTHH:mm:ss`*	Specify the start date and time for statistics collection.
`end-time` *`date/time:  yyyy-mm-ddTHH:mm:ss`*	Specify the end date and time for statistics collection.
`duration` *`duration:  #d#h#m#s`*	Specify the duration for statistics collection.
`interval` *`duration:  #d#h#m#s`*	Specify the interval for statistics collection.
`since-start`	Displays statistics from the start.
`older-than` *`duration:  #d#h#m#s`*	Displays statistics older than the specified date and time.
`within-last` *`duration:  #d#h#m#s`*	Displays statistics within the specified duration.
`name` *`switch-nic-stats-list  name`*	Specify the name of the interface.

For example, to display the NIC statistics on eth0 interface within last 2 minutes, use the command:

```
CLI  (network-admin@Leaf1)  >  switch-nic-stats-show  name  eth0  within-last  2m
```

```
layout  vertical
time:         00:22:35
name:         eth0
ipkts:        3.22M
ibytes:       1.92G
opkts:        1.10M
obytes:       277M
ierrs:        0
oerrs:        0
idrops:       0
odrops:       0
imcast:       2.08M
omcast:       119K
ibcast:       121K
obcast:       37.3K
mcast:        2.08M
bcast:        0
ioverruns:    0
rxnobuf:      0
icrcerrs:     0
iframeerrs:   0
ififoerrs:    0
ofifoerrs:    0
ilongerrs:    0
ishorterrs:   0
```

To display the statistics between a specified `start-time` and `end-time`, use the sample command:

```
CLI (network-admin@leaf1) > switch-nic-stats-show start-time 2020-07-
21T05:00:00 end-time 2020-07-21T06:00:00 layout vertical
time:         05:26:27
name:         eth0
ipkts:        25.2M
ibytes:       3.76G
opkts:        29.7M
obytes:       3.95G
ierrs:        0
oerrs:        0
idrops:       0
odrops:       0
imcast:       834K
omcast:       10.6K
ibcast:       291K
obcast:       352
mcast:        834K
bcast:        0
ioverruns:    0
rxnobuf:      0
icrcerrs:     0
iframeerrs:   0
ififoerrs:    0
ofifoerrs:    0
ilongerrs:    0
```

```
ishorterrs:   0
time:         05:26:27
```

To modify the settings for statistics collection, use the command:

```
CLI  (network-admin@Leaf1)  >  switch-nic-stats-settings-modify
```

`switch-nic-stats-settings-modify`	Modify switch NIC statistics settings.
Specify one or more of the following options:	
`enable|disable`	Enable or disable statistics collection.
`interval` *`duration : #d#h#m#s`*	Specify the time interval to collect NIC statistics. The default value is 1 minute.
`disk-space` *`disk-space-number`*	Specify the disk-space allocated for statistics (including rotated log files). The default value is 50M.

For example, use the command below to modify the interval for statistics collection:

```
CLI  (network-admin@Leaf1)  >  switch-nic-stats-settings-modify  interval  50s
```

To view the current settings, use the command:

```
CLI  (network-admin@Leaf1)  >  switch-nic-stats-settings-show
switch:      Leaf1
enable:      yes
interval:    50s
disk-space:  50M
```

Displaying Connection Statistics

You can view the statistical data collected while connected to a host by using the command:

```
CLI (network-admin@switch1) > connection-stats-show
```

`connection-stats-show`	Display connection statistics while connected to a host.
`time date/time: yyyy-mm-ddThh:mm:ss`	Specify a time to start statistics collection.
`start-time date/time: yyyy-mm-ddThh:mm:ss`	Specify the start-time for statistics collection.
`end-time date/time: yyyy-mm-ddThh:mm:ss`	Specify the end-time for statistics collection.
`duration: #d#h#m#s`	Specify the duration of statistics collection.
`interval duration: #d#h#m#s`	Specify the interval between statistics collection.
`since-start`	Displays statistics from the start of collection.
`older-than duration: #d#h#m#s`	Displays statistics older than a specified duration.
`within-last duration: #d#h#m#s`	Displays statistics within the last specified duration.
`count count-number`	Specify a count for connection statistics
`mac mac-address`	Specify the MAC address of connections
`vnet vnet-name`	Specify the vNET of connections
`vlan vlan_id`	Specify the VLAN of connections.
`ip ip-address`	Specify the IP address of connections.
`port port-number`	Specify the port number of connections.
`iconns iconns-number`	Specify the number of incoming connections.
`oconns oconns-number`	Specify the number outgoing connections.
`obytes`	Specify the number of bytes sent from the client side of the connection.
`ibytes`	Specify the number of bytes received by the client side of the connection.
`total-bytes total-bytes-number`	Specify the total number of bytes for the connection.
`first-seen date/time: yyyy-mm-ddThh:mm:ss`	Specify the time at which an entry was first seen.
`last-seen date/time: yyyy-mm-ddThh:mm:ss`	Specify the time at which an entry was last seen.
`last-seen-ago duration: #d#h#m#s`	Specify the duration since the statistics was last

	seen.

To clear connection statistics that was collected while connected to a host, use the command:

```
CLI (network-admin@switch1) > connection-stats-clear
```

To clear the history of connection statistics that was collected while connected to a host, use the command:

```
CLI (network-admin@switch1) > connection-stats-clear-history
```

To modify statistics collection settings, use the command `connection-stats-settings-modify.`

```
CLI (network-admin@switch1) > connection-stats-settings-modify
```

`connection-stats-settings-modify`	Modify connection statistics settings.
`enable|disable`	Enable or disable collecting connection statistics.
`connection-max-memory` *`connection-max-memory-number`*	Specify the maximum memory allowed for connection statistics.
`connection-backup-enable|connection-backup-disable`	Enable or disable backup for connection statistics collection. This option is disabled by default.
`connection-backup-interval duration:` *`#d#h#m#s`*	Specify backup interval for connection statistics collection.
`client-server-stats-max-memory` *`client-server-stats-max-memory-numbe`*	Specify maximum memory for client server statistics.
`client-server-stats-log-enable|client-server-stats-log-disable`	Enable or disable client server statistics logging. This option is disabled by default.
`client-server-stats-log-interval duration:` *`#d#h#m#s`*	Specify the client server statistics logging interval.
`client-server-stats-log-disk-space` *`disk-space-number`*	Specify the disk-space allocated for client server statistics logging.
`connection-stats-max-memory` *`connection-stats-max-memory-number`*	Specify the maximum memory allowed for connection statistics.
`connection-stats-log-enable|connection-stats-log-disable`	Enable or disable connection statistics logging. This option is disabled by default.
`connection-stats-log-interval duration: #d#h#m#s`	Specify the connection statistics logging interval.
`connection-stats-log-disk-space` *`disk-space-number`*	Specify the disk-space allocated for connection statistics logging.
`service-stat-max-memory` *`service-stat-max-memory-number`*	Specify the maximum memory allowed for service statistics.
`fabric-connection-max-memory` *`fabric-connection-max-memory-number`*	Specify the maximum memory allowed for fabric connection statistics.

`fabric-connection-backup-enable|fabric-connection-backup-disable`	Enable or disable backup for fabric connection statistics collection. This option is disabled by default.
`fabric-connection-backup-interval duration: #d#h#m#s`	Specify the backup interval for fabric connection statistics collection.

In NetVisor OS version 6.0.1, the logging of connection statistics to disk is disabled by default. Specifically, the options below under the command `connection-stats-settings-modify` are disabled by default.

`connection-backup-enable|connection-backup-disable`

`client-server-stats-log-enable|client-server-stats-log-disable`

`fabric-connection-backup-enable|fabric-connection-backup-disable`

`connection-stats-log-enable|connection-stats-log-disable`

You can, however, enable any of the options above for a limited duration. For example:

```
CLI (network-admin@switch1) > connection-stats-settings-modify connection-stats-log-enable
```

> **Note:** Arista recommends disabling connection statistics to minimize disk usage during normal operations.

To view the connection statistics settings, use the command `connection-stats-settings-show`:

```
CLI (network-admin@switch1) > connection-stats-settings-show
switch:                                 switch1
enable:                                 yes
connection-max-memory:                  50M
connection-max-count:                   52012
connection-current-count:               28482
connection-backup-enable:               no
connection-backup-interval:             1m
connection-backup-used-disk-space:      5.54M
client-server-stats-max-memory:         50M
client-server-stats-max-cnt:            71234
client-server-stats-cur-cnt:            132
client-server-stats-log-enable:         no
client-server-stats-log-interval:       1m
client-server-stats-log-disk-space:     50M
connection-stats-max-memory:            50M
connection-stats-max-cnt:               82956
connection-stats-cur-cnt:               108
connection-stats-log-enable:            yes
connection-stats-log-interval:          1m
connection-stats-log-disk-space:        50M
service-stat-max-memory:                1M
service-stat-max-cnt:                   4681
service-stat-cur-cnt:                   0
```

```
fabric-connection-max-memory:                10M
fabric-connection-max-count:                 8090
fabric-connection-current-count:             7234
fabric-connection-backup-enable:             no
fabric-connection-backup-interval:           1m
fabric-connection-backup-used-disk-space:    1.46
```

In this example, the software has enabled connection statistics logging for a duration of 1m.

Running Commands on a Local Switch

You can specify to run commands locally on a switch by using the `switch-local` parameter.

For instance, using `switch-local port-stats-show` displays output for the local switch ports only.

Changing Switch Setup Parameters

You can modify the following switch parameters by using the `switch-setup-modify` command:

- Switch name
- Management IPv4 and IPv6 addresses
- Management IPv4 and IPv6 netmasks
- Management IPv4 and IPv6 address assignments
- In-band IPv4 and IPv6 addresses
- In-band IPv4 and IPv6 netmasks
- Gateway IPv4 and IPv6 addresses
- Loopback IPv4 and IPv6 addresses
- Primary and secondary IPv4 addresses for DNS services
- Domain name
- NTP server and NTP secondary server
- Timezone
- End User License Agreement (EULA) acceptance and timestamp
- Password
- Date
- Enable or disable host ports
- Message of the Day (MOTD)
- Banner
- Management interface LAG: enable or disable
- LACP mode of the management interface
- NTP enable or disable
- IEEE 802.1X authentication of the management interface (when not part of a LAG)

For example if you want to change the date and time on the switch, use the command:

```
CLI (network-admin@switch-1) > switch-setup-modify date 2020-01-28
T22:14:27
```

To view the changed date (see **bold** in output), use the show command:

```
CLI (network-admin@sw-leaf1) > switch-setup-show
switch-name:                sw-leaf1
mgmt-ip:                    10.0.0.01/23
mgmt-ip-assignment:         static
mgmt-ip6:                    aa80::aab4:bbff:fffc:aa65/64
mgmt-ip6-assignment:        autoconf
mgmt-link-state:            up
mgmt-link-speed:            1g
in-band-ip:                  10.168.0.02/24
in-band-ip6:                 aa80::640a:94ff:ff52:aa9d/64
in-band-ip6-assign:         autoconf
gateway-ip:                 10.0.0.1
dns-ip:                     10.10.0.13
dns-secondary-ip:           172.10.0.4
domain-name:                 pluribusnetworks.com
ntp-server:                  0.ubuntu.pool.ntp.org
ntp-secondary-server:        1.ubuntu.pool.ntp.org
timezone:                    America/Los_Angeles
date:                       2020-01-28,22:14:27
hostid:                     184556370
location-id:                6
enable-host-ports:          yes
banner:                      * Welcome to Arista Networks Inc. Netvisor(R).
This is a monitored system.      *
*                ACCESS RESTRICTED TO AUTHORIZED USERS ONLY
    *
* By using the Netvisor(R) CLI, you agree to the terms of the Arista
Networks  *
* End User License Agreement (EULA). The EULA can be accessed via
    *
* http://www.arista.com/eula or by using the command "eula-show"
```

Configuring 802.1X Authentication During Switch Setup

An Arista switch can be connected to an out-of-band network for management purposes using the dedicated management interface.

Starting from NetVisor OS release 6.1.0, it is possible to use the `switch-setup-modify` command to configure the standard IEEE 802.1X authentication (as a supplicant) on the switch management interface. The interface needs to be connected to an authenticator device and cannot be part of a LAG for this feature to work.

An external network device used for out-of-band connectivity may be capable of running the IEEE 802.1X standard as an authenticator. In such cases, for security purposes the network administrator may want to enable the IEEE 802.1X authentication exchange between the switch management interface (as a supplicant) and the external authenticator.

Once the management interface is configured as supplicant and comes up, it sends out a special 802.1X message (*EAPoL Start*) to start the authentication process. If the authentication of the configured credentials is successful, then the interface is *authorized*. Before the interface is authenticated, only 802.1X packets are allowed and all other traffic is dropped on the authenticator device.

> **Note**: In NetVisor OS release 6.1.0 support was added for the *EAP-MD5 authentication* method only. In addition, NetVisor OS supports the 802.1X-2001 version of the standard for interoperability with older authenticators as well as the 802.1X-2004 version.

802.1X can be configured during a fresh switch install or later using the `switch-setup-modify` and `switch-setup-show` commands.

However, the 802.1X configuration requires the creation of a host profile before the feature can be enabled. That cannot be achieved within the `switch-setup-modify` command, so a two-step process is required.

First, a host profile needs to be created like so:

```
CLI (network-admin@switch) > eap-host-profile-create name profile1 mode md5
identity user1 password
password for user identity:
confirm password for user identity:
```

The profile can be created with local (or cluster) scope and then verified with the command:

```
CLI (network-admin@switch) > eap-host-profile-show
switch  name           mode identity    scope
------- ------------- ---- ----------- -----
switch  profile1       md5  user1       local
```

Then 802.1X can be enabled with the `switch-setup-modify` command by specifying the newly created profile plus an additional parameter such as the standard version to use:

```
CLI (network-admin@switch) > switch-setup-modify mgmt-dot1x-enable mgmt-
```

```
dot1x-profiles  profile1  mgmt-dot1x-version  802.1X-2004

CLI  (network-admin@switch)  >  switch-setup-show
switch-name:                 switch
mgmt-dot1x-enable:           true
mgmt-dot1x-version:          802.1X-2004
mgmt-dot1x-profiles:         profile1
mgmt-dot1x-status:           CONNECTING
mgmt-ip:                     10.14.8.90/23
mgmt-ip-assignment:          static
mgmt-ip6:                     fe80::660e:94ff:fe4c:de/64
mgmt-ip6-assignment:         autoconf
mgmt-link-state:             up
mgmt-link-speed:             1g
in-band-ip:                   192.168.3.55/24
in-band-ip6:                  fe80::640e:94ff:fe03:faf8/64
in-band-ip6-assign:          autoconf
gateway-ip:                  10.14.8.1
dns-ip:                      10.135.2.13
dns-secondary-ip:             10.20.4.1
domain-name:                  pluribusnetworks.com
ntp-server:                  10.135.2.13
ntp-secondary-server:         10.20.4.1
ntp-tertiary-server:          2.ubuntu.pool.ntp.org
timezone:                     America/Los_Angeles
date:                        2021-04-09,03:37:57
hostid:                      436207619
location-id:                 1
enable-host-ports:           yes
banner:                       * Welcome to Arista Networks Inc. Netvisor(R).
This is a monitored system.      *
*                ACCESS RESTRICTED TO AUTHORIZED USERS ONLY
    *
* By using the Netvisor(R) CLI, you agree to the terms of the Arista
Networks   *
* End User License Agreement (EULA). The EULA can be accessed via
    *
* http://www.arista.com/eula or by using the command "eula-show"
```

The `mgmt-dot1x-status` line in the output can have one of these values:

- AUTHENTICATED (authentication successful and port authorized)
- CONNECTING (connecting to the authenticator, check periodically for updates)
- UNKNOWN (unknown condition that can occur in corner cases)
- HELD (authentication failed on the authentication backend server)

After connecting, if the credentials are correctly configured both in the backend's database and in the supplicant's profile, the port gets authenticated as shown below:

```
CLI  (network-admin@switch)  >  switch-setup-show
switch-name:                 switch
mgmt-dot1x-enable:           true
```

```
mgmt-dot1x-version:          802.1X-2004
mgmt-dot1x-profiles:         profile1
mgmt-dot1x-status:           AUTHENTICATED
mgmt-ip:                     10.14.8.90/23
mgmt-ip-assignment:          static
mgmt-ip6:                    fe80::660e:94ff:fe4c:de/64
mgmt-ip6-assignment:         autoconf
mgmt-link-state:             up
mgmt-link-speed:             1g
in-band-ip:                  192.168.3.55/24
in-band-ip6:                 fe80::640e:94ff:fe03:faf8/64
in-band-ip6-assign:          autoconf
gateway-ip:                  10.14.8.1
dns-ip:                      10.135.2.13
dns-secondary-ip:            10.20.4.1
domain-name:                 pluribusnetworks.com
<snip>
```

The same two-step process can be used during a fresh switch install like so:

```
Netvisor OS Command Line Interface 6.1
         By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE
READ THE TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA)
AND AGREE TO THEM. [YES | NO | EULA]?: YES or NO?
         By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE
READ THE TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA)
AND AGREE TO THEM. [YES | NO | EULA]?: YES or NO?
         By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE
READ THE TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA)
AND AGREE TO THEM. [YES | NO | EULA]?: YES or NO?
         By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE
READ THE TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA)
AND AGREE TO THEM. [YES | NO | EULA]?:
YES or NO?
         By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE
READ THE TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA)
AND AGREE TO THEM. [YES | NO | EULA]?: YES
Switch setup required:
         Switch Name (switch):
         network-admin Password:
         Re-enter Password:
         Mgmt IP/Netmask (10.14.8.90/23):
         Mgmt IPv6/Netmask:
         In-band IP/Netmask (192.168.3.55/24):
         In-band IPv6/Netmask:
         Gateway IP (10.14.8.1):
         Gateway IPv6:
         Primary DNS IP (10.135.2.13):
         Secondary DNS IP (10.20.4.1):
         Domain name (pluribusnetworks.com):
         Automatically Upload Diagnostics (yes):
         Enable host ports by default (yes):
```

```
Switch Setup:
        Switch Name          : switch
        Mgmt 802.1x cfg      : no
         Mgmt 802.1x profiles :
        Mgmt 802.1x status   :
         Switch Mgmt IP       : 10.14.8.90/23
          Switch Mgmt IPv6     : fe80::660e:94ff:fe4c:de/64
         Switch In-band IP    : 192.168.3.55/24
          Switch In-band IPv6  : fe80::640e:94ff:fe03:faf8/64
        Switch Gateway       : 10.14.8.1
         Switch IPv6 Gateway  : ::
         Switch DNS Server    : 10.135.2.13
         Switch DNS2 Server   : 10.20.4.1
          Switch Domain Name   : pluribusnetworks.com
         Switch NTP Server    : 10.135.2.13
         Switch Timezone      : America/Los_Angeles
         Switch Date          : 2021-04-09,04:00:14
        Enable host ports    : yes
         Analytics Store      : optimized
Fabric required. Please use fabric-create/join/show
Connected to Switch switch; nvOS Identifier:0x1a000003; Ver: 6.1.0-
6010018092

CLI (network-admin@switch) > eap-host-profile-create name profile1 mode md5
identity user1 password
password for user identity:
confirm password for user identity:

CLI (network-admin@switch) > switch-setup-modify mgmt-dot1x-enable mgmt-
dot1x-profiles profile1 mgmt-dot1x-version 802.1X-2004
```

Note: If you try to delete a profile that is in use, you will get an error message:

```
CLI (network-admin@switch) > eap-host-profile-delete name profile1
eap-host-profile-delete: EAP Host profile profile1 is currently used
```

If you need to delete it, you can create and assign another profile first, then you can delete the old one:

```
CLI (network-admin@switch) > switch-setup-modify mgmt-dot1x-profiles
profile2

CLI (network-admin@switch) > eap-host-profile-delete name profile1

CLI (network-admin@switch) > eap-host-profile-show
switch  name          mode identity    scope
------- ------------- ---- ----------- -----
switch  profile2      md5  user1       local

802.1X can be disabled with the following command:

CLI (network-admin@switch) > switch-setup-modify mgmt-dot1x-disable
```

Creating Switch Groups

This feature allows you to create a switch group, and you can create as many switch groups as needed. You provide a name to a group of switches, and a switch can be a member of more than one group.
If a switch is offline when you add it to a group, the configuration fails for that switch. Online switches are added normally.

Switch groups are static and you must manually remove a switch from a group. You cannot use a switch name for the switch group name and a warning message is displayed because the configuration is invalid.

Use the following commands:

```
CLI (network-admin@Spine1) > switch-group-create
```

`name` *`name-string`*	Specify a name for the switch group.
`description` *`description-string`*	Specify a description for the switch group.

To create a switch-group with the name, **`rack-1-row-1`**, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-create name rack-1-row-1
description "datacenter rack 1"

CLI (network-admin@Spine1) > switch-group-delete
```

`name` *`name-string`*	Specify a name for the switch group.
`description` *`description-string`*	Specify a description for the switch group.

To delete a switch-group with the name, **`rack-1-row-1`**, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-delete name rack-1-row-1

CLI (network-admin@Spine1) > switch-group-modify
```

`name` *`name-string`*	Specify a name for the switch group.

To modify a switch-group with the name, **`rack-1-row-1`**, and change the description, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-modify name rack-1-row-1
description datacenter

CLI (network-admin@Spine1) > switch-group-show
```

`name` *`name-string`*	Displays the name of the switch group.

`description` *`description-string`*	Displays a description of the switch group.

To display a switch-group with the name, **`rack-1-row-1`**, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-show name rack-1-row-1

name                  description
------------  -----------
rack-1-row-1  datacenter
```

Adding Switches to Switch-Groups

```
(CLI (network-admin@Spine1) > switch-group-member-add
```

`name` *`name-string`*	Specify the name of the switch group to add the member.
`member` *`fabric-node name`*	Specify the name of the switch to add as a member.

To add switch, Leaf-1, to switch-group, **`rack-1-row-1`**, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-member-add name rack-1-row-1
member Leaf-1

CLI (network-admin@Spine1) > switch-group-member-remove
```

`name` *`name-string`*	Specify the name of the switch group to remove the member.
`member` *`fabric-node name`*	Specify the name of the switch to remove as a member.

To remove switch, Leaf-1, from switch-group, **`rack-1-row-1`**, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-member-remove name rack-1-row-1
member Leaf-1

CLI (network-admin@Spine1) > switch-group-member-show
```

`name` *`name-string`*	Displays the name of the switch group.
`member` *`fabric-node name`*	Displays the name of the switches in a group.

To display switch-group, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-group-member-show
```

```
switch   name             member
-------  ------------  ------
Spine-1  rack-1-row-1  Leaf-1
```

Support for Enabling or Disabling LLDP

This feature provides for a generic LLDP ON/OFF toggle function set at the system level.

Currently, to disable LLDP on a switch you must disable the LLDP configuration on all ports. This resets all related configurations of LLDP protocol setting and LLDP vFlows.

Use the following CLI command to enable and disable the protocol:

```
CLI (network-admin@Leaf1) > system-settings-modify [lldp|no-lldp]
```

LLDP packets are executed on the CPU with the help of LLDP vFlows.

To clear all LLDP protocol system flows use the parameter `no-lldp`.

To add all LLDP protocol system flows use the parameter `lldp`.

This approach ensures port LLDP configurations are not disturbed.

```
CLI (network-admin@Leaf1) > system-settings-show

switch:         Spine1
optimize-arps:  on
lldp:           on
```

Configuring System Settings

You can use the `system-settings-modify` command to configure a host of system settings. These settings correspond to different features and are discussed in detail in the related sections.

```
CLI  (network-admin@leaf1)  >  system-settings-modify
```

`system-settings-modify`	Use this command to modify system settings.
Specify one or more of the following options:	
`optimize-arps|no-optimize-arps`	Enable or disable ARP optimization. This feature is ON by default. When ON, the NetVisor sends all ARP requests and Gratuitous ARP requests to the local CPU of the corresponding switch, and responds to ARP requests on behalf of the intended ARP targets. This feature corresponds to an ARP-proxy function on the corresponding switch. It is assumed that ARP requests and Gratuitous ARP requests are used by NetVisor to learn the MAC address to IPv4 address mapping, as well as Unknown Unicast and Broadcast traffic when the parameters, `manage-unknown-unicast` or `manage-broadcast` are ON. When OFF, the NetVisor learns and bridges ARP requests on the switch ASIC instead of relying on the local CPU.
`lldp|no-lldp`	Enable or disable LLDP at the system level. The Link Layer Discovery Protocol (LLDP) is an open, vendor-independent protocol that advertises a device's identity, abilities, and neighboring devices connected within the Local Area Network.
`policy-based-routing|no-policy-based-routing`	Enable or disable policy-based routing. Policy-Based Routing (PBR) enables flexible packet forwarding and routing through user defined policies.
`optimize-nd|no-optimize-nd`	Enable or disable Neighbor Discovery optimization. This feature is ON by default. When ON, NetVisor sends all Neighbor Solicitations and Unsolicited Neighbor Advertisements to the local CPU of the corresponding switch, responds to Neighbor Solicitation requests on behalf of the intended ND targets themselves. This feature corresponds to an ND-proxy function on the corresponding switch. Neighbor Solicitations and Unsolicited Neighbor Advertisements are used by NetVisor to learn the MAC address to IPv6 address mapping, as well as Unknown Unicast and Broadcast traffic when tthe parameters, `manage-unknown-unicast` or `manage-broadcast` are ON. When OFF, NetVisor OS learns and bridges Neighbor Solicitations on the switch ASIC instead of relying on NetVisor.

`reactivate-mac|no-reactivate-mac`	Enable or disable reactivation of aged out MAC entries. This feature is ON by default. When ON, if a vPort entry is looked up and found as inactive, the corresponding l2-table entry is re-activated on the switch. When OFF, the traffic is flooded on the corresponding broadcast domain by NetVisor OS. Please note that l2-table entries are refreshed on the Ethernet switch ASIC every time it is used, otherwise it is removed after a time of inactivity corresponding to configured aging period. Requires unknown-unicast to be enabled.
`reactivate-vxlan-tunnel-mac|no-reactivate-vxlan-tunnel-mac`	Enable or disable reactivation of MAC entries over VXLAN tunnels.
`manage-unknown-unicast|no-manage-unknown-unicast`	Enable or disable unknown unicast management. This feature is ON by default. NetVisor sends all unknown-unicast traffic to the local CPU of the corresponding switch. NetVisor learns the source of unknown-unicast packets and floods the packets on the corresponding unknown-unicast domain. When OFF, all unknown-unicast traffic is flooded on the corresponding unknown-unicast domain by the switch instead of relying on NetVisor.
`manage-broadcast|no-manage-broadcast`	Enable or disable broadcast management. This feature is ON by default. NetVisor sends all broadcast traffic to the local CPU of the corresponding switch. NetVisor learns the source of broadcast packets and floods the packets on the corresponding broadcast domain. When OFF, all unicast traffic is flooded on the corresponding broadcast domain by the switch instead of relying on NetVisor.
`block-loops|no-block-loops`	Enable or disable loop detection. NetVisor Loop Detection exposes loops to customers using system log messages, `port-show` output, and `vport-show` output. **Note:** This parameter is available only on NSU, NRU-02, NRU-03, and NRU-S0301 platforms.
`auto-trunk|no-auto-trunk`	Enable or disable auto trunking.
`auto-host-bundle|no-auto-host-bundle`	Enable or disable auto host bundling. This feature enables auto trunking of ports between PN switches and ESXi hosts.
`cluster-active-active-routing|no-cluster-active-active-routing`	Enable or disable active-active routing on a cluster. Here, two cluster switches run vRouters with active-active VRRP in order to provide redundant Layer 3 next hops (using virtual IPs) to both upstream and downstream devices.

`fast-route-download|no-fast-route-download`	Enable or disable fast route download from routesnoop.
`fast-interface-lookup|no-fast-interface-lookup`	Enable or disable fast router interface lookup.
`routing-over-vlags|no-routing-over-vlags`	Enable or disable routing over vLAGs. This feature allows packets crossing the cluster link to be routed without being dropped when egressing vLAGs.
`source-mac-miss to-cpu|copy-to-cpu`	Specify either of the options as the unknown source MAC learn behavior.
`optimize-datapath disable|cluster-only|all`	Specify the datapath optimization for cluster, fabric and vRouter communication: • `disable`- disables the datapath optimization. • `cluster-only`- enables datapath optimization for cluster-only, where cluster traffic is redirected to cluster 4094 vNIC. • `all`- enables datapath optimization for fabric and data traffic. The default value is `all`.
`cpu-class-enable|no-cpu-class-enable`	Enable or disable CPU class. This feature enables advanced CPTP which operates over 43 independent queues (from 0 to 42) in order to be able to provide separation and granular control over different types of control plane traffic classes.
`usb-port|no-usb-port`	Enable or disable the USB port on the front of the switch. This is only applicable to switches with ONVL.
`use-igmp-snoop-l2|use-igmp-snoop-l3`	Specify whether L2 or L3 tables are to be used for IGMP snooping.
`vle-tracking-timeout <3..30>`	Set a vLE tracking timeout as a value between 3 and 30s. The default timeout is 3s.
`pfc-buffer-limit` *`pfc-buffer-limit-string`*	Specify the percent of global system buffer space allowed for PFC.
`cosq-weight-auto|no-cosq-weight-auto`	Specify either of the options to enable or disable automatic weight assignment for CoS (Class of Service) queues based on min-guarantee configuration.
`lossless-mode|no-lossless-mode`	Enable or disable lossless mode.
`stagger-queries|no-stagger-queries`	Stagger igmp/mld snooping queries.
`sfp28-port-mode 48x25G| 36x25G+12x10G/1G|24x25G+24x10G/1G`	Configure the sfp28 port mode using one of the three options. **Note**: This parameter is available only on AS7326-56X / F9480-V Platforms.

`host-refresh|no-host-refresh`	Enable or disable refreshing host ARP entries to keep L2 entries active.
`proxy-conn-retry|no-proxy-conn-retry`	Enable or disable proxy connection retry.
`proxy-conn-max-retry 0..10`	Set the maximum number of proxy connection retry attempts as a value between 0 and 10.
`proxy-conn-retry-interval 100..2000`	Set the number of milliseconds to wait between proxy connection retry attempts. This is a value between 100 and 2000.
`nvos-debug-logging|no-nvos-debug-logging`	Logging mode selection (Direct OR viz. nvlog demon)
`manage-l2-uuc-drop|no-manage-l2-uuc-drop`	Enable or disable L2 UUC (Unknown Unicast Drop) towards data port.
`xcvr-link-debug|no-xcvr-link-debug`	Enable or disable system debug to capture link information.
`fastpath-bfd|no-fastpath-bfd`	Enable or disable BFD fastpath. This feature is disabled by default.
`linkscan-interval 10000..1000000`	Specify the linkscan interval as a value between 10000µ s and 1000000µ s. The default value is 150000µ s.
`linkscan-mode software|hardware`	Specify the linkscan mode as hardware or software. Software linkscan mode is enabled by default.
`single-pass-flood|no-single-pass-flood`	Enable or disable single-pass flood.

For example, enable Policy-based routing by using the command:

```
CLI (network-switch) > system-settings-modify policy-based-routing
```

Use the system-settings-show command to display the configuration:

```
CLI (network-switch) > system-settings-show
switch:                                 eri-spine1
optimize-arps:                          off
lldp:                                   on
policy-based-routing:                   on
optimize-nd:                            off
reactivate-mac:                         on
reactivate-vxlan-tunnel-mac:            on
manage-unknown-unicast:                 off
manage-broadcast:                       off
auto-trunk:                             on
auto-host-bundle:                       off
cluster-active-active-routing:          on
routing-over-vlags:                     off
```

```
source-mac-miss:                          copy-to-cpu
optimize-datapath:                        all
cpu-class-enable:                         on
usb-port:                                 on
igmp-snoop:                               use-l3
vle-tracking-timeout:                     3
pfc-buffer-limit:                         40%
cosq-weight-auto:                         off
lossless-mode:                            off
snoop-query-stagger:                       no-stagger-queries
host-refresh:                             off
proxy-conn-retry:                         on
proxy-conn-max-retry:                     3
proxy-conn-retry-interval:                500
optimize-rxlos:                           off
xcvr-link-debug:                          disable
fastpath-bfd:                             off
linkscan-interval:                        150000
linkscan-mode:                            software
single-pass-l2-known-multicast:           off
single-pass-flood:                        off
batch-move-mac-hw-group-for-vlan-only:    off
memory-tracker:                           on
symmetric-hash:                           off
hash-suppress-unidir-fields:              off
prioritize-rx-reasons:                    off
```

About GREP Support with NetVisor OS

NetVisor OS supports filtering output and allows switch administrators to filter output "`grep|`" from the CLI. This functionality is limited to the following commands:

- `running-config-show`
- `tech-support-show`
- `help`

Installing NetVisor OS and Initial Configuration

This chapter contains information about initial configuration of your switch as well as commands to manage, upgrade, and restore configurations on a NetVisor OS switch.

Changes to the End User License Agreement (EULA)

Currently, the NetVisor OS EULA is displayed when the switch is setup.

```
Netvisor OS Command Line Interface 6.0.0
By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE READ THE
TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA) AND AGREE
TO THEM. [YES | NO | EULA]?: yes
```

If you enter the EULA option, the output displays the complete EULA text. After this action, it is not possible to confirm EULA acceptance again. In some cases, an integrator may have accepted the EULA on behalf of the actual end user.

A new command is now available to display the EULA acceptance with a timestamp of the event:

```
CLI (network-admin@pn-sw-01) > eula-show
End User License Agreement
Pluribus Networks, Inc.'s ("Pluribus", "we", or "us") software products are
designed to provide fabric networking and analytics solutions that simplify
operations, reduce operating expenses, and introduce applications online
more rapidly. Before you download and/or use any
 of our software, whether alone or as loaded on a piece of equipment, you
will need to agree to the terms of this End User License Agreement (this
"Agreement").
...
PN EULA v. 2.1

eula-show: No fabric
eula-show: Fabric required. Please use fabric-create/join/show
CLI (network-admin@pn-sw-01) >
```

Adding License Keys to NetVisor OS

Note: You can use this feature only on switches that are completely provisioned and where the EULA setup is also completed.

Caution: Do not attempt to use this feature on a newly added switch where provisioning of the switch is not done and when the switch is in fresh-install mode or new-install mode.

This feature is applicable when you want to renew an expired license or add additional licenses to an existing license on a switch that was provisioned previously.

NetVisor OS binds the license key to the serial number of the switch and when downloading or upgrading the NetVisor OS software, the Pluribus (Arista) Networks Cloud locates the serial number.

To install the license key, use the following syntax:

```
CLI (network-admin@switch) > software-license-install key <key-string>
```

The license key has the format of four words separated by commas. For example,

```
License Key: rental,deer,sonic,solace
```

Once the license key is installed, you can display information about the key using the following command:

```
CLI (network-admin@switch) > software-license-show

switch: T6001-ON
license-id: NVOS-CLD-LIC-60D
description: Pluribus Open Netvisor OS Linux Cloud Edition License
expires-on: never
status: VALID
```

Using the Serial Console Port for Initial Configuration

This procedure assumes that you have installed the switch in the desired location and it is powered on.

Warning: Do not connect any ports to the network until the switch is configured. You can accidentally create loops or cause IP address conflicts on the network.

If you are going to cable host computers to the switch, there is an option to enable or disable host ports by default.

1. Connect the console port on the rear or front (depending on the model) of the switch to your laptop or terminal concentrator using a serial cable.
2. From the terminal emulator application on your computer, log into the switch with the username **network-admin** and the default password **admin**.

Note: NetVisor OS supports both IPv4 and IPv6 addresses for the in-band interface.

Warning: Be sure to type in a static IP address for the management interface during the initial configuration. NetVisor OS initially uses DHCP to obtain an IP address, but DHCP is not supported after the initial configuration.

3. Begin the initial configuration using the initialization procedure displayed.
4. Enter the following details when prompted, an example is provided in the output below:
    - Accept the EULA agreement
    - Type-in the switch name. An example is provided in the output below.
    - Enter and re-enter the password
    - Enter the Management IP and netmask. An example is provided in the output below.
    - Enter the In-band IP and netmask. An example is provided in the output below.
    - Enter the IP address of the Gateway.
    - Enter the IP address for the primary and secondary DNS.
    - Enter the domain name.

```
switch console login: network-admin
Password: admin

root@switch-new:~# cli

Netvisor OS Command Line Interface 6.0
By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE READ THE
TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA) AND AGREE
TO THEM. [YES | NO | EULA]?: yes
Switch setup required:
Switch Name (switch-new):pn-switch-01
```

```
network-admin Password:password <return>
Re-enter Password:password <return>
Mgmt IP/Netmask (10.13.27.218/23):
Mgmt IPv6/Netmask:
In-band IP/Netmask (192.168.10.8/24):
In-band IPv6/Netmask:
Gateway IP (10.13.26.1):
Gateway IPv6:
Primary DNS IP (10.135.2.13):
Secondary DNS IP (10.20.4.1):
Domain name (pluribusnetworks.com):
Automatically Upload Diagnostics (yes):
Enable host ports by default (yes):

Switch Setup:
Switch Name           : pn-switch-01
Switch Mgmt IP        : 10.13.27.218/23
Switch Mgmt IPv6      : fe80::aa2b:b5ff:febc:c78e/64
Switch In-band IP     : 192.168.10.8/24
Switch In-band IPv6   : fe80::640e:94ff:feba:f719/64
Switch Gateway        : 10.13.26.1
Switch IPv6 Gateway   : ::
Switch DNS Server     : 10.135.2.13
Switch DNS2 Server    : 10.20.4.1
Switch Domain Name    : pluribusnetworks.com
Switch NTP Server     : 0.ubuntu.pool.ntp.org
Switch Timezone       : America/Los_Angeles
Switch Date           : 2020-07-20,19:48:20
Enable host ports     : yes
Analytics Store       : optimized
Fabric required. Please use fabric-create/join/show
Connected to Switch ghspine01-new; nvOS Identifier:0xb001dba; Ver: 6.0.0-
6000016292
CLI (network-admin@switch-new) >
```

When you setup a switch for initial configuration, the host facing ports are enabled by default. However, you can disable the host ports until you are ready to plug-in host cables to the switch. If NetVisor OS does not detect adjacency on a port during the `quickstart` procedure, the ports remain in the disabled state.

To enable the ports after plugging in cables, use the `port-config-modify port port-list host-enable` command. NetVisor OS enables host ports by default unless you specify NO during the `quickstart` procedure as displayed below.

```
Netvisor OS Command Line Interface 6.0.0
By ANSWERING "YES" TO THIS PROMPT YOU ACKNOWLEDGE THAT YOU HAVE READ THE
TERMS OF THE PLURIBUS NETWORKS END USER LICENSE AGREEMENT (EULA) AND AGREE
TO THEM. [YES | NO | EULA]?: yes
Switch setup required:
Switch Name (netvisor): pn-switch-01
network-admin Password: password <return>
Re-enter Password: ******** <return>
Mgmt IP/Netmask (dhcp): 10.14.2.42/23
Mgmt IPv6/Netmask:
```

```
In-band IP/Netmask: 12.1.165.21/24
In-band IPv6/Netmask:
Loopback IP:
Loopback IPv6:
Gateway IP (10.14.2.1):
Gateway IPv6:
Primary DNS IP: 10.135.2.13
Secondary DNS IP: 10.20.4.1
Domain name: pluribusnetworks.com
Automatically Upload Diagnostics (yes):
Enable host ports by default (yes): no
```

To verify, use the command:

```
CLI (network-admin@pn-switch-01) > port-show port 9,10

switch        port  bezel-port   status              config
------------  ----  ---------- --------------------   ------
pn-switch-01   9      3        phy-up,host-disabled    10g
pn-switch-01   10     3.2      phy-up,host-disabled    10g
```

To enable the port (s), use the command:

```
CLI (network-admin@pn-switch-01) > port-config-modify port 9,10 enable
host-enable

CLI (network-admin@pn-switch-01) > port-show port 9,10

switch        port  bezel-port  status     config
------------  ---- ----------  ----------  ------
pn-switch-01   9     3          up,vlan-up  fd,10g
pn-switch-01   10    3.2        up,vlan-up  fd,10g
```

You cannot change (enable or disable) the host-ports by using the switch setup process after the initial configuration is done. If you try to modify the host-ports, NetVisor OS displays an error as displayed in the example here:

```
CLI (network-admin@pn-switch-01) > switch-setup-modify disable-host-ports

switch-setup-modify: disable/enable host ports can be set only at initial
switch-setup time
```

During the initial configuration of the switch, if the host ports are disabled, then all ports having the same port configuration will be disabled. This can be viewed using the following command:

```
<CLI (network-admin@pn-switch-01) > port-config-show port port-list host-
enable
```

In this mode, when any port comes up physically, NetVisor OS automatically sends and receives LLDP packets to look for peer switches. If NetVisor OS does not detect an adjacency within 5 seconds, the port is flagged as `host-disabled`. With this flag set, NetVisor OS only accepts LLDP packets and does not initiate packet transmission.

```
CLI (network-admin@pn-switch-01) > port-config-show port 9,10
```

```
switch        port  bezel-port       status             config
------------  ----  ----------  ----------------------  ------
pn-switch-01   9     3          up,vlan-up,PN-other,LLDP  fd,10g
pn-switch-01   10    3.2        up,vlan-up                fd,10g
```

After completing switch discovery and fabric creation, use the `host-enable` option to enable host, server, or router traffic switching, and ports:

```
CLI (network-admin@pn-switch-01) > port-config-modify port 9 host-enable
```

Managing NetVisor OS Certificates

Arista Networks includes the NetVisor OS certificates along with the switches during shipment and you can access the certificates from /var/nvos/certs directory. These certificates are necessary for communication between switches in a fabric and hinders the transactions between fabric members if the certificate expires. You can view the validity (dates valid from and dates valid until) for NetVisor OS certificate using the `switch-info-show` command.

When you configure the alarm, the certificate is checked every 24 hours and an alarm is issued if the number of days of expiry is equal to or less than 30 days . The certificate expiry alert is enabled by default for 30 days, but can configured between 7 days through 180 days on NetVisor OS. You can disable this feature using the `cert-expiration-alert-modify no-netvisor` command.

You can view the certificate expiration alert or alarm configuration by using the `cert-expiration-alert-show` command and can schedule an alert notification before the certificate expires. You can view the alarm or alert notification in the `event.log` file and also by running the `log-alert-show command.` You can also configure a new `SNMP trap` for certificate expiry on the SNMP services.

Alarm is an event in the *event log*, an alert in `log-alert-show` command and a new SNMP trap if the trap server is configured. Frequency of alarm will be every 24 hours until the certificate has expired.

To configure the certificate expiry alert, use the command:

```
CLI (network-admin@switch01) > cert-expiration-alert-modify
```

Specify one or more of the following options:	
`netvisor|no-netvisor`	Specify whether to enable or disable NetVisor OS certificate expiration alerts.
`days-before-expiration 7..180`	Modify the number of days before expiration to send alerts (Default 30 days). The value ranges from 7 through 180 days.

To view the alert configuration for the certificate expiry, use the command:

```
CLI (network-admin@switch01) > cert-expiration-alert-show

switch:                          switch01
days-before-expiration(d):       30
```

To enable or disable the SNMP trap for certificate expiry alert, use the command:

```
CLI (network-admin@switch01) > snmp-trap-enable-modify cert-expiry|no-cert-expiry
```

where,

`cert-expiry|no-cert-expiry`	Specify whether to monitor certificate expiry or not.

To view the alert configuration details older than an hour, use the command:

```
CLI (network-admin@switch01) > log-alert-show older-than 1h

time      switch     code   name                count    last-message
--------- ---------- -----  ------------------  -----  ------------------------------
00:17:05  switch01   31008  smf_nvOSd_stop        1     SMF Service stopping nvOSd
00:17:08  switch01   11008  nvOSd_start           1     version 5.1.5010014665
00:35:49  switch01   31016  certificate_expiry    1     switch cert expiring in 19 days
```

The `switch-info-show` command displays the validity (dates valid from and dates valid until) for NetVisor OS certificate. For example,

```
CLI (network-admin@nru03-sw-1*) > switch-info-show

model:                    NRU03
chassis-serial:           1937ST9100075
cpu1-type:                Intel(R) Xeon(R) CPU D-1557 @ 1.50GHz
cpu2-type:                Intel(R) Xeon(R) CPU D-1557 @ 1.50GHz
cpu3-type:                Intel(R) Xeon(R) CPU D-1557 @ 1.50GHz
cpu4-type:                Intel(R) Xeon(R) CPU D-1557 @ 1.50GHz
system-mem:               30.6G
switch-device:            OK
fan1-status:              OK
fan2-status:              OK
fan3-status:              OK
fan4-status:              OK
fan5-status:              OK
fan6-status:              OK
fan7-status:              OK
fan8-status:              OK
fan9-status:              OK
fan10-status:             OK
fan11-status:             OK
fan12-status:             OK
ps1-status:               OK
ps2-status:               OK
disk-model:               Micron_1300_MTFDDAV256TDL
disk-firmware:            M5MU000
disk-size:                238G
disk-type:                Solid State Disk, TRIM Supported
bios-vendor:              American Megatrends Inc.
bios-version:             1.00.00
netvisor-cert-valid-from: Sep 13 07:00:00 2019 GMT
netvisor-cert-valid-till:  Sep 14 06:59:59 2039 GMT
```

Viewing the Validity of NetVisor OS Certificates

Starting with NetVisor OS version 6.1.1, you can view the validity of NetVisor OS certificates on your switch. That is, after you upgrade to version 6.1.1, you can check the validity of certificates available on NetVisor OS using the `cert-switch-show` command. You can also REST API command to verify the validity of certificates.

The command output also gets logged into `tech-support-show` command output and can be helpful during a customer support (TAC) call.

To check the validity of certificates on a switch, use the command:

```
CLI (network-admin@switch) > cert-switch-show

name:                  /usr/nvos/cert/core.pem
cert-type:             server
subject:                /C=US/ST=California/O=Pluribus Networks/OU=Core
Encryption/CN=Core  Encryption
issuer:                /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus  Update  Root  CA
serial-number:         291
valid-from:            Mar  9 10:19:06 2021 GMT
valid-to:              Mar  4 10:19:06 2041 GMT
status:                Certificate valid for 7174 days
----------------------------------------
name:                  /usr/nvos/cert/cacert.pem
cert-type:             ca
subject:               /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus  Update  Root  CA
issuer:                /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus  Update  Root  CA
serial-number:         17400531076958470354
valid-from:            Apr  9 11:00:34 2021 GMT
valid-to:              Apr  4 11:00:34 2041 GMT
status:                Certificate valid for 7205 days
----------------------------------------
name:                  /usr/nvos/cert/pkg.pem
cert-type:             server
subject:               /C=US/ST=California/O=Pluribus
Networks/OU=Package/CN=Package
issuer:                /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus  Update  Root  CA
serial-number:         31
valid-from:            Oct 15 19:24:41 2012 GMT
valid-to:              Oct 13 19:24:41 2022 GMT
status:                Certificate valid for 458 days
----------------------------------------
name:                  /usr/nvos/cert/newswitch.pem
cert-type:             server
subject:                /C=US/ST=California/O=Pluribus Networks/OU=New
Switch/CN=New  Pluribus  Switch
```

```
issuer:                 /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus  Update  Root  CA
serial-number:          290
valid-from:             Mar  9 10:18:37 2021 GMT
valid-to:               Mar  4 10:18:37 2041 GMT
status:                 Certificate valid for 7174 days
----------------------------------------
name:                   /var/nvos/cert/switch.pem
cert-type:              server
subject:                /C=US/ST=California/O=Pluribus
Networks/OU=Switch/CN=hostid:184560120   serial:2126PN8500255
issuer:                 /C=US/ST=California/O=Pluribus Networks/CN=Pluribus
Update
Server  CA
serial-number:          12343
valid-from:             Jun 21 07:00:00 2021 GMT
valid-to:               Jun 22 06:59:59 2041 GMT
status:                 Certificate valid for 7284 days
----------------------------------------
name:                   /var/nvos/cert/cacerts.pem
cert-type:              ca
subject:                /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus  Update  Root  CA
issuer:                 /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus  Update  Root  CA
serial-number:          17400531076958470354
valid-from:             Apr  9 11:00:34 2021 GMT
valid-to:               Apr  4 11:00:34 2041 GMT
status:                 Certificate valid for 7205 days
----------------------------------------
name:                   /var/nvos/cert/cacerts.pem
cert-type:              ca
subject:                /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=Pluribus  Update  DNI  CA
issuer:                 /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus  Update  Root  CA
serial-number:          295
valid-from:             Mar  9 14:34:32 2021 GMT
valid-to:               Mar  4 14:34:32 2041 GMT
status:                 Certificate valid for 7174 days
----------------------------------------
name:                   /var/nvos/cert/cacerts.pem
cert-type:              ca
subject:                /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=Pluribus  Update  Celestica  CA
issuer:                 /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus  Update  Root  CA
serial-number:          296
valid-from:             Mar  9 14:40:07 2021 GMT
valid-to:               Mar  4 14:40:07 2041 GMT
status:                 Certificate valid for 7174 days
----------------------------------------
name:                   /var/nvos/cert/cacerts.pem
```

```
cert-type:           ca
subject:               /C=US/ST=California/O=Pluribus Networks/CN=Pluribus
Update
Leadman CA
issuer:               /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
serial-number:       298
valid-from:          Mar  9 14:45:58 2021 GMT
valid-to:            Mar  4 14:45:58 2041 GMT
status:               Certificate valid for 7174 days
----------------------------------------
name:                 /var/nvos/cert/cacerts.pem
cert-type:           ca
subject:               /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=cb-update.pluribusnetworks.com
issuer:               /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
serial-number:       297
valid-from:          Mar  9 14:43:55 2021 GMT
valid-to:            Mar  4 14:43:55 2041 GMT
status:               Certificate valid for 7174 days
----------------------------------------
name:                 /var/nvos/cert/cacerts.pem
cert-type:           ca
subject:               /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=celestica-update2.pluribusnetworks.com
issuer:               /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
serial-number:       299
valid-from:          Mar  9 14:47:48 2021 GMT
valid-to:            Mar  4 14:47:48 2041 GMT
status:               Certificate valid for 7174 days
----------------------------------------
name:                 /var/nvos/cert/cacerts.pem
cert-type:           ca
subject:               /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=celestica-update.pluribusnetworks.com
issuer:               /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
serial-number:       300
valid-from:          Mar  9 14:48:29 2021 GMT
valid-to:            Mar  4 14:48:29 2041 GMT
status:               Certificate valid for 7174 days
----------------------------------------
name:                 /var/nvos/cert/cacerts.pem
cert-type:           ca
subject:               /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=ct-update2.pluribusnetworks.com
issuer:               /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
serial-number:       301
valid-from:          Mar  9 14:49:17 2021 GMT
valid-to:            Mar  4 14:49:17 2041 GMT
```

```
status:                Certificate valid for 7174 days
----------------------------------------
name:                  /var/nvos/cert/cacerts.pem
cert-type:             ca
subject:                /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=ct-update.pluribusnetworks.com
issuer:                /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
serial-number:         302
valid-from:            Mar  9 14:49:56 2021 GMT
valid-to:              Mar  4 14:49:56 2041 GMT
status:                Certificate valid for 7174 days
----------------------------------------
name:                  /var/nvos/cert/cacerts.pem
cert-type:             ca
subject:                /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=staging-update.pluribusnetworks.com
issuer:                /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
serial-number:         303
valid-from:            Mar  9 14:50:26 2021 GMT
valid-to:              Mar  4 14:50:26 2041 GMT
status:                Certificate valid for 7174 days
----------------------------------------
name:                  /var/nvos/cert/cacerts.pem
cert-type:             ca
subject:                /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=Pluribus Update Manufacturing CA
issuer:                /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
serial-number:         110
valid-from:            Nov  4 01:08:14 2014 GMT
valid-to:              Nov  1 01:08:14 2024 GMT
status:                Certificate valid for 1207 days
----------------------------------------
name:                  /var/nvos/cert/cacerts.pem
cert-type:             ca
subject:                /C=US/ST=California/O=Pluribus Networks/CN=Pluribus
Update
Server CA
issuer:                /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
serial-number:         304
valid-from:            Mar  9 14:51:10 2021 GMT
valid-to:              Mar  4 14:51:10 2041 GMT
status:                Certificate valid for 7174 days
----------------------------------------
name:                  /var/nvos/cert/cacerts.pem
cert-type:             ca
subject:                /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=Pluribus Update Advantech CA
issuer:                /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
```

```
serial-number:        344
valid-from:           Apr 21 12:02:10 2021 GMT
valid-to:             Apr 16 12:02:10 2041 GMT
status:               Certificate valid for 7217 days
----------------------------------------
name:                 /var/nvos/cert/cacerts.pem
cert-type:            ca
subject:              /C=US/ST=California/O=Pluribus Networks/OU=Update
Servers/CN=Pluribus Update staging CA
issuer:               /O=Pluribus Networks/L=Palo
Alto/ST=California/C=US/CN=Pluribus Update Root CA
serial-number:        101
valid-from:           Jun  2 22:30:14 2014 GMT
valid-to:             May 30 22:30:14 2024 GMT
status:               Certificate valid for 1053 days
```

To view the certificate expiration alert, use the command:

```
CLI (network-admin@switch) > cert-expiration-alert-show
switch:                    sff-pvt-ptf
netvisor:                  yes
days-before-expiration(d): 180
```

Enabling Administrative Services

There are many features of the Arista Networks fabric that require or can be enhanced using remote access. For example, when packets are written to a log file, you may want to transfer that file from a switch to a different system for analysis. Also, if you are creating a NetVM environment, an IOS image of the guest OS must be loaded on the switch.

You can enhance or modify several services such as SSH, NFS, Web, SNMP, SFTP.

To check the status of various services, use the following command:

```
CLI  (network-admin@Leaf-1)  >  admin-service-show

switch:             Leaf-1
if:                 mgmt
ssh:                on
nfs:                on
web:                on
web-ssl:            off
web-ssl-port:       443
web-port:           80
web-log:            off
snmp:               on
net-api:            on
icmp:               on
switch:             Leaf-1
if:                 data
ssh:                on
nfs:                on
web:                on
web-ssl:            off
web-ssl-port:       443
web-port:           80
web-log:            off
snmp:               on
net-api:            on
icmp:               on
```

To modify administrative services, use the command:

```
CLI  (network-admin@Leaf-1)  >  admin-service-modify
```

`admin-service-modify`	Modifies services on the switch.
`if` *`if-string`*	Specify the administrative service interface. The options are `mgmt` or `data`.
`ssh|no-ssh`	Specify if you want to enable or disable SSH.
`web|no-web`	Specify if you want to enable web management. Use this option to enable REST API access over HTTP.

`web-ssl|no-web-ssl`	Specify if you want to use SSL and certificates for web services. Use this option to enable REST API access over HTTPS.
`web-ssl-port` *`web-ssl-port-number`*	Specify the web SSL port.
`web-port` *`web-port-number`*	Specify the port for web management.
`web-log|no-web-log`	Specify if you want to turn web logging on or off. **Note:** This option is for use in debugging with Arista support's guidance.
`vrrp|no-vrrp`	Specify if you want to enable or disable VRRP.
`snmp|no-snmp`	Specify if you want to enable or disable SNMP.
`net-api|no-net-api`	Specify if you want to enable or disable NetVisor API.
`icmp|no-icmp`	Specify if you want to enable or disable Internet Message Control Protocol (ICMP).

NetVisor OS supports the file transfer method, SFTP and SFTP is enabled by default on NetVisor OS. Because SFTP relies on Secure Shell (SSH), you must enable SSH before enabling SFTP.

To enable SSH, use the following command

```
CLI (network-admin@Leaf1) > admin-service-modify nic mgmt ssh
```

To enable SFTP, use the following command:

```
CLI (network-admin@Leaf1) > admin-sftp-modify enable

sftp password: <password>
confirm sftp password: <password>
```

The default SFTP username is sftp and the password can be changed using the `admin-sftp-modify` command:

```
CLI (network-admin@Leaf1) > admin-sftp-modify

sftp password: <password>
confirm sftp password: <password>
```

To display the details, use the following commands:

```
CLI (network-admin@Leaf-1) > admin-service-show

switch if   ssh nfs web web-ssl web-ssl-port web-port snmp net-api icmp
------ ---- --- --- --- ------- ------------ -------- ---- ------- ----
Leaf-1 mgmt on  off off off     443          80       on   off     on
Leaf-1 data on  off off off     443          80       on   off     on

admin-service-show: Fabric required. Please use fabric-create/join/show
```

```
CLI (network-admin@Leaf1) > admin-sftp-show

switch:     Leaf1
sftp-user:  sftp
enable:     yes
```

Use SFTP from a host to the switch, and login with the username **sftp** and the password configured for SFTP. Then you can download the available files or upload files to the switch.

```
CLI (network-admin@Leaf1) > admin-service-show

switch nic  ssh nfs web web-port snmp net-api icmp
------ ---- --- --- --- -------- ---- ------- ----
Leaf1  mgmt on  off on  80       off  on      on
```

Configuring Administrative Session Timeout

NetVisor OS sets the administrator sessions to timeout after 60 minutes (by default) of idle time or no activity, but allows you to change the timeout value to a user desirable time. During the session timeout, you are logged out of the CLI and the Shell prompt and your privileges changes to *root user*. To access the switch, you must login again using the CLI or Shell prompt.

To verify the default or user configured session timeout value, use the command:

```
CLI (network-admin@Spine1) > admin-session-timeout-show

switch: Spine1
timeout: 1h
```

To modify the timeout value, use the command:

```
CLI (network-admin@Spine1) > admin-session-timeout-modify
```

`timeout duration: #d#h#m#s`	Specify the maximum time to wait for user inactivity before terminating login session.

Confirming Connectivity on the Network

After connecting your switch, take the time to ensure connectivity by pinging an external IP address (supports both IPv4 and IPv6), and pinging a domain to ensure domain name resolution.

To ping the external network from the switch, use the `ping` command:

```
CLI (network-admin@switch) > ping 2010::2

PING 2010::2(2010::2) 56 data bytes
64 bytes from 2010::2: icmp_seq=1 ttl=64 time=1.69 ms
64 bytes from 2010::2: icmp_seq=2 ttl=64 time=0.412 ms
64 bytes from 2010::2: icmp_seq=3 ttl=64 time=0.434 ms
64 bytes from 2010::2: icmp_seq=4 ttl=64 time=0.418 ms
```

To ping an IP address from the switch, use the ping command:

```
CLI (network-admin@switch) > ping 98.138.253.109 : 56 data bytes

PING 98.138.253.109 (98.138.253.109) 56(84) bytes of data.
64 bytes from 98.138.253.109: icmp_seq=1 ttl=47 time=51.8 ms
64 bytes from 98.138.253.109: icmp_seq=2 ttl=47 time=51.9 ms
64 bytes from 98.138.253.109: icmp_seq=3 ttl=47 time=53.6 ms
```

```
To ping a domain, use the ping command again:

CLI (network-admin@Leaf1) > ping yahoo.com

PING yahoo.com (98.138.253.109) 56(84) bytes of data.
64 bytes from ir1.fp.vip.ne1.yahoo.com (98.138.253.109): icmp_seq=1 ttl=47 time=52.2 ms
64 bytes from ir1.fp.vip.ne1.yahoo.com (98.138.253.109): icmp_seq=2 ttl=47 time=52.5 ms
64 bytes from ir1.fp.vip.ne1.yahoo.com (98.138.253.109): icmp_seq=3 ttl=47 time=51.9 ms
64 bytes from ir1.fp.vip.ne1.yahoo.com (98.138.253.109): icmp_seq=4 ttl=47 time=51.8 ms
```

Setting the Date and Time

You can set the date and time on a switch by modifying the switch configuration using the `switch-setup-modify` command. For example, to change the date and time to September 24, 2019, 09:30:00, use the following command syntax:

```
CLI (network-admin@Leaf1) > switch-setup-modify date 2019-09-24T09:30:00
```

To display the configured setting, use the `switch-setup-show` command:

```
CLI (network-admin@Leaf2) > switch-setup-show
switch-name:                Leaf2
mgmt-ip:                    10.14.30.18/23
mgmt-ip-assignment:         static
mgmt-ip6:                   2721::3617:ebff:fef7:94c4/64
mgmt-ip6-assignment:        autoconf
mgmt-link-state:            up
mgmt-link-speed:            1g
in-band-ip:                 192.168.101.7/24
in-band-ip6:                fe80::640e:94ff:fe83:cefa/64
in-band-ip6-assign:         autoconf
gateway-ip:                 10.14.30.1
dns-ip:                     10.20.4.1
dns-secondary-ip:           172.16.1.4
domain-name:                pluribusnetworks.com
ntp-server:                 0.us.pool.ntp.org
ntp-secondary-server:       0.ubuntu.pool.ntp.org
timezone:                   America/Los_Angeles
date:                       2019-09-24,09:30:00
hostid:                     184555395
location-id:                7
enable-host-ports:          yes
banner:                     * Welcome to Arista Networks Inc. Netvisor(R). This is a monitored
system.     *
*                ACCESS RESTRICTED TO AUTHORIZED USERS ONLY                    *
* By using the Netvisor(R) CLI, you agree to the terms of the Arista Networks  *
* End User License Agreement (EULA). The EULA can be accessed via              *
* http://www.arista.com/eula or by using the command "eula-show"
```

Changing the Default Timezone

By default, NetVisor sets the default timezone to US/Pacific Standard Time (PST). To change the timezone, use the switch-setup-modify command:

```
CLI (network-admin@Leaf1) > switch-setup-modify timezone time-zone name
```

Viewing User Sessions on a Switch

NetVisor OS enables you to view the user sessions on a specified switch and displays all currently logged-in users along with the IP address of the user and login time when you run the command, `mgmt-session-show`. This information is useful for troubleshooting purposes or while dealing on issues with Arista Customer Support teams.

```
CLI (network-admin@Leaf1) > mgmt-session-show
```

Specify any of the following:

`user` *`user-string`*	Displays the user name.
`cli-user` *`cli-user-string`*	Displays the name used to log into the switch.
`pid` *`pid-number`*	Displays the process ID.
`terminal` *`terminal-string`*	Displays the terminal ID
`from-ip` *`ip-address`*	Displays the IP address of the user.
`login-time date/time:` *`yyyy-mm-ddTHH:mm:ss`*	Displays the time and date that the user logged into the switch.
`remote-node` *`remote-node-string`*	Displays the name of the remote node.
`vnet` *`vnet-string`*	Displays the vNET assigned to the user.
`type cli|api|shell`	Displays the type of login session.

For example,

```
CLI (network-admin@Leaf1) > mgmt-session-show

user   cli-user       pid    terminal  from-ip      login-time     type
----   -------        ---    -------  --------      ---------      ----
admin  network-admin  13805  pts/3    10.60.1.216   11:20:52         cli
root   network-admin  8589   pts/2    10.14.20.109  11-15,17:16:17   cli
       network-admin                                08:24:10         cli
root                 19139   pts/1    10.14.22.54   11-15,11:01:08   shell
```

In this example, the `root` user represents the user who has all access by default, while the admin user has only customized access privileges.

Archiving NetVisor OS Log Files Outside the Switch

NetVisor OS enables you to archive the nvOSd log files to an external file server periodically and these log files may be helpful for troubleshooting purposes. As a network administrator, you can configure the following parameters to enable archiving of the log files:

- Server IP address and hostname
- Username and password
- Log archival interval (minimum interval is 30 minutes and the default value is 24 hours)

On configuring this feature, the log archival configuration parameters are saved in the *log_archival_config.xml* file with an encrypted password string. A binary file deciphers the configuration parameters and also the files that are to be archived. NetVisor OS sends an empty *time-stamped directory* to the configured remote server path and subsequently, all the log files are archived to the newly created remote directory. The new directory in the remote server is created in the *nvOS_archive.yyyymmdd_hh.mm.ss* format.

NetVisor OS uses the Secure Copy Protocol (SCP) to archive the log files from the switch to the remote external server at specific intervals. Using the `enable` or `disable` parameter in the CLI command, you can start or stop archiving of the log files. You can archive regular log files, a set pattern of log files, or a whole directory from one of the following paths only. If you add files from other paths than the directories specified here, NetVisor OS displays an error.

- /var/nvOS/log/*
- /nvOS/log/*
- /var/log/*

Use the below CLI commands to configure the log archival parameters and schedule the archival interval.

To modify the archival schedule parameters for the log files, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-modify
```

`enable|disable`	Specify to enable or disable the log archival schedule.
`Specify one or many of the following options:`	
`archive-server-username` *`<string>`*	Specify the SCP username of log archival server.
`archive-server` *`<string>`*	Specify the IP address or hostname of the log archival server.
`archive-server-path` *`<string>`*	Specify the SCP server path to archive the log files in.
`archive-interval` *`<30..4294967295>`*	Specify the log archival interval in minutes. The range varies from 30 minutes to 4294967295 minutes with a default value of 1440 minutes (one day).
`archive-server-password` *`<string>`*	Enter the SCP server password.

For example, if you had modified the log-archival-schedule by specifying the archive-server-username as `pn-user`, archive-server as `pn-server`, and `archive-server-path` as `/home/pn-server/workspace/log_archival_tests`, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-modify archive-server-username pn-user archive-server pn-server archive-server-path /home/pn-server/workspace/log_archival_tests
```

To display the modified configuration, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-show

switch:                  switch-1
archive-server-username:  pn-user
archive-server:           pn-server
archive-server-path:       /home/pn-server/workspace/log_archival_tests
enable:                  no
archive-interval(m):     1440
```

To add the log files to the archival list, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-files-add log-file log-file-string
```

`log-file` *`log-file-string`*	Specify a comma-separated list of log file names to add to the archive list.

For example, to add the `nvOSd.log` or `audit.log` or all *log* files or a whole directory, use the following commands:

```
CLI (network-admin@switch-1) > log-archival-schedule-files-add log-file /var/nvOS/log/nvOSd.log

CLI (network-admin@switch-1) > log-archival-schedule-files-add log-file /var/nvOS/log/*.log

CLI (network-admin@switch-1) > log-archival-schedule-files-add log-file /var/log

CLI (network-admin@switch-1) > log-archival-schedule-files-add log-file /nvOS/log/audit.log
```

To view the list of log files that you had scheduled to be archived, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-files-show

/var/nvOS/log/nvOSd.log
/var/nvOS/log/*.log
/var/log
/nvOS/log/audit.log
```

If you try to add an unsupported file or directory, an error message is displayed. For example,

```
CLI (network-admin@switch-1) > log-archival-schedule-files-add log-
file /var/nvOS

/var/nvOS, not from valid logs supported
```

To remove the log files or a list of log files from the archival list, use the command,

```
CLI (network-admin@switch-1) > log-archival-schedule-files-remove log-file
log-file-string
```

Guidelines and Tips

- When the `log-archival-schedule` is enabled and if all files are removed from the archival list, the log-archival-schedule gets disabled.
- If the systemd timer expires before the previous log-archival process is finished, then the systemd waits for the process to complete before starting the new process.

Displaying and Managing Boot Environment (BE) Information

NetVisor OS provides two boot environments (BEs): the current boot environment, and the previous boot environment. The presence of two BEs enable you to rollback or rollforward the software versions or configurations.

During software upgrade, a new boot environment is automatically created to install the new software and after the software upgrade, the switch boots into the newly created BE that is running the new software. Also, during software upgrade the older BE (which was not active) is deleted, and the current BE from where software upgrade started is preserved, and then the switch boots into newer BE with new software. This process ensures that there are only two boot environments (BEs) after upgrade.

To display boot environment information, use the following command:

```
CLI (network-admin@switch) > bootenv-show

name       version     current reboot space created        apply-current-config
--------- ----------- ------- ------ ----- -------------- --------------------
netvisor-1 5.2.1-15718  no      no     0    02-23,04:22:46      false
netvisor-2 6.1.0-17888  yes     yes    0    03-06,20:47:13      false
```

To reboot the switch into one of the boot environments other than current boot environment, use the following command:

```
CLI (network-admin@switch) > bootenv-activate-and-reboot name netvisor-1
```

To delete a boot environment, use the following syntax:

```
CLI (network-admin@switch) > bootenv-delete name netvisor-2
```

You can display information about different boot environments on the switch.

Exporting Configurations Using Secure Copy Protocol (SCP)

NetVisor OS supports export of switch configuration to a configuration bundle. This functionality allows the network administrator to create configuration backups that can be used in cases when the administrator needs to restore the switch configuration to a previous revision of the configuration.

The switch config export functionality in NetVisor OS can export the configuration bundle to a local disk on the switch or can upload the configuration bundle to an external server using Secure Copy Protocol(SCP). To upload the configuration bundle to an external server using SCP, the upload server should support SCP protocol.

To export a switch configuration, use the command:

```
CLI (network-admin@Leaf1) > switch-config-export
```

Specify any of the following options:	
`export-file switch-config` *`export-file`*	File name for exported configuration bundle.
`upload-server` *`upload-server-string`*	DNS Name or IP address of upload server and path to store configuration bundle on the upload server.Uploads the config file to server via SCP

For example,

```
CLI (network-admin@switch-crater) > switch-config-export export-file
crater-backup-04142011 upload-server root@server-test-67:/var/tmp/
server password:
Uploaded configuration to server at /var/tmp/
CLI (network-admin@switch-crater) >
server password:
Uploaded configuration to server at /root
CLI (network-admin@switch-crater) >
```

During the software upgrade process, NetVisor OS exports the switch configuration and makes it available at a /export directory that is accessible when the admin connects to the switch using SFTP client. NetVisor OS stores a maximum of three configuration archives on the switch. All older configurations are deleted.

Similar to NetVisor configuration export functionality, during software upgrade NetVisor can optionally upload the configuration bundle to an external upload server using SCP protocol to create a configuration back up. Similar to switch-config-export command, admin can use `upload-server` parameter of software-upgrade command to specify details of upload server to upload configurations to the external server.

```
CLI (network-admin@switch-crater) > software-upgrade package nvOS-6.1.0-
6010018109-onvl.pkg upload-server root@server-test-67:/var/tmp/
server password:
Scheduled background update.
* software-upgrade-status-show to check the progress
* software-upgrade-instant-status-show to check the instant upgrade status
```

```
Switch will reboot itself.DO NOT reboot manually.
CLI (network-admin@switch-crater) >
```

On upload server, configuration bundle is saved as below(`upgrade-<hostname>-<bename>-<sw version>-<timestamp>.tar.gz`):

```
root@server-test-67:/var/tmp# ls -l | grep crater
-rw-r--r--  1 root     root           16219 Apr 14 14:19 upgrade-switch-crater-crater-rfc-6.1.0-6010018118.2021-04-14T14.19.59.tar.gz
root@server-test-67:/var/tmp#
```

Upgrading the NetVisor OS

Software upgrades are a routine maintenance procedure that must be completed often. However, there are some guidelines to consider before you start the upgrade procedures.

Before you start the upgrade process, obtain the required upgrade software. You can download the software manually and copy it to a switch before beginning the upgrade procedures.

> **Caution**: If you had enabled split brain script on any nodes of your cluster or fabric, we recommend disabling the *split brain* script before performing any software upgrades on the switches in the cluster.

Below are the different upgrade scenarios and respective process:

Upgrading from NetVisor OS Version 3.0.4 or Earlier

If you are upgrading from NetVisor OS version 3.0.4 or earlier to NetVisor OS version 5.x.x, 6.x.x, or 7.x.x, then, you must first upgrade to NetVisor OS version 3.1.1 before upgrading to a later release. The direct upgrade from version 3.0.4 or earlier to version 5.x.x/6.x.x/7.x.x is not supported.

Upgrading from NetVisor OS Version 6.x.x

If you are upgrading from NetVisor OS version 6.x.x version to 7.0.0 GA, 7.0.0 HF, or 7.0.1 GA, the upgrade process also includes upgrading the Linux OS from Ubuntu version 16.04 to 20.04. Hence, you must use the focal upgrade image, that is, *nvOS-focalupg-7.0.1-70001xxxxx-onvl.pkg* by using the command:

```
CLI> software-upgrade package nvOS-focalupg-7.0.1-70001xxxxx-onvl.pkg
```

Upgrading from NetVisor OS Version 7.0.0 GA/HF1 version to 7.0.1 or above

If you are upgrading from NetVisor OS version 7.0.0 GA or 7.0.0 HF1 to 7.0.1 GA or above, you need to force a release upgrade to include the kernel upgrade. Even though NetVisor version 7.0.0 runs Ubuntu 20.04, the kernel version is different in NetVisor 7.0.1 release. To upgrade to version 7.0.1:

1. Download the focal upgrade image (for example, `nvOS-focalupg-7.0.1-70001xxxxx-onvl.pkg`)to `/sftp/import` directory.
2. Rename the file to `nvOS-force-7.0.1-70001xxxxx-onvl.pkg`
    1. `CLI> role-modify name network-admin shell sudo`
    2. `CLI> shell sudo`
    3. `CLI# cd /sftp/import`
    4. `CLI# mv nvOS-focalupg-7.0.1-70001xxxxx-onvl.pkg nvOS-force-7.0.1-70001xxxxx-onvl.pkg`
    5. `CLI# exit`
3. Upgrade the version by using the command:

    ```
    CLI> software-upgrade package nvOS-force-7.0.1-70001xxxxx-onvl.pkg
    ```

Note: The *force* rename option is relevant to focalupg image only. Please do not rename regular upgrade image with *force* rename option.

General Upgrade Procedure

Software and fabric upgrade process comprises of two distinct phases: (i) the installation (upgrade) of the new software and (ii) a switch reboot to activate the new software.

Read this section completely before starting the software upgrade procedure.

To upgrade the software on a switch, use the software-upgrade command.

```
CLI (network-admin@switch) > software-upgrade
```

`software-upgrade` `package upgrade-package-name.pkg`	Starts software upgrade on local switch using software bundle from /sftp/import directory.

`software-upgrade`
`package upgrade-package-name.pkg`
`auto-reboot|no-auto-reboot`
	Starts software upgrade on local switch using software bundle from /sftp/import directory. Use the `auto-reboot` or `no-auto-reboot` parameter to specify whether the switch needs to be automatically rebooted after software upgrade or whether admin will manually reboot the switch at a later time to start with new upgraded software. The default value is auto-reboot. This parameter is added in NetVisor OS 6.1.0.

`software-upgrade` *`package upgrade-package-name.pkg`* *`upload-server upload-server-name/ip:/path/to/upload/to`*	Starts software upgrade on local switch using software bundle from /sftp/import directory and uploads switch configuration backup file to the specified path on server using SCP.
`software-upgrade` *`package upgrade-package-name.pkg`* *`auto-reboot|no-auto-reboot`* *`upload-server upload-server-string`*	Starts software upgrade on local switch using software bundle from /sftp/import directory. Use the `auto-reboot` or `no-auto-reboot` parameter to specify whether switch needs to be automatically rebooted after software upgrade or not. The default value is `auto-reboot`. The presence of `upload-server` parameter in software-upgrade command uploads switch configuration bundle to the specified path on server using SCP.
`software-upgrade-abort`	Aborts an *in progress* software upgrade. Note that

	software upgrade will be aborted when upgrade process completes its current step and reaches logical next step. This command is added in NetVisor OS 6.1.0.
software-upgrade-reboot	Indicates the software upgrade process to reboot the switch after software upgrade is complete. This command is used when `no-auto-reboot` parameter is specified when software upgrade is started. This command is added in NetVisor OS 6.1.0.
software-upgrade-instant-status-show	Displays the current (most recent status of software upgrade during a fabric-wide upgrade process. This command accepts various format options of NetVisor CLI. This command is introduced in NetVisor OS 6.1.0
software-upgrade-status-show	Displays the current status of software upgrade that is in progress or completed.

Starting with NetVisor OS 6.1.0 release, additional options such as `auto-reboot|no-auto-reboot`, `abort`, and `manual reboot` are supported on the `software-upgrade` command. By using these options, you can control the automatic reboot of switch after software upgrade completes or you can abort an upgrade that is in progress.

If you start software upgrade by specifying `auto-reboot` parameter for `software-upgrade` command, switch automatically reboots after software upgrade completes and switch boots up with new software version.

However, if you start the software upgrade by specifying the `no-auto-reboot` parameter for `software-upgrade` command, switch completes the software upgrade and waits for administrator to manually reboot the switch to boot into upgraded software. After determining that upgrade is complete, you must issue a `software-upgrade-reboot` command to reboot the switch to boot with upgraded software.

While software upgrade is in progress or software upgrade is complete and switch is waiting for administrator to issue `software-upgrade-reboot` command, if you want to abort the upgrade and keep switch in current software version, you can use `software-upgrade-abort` command to abort the upgrade process. This scenario is explained further in the examples below.

Note: The software-upgrade-abort and software-upgrade-reboot commands are supported only if upgrade is started using software-upgrade command. Do not run these commands if you started fabric wide upgrade using fabric-upgrade command. An error message is displayed if you do so.

To start a software upgrade follow the steps below:

1. View the current version of NetVisor OS on the switch by using command:

   For example,

```
CLI network-admin@switch > software-show
version: 6.1.0-6010017911
```

2. Identify the software package. Depending on the version of NetVisor OS running on your switch, you should use appropriate software upgrade package. Select the appropriate upgrade bundle from the following upgrade matrix table based on the current version on your switch:

Table 2-1: Upgrade Matrix

Current Software	Target Release for Upgrade	Upgrade Type	Upgrade Software Package
3.1.x GA	7.0.2 GA	Software Upgrade using release upgrade bundle	nvOS-focalupg-7.0.2-7000219757-onvl.pkg
5.x.x OR 6.x.x OR	7.0.2 GA	Software Upgrade using regular offline upgrade bundle	nvOS-focalupg-7.0.2-7000219757-onvl.pkg
7.0.0 GA 7.0.0 HF1 7.0.1 GA	7.0.2 GA		nvOS-**force**-7.0.2-7000219757-onvl.pkg*

***Note**: The *force* rename option is relevant to focalupg image only. Please do not rename regular upgrade image with *force* rename option.

3. Copy the upgrade package to the switch, to do:
   a) Enable Secure File Transfer Protocol (SFTP) on the switch:

```
CLI (network-admin@switch)> admin-sftp-modify enable
sftp password:
confirm sftp password:
CLI (network-admin@switch)>
```

   b) Upload the Software package to the switch:

```
root@server-os-9:~/# sftp sftp@switch
The authenticity of host 'switch (10.0.0.02)' can't be established.
RSA key fingerprint is
SHA256:SI8VQZgJCppbrF4sRcby36Fx7rz3Hh5EJllPPyScLZU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'switch1, 10.0.0.02 (RSA) to the list of
known hosts.
* Welcome to Arista Networks Inc. Netvisor(R). This is a monitored
system.      *
*                ACCESS RESTRICTED TO AUTHORIZED USERS ONLY
           *
* By using the Netvisor(R) CLI, you agree to the terms of the Arista
Networks   *
* End User License Agreement (EULA). The EULA can be accessed via
           *
* http://www.arista.com/eula or by using the command "eula-show"
Password:
```

```
Connected to switch
sftp> cd import
sftp> put nvOS- nvOS-7.0.0-7000019033-onvl.pkg
Uploading nvOS- nvOS-7.0.0-7000019033-onvl.pkg
nvOS- nvOS-7.0.0-7000019033-onvl.pkg
nvOS- nvOS-7.0.0-7000019033-onvl.pkg 100% 332MB 7.5MB/s 04:00
```

4. Start the upgrade process by using the software-upgrade command with package parameter, which allows you to specify the name of the upgrade file. The switch gets automatically rebooted after software upgrade is complete.

```
CLI (network-admin@switch) > software-upgrade package nvOS-7.0.0-
7000019033-onvl.pkg
Scheduled background update. Use software-upgrade-status-show to check.
Switch will reboot itself. DO NOT reboot manually.
```

> **Caution**: Do not reboot or power off the switch during the upgrade procedure. When software upgrade is complete, switch reboots automatically.

5. Monitor the upgrade process using the software-upgrade-status-show command:

```
CLI (network-admin@switch)>software-upgrade-status-show show-interval 5

[Apr11.21:30:41] Starting software upgrade ...
[Apr11.21:30:42] Cleaning old package bundles
[Apr11.21:30:42] Checking available disk space...
[Apr11.21:30:42] Avbl free space: 12.69G, Required: 0.62G
[Apr11.21:30:42] Unpacking local package bundle...
[Apr11.21:30:42] Extracting initial bundle.
.
.
log
-----------------------------------------------------------------
[Apr11.21:30:41] Starting software upgrade ...
[Apr11.21:30:42] Cleaning old package bundles
[Apr11.21:30:42] Checking available disk space...
[Apr11.21:30:42] Avbl free space: 12.69G, Required: 0.62G
[Apr11.21:30:42] Unpacking local package bundle...
[Apr11.21:30:42] Extracting initial bundle.
[Apr11.21:30:50] Decrypting signed bundle.
[Apr11.21:30:52] Extracting signed bundle.
[Apr11.21:31:00] Extracting packages.
[Apr11.21:31:09] Fetching repository metadata.
[Apr11.21:31:09] Skipping dpkg update in current boot image
[Apr11.21:31:11] Computing package update requirements.
[Apr11.21:31:12] Upgrade agent version: 6.0.1-6000116966
[Apr11.21:31:12] Upgrading software upgrade framework
[Apr11.21:31:16] Fetching repository metadata.
[Apr11.21:31:17] Skipping dpkg update in current boot image
[Apr11.21:31:17] Computing package update requirements.
[Apr11.21:31:17] Upgrade agent version: 7.0.0-7000019033
[Apr11.21:31:17] Upgrading nvOS 6.0.1-6000116966 -> 7.0.0-7000019033
```

```
[Apr11.21:32:28] Cleaning up old BEs.
[Apr11.21:32:30] Upgrading nvOS 6.0.1-6000116966 -> 7.0.0-7000019033
[Apr11.21:32:30] Software upgrade completed. Rebooting.
```

After the switch reboots and you see following message in serial console of the switch, you can SSH to switch as the *network-admin*:

```
nvOS system info:
serial number: 1XXXXXXX00059
hostid: 090XXX9d
device id:
[ OK ] Started NetVisor Operating System.
Starting nvOSd Monitor...
[ OK ] Started nvOSd Monitor.
[ OK ] Reached target Multi-User System.
[ OK ] Reached target Graphical Interface.
[ OK ] Started Stop ureadahead data collection 45s after completed
startup.
Starting Update UTMP about System Runlevel Changes...
[ OK ] Started Update UTMP about System Runlevel Changes.
* Welcome to Arista Networks Inc. Netvisor(R). This is a monitored
system.     *
*                ACCESS RESTRICTED TO AUTHORIZED USERS ONLY
      *
* By using the Netvisor(R) CLI, you agree to the terms of the Arista
Networks  *
* End User License Agreement (EULA). The EULA can be accessed via
      *
* http://www.arista.com/eula or by using the command "eula-show"
switch1 login: network-admin
Password:
Netvisor OS Command Line Interface 6.1
Connected to Switch switch; nvOS Identifier:0x90XXXX9; Ver: 7.0.0-
7000019033
CLI (network-admin@switch) > software-show
version:  7.0.0-7000019033
```

To verify that a standalone (non-cluster) switch is fully operational after the software upgrade, you can use the `log-event-show` command. The 'System is up for service' log message indicates that the switch is ready for forwarding. For example:

```
CLI (network-admin@Leaf1)> log-event-show

category time                      name          code  level event-type message
-------- ------------------------- ------------- ----- ----- ---------- -----------------------
event    2021-03-10,23:37:40.572119 systemup_alert 11513 note  system     System is up for service
```

Note: You can also use the 'System is up for service' log message to verify that a switch is functional after a reboot or restart triggered by the following operations: `fabric-upgrade`, `switch-reboot`, `nvos-restart`, `fabric-join`, `cluster repeer`, and `config import` commands.

Apart from upgrading NetVisor OS on individual switches, starting from NetVisor OS version 6.1.0, you can perform software upgrade on all switches in a fabric using `software-upgrade` from one switch and reboot the switches to perform fabric-wide software upgrade (software upgrade on all the nodes in a fabric).

You can choose one of the below options to perform software upgrade on all switches from one switch:

- Start software upgrade on entire fabric and reboot the switches sequentially

OR

- Start software upgrade on entire fabric and reboot all switches, but not sequentially

Start software upgrade on entire fabric and reboot the switches sequentially

Follow the steps to perform software upgrade on entire fabric and reboot the switches sequentially:

1. Enable SFTP on all switches and provide the credentials for each switch using the CLI command:
   ```
   CLI (network-admin@switch)> switch * admin-sftp-modify enable
   ```

2. Enable shell access for *network-admin* role on all switches in the fabric:
   ```
   CLI (network-admin@switch)> switch * role-modify name network-admin shell
   ```

3. Download the software upgrade bundle to each switch by entering the below command only on one switch:
   ```
   CLI (network-admin@switch)> switch * shell cd /sftp/import;wget
   http://artifactory.pluribusnetworks.com/offline-pkgs/onvl/nvOS-
   7.0.0/nvOS-7.0.0-7000019033-onvl.pkg
   ```

4. Start the software upgrade process on all the switches using the command:
   ```
   CLI (network-admin@switch)> switch * software-upgrade package nvOS-7.0.0-
   7000019033-onvl.pkg no-auto-reboot
   ```

5. Verify that all nodes display the "*Waiting for software-upgrade-abort/software-upgrade-reboot*" message by using the command individually on each switch:
   ```
   CLI (network-admin@switch)> software-upgrade-status-show
   ```

6. Issue the *software-upgrade-reboot* command on each switch in the fabric one by one based on the reboot sequence that best works for your deployment.

Start software upgrade on entire fabric and reboot all switches non-sequentially

Follow the steps to perform fabric-wide software upgrade without reboot sequence:

1. Enable SFTP on all switches and provide the credentials for each switch using the CLI command:
   ```
   CLI (network-admin@switch)> switch * admin-sftp-modify enable
   ```

2. Enable shell access for *network-admin* role on all switches in the fabric:

```
CLI (network-admin@switch)> switch * role-modify name network-admin shell
```

3. Download the software upgrade bundle to each switch by entering the below command only on one switch:

```
CLI (network-admin@switch)> switch * shell cd /sftp/import;wget
http://artifactory.pluribusnetworks.com/offline-pkgs/onvl/nvOS-
7.0.0/nvOS-7.0.0-7000019033-onvl.pkg
```

4. Start the software upgrade process on all the switches with no-auto-reboot command parameter:

```
CLI (network-admin@switch)> switch * software-upgrade package nvOS-7.0.0-
7000019033-onvl.pkg  no-auto-reboot
```

5. Verify that all nodes display the "*Waiting for software-upgrade-abort/software-upgrade-reboot*" message by using the command individually on each switch

```
CLI  (network-admin@switch)>  software-upgrade-status-show
```

6. Reboot all the switches except the node where you are running the commands. This step ensures that the `software-upgrade-reboot` command is executed by all the switches before the node where you originally started the upgrade process goes down.

```
CLI (network-admin@switch)> switch <comma separated non controller
switches>  software-upgrade-reboot
```

7. Reboot the node where you started the upgrade and complete the upgrade process of the fabric:

```
CLI (network-admin@switch)> switch <controller node> software-upgrade-
reboot
```

Below are some examples of the software upgrade that can be used when you want the switch to not reboot automatically after upgrade is complete, but wants to reboot manually or if you want to abort the software upgrade that is in progress.

- Usage of the `software-upgrade` command with `no-auto-reboot` option and later issuing an `abort` command to abort the upgrade process:

```
CLI (network-admin@switch) > software-upgrade package nvOS-7.0.0-
7000019033-onvl.pkg  no-auto-reboot

CLI (network-admin@ switch) > software-upgrade-status-show
log
-------------------------------------------------------------------------
[Mar11.07:47:21] Starting software upgrade ...
[Mar11.07:47:22] Checking available disk space...
[Mar11.07:47:22] Avbl free space: 82.67G, Required: 1.28G
[Mar11.07:47:22] Unpacking local package bundle...
[Mar11.07:47:22] Extracting initial bundle.
[Mar11.07:47:38] Decrypting signed bundle.
[Mar11.07:47:39] Extracting signed bundle.
[Mar11.07:47:56] Extracting packages.
[Mar11.07:48:13] Fetching repository metadata.
[Mar11.07:48:13] Skipping dpkg update in current boot image
[Mar11.07:48:13] Computing package update requirements.
```

```
[Mar11.07:48:13] Upgrade agent version: 7.0.0-7000019033
[Mar11.07:48:13] Upgrading nvOS 6.1.0-6010017911 -> 7.0.0-7000019033
[Mar11.07:49:01] Waiting for software-upgrade-abort/software-upgrade-reboot

CLI (network-admin@switch) > software-upgrade-abort
Upgrade running, scheduled abort

CLI (network-admin@switch) > software-upgrade-status-show
log
---------------------------------------------------------------------
[Mar11.07:47:21] Starting software upgrade ...
[Mar11.07:47:22] Checking available disk space...
[Mar11.07:47:22] Avbl free space: 82.67G, Required: 1.28G
[Mar11.07:47:22] Unpacking local package bundle...
[Mar11.07:47:22] Extracting initial bundle.
[Mar11.07:47:38] Decrypting signed bundle.
[Mar11.07:47:39] Extracting signed bundle.
[Mar11.07:47:56] Extracting packages.
[Mar11.07:48:13] Fetching repository metadata.
[Mar11.07:48:13] Skipping dpkg update in current boot image
[Mar11.07:48:13] Computing package update requirements.
[Mar11.07:48:13] Upgrade agent version: 7.0.0-7000019033
[Mar11.07:48:13] Upgrading nvOS 6.1.0-6010017911 -> 7.0.0-7000019033
[Mar11.07:49:01] Waiting for software-upgrade-abort/software-upgrade-reboot
[Mar11.07:50:39] User indicated software-upgrade-abort
[Mar11.07:50:40] User aborted the upgrade.
[Mar11.07:50:42] Software upgrade aborted
[Mar11.07:50:42] ERR: Upgrade failed: User aborted the upgrade.
```

- Usage of the `software-upgrade` command with `no-auto-reboot` option and later issuing a `software-upgrade-reboot` command to complete the upgrade process:

```
CLI (network-admin@switch) > software-upgrade package nvOS-7.0.0-
7000019033-onvl.pkg  no-auto-reboot

CLI (network-admin@switch) > software-upgrade-status-show
log
---------------------------------------------------------------------
[Mar11.07:52:59] Starting software upgrade ...
[Mar11.07:52:59] Checking available disk space...
[Mar11.07:52:59] Avbl free space: 82.67G, Required: 1.28G
[Mar11.07:52:59] Unpacking local package bundle...
[Mar11.07:52:59] Extracting initial bundle.
[Mar11.07:53:16] Decrypting signed bundle.
[Mar11.07:53:17] Extracting signed bundle.
[Mar11.07:53:33] Extracting packages.
[Mar11.07:53:50] Fetching repository metadata.
[Mar11.07:53:50] Skipping dpkg update in current boot image
[Mar11.07:53:50] Computing package update requirements.
[Mar11.07:53:51] Upgrade agent version: 7.0.0-7000019033
[Mar11.07:53:51] Upgrading nvOS 6.1.0-6010017911 -> 7.0.0-7000019033
[Mar11.07:54:38] Waiting for software-upgrade-abort/software-upgrade-reboot
```

```
CLI (network-admin@switch) > software-upgrade-reboot
Upgrade running, scheduled reboot

CLI (network-admin@switch) > Shared connection to switch closed.
```

Check the status on the new BE:

```
CLI (network-admin@switch) > software-upgrade-status-show
log
------------------------------------------------------------
[Mar11.07:52:59] Starting software upgrade ...
[Mar11.07:52:59] Checking available disk space...
[Mar11.07:52:59] Avbl free space: 82.67G, Required: 1.28G
[Mar11.07:52:59] Unpacking local package bundle...
[Mar11.07:52:59] Extracting initial bundle.
[Mar11.07:53:16] Decrypting signed bundle.
[Mar11.07:53:17] Extracting signed bundle.
[Mar11.07:53:33] Extracting packages.
[Mar11.07:53:50] Fetching repository metadata.
[Mar11.07:53:50] Skipping dpkg update in current boot image
[Mar11.07:53:50] Computing package update requirements.
[Mar11.07:53:51] Upgrade agent version: 7.0.0-7000019033
[Mar11.07:53:51] Upgrading nvOS 6.1.0-6010017911 -> 7.0.0-7000019033
[Mar11.07:54:38] Waiting for software-upgrade-abort/software-upgrade-reboot
[Mar11.07:57:07] User indicated software-upgrade-reboot
[Mar11.07:57:08] User issued software-upgrade-reboot, rebooting the switch.
[Mar11.07:57:08] Upgrading nvOS 6.1.0-6010017911 -> 7.0.0-7000019033
[Mar11.07:57:08] Software upgrade completed. Rebooting.
[Mar11.07:59:35] Upgrade agent is listening.
```

- Usage of the `software-upgrade` command in default mode:

```
CLI (network-admin@switch) > software-upgrade package nvOS-7.0.0-
7000019033-onvl.pkg

log
------------------------------------------------------------
[Mar11.09:32:10] Starting software upgrade ...
[Mar11.09:32:10] Checking available disk space...
[Mar11.09:32:10] Avbl free space: 81.89G, Required: 1.28G
[Mar11.09:32:10] Unpacking local package bundle...
[Mar11.09:32:10] Extracting initial bundle.
[Mar11.09:32:27] Decrypting signed bundle.
[Mar11.09:32:28] Extracting signed bundle.
[Mar11.09:32:44] Extracting packages.
[Mar11.09:33:01] Fetching repository metadata.
[Mar11.09:33:01] Skipping dpkg update in current boot image
[Mar11.09:33:02] Computing package update requirements.
[Mar11.09:33:02] Upgrade agent version: 7.0.0-7000019033
[Mar11.09:33:02] Upgrading nvOS 6.1.0-6010017911 -> 7.0.0-7000019033
[Mar11.09:33:51] Upgrading nvOS 6.1.0-6010017911 -> 7.0.0-7000019033
[Mar11.09:33:51] Software upgrade completed. Rebooting.
```

```
Shared connection to switch closed.
```

- Usage of the `software-upgrade` command in default mode and later issuing an abort command to abort the upgrade process. The `software-upgrade-abort` command is also supported in the default mode (same as `auto-reboot` mode):

```
CLI (network-admin@switch) > software-upgrade package nvOS-7.0.0-
7000019033-onvl.pkg

CLI (network-admin@switch) > software-upgrade-abort

CLI (network-admin@switch) > software-upgrade-status-show
log
------------------------------------------------------------
[Mar11.09:43:04] Starting software upgrade ...
[Mar11.09:43:04] Checking available disk space...
[Mar11.09:43:04] Avbl free space: 81.15G, Required: 1.28G
[Mar11.09:43:04] Unpacking local package bundle...
[Mar11.09:43:04] Extracting initial bundle.
[Mar11.09:43:13] User indicated software-upgrade-reboot <============ time
at which user initiated the command
[Mar11.09:43:21] Decrypting signed bundle.
[Mar11.09:43:22] Extracting signed bundle.
[Mar11.09:43:38] Extracting packages.
[Mar11.09:43:55] Fetching repository metadata.
[Mar11.09:43:56] Skipping dpkg update in current boot image
[Mar11.09:43:56] Computing package update requirements.
[Mar11.09:43:56] Upgrade agent version: 7.0.0-7000019033
[Mar11.09:43:56] User aborted the upgrade. <================ place at
which actual abort is performed (consistent state)
[Mar11.09:43:56] Software upgrade aborted
[Mar11.09:43:56] ERR: Upgrade failed: User aborted the upgrade.
```

Another new command added in NetVisor OS version 6.1.0 is the `software-upgrade-instant-status-show` command, which displays the most current status of the upgrade process of all the switches in the fabric. This command is different from the `software-upgrade-status-show` command, where all details of the upgrade process is displayed.

Use this command to view the most recent status on each switch in the fabric when you perform the upgrade process on all switches of the fabric (fabric-wide upgrade):

```
CLI (network-admin@switch) > switch * software-upgrade-instant-status-show
show-interval 1

switch      log
-------  ------------------------------------------------------------
Switch1 [Mar11.07:48:13] Upgrading nvOS 6.0.1-6000116966 -> 6.1.0-6010017911
Switch2 [Mar11.07:48:13] Upgrading nvOS 6.0.1-6000116966 -> 6.1.0-6010017911
```

The `switch *` in the above command indicates all the switches in that fabric which is undergoing the software upgrade process.

Implementing a Fabric Upgrade

A switch that is part of a fabric can be upgraded locally using software-upgrade process or you can start a fabric-wide upgrade of all nodes in the fabric.
While performing a fabric wide upgrade, the switch on which fabric-upgrade command is issued acts as the controller node. It is mandatory to copy the package to /sftp/import/ directory of the controller node.

NetVisor OS copies the upgrade package to other nodes in the fabric as part of fabric-wide upgrade. The controller node monitors the progress of the upgrade on each node and you can view the status of the upgrade using the fabric-upgrade-status-show command. The controller node is identified by an "*" after the switch name in the status output.

NetVisor OS enables you to implement a fabric-wide upgrade and reboot the switches at the same time or in a sequential order.

Upgrading the Fabric

Follow the tasks explained here to upgrade all switches in the fabric:

Upgrade Commands

Following are the commands that control the fabric upgrade process:

- **`fabric-upgrade-start`** – begin the upgrade process on entire fabric by specifying the package name
- **`fabric-upgrade-status-show`** – monitor the progress of the upgrade for each node in the fabric
- **`fabric-upgrade-finish`** – finalize when upgrade is complete
- **`fabric-upgrade-abort`** – abort the entire upgrade process and return switches to their prior state

The `fabric-upgrade-start` command defines all the future behavior of the upgrade process, that is, any optional settings need to be defined with the ***`start`*** command. In addition, the `fabric-upgrade-start` command acquires a configuration lock from all the members of the fabric. No configuration changes are permitted during the upgrade process.

The `fabric-upgrade-start` command includes the following options:

```
CLI (network-admin@switch) > fabric-upgrade-start
```

`fabric-upgrade-start`	Starts the software upgrade or prepare process on entire fabric.
`packages sftp-files name`	Comma separate list of software bundles.
Specify between 0 and 7 of the following options:	
`auto-finish|no-auto-finish`	Automatically starts the software upgrade on the entire fabric. The default option is `no-auto-finish`.
`abort-on-failure|no-abort-on-failure`	Whether to abort fabric upgrade if a node fails or not. The default option is `no-abort-on-failure`.

`manual-reboot|no-manual-reboot`	Whether to defer to user for reboot after upgrade.
`download-count 1..5`	Number of concurrent downloads. The default value is 5 (maximum). This option is introduced in version 6.1.0.
`prepare|no-prepare`	Perform setup steps for the actual upgrade.
`upload-server` *`upload-server-string`*	Upload config file to server via SCP.
`server-password`	SCP host password.

During a fabric upgrade, all members of fabric downloads the upgrade bundle from controller node. By default, fabric upgrade allows a maximum of 5 switches in the fabric to download the upgrade bundle from controller at a given time.

However, this can cause issues if there is bandwidth constraint or can overwhelm the controller node if the controller is of a lower hardware specification switch. To address this issue, starting with NetVisor OS version 6.1.0, you can use the `download-count` parameter of `fabric-upgrade` command to reduce the number of concurrent downloads depending upon your network conditions and hardware capabilities of the controller node. By default, the `download-count` is five.

For example, to set the download count to 2, use the command:

```
CLI (network-admin@switch) > fabric-upgrade-start packages nvOS-6.0.1-
6010017911-onvl.pkg download-count 2
```

Before you start the fabric-wide upgrade

1. Copy image to */sftp/import/* directory of controller node.
2. Ensure there is a reliable in-band and/or out-off-band connectivity between fabric members, which helps to distribute the software for the upgrade and monitor the progress of the upgrade process. The distribution of software to the nodes of the fabric is done in parallel, that is, each node receives the software approximately at the same time. An independent communications link is established over the fabric communications path to distribute the software to each node in the fabric.
3. Console access to switches are recommended.
4. Switches do not accept any configuration commands once upgrade starts, so plan accordingly.

Copying Image to the Switch

To copy the image:

- First, enable Secure File Transfare Protocol (SFTP) service on all switches by using the following command and create an /sftp/import directory:

```
CLI (network-admin@switch)>switch* admin-sftp-modify enable
sftp password:
confirm sftp password:
```

```
CLI  (network-admin@switch)>
```

OR

Enable shell access on all the switches to copy the file to the folder by using the command:

```
CLI(admin@netvisor) > switch* role-modify name network-admin shell
```

And access the shell:

```
CLI(admin@netvisor) > shell
network-admin@netvisor:~$ cd /sftp/import
 network-admin@netvisor:/sftp/import$
```

- Copy the image to */sftp/import* directory

```
root@server-os-9:~/# sftp sftp@switch
The authenticity of host 'switch (10.0.0.02)' can't be established.
RSA key fingerprint is
SHA256:SI8VQZgJCppbrF4sRcby36Fx7rz3Hh5EJllPPyScLZU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'switch, 10.0.0.02 (RSA) to the list of known
hosts.
* Welcome to Arista Networks Inc. Netvisor(R). This is a monitored
system.     *
*                ACCESS RESTRICTED TO AUTHORIZED USERS ONLY
     *
* By using the Netvisor(R) CLI, you agree to the terms of the Arista
Networks  *
* End User License Agreement (EULA). The EULA can be accessed via
     *
* http://www.arista.com/eula or by using the command "eula-show"
Password:
Connected to switch
sftp> cd import
sftp> put nvOS-6.1.0-6010018118-onvl.pkg
Uploading nvOS-6.1.0-6010018118-onvl.pkg
nvOS-6.1.0-6010018118-onvl.pkg
nvOS-6.1.0-6010018118-onvl.pkg 100% 332MB 7.5MB/s 04:00
```

Fabric upgrade with *manual-reboot* option

This option completes in three phases:

- Copy upgrade package to switches in fabric and start upgrade with `fabric-upgrade-start` command.
- Finish or abort fabric upgrade with `fabric-upgrade-finish` or `fabric-upgrade-abort` commands.
- Manually reboot switches with the `switch-reboot` command.

Starting the Fabric Upgrade

> Before starting the upgrade process, ensure that all the nodes of the fabric are online, you can use the command `fabric-node-show` and check that the `state` is online for all the nodes.

Use the following command to copy the upgrade package from controller switch to all other switches in the fabric and start the upgrade process.

Run the `fabric-upgrade-finish` command to reboot the fabric and complete the upgrade process:

```
CLI network-admin@switch >fabric-upgrade-start packages <image> manual-reboot
```

The `fabric-upgrade-start` command defines all behavior of the upgrade process during the upgrade, that is, any optional settings need to be defined with the "start" command (see optional settings below). In addition, the `fabric-upgrade-start` command acquires a configuration lock from all the members of the fabric. No configuration changes are permitted during the upgrade process.

The optional setting parameters for the `fabric-upgrade-start` command includes:

- `auto-finish` — specify to start software upgrade on the entire fabric. The default is `no-auto-finish.`
- `abort-on-failure` — specify if you want the upgrade to stop if there is a failure during the process.
- `manual-reboot` — specify if you want to manually reboot individual switches after the upgrade process. If you specify no-manual-reboot, all switches reboot automatically after the upgrade is complete.
- `prepare` — specify if you want to perform setup steps prior to performing the upgrade. This step copies the offline software package and then extracts and prepares for the final upgrade process. Once you begin the prepare process, you cannot add new switches to the fabric.

A sample upgrade process is explained below. Start the upgrade process by using the command:

```
CLI (network-admin@switch) > fabric-upgrade-start packages nvOS-6.1.0-6010018118-onvl.pkg auto-finish manual-reboot
Warning: This will start software upgrade on your entire fabric.
Please confirm y/n (Default: n):y
Scheduled background update.
Use:
* fabric-upgrade-status-show to check progress
* fabric-upgrade-finish to finalize when complete
* fabric-upgrade-abort to cancel cleanly
* switch-reboot on each switch in fabric to reboot manually when complete
```

Monitoring the Upgrade Process

The controller node monitors the progress of the upgrade on each node and reports the status of the upgrade by using the `fabric-upgrade-status-show` command. There are many interim steps to the upgrade process and to continually monitor the upgrade process use the *show-interval (in seconds)* option with the `fabric-upgrade-status-show` command:

Use the following commands to:

- To monitor the progress of the upgrade for each node in the fabric:

```
CLI (network-admin@switch) > fabric-upgrade-status-show
```

For example,

```
CLI (network-admin@switch) > fabric-upgrade-status-show show-interval 5

log                                switch          state              cluster
---------------------------------  --------------  -----------------  ---------------------
(0:00:36)Agent needs restart       eq-colo-7       Agent restart wait aqr07-08(sec)
(0:00:34)Agent needs restart        tucana-colo-7   Agent restart wait spine-cl(sec)
(0:03:57)Extracting signed bundle. aquarius-test-1 Running             aquarius-test-1-2(sec)
(0:00:45)Agent needs restart        dorado-test-3   Agent restart wait dorado-test-2-3(sec)
(0:03:57)Extracting signed bundle. aqr08           Running            aqr07-08(pri)
(0:00:28)Agent needs restart       switch*         Agent restart wait spine-cl(pri)
(0:03:57)Extracting signed bundle. aquarius-test-2 Running             aquarius-test-1-2(pri)
(0:00:38)Agent needs restart        dorado-test-2   Agent restart wait dorado-test-2-3(pri)
(0:01:00)Agent needs restart       scorpius10      Agent restart wait none
(0:00:47)Agent needs restart       vnv-mini-1      Agent restart wait none
log                                switch          state              cluster
---------------------------------  --------------  -----------------  ---------------------
(0:00:36)Agent needs restart       eq-colo-7       Agent restart wait aqr07-08(sec)
(0:00:34)Agent needs restart        tucana-colo-7   Agent restart wait spine-cl(sec)
(0:04:02)Extracting packages.      aquarius-test-1 Running            aquarius-test-1-2(sec)
(0:00:45)Agent needs restart        dorado-test-3   Agent restart wait dorado-test-2-3(sec)
(0:04:02)Extracting signed bundle. aqr08           Running            aqr07-08(pri)
(0:00:28)Agent needs restart       switch*         Agent restart wait spine-cl(pri)
(0:04:02)Extracting packages.      aquarius-test-2 Running            aquarius-test-1-2(pri)
(0:00:38)Agent needs restart    dorado-test-2  Agent restart wait dorado-test-2-3(pri)
(0:01:00)Agent needs restart   scorpius10     Agent restart wait none
(0:00:47)Agent needs restart   vnv-mini-1     Agent restart wait none
.
.
log                                                          switch          state
cluster
-----------------------------------------------------------  --------------  ---------------
----------------------
(0:01:53)Waiting for completion processing                   eq-colo-7       Upgrade complete
aqr07-08(sec)
(0:01:25)Waiting for completion processing                   tucana-colo-7   Upgrade complete
spine-cl(sec)
(0:06:24)Waiting for completion processing                   aquarius-test-1 Upgrade complete
aquarius-test-1-2(sec)
(0:02:29)Waiting for completion processing                   dorado-test-3   Upgrade complete
dorado-test-2-3(sec)
(0:06:43)Waiting for completion processing                   aqr08           Upgrade complete
aqr07-08(pri)
(0:01:23)Waiting to reboot                                   tucana-colo-6*  Upgrade complete
spine-cl(pri)
(0:06:16)Waiting for completion processing                   aquarius-test-2 Upgrade complete
```

```
aquarius-test-1-2(pri)
(0:02:19)Waiting for completion processing                  dorado-test-2   Upgrade complete
dorado-test-2-3(pri)
(0:06:09)Waiting for completion processing                  scorpius10      Upgrade complete
none
(0:08:09)Upgrading nvOS 6.0.1-6000116966 -> 6.1.0-6010017911 vnv-mini-1      Running
none
.
.
log                                          switch          state            cluster
-------------------------------------  --------------  ----------------
----------------------
(0:01:53)Current/Reboot BE: netvisor-16           eq-colo-7       Upgrade complete aqr07-
08(sec)
(0:01:25)Waiting for completion processing         tucana-colo-7   Upgrade complete spine-
cl(sec)
(0:06:24)Waiting for completion processing         aquarius-test-1 Upgrade complete aquarius-
test-1-2(sec)
(0:02:29)Destroy BE: netvisor-45                  dorado-test-3   Upgrade complete dorado-
test-2-3(sec)
(0:06:43)Waiting for completion processing        aqr08           Upgrade complete aqr07-
08(pri)
(0:01:23)Waiting to reboot                        switch*  Upgrade complete spine-cl(pri)
(0:06:16)Current/Reboot BE: netvisor-10            aquarius-test-2 Upgrade complete aquarius-
test-1-2(pri)
(0:02:19)Software upgrade done. Waiting for reboot dorado-test-2   Upgrade complete dorado-
test-2-3(pri)
(0:06:09)Waiting for completion processing        scorpius10      Upgrade complete none
(0:13:17)Waiting for completion processing        vnv-mini-1      Upgrade complete none
-----------------------------------------  ----------------  ----------------------
--------------
(0:01:53)Upgrade complete                      eq-colo-7       Reboot wait      aqr07-08(sec)
(0:01:25)Upgrade complete                      tucana-colo-7   Reboot wait      spine-cl(sec)
(0:06:24)Upgrade complete                      aquarius-test-1 Reboot wait      aquarius-
test-1-2(sec)
(0:02:29)Upgrade complete                      dorado-test-3   Reboot wait      dorado-test-
2-3(sec)
(0:06:43)Upgrade complete                      aqr08           Reboot wait      aqr07-08(pri)
(0:01:23)Sending Reboot wait message to handler switch*  Reboot wait       spine-cl(pri)
(0:06:16)Upgrade complete                      aquarius-test-2 Reboot wait      aquarius-
test-1-2(pri)
(0:02:19)Upgrade complete                      dorado-test-2   Reboot wait      dorado-test-
2-3(pri)
(0:06:09)Upgrade complete                      scorpius10      Reboot wait      none
(0:13:17)Waiting for completion processing     vnv-mini-1      Upgrade complete none
Connection to switch closed by remote host.
Connection to switch closed.
```

The first entry in the log is the elapsed time of the upgrade process. It does not include waiting time. The switch with the asterisk (*) is the upgrade controller node where the `fabric-upgrade-start` command was issued.

During a fabric-wide upgrade, the messages displayed by the `fabric-upgrade-status-show` command, based on the current progress status is described in table below:

Table 2-1: Fabric Upgrade Status Description

Message	Description

`Downloading package bundle`	The upgrade package is downloaded from the initial node to all the other nodes.
`Extracting initial bundle`	Once successfully downloaded, the offline bundle is extracted.
`Extracting signed bundle`	The signature of the package is verified.
`Extracting packages`	The packages are extracted and readied to install.
`Agent needs restart`	The nodes wait for the package to be extracted on all nodes of the fabric.
`Upgrading nvOS *`	The switch upgrades NetVisor from the older version to the newer one
`Waiting for fabric-upgrade-finish/abort`	The switches wait for the user to complete the upgrade once it completes using either of the commands mentioned above.

- Once the upgrade package is copied to all switches by fabric upgrade process and the upgrade process is completed, run the `fabric-upgrade-finish` or `fabric-upgrade-abort` command to either finish the upgrade or abort it.

```
CLI (network-admin@switch) > fabric-upgrade-finish
```

Once the upgrade phase is complete, all switches display the *Upgrade complete* message in the log field. You can then reboot the fabric. Following is an example:

```
CLI (network-admin@switch) > fabric-upgrade-finish

       log                                           switch   state              cluster
----------------------------------------------   ------   ---------------     ------------
(0:13:00)Waiting for fabric-upgrade-finish/abort   sw2     Upgrade complete    spine(sec)
(0:12:04)Waiting for fabric-upgrade-finish/abort   sw1*    Upgrade complete    spine(pri)
(0:16:49)Waiting for fabric-upgrade-finish/abort   sw1     Upgrade complete    none
(0:15:27)Waiting for fabric-upgrade-finish/abort   sw2     Upgrade complete    none

Finalizing upgrade. Manual reboot of nodes required.
```

- Manual reboot: each switch in the fabric need to be manually rebooted after the upgrade is completed. The `fabric-upgrade-status-show` command displays the status as *switch waiting to reboot*. For example,

```
CLI (network-admin@switch) > fabric-upgrade-status-show
fabric-upgrade-status-show: Switch waiting to reboot
```

At this point, upgrade is completed on all switches, reboot switches one at a time by the following command:

```
CLI (network-admin@switch) > switch-reboot
```

> **Note**: You should **reboot** the *controller switch* at the end only.

Note: All the nodes of the fabric should be running the same software version for the NetVisor OS features to work correctly.

- During the installation, if there is any issue, the upgrade process can be rolled back using the command `fabric-upgrade-abort`. To abort the upgrade process and return the switches to their prior state (no reboot needed):

```
CLI (network-admin@switch) > fabric-upgrade-abort
```

Aborts the fabric upgrade process. All changes to the switches are cleaned up and the server-switches do not reboot. The configuration lock on the fabric is also released. If you issue the `fabric-upgrade-abort` command during the upgrade process, it may take some time before the process stops because the upgrade has to reach a logical completion point before the changes are rolled back on the fabric. This allows the proper cleanup of the changes.

Warning: DO NOT use the switch-reboot command to reboot the switch while upgrade is in progress.

Note: During the fabric-upgrade process, the fabric configuration is locked throughout the entire process and you cannot change any configurations during the process.

Related Command:

Other related commands for fabric-upgrade includes:

- `fabric-upgrade-prepare-cancel` — cancels a fabric upgrade that was prepared earlier.
- `fabric-upgrade-prepare-resume` — resume a fabric upgrade that was prepared earlier.
- `fabric-upgrade-prepare-show` — displays the status of prepared upgrades on the fabric nodes.

Review bootenv

A new boot environment is built during the upgrade process. Upon reboot this new boot environment becomes active and the new software is up-and-running on the switch. Generally, it is not required to interact with the boot environments during the upgrade process. It may be necessary to review the boot environments using the command `bootenv-show` if there is some failure during the upgrade process.

Modifying the Fabric Password

NetVisor OS version 6.0.0 offers support for changing and resetting the fabric password. Use the `fabric-local-modify` command to change the current fabric password or reset the password forcefully.

```
CLI (network-admin@switch) > fabric-local-modify
```

`fabric-local-modify`	Modify fabric local information.
Specify one or more of the following options:	
`vlan` *`0..4095`*	Specify the VLAN assigned to the fabric.
`change-password` *`change-password-string`*	Use this option to change the fabric password after confirming the current password.
`reset-password` *`reset-password-string`*	Use this option to reset the fabric password forcefully. **Note:** Resetting the password requires assistance from Arista support.
`fabric-network in-band|mgmt|vmgmt`	Specify the type of interface over which the fabric administration network communicates.
`control-network in-band|mgmt|vmgmt`	Specify the type of interface over which the control plane network communicates.
`fabric-advertisement-network inband-mgmt|inband-vmgmt|inband-only|mgmt-only`	Specify the interface to send fabric advertisement packets on.

For example, to change the current fabric password, use the command:

```
CLI (network-admin@switch) > fabric-local-modify change-password
Please note this will disrupt fabric communications affecting user traffic
till the password is changed on ALL nodes.
Please confirm y/n (Default: n):<y>
current password: <current-password>
fabric password: <fabric-password>
confirm fabric password: <fabric-password>
Password changed. Please ensure that the password is changed on all nodes
of the fabric.
```

You can leave the `<current-password>` field empty, if the fabric does not have a pre-existing password. Enter the new password in the `<fabric-password>` field.

If you forget the fabric password, you can reset the same with assistance from Arista support. For example:

```
CLI (network-admin@switch) > fabric-local-modify reset-password
Please note this will disrupt fabric communications affecting user traffic
till the password is changed on ALL nodes.
Please confirm y/n (Default: n):<y>
```

```
fabric password: <fabric-password>
confirm fabric password: <fabric-password>
fabric-local-modify: Contact Pluribus Networks to reset password
```

Note: Changing or resetting the fabric password on one of the nodes disrupts fabric communication and can affect the user traffic as well. You must change or reset the password individually on all fabric nodes to re-establish the communication.

If you modify the password on one of the nodes in a fabric, the `fabric-node-show` output displays the state of other nodes in the fabric as 'offline'.

For example, if you change the password on switch1, while switch2 and switch3 are other nodes in the fabric, the `fabric-node-show` output is:

```
CLI (network-admin@switch1) > fabric-node-show layout vertical
name:                   switch1
fab-name:               fabric10
mgmt-ip:                 10.10.10.29/23
in-band-ip:              192.168.0.33/24
in-band-vlan-type:      public
version:                 6.0.0-6000016140
state:                  online
firmware-upgrade:        not-required
device-state:           ok
name:                   switch2
fab-name:               fabric10
mgmt-ip:                 10.10.10.14/23
in-band-ip:              192.168.0.227/24
in-band-vlan-type:      public
version:                 6.0.0-6000016140
state:                  offline
firmware-upgrade:        not-required
device-state:           ok
name:                   switch3
fab-name:               fabric10
mgmt-ip:                 10.10.10.16/23
in-band-ip:              192.168.0.223/24
in-band-vlan-type:      public
version:                 6.0.0-6000016140
state:                  offline
firmware-upgrade:        not-required
device-state:           ok
```

To add a switch to a fabric, you must authenticate the `fabric-join` operation by using the fabric password. For example:

```
CLI (network-admin@switch1) > fabric-join name fabric1 password
fabric password: <fabric-password>
confirm fabric password: <fabric-password>
Joined fabric fabric1. Restarting nvOS...
```

If a cluster node exits a fabric, and the fabric password changes thereafter, you must authenticate the

cluster re-peer process by using the new password. For example, to re-peer node1 with node2, use the command:

```
CLI (network-admin@node1) > fabric-join repeer-to-cluster-node node2
password
fabric password: <fabric-password>
confirm fabric password: <fabric-password>
Joined fabric fabric1. Restarting nvOS...
```

Copying and Importing Configuration Files

You can create a configuration file to import to another switch by using the `switch-config-copy-to-import` command. To create a configuration file with the name config-092613 to import on another switch, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-config-copy-to-import export-file
config-092613
```

After you create the configuration file, you can export it to `/nvOS/export/` directory, and SFTP to it from the target switch.

To review the available files for import and export, use the following syntax:

```
CLI (network-admin@Leaf1) > switch-config-show

switch        export-file
pbg-nvos      config-092613.tar.gz
```

Depending on the available remote access services, you can now copy the configuration file to a different switch. For example, you can SFTP to another switch using the IP address of the switch, login as SFTP with the password that you previously set, `cd /nvOS/import` and `get` the configuration file.

To upload the configuration file to the target switch and set the configuration from the configuration file, transfer the configuration file to the target switch with the IP address, 192.168.3.35.

To export a configuration to a server, use the `switch-config-export` command:

```
CLI (network-admin@Leaf1) > switch-config-export
```

Auto-configuration of IPv6 Addresses on the Management Interface Support

IPv6 Stateless Address Auto-Configuration (SLAAC)

Like IPv4 addresses, you can configure hosts in a number of different ways for IPv6 addresses. Dynamic Host Configuration Protocol (DHCP) assigns IPv4 addresses dynamically and static addresses assign fixed IP addresses. DHCP provides a method of dynamically assigning addresses, and provides a way to assign the host devices other service information like DNS servers, domain names, and a number of different custom information.

SLAAC allows you to address a host based on a network prefix advertised from a local network router using Router Advertisements (RA). RA messages are sent by default by IPv6 router.

These messages are sent out periodically by the router and include following details:

- One or more IPv6 prefixes (Link-local scope)
- Prefix lifetime information
- Flag information
- Default device information (Default router to use and its lifetime)

NetVisor OS enables SLAAC by default on the switch.

When you configure IPv6 address on the management interface during setup, the parameter, **assignment**, has two options:

- **none** — Disables IPv6 addresses.
- **autoconf** — Configure the interface with SLAAC.

Support for Local Loopback IP Addresses

NetVisor OS uses the loopback interface as an always up and available virtual interface, and you can assign it a unique IPv4 or IPv6 address. NetVisor OS uses a loopback interface as a termination address for some routing protocols, because of the availability of the interface. NetVisor OS allows you to configure a loopback address for a global zone.

- Send a dedicated ping to loopback interface
- Create a BGP neighbor using the loopback Interface with OSPF so reach-ability is there for BGP and BGP next hop self
- Make sure log messages do not show any issues

NetVisor OS deploys the loopback IP address as persistent in the configuration and not affected by a reboot or reset of NetVisor OS.

To add a loopback IPv4 or IPv6 address or both to an existing configuration, use the following syntax:

```
CLI (network-admin@switch1) > switch-setup-modify loopback-ip ip-address loopback-ipv6 ipv6-address
```

For example, to add the IPv4 address, 12.1.1.1, and the IPv6 address, 1212::1, use the following syntax:

```
CLI (network-admin@switch1) > switch-setup-modify loopback-ip 12.1.1.1 loopback-ip6 1212::1

CLI (network-admin@switch1) > switch-setup-show format in-band-ip,in-band-ip6,loopback-ip,loopback-ip6, layout horizontal

in-band-ip     in-band-ip6  loopback-ip  loopback-ip6
-------------  -----------  -----------  ------------
150.1.1.1/24  2001::1/96   12.1.1.1      1212::1
150.1.1.2/24  2001::2/96   12.1.1.2      1212::2
```

After configuring the loopback address, you can SSH to the switch over the management, in-band, or loopback interface using the following syntax:

```
CLI (network-admin@switch1) > ssh network-admin@<mgmt/inband/loopback ip-address>
```

Then from CLI, execute the `shell` command to access the switch shell:

```
CLI (network-admin@switch1) > network-admin@switch:~$
```

Configuring REST API Access

NetVisor OS enables you to use REST API over HTTP and HTTPS to manage the switches in a fabric, in addition to using the CLI. Though REST API access over HTTP is simpler to configure, Arista recommends using HTTPS for security reasons. The vREST web application that runs on the switch enables the REST API client to access the resources on the switch.

Follow the steps below to configure REST API access over HTTP:

- Enable the web service using the command `admin-service-modify`.

```
CLI (network-admin@switch1) admin-service-modify if mgmt web
```

`admin-service-modify`	Modifies services on the switch.
`if` *`if-string`*	Specify the administrative service interface. The options are `mgmt` or `data`.
`web|no-web`	Specify if you want to enable web management. **Use this option to enable REST API access over HTTP.**
`web-ssl|no-web-ssl`	Specify if you want to use SSL and certificates for web services. **Use this option to enable REST API access over HTTPS.**
`web-ssl-port` *`web-ssl-port-number`*	Specify the web SSL port.
`web-port` *`web-port-number`*	Specify the port for web management.
`web-log|no-web-log`	Specify if you want to turn on or off web logging.

- Verify configuration using the command `admin-service-show:`

```
CLI (network-admin@switch1) admin-service-show

switch      if   ssh nfs web web-ssl web-ssl-port web-port snmp net-api icmp
----------- ---- --- --- --- ------- ------------ -------- ---- ------- ----
switch1     mgmt on  off on  on      443          80       on   off     on
switch1     data on  off on  off     443          80       on   off     on
```

- Download Swagger tool onto the desktop and open index.html file.
- Paste the URL http://<ip-address>/vRest/api-docs into the URL field.

  <ip-address> should be the hostname or management IP address of your switch.
- Click 'Explore'. Commands are populated as below:

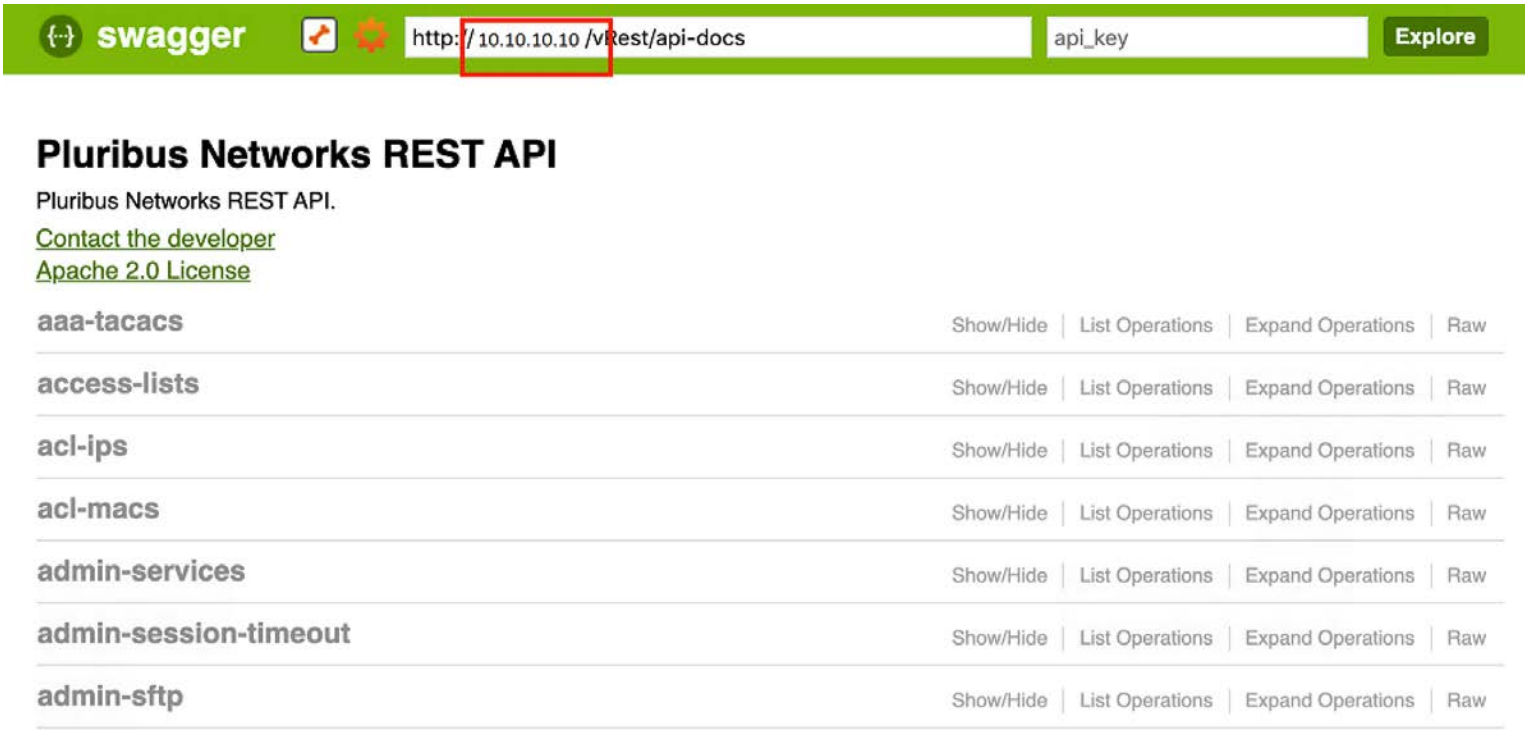


Figure 2- 1: Swagger UI for Pluribus REST API

- Right click to enable Chrome Inspect, and view the network tab to ensure that all the commands are loaded.
- Expand one of the CLI commands to see the corresponding GET/POST/DELETE/PUT options.
- Click on the model schema to view the list of configurable options supported under that model (vRest API) which are populated as key-value pairs. Make the required modifications to the values and click 'Try it out!'.

Using cURL with REST API

To create a VLAN, use the vREST API:

```
$ curl -u network-admin:pluribus!23 -H "Content-Type:application/json" -X POST
http://switch1/vRest/vlans -d
   '{
    "scope": "local",
    "id": 1111,
    "description": "hello world"
    }'
```

By default, all the vRest APIs provide fabric level information. To specifically access the resources of the

switch (scope : local), the switch ID needs to be specified in the URL.

For switch ID specific information, use the command:

```
$ curl -u network-admin:pluribus!23 http://switch1/vRest/vlans?api.switch={hostid} |
python -m json.tool
```

as in the following example:

```
$ curl -u network-admin:pluribus!23 http://10.10.10.10/vRest/vlans?
api.switch=201327131 | python -m json.tool
```

For switch information listing a local scope:

```
$ curl -u network-admin:pluribus!23 http://switch1/vRest/vlans?api.switch=fabric |
python -m json.tool
```

or

```
$ curl -u network-admin:pluribus!23 http://switch1/vRest/vlans | python -m json.tool
```

as in the following example:

```
$ curl -u network-admin:pluribus!23 http://10.110.0.48/vRest/vlans | python -m json.tool
```

Configuring REST API Access over HTTPS

To enable HTTPS communication between a REST API client and NetVisor vREST web service, you have two options:

1. You can generate a self-signed certificate using NetVisor CLI and use this certificate for REST web service.

2. After creating a self-signed certificate using NetVisor CLI, create a certificate request, get the certificate request signed by a trusted Certificate Authority (CA), import the signed certificate and CA certificate into NetVisor OS and use the certificates for REST web service.

Follow the steps below to create the certificates and deploy them:

- Generate self-signed certificate (the private key and the certificate file, in PEM format) using the `web-cert-self-signed-create` command.

```
CLI  (network-admin@switch1)  >  web-cert-self-signed-create
```

`web-cert-self-signed-create`	This command creates a self-signed certificate and deletes any existing certificates.
`country  country-string`	Specify the contact address of the organization, starting with the country code.
`state  state-string`	Specify the state or province.
`city` *`city-string`*	Specify the city.

`organization` *`organization-string`*	Specify the name of the organization.
`organizational-unit` *`organizational-unit-string`*	Specify the organizational unit.
`common-name` *`common-name-string`*	Specify the common name. The common name must precisely match the hostname where the certificate is installed.

For example:

```
CLI (network-admin@switch1) > web-cert-self-signed-create country US state
California city "Santa Clara" organization "Pluribus Networks Inc"
organizational-unit Engineering common-name switch1.pluribusnetworks.com
Successfully generated self-signed certificate.
```

- ·If you want to get the certificate signed by a trusted Certificate Authority(CA), generate a CSR from the self-signed certificate by using the command `web-cert-request-create.`

```
CLI (network-admin@switch1) > web-cert-request-create
Certificate signing request successfully generated
at   /sftp/export/switch1.pluribusnetworks.com.csr.
```

- To view the CSR, use the command `web-cert-request-show.`

```
CLI (network-admin@switch1) > web-cert-request-show
```

`web-cert-request-show`	Displays the certificate signing request.
`cert-request` *`cert-request-string`*	Specify the name of the CSR.

For example:

```
CLI (network-switch1) > web-cert-request-show

cert-request
----------------------------------------------------------------
-----BEGIN CERTIFICATE REQUEST-----
MIICnDCCAYQCAQEwVzELMAkGA1UEBhMCVVMxCzAJBgNVBAgMAkNBMQswCQYDVQQH
DAJTSjELMAkGA1UECgwCUE4xDTALBgNVBAsMBEVuZ2cxEjAQBgNVBAMMCWVxLWNv
bG8tMTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALMmrZ8hvZ5J+FRs
Lo1sfVtwmmLEaxyhaxD/HNVdXSRhzbQDT20+qySfOudxtWGKyCsuCFFbgMUz7rgu
H1Xle8uwPSoxgTjLGq20sgBQIfNBT5UwDLDuzUUPzMEEjFb3/9Cg1VWju2t1KPim
Gqg3rcA3PCsMeCr/q+9Gz6gfLe6Rfx91yxTA44ZWsOWnvgDdXAPfHOLZ5zBWG8a3
ohgOwMLjy21ytDTA6aR1M9I12MkJwev3t0y6n/CLp6Zigp5wXiArPPnR9sZ+E7so
MqpEzz0rjFDfrNwNAGMzT3WPcmlYRjYrUJ0QsOEQ+O1uHJaNbw1pJEmK2jm97kbk
/HvEFmMCAwEAAaAAMA0GCSqGSIb3DQEBBQUAA4IBAQCnlgEwzoesbuiCYG7HZJN/
Rxm/NcznpvJXxdlTAdzSbTWWLswrZMyX6bQqUTWEb3qvVccD4tIZShyIGiR0CpCD
22m8LD4+e6/FA6NijjanHkKsRW9Z7ka97TFpsUaH27sUTtfFDDkDImwRIGfns+nu
kTRNMuNiyC/+uHovsvCxS8is3OasQtS1lkG28sZgxisvP17qmfjlb9fQC3pcvR4t
K8GciPMUfgcIA5qLDmCZAg1A6JBMb/UHtUuEnztLrLz4qjWqJJK3pWvdLWZcKDEz
C0t5Dre9ByJ2RT75GdUq2c16xYBGAwZNCzjdhParyBnvn00Mwb6PpPmLGcBQiRNn
-----END CERTIFICATE REQUEST-----
```

- Send the CSR to your trusted CA. You can copy the `web-cert-request-show` output and send it to

the CA for signing the certificate.

You can also connect to the switch by using SFTP and copy the certificate file from /sftp/export location and send it to the CA.

If disabled, use the command `admin-sftp-modify enable` to enable SFTP.

- Upload the signed certificate, the CA root certificate, and the intermediate CA certificate (if the certificate is signed by an intermediate CA) to /sftp/import directory on the switch using SFTP.

For example, to upload the file server-cert.pem to the /sftp/import directory, follow the steps below:

```
$ sftp sftp@switch1

Password:

sftp> cd /sftp/import

sftp> put server-cert.pem
```

- Import the signed server certificate, CA root certificate, and the intermediate certificate (if available) onto the switch using the `web-cert-import` command:

```
CLI (network-admin@switch1) > web-cert-import
```

`web-cert-import`	This command imports certificates from /sftp/import directory.
`file-ca` *`file-ca-string`*	Specify the name of the CA certificate file.
`file-server` *`file-server-string`*	Specify the name of server certificate file (signed by CA).
`file-inter` *`file-inter-string`*	Specify the name of intermediate CA certificate file.

```
CLI (network-admin@switch1) > web-cert-import file-ca ca.pem file-server
server-cert.pem file-inter intermediate.pem
Successfully imported certificates.
```

- After the import is successful, enable `web-ssl` using the `admin-service-modify` command.

```
CLI (network-admin@switch) > admin-service-modify if mgmt web-ssl
```

- Download the Swagger tool and follow the general steps above to complete the configuration of REST API access.

Related Commands

- `web-cert-clear`

Use this command to delete previously generated certificates.

For example:

```
CLI (network-admin@switch1) > web-cert-clear
```

```
Successfully deleted all certificate files.
```

- `web-cert-info-show`

Use this command to display web certificate information.

```
CLI  (network-admin@switch1)  web-cert-info-show
```

`web-cert-info-show`	Displays the web certificate information.
Specify any of the following options:	
`cert-type  ca|intermediate|server`	Specify the one among the options as the certificate type.
`subject` *`subject-string`*	Specify the the subject of the certificate.
`issuer` *`issuer-string`*	Specify the issuer of the certificate.
`serial-number` *`serial-number`*	Specify the serial number of the certificate.
`valid-from` *`valid-from-string`*	Specify the time from which the certificate is valid.
`valid-to` *`valid-to-string`*	Specify the time at which the certificate expires and is no longer valid.

For example:

```
CLI  (network-admin@switch1)  web-cert-info-show

switch:         switch1
cert-type:      ca
subject:          /C=US/ST=CA/L=SJ/O=PN/OU=Engg/CN=switch1
issuer:           /C=US/ST=CA/L=SJ/O=PN/OU=Engg/CN=switch1
serial-number:  1
valid-from:     May   7 18:16:10 2019 GMT
valid-to:       May   6 18:16:10 2020 GMT
----------------------------------------
switch:         switch1
cert-type:      server
subject:          /C=US/ST=CA/L=SJ/O=PN/OU=Engg/CN=switch1
issuer:           /C=US/ST=CA/L=SJ/O=PN/OU=Engg/CN=switch1
```

Using cURL to Implement SSL Certificates

Use cURL to automate the upload of the CA root, CA intermediate, and signed switch certificates.

Run the following command for each of the PEM formatted certificates:

```
awk 'NF {sub(/\r/, ""); printf "%s\\n",$0;}'  <file-name>.pem
```

For example:

```
$ awk 'NF {sub(/\r/, ""); printf "%s\\n",$0;}'  /tmp/server-cert.pem.bkp
```

```
-----BEGIN CERTIFICATE-----
\nMIIDHDCCAgQCAQEwDQYJKoZIhvcNAQELBQAwVDELMAkGA1UEBhMCSU4xCzAJBgNV\nBAgMAktBMQwwCgYD
VQQHDANCTFIxCzAJBgNVBAoMAlBOMQwwCgYDVQQLDANFTkcx\nDzANBgNVBAMMBlNQSU5FMTAeFw0yMDA1MD
QxODM4NTZaFw0yMTA1MDQxODM4NTZa\nMFQxCzAJBgNVBAYTAklOMQswCQYDVQQIDAJLQTEMMAoGA1UEBwwD
QkxSMQswCQYD\nVQQKDAJQTjEMMAoGA1UECwwDRU5HMQ8wDQYDVQQDDAZTUElORTEwggEiMA0GCSqG\nSIb3
DQEBAQUAA4IBDwAwggEKAoIBAQCot16ddH0LNHOrWZt63FHYiArVXYIYbCDh\nWCY6MX3suoXYvKstvRgJkU
e/6G5As+vYtwRi2bDqdDsgTC5+Qo4SnjrdTcTM98F+
\n0Qzqv02c+dbzk5GclUkljqq0PHGXRPGOMhou8B/6LI9Hg5XkG+FSfaDGQTM39uj2\nzzvrMOFn96gzpTBo
h40sMoIpnKQLrWeGjlNxaBxhM342c1jn1CVmXss/uHMQeang\nsVhPTynikyxIrDwl9gh/2X1EwzVzpAnUBT
UZvJ9rgrceC9GcuGmiPZgxxSruNb0w\nK8xsyH8/hLwhK4Axgu3a+lfmmKFmSWjywmcxlmQl+jwiMPA/Ty55
AgMBAAEwDQYJ\nKoZIhvcNAQELBQADggEBAEG0D/2FcNU6Z6w/6eKbyH855kHSrJyqeU8eoCW9rnOb\nqdnA
sFX3aYwiUCjzSFXpWA3bRr3L7X0Y01x7VSvwITuDvwO43llK29rQfrSvoPiw\nf7fhU7bszlUc2GAumU9OEd
YBnSI1DzfBawUcPmbDmm+ci27k0po53KDWTbxkBIZR\n2Oh25LXkmq8ZBzE4vgS+mAw436nToazB1/vDTMWo
BuLVzOUlU8cdcjJUnJBevTbX\nThP691sHVMED8B8Fhl08BzIJmQQ9qp1tjplFq1Ea9oEFnT5U5gKvJYy48q
EPlW+r\nhRIHysvZXF/dghtrLXDMSBWLlLofUsDQsh+qLxpo1+k=\n-----END CERTIFICATE-----\n
```

Copy the output into the JSON payload.

> **Note:** The escape character syntax of \n" must be used as highlighted in red in the example below. Otherwise, the script will fail, and the certificates will not install.

```
$ curl -u network-admin:test123 http://10.100.64.5/vRest/web-certs/upload -H
"content-type:application/json" -v -X POST -d '{"cert-ca":"-----BEGIN
CERTIFICATE-----
\nMIIDHDCCAgQCAQEwDQYJKoZIhvcNAQELBQAwVDELMAkGA1UEBhMCSU4xCzAJBgNV\nBAgMAktBMQwwCgYD
VQQHDANCTFIxCzAJBgNVBAoMAlBOMQwwCgYDVQQLDANFTkcx\nDzANBgNVBAMMBlNQSU5FMTAeFw0yMDA1MD
QxODM4NTZaFw0yMTA1MDQxODM4NTZa\nMFQxCzAJBgNVBAYTAklOMQswCQYDVQQIDAJLQTEMMAoGA1UEBwwD
QkxSMQswCQYD\nVQQKDAJQTjEMMAoGA1UECwwDRU5HMQ8wDQYDVQQDDAZTUElORTEwggEiMA0GCSqG\nSIb3
DQEBAQUAA4IBDwAwggEKAoIBAQCot16ddH0LNHOrWZt63FHYiArVXYIYbCDh\nWCY6MX3suoXYvKstvRgJkU
e/6G5As+vYtwRi2bDqdDsgTC5+Qo4SnjrdTcTM98F+
\n0Qzqv02c+dbzk5GclUkljqq0PHGXRPGOMhou8B/6LI9Hg5XkG+FSfaDGQTM39uj2\nzzvrMOFn96gzpTBo
h40sMoIpnKQLrWeGjlNxaBxhM342c1jn1CVmXss/uHMQeang\nsVhPTynikyxIrDwl9gh/2X1EwzVzpAnUBT
UZvJ9rgrceC9GcuGmiPZgxxSruNb0w\nK8xsyH8/hLwhK4Axgu3a+lfmmKFmSWjywmcxlmQl+jwiMPA/Ty55
AgMBAAEwDQYJ\nKoZIhvcNAQELBQADggEBAEG0D/2FcNU6Z6w/6eKbyH855kHSrJyqeU8eoCW9rnOb\nqdnA
sFX3aYwiUCjzSFXpWA3bRr3L7X0Y01x7VSvwITuDvwO43llK29rQfrSvoPiw\nf7fhU7bszlUc2GAumU9OEd
YBnSI1DzfBawUcPmbDmm+ci27k0po53KDWTbxkBIZR\n2Oh25LXkmq8ZBzE4vgS+mAw436nToazB1/vDTMWo
BuLVzOUlU8cdcjJUnJBevTbX\nThP691sHVMED8B8Fhl08BzIJmQQ9qp1tjplFq1Ea9oEFnT5U5gKvJYy48q
EPlW+r\nhRIHysvZXF/dghtrLXDMSBWLlLofUsDQsh+qLxpo1+k=\n-----END CERTIFICATE-----\n",
"cert-server":"-----BEGIN CERTIFICATE-----
\nMIIDHDCCAgQCAQEwDQYJKoZIhvcNAQELBQAwVDELMAkGA1UEBhMCSU4xCzAJBgNV\nBAgMAktBMQwwCgYD
VQQHDANCTFIxCzAJBgNVBAoMAlBOMQwwCgYDVQQLDANFTkcx\nDzANBgNVBAMMBlNQSU5FMTAeFw0yMDA1MD
QxODM4NTZaFw0yMTA1MDQxODM4NTZa\nMFQxCzAJBgNVBAYTAklOMQswCQYDVQQIDAJLQTEMMAoGA1UEBwwD
QkxSMQswCQYD\nVQQKDAJQTjEMMAoGA1UECwwDRU5HMQ8wDQYDVQQDDAZTUElORTEwggEiMA0GCSqG\nSIb3
DQEBAQUAA4IBDwAwggEKAoIBAQCot16ddH0LNHOrWZt63FHYiArVXYIYbCDh\nWCY6MX3suoXYvKstvRgJkU
e/6G5As+vYtwRi2bDqdDsgTC5+Qo4SnjrdTcTM98F+
\n0Qzqv02c+dbzk5GclUkljqq0PHGXRPGOMhou8B/6LI9Hg5XkG+FSfaDGQTM39uj2\nzzvrMOFn96gzpTBo
h40sMoIpnKQLrWeGjlNxaBxhM342c1jn1CVmXss/uHMQeang\nsVhPTynikyxIrDwl9gh/2X1EwzVzpAnUBT
UZvJ9rgrceC9GcuGmiPZgxxSruNb0w\nK8xsyH8/hLwhK4Axgu3a+lfmmKFmSWjywmcxlmQl+jwiMPA/Ty55
AgMBAAEwDQYJ\nKoZIhvcNAQELBQADggEBAEG0D/2FcNU6Z6w/6eKbyH855kHSrJyqeU8eoCW9rnOb\nqdnA
sFX3aYwiUCjzSFXpWA3bRr3L7X0Y01x7VSvwITuDvwO43llK29rQfrSvoPiw\nf7fhU7bszlUc2GAumU9OEd
YBnSI1DzfBawUcPmbDmm+ci27k0po53KDWTbxkBIZR\n2Oh25LXkmq8ZBzE4vgS+mAw436nToazB1/vDTMWo
BuLVzOUlU8cdcjJUnJBevTbX\nThP691sHVMED8B8Fhl08BzIJmQQ9qp1tjplFq1Ea9oEFnT5U5gKvJYy48q
EPlW+r\nhRIHysvZXF/dghtrLXDMSBWLlLofUsDQsh+qLxpo1+k=\n-----END CERTIFICATE-----\n"}'

Note: Unnecessary use of -X or --request, POST is already inferred.
* Trying 10.100.64.5...
* TCP_NODELAY set
* Connected to 10.100.64.5 (10.100.64.5) port 80 (#0)
* Server auth using Basic with user 'network-admin'
> POST /vRest/web-certs/upload HTTP/1.1
> Host: 10.100.64.5
> Authorization: Basic bmV0d29yay1hZG1pbjp0ZXN0MTIz
> User-Agent: curl/7.54.0
> Accept: */*
> content-type:application/json
> Content-Length: 2348
> Expect: 100-continue
>
< HTTP/1.1 100 Continue
* We are completely uploaded and fine
< HTTP/1.1 200 OK
< Server: Apache-Coyote/1.1
< Access-Control-Allow-Origin: *
< Access-Control-Allow-Methods: GET, POST, DELETE, PUT
< Set-Cookie: JSESSIONID=C52C3170DEEAC8E4996FF428D152BF25; Path=/vRest/; HttpOnly
< Date: Tue, 05 May 2020 19:34:05 GMT
```

```
< Content-Type: application/json
< Content-Length: 162
<
* Connection #0 to host 10.100.64.5 left intact

{"result":{"status":"Success","result":[{"api.switch-
name":"local","scope":"local","status":"Success","code":0,"message":"Successfully
uploaded certificates."}]}}
```

Running Shell Commands or Scripts Using REST API

NetVisor OS version 5.1.0 provides the ability to run shell commands or scripts using REST API or through CLI commands. As a network administrator or as an admin user, you can run the scripts from the directories */opt/nvOS/bin/pn-scripts* (directory and all files are delivered as part of pn-upgrade-agent package) and */usr/bin/pn-scripts* (backup directory for running custom scripts).

The commands introduced to enable this feature are: `pn-script-show` (to view all the available scripts) and `pn-script-run name <script-name>` (to run a specified script).

Usage Guidelines:

To run a custom script,

- You should have permission to run the script.
- You should not have any duplicate scripts in the directories, */opt/nvOS/bin/pn-scripts* and */usr/bin/pn-scripts*. In case of duplicate scripts, the script from the directory, */opt/nvOS/bin/pn-scripts* takes precedence.
- It is not recommended to execute any scripts that are manually copied to the directory.

You can use the CLI commands or the vREST API to run the scripts. To run the scripts using the CLI commands, for example:

To display the available scripts using the CLI command:

```
CLI (network-admin@switch) > pn-script-show

name
-----------------
storm.c
testscript.sh
block_learning.pl
cint.sh
```

To display the scripts using vREST API:

```
$ curl  -s -u network-admin:test123 http://leo-ext-leaf1/vRest/pn-scripts
{"data":[{"name":"storm.c"},{"name":"testscript.sh"},
{"name":"block_learning.pl"},{"name":"cint.sh"}],"result":
{"status":"Success","result":[{"api.switch-
name":"local","scope":"local","status":"Success","code":0,"message":""}]}}%
```

To run the script using the CLI command:

```
CLI (network-admin@switch) > pn-script-run name testscript.sh

Executing  /opt/nvOS/bin/pn-scripts/testscript.sh:
Executing Test PN script!
```

To run the scripts using vREST API, use the following API call:

```
$ curl  -s -u network-admin:test123 -X POST http://leo-ext-leaf1/vRest/pn-
scripts/run -d '{ "name" : "testscript.sh" }' -H "Content-Type:
application/json"
{"result":{"status":"Success","result":[{"api.switch-
name":"local","scope":"local","status":"Success","code":0,"message":"Execut
ing /opt/nvOS/bin/pn-scripts/testscript.sh:\nExecuting Test PN script!
\n"}]}}%
```

To display the API docs of pn-scripts-*, use the following API call:

```
$ curl  -s -u network-admin:test123 http://leo-ext-leaf1/vRest/api-docs/pn-
scripts
{"apiVersion":"1.0.0","swaggerVersion":"1.2","basePath":"/vRest","resourceP
ath":"/pn-scripts","produces":["application/json","application/x-
ndjson"],"consumes":["application/json"],"apis":[{"path":"/pn-
scripts","operations":[{"method":"GET","summary":"","notes":"","type":"pn-
script-show-response","nickname":"showPnScripts","consumes":
["application/x-www-form-urlencoded"],"parameters":[]}]},{"path":"/pn-
scripts/run","operations":
[{"method":"POST","summary":"","notes":"","type":"result-
list","nickname":"runPnScript","consumes":
["application/json"],"parameters":
[{"name":"body","required":false,"type":"pn-script-
run","paramType":"body","allowMultiple":false}]}]},{"path":"/pn-scripts/
{name}","operations":[{"method":"GET","summary":"","notes":"","type":"pn-
script-show-response","nickname":"showPnScriptByName","consumes":
["application/x-www-form-urlencoded"],"parameters":
[{"name":"name","required":true,"type":"string","paramType":"path","allowMu
ltiple":false}]}]}],"models":{"result-list":{"id":"result-list","required":
["status","result"],"properties":{"status":{"type":"string","enum":
["Success","Failure"]},"result":{"type":"array","items":
{"$ref":"result"}}}},"result":{"id":"result","required":["api.switch-
name","scope","status","code"],"properties":{"api.switch-name":
{"type":"string"},"scope":{"type":"string","enum":
["local","fabric"]},"status":{"type":"string","enum":
["Success","Failure"]},"code":
{"type":"integer","format":"int32"},"message":{"type":"string"}}},"pn-
script-show-response":{"id":"pn-script-show-response","required":
["data","result"],"properties":{"data":{"type":"array","items":{"$ref":"pn-
script-show"}},"result":{"$ref":"result-list"}}},"pn-script-run":{"id":"pn-
script-run","description":"Run PN script","required":["name"],"properties":
{"name":{"type":"string","description":"desc=Script to execute:pattern=^[a-
zA-Z0-9_.:-]+$:pattern-help=letters, numbers, _, ., :, and -"}}},"pn-
script-show":{"id":"pn-script-show","description":"Show PN
scripts","required":["name"],"properties":{"api.switch-name":
{"type":"string"},"name":{"type":"string","description":"desc=Script to
execute"}}}}}%
```

Managing RMAs for Switches

In the event of a switch failure, you must raise an RMA request with the respective hardware vendor. After the RMA is received, do the following to reconnect the switch back in the fabric:

1. Power on the switch
2. Connect to the Console window
3. Configure the initial switch settings
4. Enable SFTP
5. Install or upgrade the software to the same version that was available on the failed switch
6. Upload or import the previously backed up Configuration file

In Arista Unified Cloud Fabric, the configurations can be on per scope basis: `Local`, `Cluster`, or `Fabric` scoped configurations. The scope defines the set of nodes participating in a transaction:

- Local — only the local node (switch) participates in the transaction
- Cluster — only two cluster peer nodes participate in the transaction
- Fabric — all nodes in the same fabric instance participate in the transaction

Based on the switch configurations, use the below processes to restore the configurations on an RMAed switch.

Cluster Re-Peer Process

When the failed switch is part of a cluster, Arista recommends the use the cluster re-peer process to restore configuration on the failed switch . Use the `fabric-join repeer-to-cluster-node` command if one of the member-nodes of the cluster is active and none of the nodes in fabric is offline, except the node that is faulty, for which the configuration needs to be restored. In this case, using the `fabric-join repeer-to-cluster-node` command is helpful because each cluster node automatically backs-up the other node's configuration. If one node in cluster fails, the above command restores the configuration from the other node. For details, see the *Performing the Cluster Re-peer Proces*s in the *Configuring High Availability* chapter.

However, if both member-nodes of a cluster fail or is being RMAed, or in the cases where multiple fabric member-nodes are offline, then use the `switch-config-import` command to restore the configuration on the cluster member-node.

Switch Config Export and Import process

In the case of a standalone switch or a non-cluster node (switch) failure, to restore back the configuration, Arista recommends the Switch Config Export-Import process. For details, see the Configuring the Export and Import of the Switch Configurations for RMA in the *Installing NetVisor OS and Initial Configuration* chapter.

> **Note**: As a best practice, Arista recommends to periodically backup the configurations of all fabric nodes (including cluster nodes) and save the configurations on to a external server.

Configuring the Export and Import of the Switch Configurations for RMA

A switch contains fabric configuration information and local configuration information such as port settings. When a switch is replaced, removed, or otherwise disrupted, you can restore the local configuration information if the configuration was saved earlier.

The information that is saved and restored on the local switch includes the following:

- VNETs with vNET manager running on the switch
- Port VLAN associations
- Network services running on the switch

To display a full list of the current configuration details on a switch, use the `running-config-show` command.

To save and restore the switch configuration, the following process is involved:

- Exporting (saving) the configuration
- Importing the saved configuration on the replaced (new) switch.

> **Note**: Arista recommends you to take periodic backup of the switch by exporting the configurations. In the event of a hardware failure, the exported configuration can be imported onto a new replaced switch.

Exporting the Configuration

To save or export the switch configuration to a file, use the following command:

```
CLI (network-admin@Leaf1) > switch-config-export
```

`switch-config-export`	Use this command to export the switch configuration
`Specify any of the following:`	
`export-file switch-config` *`export-file`*	Specify a file name for the exported configuration file. See *Note* below for more details.
`upload-server` *`upload-server-string`*	Specify this parameter to upload or save the configuration file to an SCP server (that is, on a server outside the switch). The syntax for the `upload-server` parameter is: *`<username>@<servername>:/<path to directory on server>/`*. See the example below.

The file created from the above command is a *tar* file, which includes a number of configuration files for the switch. The file is created under */nvOS/export* directory.

Configuration File Naming

If you run the `switch-config-export` command without including any additional parameters, the file created is in the format: <*switch-name-fabric-name.timestamp.tar.gz*>. However, if you mention the filename for the configuration file, for example, `switch-config-export export-file switch-config export-file` command, then the configuration file format is: filename followed by timestamp format.

For example, to export the configuration on a switch (*Leaf1*), which is part of a fabric (*multi-VRF*), use the command:

```
CLI (network-admin@Leaf1) > switch-config-export
Exported configuration to /nvOS/export/Leaf1-multi-vrf.2020-10-
11T23.46.38.tar.gz
```

where, *`Leaf1`* is the switch name and *`multi-vrf`* is the already configured fabric name.

To export the configuration specifying the filename, use the command:

```
CLI (network-admin@Leaf1) > switch-config-export export-file Leaf1
Exported configuration to /nvOS/export/Leaf1.2020-10-11T23.48.49.tar.gz
```

Exporting the Configuration to an External Server using Direct Command:

Use the `switch-config-export` command with the `upload-server` parameter to upload (save) the configuration file to an external SCP server by using the command syntax:

```
CLI (network-admin@Leaf1) > switch-config-export export-file <file-name>
upload-server <username>@<servername>:/<path to directory on server>/
```

For example, to save the configuration file with a name (here, *`leaf1-config`*), to the *root* directory of an external server (here *server-test1),* use the command and enter the password:

```
CLI (network-admin@Leaf1) > switch-config-export export-file leaf1-config
upload-server root@server-test1:/root/
server password:
Uploaded configuration to server at /root/
```

> **Note**: If you specify the `upload-server` parameter while exporting the configuration file, you do not have to follow the secure FTP (SFTP) process described below to upload the file to an external server.
>
> Arista recommends to use the `switch-config-export` command with the `upload-server` parameter as this process is faster, easier, and also safer during a switch failure.

To export the configurations from all the nodes (switches) in the fabric, use the `switch * switch-config-export upload-server` command. For example, if you have six switches in a fabric, use this command to export the configurations from all six nodes to */build/diags/<name>/tmp* directory :

```
CLI (network-admin@switch) > switch * switch-config-export upload-server
```

```
root@server:/build/diags/<name>/tmp
server password:
switch-config-export [switch-leaf1]: switch-leaf1: Uploaded configuration
to server at /build/diags/<name>/tmp
server password:
switch-config-export [switch-leaf2]: switch-leaf2: Uploaded configuration
to server at /build/diags/<name>/tmp
server password:
switch-config-export [switch-leaf3]: switch-leaf3: Uploaded configuration
to server at /build/diags/<name>/tmp
server password:
switch-config-export [switch-leaf4]: switch-leaf4: Uploaded configuration
to server at /build/diags/<name>/tmp
server password:
switch-config-export [switch-leaf4]: switch-leaf4: Uploaded configuration
to server at /build/diags/<name>/tmp
```

To view the details of the saved configuration file, use the following command:

```
CLI (network-admin@Leaf1) > switch-config-show

switch   export-file
------    ------------------------------
Leaf1     leaf1.2020-10-11T23.48.49.tar.gz
```

To view the software version installed on the switch, use the command:

```
CLI (network-admin@Leaf1) > software-show
version:            6.0.1-6000116966
```

Copying the Configuration to an External Server using SFTP

For any reason if you had missed using the `switch-config-export` command with the `upload-server` parameter to upload the file to an external server, you can still copy the configuration file to a different server using SFTP (secure FTP). You must enable the transfer protocol client first before copying the files:

> Note: Arista recommends to use the direct command as described in the *Exporting the Configuration to an External Server using Direct Command* section.

To enable SFTP, use the following command:

```
CLI (network-admin@Leaf1) > admin-sftp-modify enable
password: *******
password: *******
CLI (network-admin@Leaf1)
sftp-user: sftp
enable:    yes
```

```
CLI  (network-admin@Leaf1)
```

Then copy the configuration file to a backup server by using the following command:

```
backup-user@backupserver:~$  sftp  network-admin@Leaf1
Connecting  to  Leaf1
Password:

sftp> cd  /nvOS/export
sftp> ls
leaf1.2020-10-11T23.48.49.tar.gz

sftp> get   leaf1.2020-10-11T23.48.49.tar.gz
sftp>  bye
backup-user@backupserver:~$
```

Note: Whenever you reset a switch by using the `switch-config-reset` command, NetVisor takes a back-up the configuration file and the file is stored in the same location (*/nvOS/export*).

Importing the Saved Configuration on the Replaced (new) Switch

After the switch configuration is exported (see above section) , you can import or retrieve the configuration by using the `switch-config-import` command. Before importing the saved configuration, you must copy (upload) the file back from the backup location.

Note: Please ensure to install the same software version (that was available on the replaced switch) onto the new switch.

To copy the configuration file from the backup location, use the following command:

```
backup-user@backupserver:~$  sftp  network-admin@Leaf1
Connecting  to  Leaf1
Password:

sftp> cd  /nvOS/import
sftp> put   leaf1.2020-10-11T23.48.49.tar.gz

sftp> ls
leaf1.2020-10-11T23.48.49.tar.gz
sftp>  bye
```

This command copies the configuration *tar* file from the backup location to the */nvOS/import* directory. Once in the */nvOS/import* directory, by using the `switch-config-import` command, you can import the previously saved switch configuration.

Import the previously saved configuration on the new switch by using the `switch-config-import` command:

```
CLI (network-admin@Leaf1) > switch-config-import import-file <file-name>
```

`switch-config-import`	Use this command to import the previously saved (exported) switch configuration
`import-file` *`switch-config import-file`*	Specify the filename of the configuration file that you want to import.
`Specify Optional parameters:`	
`apply-system-config|ignore-system-config`	This is an optional parameter. Choose: • `apply-system-config`: if you want to import (along with the configuration file) the additional switch settings such as the management IP address, in-band IP address, IPv4 and IPv6 addresses, gateway, etc to the new switch. • `ignore-system-config`: if you do not want to import any additional setting, import only the configuration file. The default option is `ignore-system-config.`

For example, if you want to import the previously saved configuration file (*leaf1.2020-10-11T23.48.49.tar.gz*) on the switch *Leaf1*, use the command:

```
CLI (network-admin@Leaf1) > switch-config-import import-file leaf1.2020-10-
11T23.48.49.tar.gz

New configuration imported. Restarting nvOS...
Connected to Switch Leaf1; nvOS Identifier:0xb000011; Ver: 6.0.1-6000116966
```

Usage Guidelines for Switch Export-Import Process

- Arista recommends you to backup the configurations on all switches to an external server using the `switch-config-export upload-server` command to retrieve the configurations in case of a hardware failure.
- The configuration file must use the *`*.tar.gz`* extension to be recognized by nvOS.
- Importing the configuration file causes nvOS to restart which results in a brief interruption in traffic. Hence a scheduled maintenance window is advised.
- The `switch-config-import` command imports the last backed-up configuration and if the configuration has changed after the last backup, you must manually make those configuration changes after restoring the previously saved configuration.
- There are two options that allow you to control how the `switch-config-import` command modifies the switch:
    - `apply-system-config` — While replacing a faulty switch, apply this parameter to the `switch-config-import` command if you want all the switch settings (displayed in the `switch-setup-show` command output) including hostname, mgmt IP address, in-band IP address, etc., from the old switch to be restored onto the new switch.
    - `ignore-system-config` — Use this parameter if you do not want to restore the additional switch settings from the old switch on to the new switch. This is the default setting.
- The `switch-config-reset`, `software-upgrade`, and `fabric-upgrade-start` commands

automatically back-up the configuration before executing these commands.

- To restore the configuration on a non-cluster node, use the `switch-config-import` process described in this section.
- To restore the configuration on a cluster member-node, Arista recommends to use the cluster re-peer process. Use the `fabric-join repeer-to-cluster-node` command if one of the member-nodes of the cluster is active and none of the nodes in fabric is offline except the node that is faulty, for which the configuration needs to be restored. This is because each cluster node automatically backs-up the other node's configuration. If one node in cluster fails, the above command restores the configuration from the other node. For details, see the *Performing the Cluster Re-peer Process* in the *Configuring High Availability* chapter.

  However, if both member-nodes of a cluster fail or is being RMAed, or in the cases where multiple fabric member-nodes are offline, then use the `switch-config-import` command to restore the configuration on the cluster member-node.

Note: As a best practice, Arista recommends to periodically backup the configurations on all fabric nodes, including cluster nodes and save them to an external server.

Contacting Technical Assistance for Troubleshooting Purposes

While configuring and using the NetVisor OS fabric, you can contact the Technical Assistance Team for support. Before you contact the TAC team, gather all relevant details regarding the issue.

Use the `tech-support-show` command to view all details of the running configuration that can help with TAC troubleshooting assistance. You can save and export the log file using SFTP with TAC team.

```
CLI (network-admin@switch) > tech-support-show

Netvisor OS Command Line Interface 3.1
Connected to Switch leafsw01.xyz; nvOS Identifier:0xb000d95; Ver:
3.1.3010113816
====================  admin-service-show  ====================
if    ssh nfs web web-ssl web-ssl-port web-port snmp net-api icmp
----  --- --- --- ------- ------------ -------- ---- ------- ----
mgmt on   on  off off     443          80       on   on      on
data on   on  off off     443          80       on   on      on
====================  admin-session-timeout-show  ==============
timeout: 10m
====================  admin-sftp-show  ========================
sftp-user:     sftp
enable:        yes
====================  cluster-bringdown-show  ==================
vlag-port-staggered-interval:  0s
====================  cluster-bringup-show  ====================
state:                                  ports-enabled
l3-port-bringup-mode:                   staggered
l3-port-staggered-interval:             3s
vlag-port-bringup-mode:                 staggered
vlag-port-staggered-interval:           3s
maximum-sync-delay:                     1m
l3-to-vlag-delay:                       15s
l3-to-vlan-interface-delay:             0s
port-defer-bringup-delay:               30s
port-defer-bringup-mode:                staggered
port-defer-bringup-staggered-interval:  0s
.
<snip>
.
====================  vxlan-stats-settings-show  ===============
enable:     yes
interval:   30m
disk-space: 50M
diags@jerry:/build/diags/name$
```

Configuring Switch Ports

This section contains information about the configuration and management of switch ports.

Introduction

Switch ports are physical interfaces (typically on the front-panel of the device) that are associated with the switch forwarding engine(s) (that is, the forwarding ASICs). They can carry Layer 2 and Layer 3 traffic for data plane and control plane/management purposes.

NetVisor OS offers numerous techniques to manage and monitor the traffic that traverses switch ports. These features include port statistics, port storm control, port buffering, Class of Service (CoS), Forward Error Correction (FEC), jumbo frame support, port link status detection, and others. Below are the major features discussed in this chapter:

Port Statistics: Provide useful information such as the number of incoming and outgoing packets (unicast, multicast, or broadcast), discarded or dropped packets, and errors.

Port Speed Configuration: Enables you to configure switch ports with different speed values up to 100 Gbps (depending on the transceiver type).

Port Storm Control: This feature limits the percentage of the total available port bandwidth that can be used by broadcast, multicast, or flooded unicast traffic. It can be enabled to prevent excessive flooded traffic from degrading network performance.

Configuring Port Link Status Detection: Enables you to configure port link status detection checks on Arista switches.

Port Buffering: Allocates storage space for packets that cannot be immediately forwarded on a port.

Class of Service-based Queuing: Lets you segregate traffic into different queues based on Class of Service priority.

Forward Error Correction: A technique to detect and correct a limited number of errors in the transmitted data without the need for re-transmission.

Static Pre-emphasis: Static pre-emphasis settings help shape the outputs of ports into a well-defined, clean signal.

Link Scan Mode: The Link Scan feature detects physical link state changes. You can configure interrupt-based detection as an alternative to the software-based link scan process.

Jumbo Frame Support: Enables ports to accept and forward jumbo frames (frames with an MTU size greater than 1500 bytes)

Displaying Port Information

The NetVisor OS CLI features multiple commands that display relevant switch port information including transceiver information, physical-to-logical port mappings, Layer 2 configuration of physical ports.

Use the `port-show` command to display a host of parameters for all the ports with active links. Information displayed for each port includes the IP addresses and MAC addresses of hosts connected to that port. There can be more than one host per port if a network device such as an external switch is connected to it.

The `port-show` command also displays port status, VLAN ID, VXLAN ID, and configuration details.

```
CLI (network-admin@Leaf1) > port-show
```

`port-show`	Displays port information.
`port` *`port-list`*	Specify the switch network data port number, list of ports, or range of ports. The range of valid port numbers depends on the platform's hardware configuration.
`bezel-port` *`bezel-port-string`*	The port number printed on the device bezel above the physical port's receptacle.
`ip` *`ip-address`*	IP address of a host connected to a switch data port.
`mac` *`mac-address`*	MAC address of a host connected to a switch data port.
`vnet` *`vnet-name`*	Name of the vNET.
`bd` *`bridge-domain-name`*	Name of the bridge domain.
`vlan` *`vlan-id`*	VLAN identifier as a value between 0 and 4095.
`vxlan` *`vxlan-id`*	VXLAN identifier as a value between 0 and 16777215.
`hostname` *`hostname`*	Name of a host connected to a switch data port.
`status phy-up|up|disabled|hw-nat-loop|mirror-loop|mirror-to|inuse| PN-switch|PN-fabric|PN-other|PN-cluster|PN-internal|PN-hypervisor| PN-guest|snmp-host|host|uplink| drop-pkts|no-pktin|no-fwd|no-flood| STP-BPDUs|LLDP|trunk|l3-port| remote-l3-port|vdp|dhcp| dhcpsvrblocked|no-BPDU|LACP-PDUs| vlag-active|vlag-blocked|stp-edge-`	Status of a switch data port.

`port|LACP-wait|adjacency-wait| adjacency-check|vlag-wait| multicast-router|host-disabled| loop|vxlan-loopback|vlan-up|vle| vle-wait|phy-down|down|enabled|err-disabled|err-bpdu-guard|mac-violation|stp-bpdu-guard|stp-root-guard|defer-bringup-wait`	
`loop-vlans vlan-list`	VLANs looping on the ports.
`rem-ip` *`ip-address`*	IP address of the remote switch.
`lport` *`lport-number`*	Logical port number on the switch.
`rport` *`rport-number`*	Port number on the remote switch.
`config fd|hd|10m|100m|1g|2.5g|10g| 25g|40g|50g|100g|loopback|mirror-only|autoneg|fiber|copper|qos| jumbo|pause|asymetric-pause|vxlan-termination|no-local-switching|fec`	Configured port features.
`description` *`description-string`*	Description of the port.
`trunk` *`trunk-name`*	ID of the trunk that a switch data port is a member of.
`recover-time duration: #d#h#m#s`	Time left for recovery from err-disable state.
`err-bpduguard` *`err-bpduguard-number`*	Total count of err-disables by BPDU guard.
`err-maclimit` *`err-maclimit-number`*	Total count of err-disables by MAC address limit.
`hide-connections`	Specifies whether the connections on switch data ports should be displayed.
`link-detail`	Details of the link such as PHY, link, status etc.

For example:

```
CLI (network-admin@switch*) > port-show port 2 layout vertical
switch:           switch
port:             2
bezel-port:       2
ip:               200.1.23.2
mac:              66:0e:94:b7:90:6f
vlan:             4091
hostname:         taurus-dev-spine2
status:           up,PN-switch,PN-other,STP-BPDUs,LLDP,l3-port,remote-l3-port,vlan-up
config:           fd,10g
```

Displaying Port Numbering

Bezel interface numbers are the labels for ports on the face plate of a switch. As NetVisor OS supports flexing of ports (breaking a high speed port to lower speed ports) the logical port numbers in the software may not match the bezel interface numbers. The bezel interface numbers may deviate from the logical port numbers in order to represent each logical port uniquely. To display the mapping of logical ports to bezel interface numbers on NetVisor OS platforms, use the command:

```
CLI (network-admin@switch) > bezel-portmap-show
```

`bezel-portmap-show`	Display the mapping of logical ports to bezel interface numbers.
`port` *`port-number`*	Logical port number.
`bezel-intf` *`bezel-intf-string`*	Bezel interface number.

```
CLI (network-admin@switch) > bezel-portmap-show
switch port bezel-intf
------ ---- ----------
switch 1    1
switch 2    1.2
switch 3    1.3
switch 4    1.4
switch 5    2
switch 6    3
switch 7    3.2
switch 8    3.3
switch 9    3.4
switch 10   4
switch 11   4.2
switch 12   4.3
switch 13   4.4
```

As seen from the above output, the bezel interface numbers 1, 1.2, 1.3, and 1.4 represent a physical port's receptacle connected to a breakout cable with four transceivers. Such transceivers are mapped to logical ports 1, 2, 3 and 4 respectively. For example, it can be a 40 GbE QSFP+ receptacle that is divided into four 10 GbE SFP+ ports or a 100 GbE QSFP28 receptacle that is divided into four 25 GbE SFP28 ports. Also, you can infer from the output that bezel interface 2 is not connected to a breakout cable as there is only a single corresponding logical interface.

Displaying Physical Port Information

You can display physical port information by using the `port-phy-show` command.

The command displays port state and mode, speed and auto-negotiation settings, Ethernet mode, default VLAN, maximum frame size, and status information.

```
CLI (network-admin@switch) > port-phy-show
```

`port-phy-show`	Displays physical port information.
`port` *`port-list`*	Specify the switch data port number, list of ports, or range of ports. Port numbers must be in the range of 1-64.
`bezel-port` *`bezel-port-string`*	The bezel port number.
`state off|admin-down|adminpwrdown| bist|dfe-tuning|down|local-fault| partially-up|remote-fault|up|failed`	State of physical ports.
`autoneg` *`autoneg-string`*	Auto-negotiation setting of physical ports
`speed` *`speed-number`*	Speed of physical ports.
`eth-mode unconfigured|disabled| 1000base-x|sgmii|1000base-kx| 2500base-x|6Gbase-kr|6Gbase-cr| 10Gbase-kr|10Gbase-cr|10Gbase-sr| AN-73|Xaui|10Gbase-kx4|10Gbase-cx4| 24Gbase-kr4|24Gbase-cr4|40Gbase-kr4|xlaui|40Gbase-cr4|40Gbase-sr4| no-phy-present|MII|gmii|TBI|10G-XGMII|1G-RGMII|sfi|xfi|cx|kr4|sr4| lr4|er4|cr4|cx4|kx4|caui4|kr2|er2| cr2|sr2|lr2|xlaui2|cx2|sr|lr|er|cr| kx|kr|zr|base-x|sys-default|caui| aoc|xlppi|psm4|base_t|base_cx| base_lx|base_sx`	Ethernet mode.
`max-frame` *`max-frame-number`*	The Maximum Transmission Unit (MTU) for the port.
`def-vlan` *`def-vlan-number`*	Default VLAN identifier of the physical port.

```
CLI (network-admin@switch) > port-phy-show format all
port  bezel-port  state  autoneg  speed   eth-mode  max-frame  def-vlan
----  ----------  -----  -------  ------  --------  ---------  --------
1     1           up     none     100000  caui4     9412       1
5     5           down   none     100000  caui4     1540       1
9     9           down   none     100000  caui4     1540       1
13    13          down   none     100000  caui4     1540       1
17    17          up     none     100000  caui4     1540       4091
21    21          down   none     100000  caui4     1540       1
25    25          down   none     100000  caui4     1540       1
```

Displaying Transceiver Information

To display information on the transceivers connected to a switch, use the `port-xcvr-show` command. The displayed information includes vendor name, part number, and the serial number of the transceiver.

`CLI (network-admin@switch) > port-xcvr-show`

`port-xcvr-show`	Display port transceiver information.
`port` *`port-list`*	List of ports.
`bezel-port` *`bezel-port-string`*	The bezel port number.
`vendor-name` *`vendor-name-string`*	Vendor name of the transceiver.
`part-number` *`part-number-string`*	Part number of the transceiver.
`serial-number` *`serial-number-string`*	Serial number of the transceiver.
`type unknown|sfp|qsfp`	Type of the transceiver.
`eth-mode unconfigured|disabled| 1000base-x|sgmii|1000base-kx| 2500base-x|6Gbase-kr|6Gbase-cr| 10Gbase-kr|10Gbase-cr|10Gbase-sr| AN-73|Xaui|10Gbase-kx4|10Gbase-cx4| 24Gbase-kr4|24Gbase-cr4|40Gbase-kr4|xlaui|40Gbase-cr4|40Gbase-sr4| no-phy-present|MII|gmii|TBI|10G-XGMII|1G-RGMII|sfi|xfi|cx|kr4|sr4| lr4|er4|cr4|cx4|kx4|caui4|kr2|er2| cr2|sr2|lr2|xlaui2|cx2|sr|lr|er|cr| kx|kr|zr|base-x|sys-default|caui| aoc|xlppi|psm4|base_t|base_cx| base_lx|base_sx`	Transceiver Ethernet mode.
`cbl-len cbl-len-number`	Active Optical Cable (AOC) or Direct Attach Cable (DAC) length.
`temp[C] temp[C]-string`	Transceiver temperature in degrees Centigrade.
`vcc33[V] vcc33[V]-string`	3.3V supply voltage.
`tx-bias1[mA] tx-bias1[mA]-string`	Transmit bias current (mA).
`tx-pwr[dBm] tx-pwr[dBm]-`*`string`*	Transmit power (dBm).
`rx-pwr[dBm] rx-pwr[dBm]-`*`string`*	Receive power (dBm).

```
CLI (network-admin@switch) > port-xcvr-show
switch port bezel-port vendor-name       part-number   serial-number
------ ---- ---------- ---------------- ------------ --------------
switch 25   25         Amphenol         123456789    ABC00000000DEF
switch 27   27         Amphenol         123456789    ABC00000000DEF
switch 28   28         Amphenol         123456789    ABC00000000DEF
switch 45   45         PluribusNetworks 123456780    HIJ00001
switch 46   46         PluribusNetworks 123456780    HIJ00001
switch 47   47         PluribusNetworks 123456780    HIJ00001
```

```
switch 48   48         PluribusNetworks 123456780     HIJ00001
switch 49   49         Amphenol         123456789     ABC00000000DEF
switch 50   50         Amphenol         123456789     ABC00000000DEF
switch 51   51         Amphenol         123456789     ABC00000000DEF
switch 52   52         Amphenol         123456789     ABC00000000DEF
switch 53   53         PluribusNetworks 123456780     HIJ00001
switch 54   54         PluribusNetworks 123456780     HIJ00001
switch 55   55         PluribusNetworks 123456780     HIJ00001
switch 56   56         PluribusNetworks 123456780     HIJ00001
switch 65   65         PluribusNetworks 123456780     HIJ00001
switch 66   66         PluribusNetworks 123456780     HIJ00001
switch 67   67         PluribusNetworks 123456780     HIJ00001
switch 68   68         PluribusNetworks 123456780     HIJ00001
```

Displaying Port Statistics

The port statistics information is useful for understanding the data traffic on switch ports. Port statistics information includes the number of incoming and outgoing packets (classified as unicast, multicast, and broadcast), bytes dropped, bytes discarded, Head-End-Replicated (HER) packets, and errors. Use the `port-stats-show` command to display the port statistics for all active ports on a switch:

```
CLI (network-admin@switch) > port-stats-show
```

`port-stats-show`	Displays port statistics.
`time date/time: yyyy-mm-ddThh:mm:ss`	Specify a time for the packet count statistics using the timestamp format yyyy-mm-ddThh:mm:ss.
`start-time` *`start-time`*	Specify a start time for the packet count statistics using the timestamp format yyyy-mm-ddThh:mm:ss.
`end-time` *`end-time`*	Specify an end time for the packet count statistics using the timestamp format YYYY-MM-DDTHH:MM:SS
`duration` *`duration`*	Specify the duration in seconds.
`interval duration: #d#h#m#s`	Specify the interval between statistics collection.
`since-start|no-since-start`	Specify if the statistics are collected from the start of collecting statistics.
`older-than duration: #d#h#m#s`	Specify if the statistics are older than the duration in days, hours, minutes, and seconds.
`within-last duration: #d#h#m#s`	Specify if the statistics are within the duration in days, hours, minutes, and seconds.
`port` *`port-list`*	Specify one or more switch network data port numbers. Multiple ports can be specified as a comma-separated list of numbers or a range (-).
`bezel-port` *`bezel-port-string`*	Specify the bezel ports.
`description` *`description`*	Displays the port description.
`counter` *`counter-number`*	Displays the counter number.
`ibytes ibytes-number`	Displays the incoming number of bytes.
`ibits` *`ibits-number`*	Displays the number of incoming bits.
`iUpkts` *`iUpkts-number`*	Displays the number of incoming unicast packets.

iBpkts *iBpkts-number*	Displays the number of incoming broadcast packets.
iMpkts *iMpkts-number*	Displays the number of incoming multicast packets.
iPauseFs *iPauseFs-number*	Displays the number of incoming pause frames.
iCongDrops *iCongDrops-number*	Displays the number of incoming packets dropped due to congestion.
idiscards *idiscards-number*	Displays the number of incoming packets discarded.
ierrs *ierrs-number*	Displays the number of incoming packets with errors.
obytes *obytes-number*	Displays the number of outgoing bytes.
oUpkts *oUpkts-number*	Displays the number of outgoing unicast packets.
oBpkts *oBpkts-number*	Displays the number of outgoing broadcast packets.
oMpkts *oMpkts-number*	Displays the number of outgoing multicast packets.
oPauseFs *oPauseFs-number*	Displays the number of outgoing pause frames.
oCongDrops *oCongDrops*-number	Displays the number of outgoing packets dropped due to congestion.
odiscards *odiscards-number*	Displays the number of outgoing discarded packets.
oerrs *oerrs-number*	Displays the number of outgoing packets with errors.
mtu-errs *mtu-errs-number*	Displays the number of MTU errors.
HER-packets *HER-pts-number*	Displays the number of Head End Replicated (HER) packets.
HER-bytes *HER-bytes-number*	Displays the number of Head End Replicated (HER) bytes.
HER-bits HER-bits-number	Displays the number of Head End Replicated (HER) bits.

```
CLI (network-admin@switch) > port-stats-show port 1 format all layout
vertical
time:         22:45:12
port:         1
bezel-port:   1
description:
counter:      0
ibytes:       170K
ibits:        1.39M
iUpkts:       165
```

```
iBpkts:       22
iMpkts:       402
iPauseFs:     0
iCongDrops:   0
idiscards:    2
ierrs:        46
obytes:       144K
obits:        1.18M
oUpkts:       158
oBpkts:       4
oMpkts:       249
oPauseFs:     0
oCongDrops:   0
odiscards:    48
oerrs:        0
mtu-errs:     0
HER-pkts:     0
HER-bytes:    0
HER-bits:     0
```

Configuring Port Speed

NetVisor OS enables you to configure switch ports with different speed values up to 100 Gbps depending on the specific transceiver. For example, you must use SFP transceivers for ports configured with speeds of 1 Gbps, and SFP+ or QSFP transceivers for ports configured with speeds of 10 Gbps or higher.

With a transceiver that is capable of breakout configuration, you can divide a 40 Gigabit Ethernet (GbE) port into four logical 10 GbE ports or a 100 Gigabit Ethernet (GbE) port into four logical 25 GbE ports. You must first disable a port before creating this configuration.

Use the `port-config-modify` command to configure the desired throughput on a port.

```
CLI (network-admin@switch) > port-config-modify
```

`port-config-modify`	Modify port configuration.
Specify one among the two following parameters:	
`port` *`port-list`*	Specify a single port or a list of ports. **Note:** If more than one port is specified, the list must be comma-separated without spaces.
`description` *`description-string`*	Specify the port description.
Specify any of the following options:	
`speed disable|10m|100m|1g|2.5g|10g|25g|40g|50g|100g`	Specify the port speed. Ports configured for speeds of 1G must be equipped with SFP transceivers. Ports configured for 10G or higher must be equipped with SFP+ or QSFP+ transceivers.
`egress-rate-limit unlimited`	Specify an egress rate limit for the port.
`eth-mode 1000base-x|sgmii|xlaui|gmii|sfi|xfi|cx|kr4|sr4|lr4|er4|cr4|cx4|kx4|caui4|kr2|er2|cr2|sr2|lr2|xlaui2|cx2|sr|lr|er|cr|kx|kr|zr|base-x|sys-default|caui|aoc|xlppi|psm4|base_t|base_cx|base_lx|base_sx`	Select one of the ethernet modes from the available options.
`autoneg|no-autoneg`	Specify if the port auto-negotiates the port speed with a peer.
`jumbo|no-jumbo`	Specify if the port forwards jumbo frames. Jumbo frames are Ethernet frames with more than 1500 bytes of payload.
`enable|disable`	Specify if the port is enabled and forwards or drops frames.
`lacp-priority` *`integer`*	Specify the LACP priority for the port. If you specify a low value for the priority, the port has a

	higher priority. This is a value between 1 and 65535.
`reflect|no-reflect`	Specify if the port reflects frames received for loopback testing.
`edge-switch|no-edge-switch`	Specify if the port connects to another Arista Networks device or is an uplink to a third-party switch or host.
`pause|no-pause`	Specify if you want to pause traffic on the port.
`description` *`description-string`*	Specify a description for the port.
`loopback|no-loopback`	Specify to use loopback or no loopback.
`vxlan-termination|no-vxlan-termination`	Specify if a VXLAN terminates on a port.
`mirror-only|no-mirror-receive-only`	Specify if the port receives mirrored traffic only.
`port-mac-address mac-address`	Specify the MAC address for the physical port.
`send-port` *`send-port-number`*	Specify the port to send traffic.
`loop-vlans vlan-list`	Specifies VLANs that are looping on the network.
`routing|no-routing`	Specify if the port is participating in routing on the switch.
`defer-bringup|no-defer-bringup`	Specify if you want to delay port activation.
`host-enable|host-disable`	Specify if you want host-facing ports enabled or disabled.
`crc-check-enable|crc-check-disable`	Specify if you want CRC checking performed by the port. This is applicable to a switch in Virtual Wire mode. The default value is `crc-check-disable`.
`dscp-map dscp-map name|none`	Specify a DSCP map name to enable on port.
`local-switching|no-local-switching`	Specify if local switching is allowed on the port.
`allowed-tpid vlan|q-in-q|q-in-q-old`	Specify the allowed TPID in addition to 0x8100 on VLAN header
`fabric-guard|no-fabric-guard`	Specify if you want loop detection enabled on the port.
`fec|no-fec|fec-auto`	Specify to enable port forward correction error (FEC) mode.

For example, to configure a 40 GbE port as four 10 GbE ports, use the following command:

```
CLI (network-admin@Leaf1) > port-config-modify port 49-52 speed 10g
```

To unify the four logical ports and to set the port configuration back to 40 GbE, use the following command:

```
CLI (network-admin@Leaf1) > port-config-modify port 49 speed 40g
```

Configuring Port Storm Control

A traffic storm occurs when packets flood the LAN, creating excessive load and degrading network performance. When enabled, port storm control discards the excess unicast, multicast, or broadcast traffic that is detected on a port.

Use the `port-storm-control-modify` command to modify the percentage of total available bandwidth that can be used by broadcast, multicast, or unicast traffic.

```
CLI (network-admin@switch) > port-storm-control-modify
```

`port-storm-control-modify`	Modify port storm control configuration.
`port` *`port-list`*	Specify a single port or a list of ports.
`unknown-ucast-level` *`unknown-ucast-level-string`*	Specify the bandwidth to be allotted for unknown unicast as a percentage of the port link speed. The default is 30%.
`unknown-mcast-level` *`unknown-mcast-level-string`*	Specify the bandwidth to be allotted for unknown multicast as a percentage of the port link speed. The default is 30%.
`broadcast-level` *`broadcast-level-string`*	Specify the bandwidth to be allotted for broadcast as a percentage of the port link speed. The default is 30%.

To set the available bandwidth for broadcast traffic to 10% of port link speed on port 5, use the command:

```
CLI (network-admin@switch) > port-storm-control-modify port 5 broadcast-level 10
```

Use the `port-storm-control-show` command to display the configuration:

```
CLI (network-admin@switch) > port-storm-control-show
switch port speed unknown-ucast-level unknown-mcast-level broadcast-level
------ ---- ----- ------------------- ------------------- ---------------
Leaf1  1    100g  30%                 30%                 30%
Leaf2  2    25g   30%                 30%                 30%
Leaf3  3    25g   30%                 30%                 30%
Leaf4  4    25g   30%                 30%                 30%
Leaf5  5    100g  30%                 30%                 10%
Leaf6  6    25g   30%                 30%                 30%
```

This setting discards broadcast traffic if it exceeds 10% of the link speed on port 5.

Configuring 10/100M Override Using Bel-Fuse SFP-1GBT-05 on F9372-X Platforms

Note: This feature is applicable for F9372-X or AS5812-54X platforms.

Auto-negotiation function between two connected devices stipulates a set of shared parameters - link speed, duplex mode, and flow control. The F9372-X or AS5812-54X platform, with the Bel-Fuse SFP-1GBT-05 copper transceiver plugged into one of its ports, can support legacy 10M/100M devices that do not have auto-negotiation capability. As the SFP-1GBT-05 transceiver performs auto-negotiation by default, this function should be disabled on the F9372-X switch through nvOS. An operating link speed of either 10M or 100M can be configured. With the exception of F9372-X, the 10/100M override capability is not available on other switch platforms and is supported only on the SFP-1GBT-05 transceiver.

Use the command below for disabling auto-negotiation and overriding the link speed on F9372-X platform. Enter the *port number* on which the transceiver is connected in the command:

```
CLI (network-admin@Leaf1) > port-config-modify port <port number> speed
<10m|100m> no-autoneg
```

`port-config-modify`	Use this command to update port configuration.
`port <port-list>`	Specify the ports that need to be configured as a list separated by commas. **Note**: For 10/100M override, specify the port number on which the transceiver is connected.
`Specify one or more of the following options:`	
`speed <disable|10m|100m|1g>`	Specify one among the options as the speed at which the port should operate. **Note**: specify the speed as either 10m or 100m for legacy device support.
`autoneg|no-autoneg`	Specify either of the options to enable auto-negotiation or to disable it. **Note**: auto-negotiation has to be disabled to achieve 10/100M override.

For example, to disable auto-negotiation on port 38 and set a speed of 10M, use the command below:

```
CLI (network-admin@Leaf1) port-config-modify port 38 speed 10m no-autoneg
```

However, the device do not get reprogrammed upon reinsertion. To recover from this, remove and re-add the 10M/100M override port configuration. [PR30110]

Configuring Fabric and vRouter Communication through KNET-vNICs

Note: This feature is available only on Dell and whitebox platforms that do not have rear-facing NICs.

From NetVisor OS 6.0.1 release onwards, NetVisor provides support for using KNET-vNIC as the OVS uplink from the CPU to OVS. The KNET exposes a standard network interface that is connected as an uplink port to OVS for forwarding fabric, cluster, or vRouter packets. This implementation avoids copying of traffic between OVS and nvOSd and ensures direct transmission of fabric, cluster, and vRouter NIC packets between OVS and the CPU. Therefore, communication through KNET-vNICs results in improved CPU datapath performance.

To support cluster, fabric, and vRouter traffic using KNET-vNIC, you must enable KNET on Whitebox platforms that do not have rear-facing NICs by using the `system-settings-modify` command.

To enable or disable KNET, use the command:

```
CLI (network-admin@switch1) > system-settings-modify optimize-datapath disable|cluster-only|all
```

`optimize-datapath disable|cluster-only|all`	Specify the datapath optimization for cluster, fabric, and vRouter communication: • `disable`: disables datapath optimization. • `cluster-only`: enables datapath optimization for cluster-only, where cluster traffic is redirected to cluster 4094 vNIC. • `all`: enables datapath optimization for fabric and data traffic. The default value is `all`.

To view the existing configuration on a local switch, use the `system-settings-show` command:

```
CLI (network-admin@switch1) > system-settings-show format optimize-datapath
optimize-datapath:  all
```

To view the settings on all the switches in the fabric, use the `system-settings-show` command prefixed with `switch *` as below:

```
CLI (network-admin@switch1) > switch * system-settings-show format
optimize-datapath
optimize-datapath:  all
optimize-datapath:  all
optimize-datapath: all
optimize-datapath:  all
optimize-datapath:  all
optimize-datapath:  all
optimize-datapath:  all
```

Note: Arista recommends rebooting the switch by using the `switch-reboot` command after running the `system-settings-modify optimize-datapath` command.

Using Port Buffering

Port buffering allocates storage for packets that cannot be immediately forwarded or processed on a port. To display port buffer information, use the `port-buffer-show` command. This command displays ingress and egress buffer utilization for each port.

```
CLI (network-admin@switch) > port-buffer-show
```

`port-buffer-show`	Display port buffer information.
`time date/time: yyyy-mm-ddThh:mm:ss`	Time to start statistics collection.
`start-time date/time: yyyy-mm-ddThh:mm:ss`	Start time of statistics collection
`end-time date/time: yyyy-mm-ddThh:mm:ss`	End time of statistics collection
`duration duration: #d#h#m#s`	Duration of statistics collection
`interval duration: #d#h#m#s`	Interval between statistics collection.
`since-start`	Specify to display port buffering since start.
`older-than duration: #d#h#m#s`	Specify an older-than time to display port buffering.
`within-last duration: #d#h#m#s`	Specify a within-last time to display port buffering.
`port port-list`	The list of ports.

```
CLI (network-admin@switch) > port-buffer-show
switch port ingress-used-buf ingress-used-buf-val egress-used-buf egress-used-buf-val
------ ---- ---------------- -------------------- --------------- -------------------
Leaf1  25   0%               0                    0%              0
Leaf1  27   0%               0                    0%              0
Leaf1  28   0%               0                    0%              0
Leaf1  45   0%               0                    0%              0
Leaf1  49   0%               0                    0%              0
Leaf1  53   0%               0                    0%              0
Leaf1  65   0%               0                    0%              0
Leaf1  69   0%               0                    0%              0
Leaf1  70   0%               0                    0%              0
Leaf1  71   0%               0                    0%              0
Leaf1  72   0%               0                    0%              0
```

Use the `port-buffer-settings-modify` command to enable the collection of port buffer information and to modify the logging interval and buffer size for the command output above.

```
CLI (network-admin@switch) > port-buffer-settings-modify
```

`port-buffer-settings-modify`	Modify port buffer settings.
`enable|disable`	Specify if you want to enable or disable the collection of port buffer information.
`interval duration: #d#h#m#s`	Specify the interval for the logging of port buffer information.

`disk-space` *`disk-space-number`*	Specify the amount of memory to be allotted for the collection of port buffering information. The default value is 50MB.

For example, to set an interval of 2 minutes for logging port buffer information, use the command:

```
CLI (network-admin@switch) > port-buffer-settings-modify interval 2m
```

To display the port buffering settings, use the `port-buffer-settings-show` command:

```
CLI (network-admin@switch) > port-buffer-settings-show
switch: switch
enable: yes
interval: 2m
disk-space: 50M
```

Configuring Queues 8 and 9 for Control Traffic

During network traffic congestion, if all traffic is dispatched with the same priority, mission critical traffic may get dropped with the same probability as lower priority traffic. To mitigate this issue, NetVisor OS allows you to configure traffic prioritization on a per port basis through Quality of Service (QoS) queues. QoS queues allow the switch to differentiate and prioritize various traffic types and avoid random loss of data. The hardware supports Strict Priority (SP) and Deficit Weighted Round Robin (DWRR) scheduling for the QoS queues.

NetVisor OS assigns default weights to each queue. However, you can view and modify these weights to suit your requirements. You can also configure data rate limits, minimum guaranteed bandwidth, and maximum bandwidth on a queue basis.

Note: This feature is available only on NRU01, NRU02, NRU03, and NRU-S0301 platforms.

NetVisor OS version 6.0.1 introduces queues 8 and 9 to isolate user traffic from internal control traffic. In previous versions of NetVisor OS, fabric and control traffic used queues 6 and 7 which were also used by user traffic. This release eliminates the contention between user traffic and control traffic, as fabric and control messages are now moved from queues 6 and 7 to queues 8 and 9. The queues 8 and 9 are mapped to system vFlow classes `control2` and `control3` respectively. This feature provides improved stability and resiliency to the network especially in cases where the network is overloaded.

To enable queues 8 and 9, use the command:

```
CLI (network-admin@switch) > system-settings-modify cos8-and-9-queue
!!!! Please reboot the system for cos 8 and 9 queue setting to take effect
correctly !!!!
```

Note: You must reboot the system for the queues 8 and 9 to become operational.

To view all flow classes, use the command:

```
CLI (network-admin@switch) > vflow-class-show
name           scope   type    priority cos
-------------  ------  ------  -------- ---
meter          fabric system 0        0
class0         fabric system 0        0
class1         fabric system 1        1
class2         fabric system 2        2
class3         fabric system 3        3
class4         fabric system 4        3
class5         fabric system 5        4
class6         fabric system 6        4
class7         fabric system 7        5
guaranteed_bw  fabric system 9        6
class8         fabric system 10       6
lossless       fabric system 10       6
control        fabric system 11       7
control2       fabric system 12       8
control3       fabric system 13       9
```

Configuring Port Rate Limit

By using the command `port-cos-rate-setting-modify`, you can stipulate a port speed in packets per second (pps) to regulate the traffic corresponding to each QoS queue. This command is used for rate limiting the traffic of the port queues.

```
CLI (network-admin@switch) > port-cos-rate-setting-modify
```

`port-cos-rate-setting-modify`	Modify port queue rate limit.
`port control-port|data-port|span-ports`	Specify the port type to set QoS rate limit.
`cos0-rate unlimited|0..10000000`	Specify the rate limit (pps) for the queue.
`cos1-rate unlimited|0..10000000`	
`cos2-rate unlimited|0..10000000`	
`cos3-rate unlimited|0..10000000`	
`cos4-rate unlimited|0..10000000`	
`cos5-rate unlimited|0..10000000`	
`cos6-rate unlimited|0..10000000`	
`cos7-rate unlimited|0..10000000`	
`cos8-rate unlimited|0..10000000`	
`cos9-rate unlimited|0..10000000`	

For example:

```
CLI (network-admin@switch) > port-cos-rate-setting-modify port data-port
cos0-rate 50000
```

To display the port rate limit configuration, use the command:

```
CLI  (network-admin@switch)  >  port-cos-rate-setting-show
port             ports  cos0-rate(pps)  cos1-rate(pps)  cos2-rate(pps)  cos3-rate(pps)  cos4-rate(pps)  cos5-rate(pps)
cos6-rate(pps)  cos7-rate(pps)  cos8-rate(pps)  cos9-rate(pps)
------------  -----  --------------  --------------  --------------  --------------  --------------  --------------
--------------  --------------  --------------  --------------
control-port 0     100000          100000          100000          100000          100000          100000
 100000          100000          100000          100000
data-port     129    50000           100000          100000          100000          100000          100000
 100000          100000          100000          100000
span-ports    130   100000          100000          100000          100000          100000          100000
 100000          100000          100000          100000
```

The output displays the port rate limit configuration for control ports, data ports, and span ports. The `data` ports and `span` ports are platform specific. The default rate for each queue is 100000 packets per second. The queues 8 and 9 must be enabled for the columns `cos8-rate(pps)` and `cos9-rate(pps)` to be displayed.

Note:

- The `port-cos-rate-setting-modify` and `port-cos-rate-setting-show` commands are available only when CPTP is disabled
- Queues 8 and 9 are available only on NRU01, NRU02, NRU03, and NRU-S0301 platforms.

Configuring Minimum and Maximum Bandwidth on Ports

In older releases, NetVisor OS allowed rate limiting only for CPU facing (PCIe, data and span) ports. The current version of NetVisor OS support bandwidth guarantees on switch ports. You can configure bandwidth guarantees at egress queue level and manage prioritized traffic. Use this feature for setting Service Level Agreements (SLAs). In addition to the support for configuring maximum bandwidth policing at the vFlow level, you can also set a guaranteed minimum bandwidth.

NetVisor OS supports the configuration of minimum and maximum bandwidth at a per port level. You can configure bandwidth guarantees as a percentage of port speed, and NetVisor OS determines the data rate internally upon command execution. Additionally, when you update a port speed, the port configuration internally re-adjusts the minimum or maximum bandwidths on the applicable ports. However, when you logically divide a breakout cable into multiple ports of lower bandwidth, you need to adjust the port queue bandwidths manually.

Use the `port-cos-bw-modify` command to configure minimum and maximum bandwidth:

```
CLI (network-admin@switch) > port-cos-bw-modify
```

`port-cos-bw-modify`	Modify port CoS bandwidth settings.
`cos` *`integer`*	Specify the CoS priority between 0 and 9.
`port` *`port-list`*	Specify the physical port(s).
`min-bw-guarantee` *`min-bw-guarantee-string`*	Specify the minimum bandwidth as a percentage.
`max-bw-limit` *`max-bw-limit-string`*	Specify the maximum bandwidth as a percentage.
`weight priority|no-priority`	Specify to enable or disable weight scheduling after the bandwidth guarantee is met.

For example, to configure a minimum bandwidth guarantee of 10 percent on ports 2-5 with a QoS queue of 5, use the command:

```
CLI (network-admin@switch) > port-cos-bw-modify port 2-5 cos 5 min-bw-guarantee 10
```

Use the `port-cos-bw-show` command to view the configuration.

```
CLI (network-admin@Spine1) > port-cos-bw-show
```

`cos` *`integer`*	Specify the CoS priority.
`port` *`port-list`*	Specify the physical port(s).

```
CLI (network-admin@switch) > port-cos-bw-show
switch   cos port   min-bw-guarantee max-bw-limit weight
-------- --- ------ ---------------- ------------ ------
switch   0   1-72   0%               100%         16
switch   1   1-72   0%               100%         32
switch   2   1-72   0%               100%         32
switch   3   1-72   0%               100%         32
switch   4   1-72   0%               100%         32
switch   5   1,6-72 0%               100%         32
switch   5   2-5    10%              100%         32
switch   6   1-72   0%               100%         64
switch   7   1-72   0%               100%         127
```

Note: The `port-cos-bw-show` command displays only modified port configurations. Ports not displayed in the show command output default to the settings of 100% link capacity, and no minimum guarantee for each QoS queue.

Configuring a maximum bandwidth limit helps in rate limiting or shaping the traffic on the egress queue. For example, to configure a minimum bandwidth of 20% and a maximum bandwidth of 80% on ports 11-13 for queue 4, use the command:

```
CLI (network-admin@switch) > port-cos-bw-modify port 11-13 cos 4 min-bw-
guarantee 20 max-bw-limit 80

CLI (network-admin@switch) > port-cos-bw-show
switch   cos port   min-bw-guarantee max-bw-limit weight
-------- --- ------ ---------------- ------------ ------
switch   0   1-72   0%               100%         16
switch   1   1-72   0%               100%         32
switch   2   1-72   0%               100%         32
switch   3   1-72   0%               100%         32
switch   4   1-72   0%               100%         32
switch   4   11-13  20%              80%          32
switch   5   1,6-72 0%               100%         32
switch   5   2-5    10%              100%         32
switch   6   1-72   0%               100%         64
switch   7   1-72   0%               100%         127
```

Changing the port settings to new values overrides the previous settings.

Configuring QoS Queue Weights

NetVisor OS automatically weights QoS queues in accordance with the minimum bandwidth guarantees. When you configure minimum bandwidth for a queue, the remaining bandwidth is assigned to the rest of the queues in the same ratio as the minimum bandwidth. For information on configuring minimum bandwidth, see *Configuring Minimum and Maximum Bandwidth on Ports* section.

When you configure a minimum bandwidth without specifying a weight value, the weight for the port and queue is automatically set on a scale of 1 to 100. For example, if you configure queue 1 with a minimum bandwidth of 10% and queue 2 with 30%, the weights are set to 10 and 30, respectively.

If auto-configuration of weights is disabled, you can also assign desired weight values to the queues. When you configure queue weights manually, the bandwidth that remains after meeting the minimum bandwidth guarantee is proportioned among the queues in accordance with the assigned weights.

To enable or disable automatic weight assignment for QoS queues, use the command:

```
CLI (network-admin@switch) > system-settings-modify cosq-weight-auto|no-cosq-weight-auto
```

Note:

- If you enable auto configuration of queue weights, you cannot change the weights using the `port-cos-bw-modify` command thereafter. You can still use the command to enable strict priority queue.
- When you enable this feature, weights for all the queues that do not have minimum bandwidth configured are set to the default value of 1.

Use the `port-cos-weight-modify` command to change the queue weight of ports on the control panel (PCIe and CPU ports). This command assigns weights on a queue basis only and does not allow a per port configuration of weights.

```
CLI (network-admin@switch) > port-cos-weight-modify
```

`port-cos-weight-modify`	Modify port queue weight.
`cos0-weight priority|no-priority`	
`cos1-weight priority|no-priority`	
`cos2-weight priority|no-priority`	
`cos3-weight priority|no-priority`	
`cos4-weight priority|no-priority`	Specify if you want to enable or disable strict priority. Or, specify the weight for the queue as an integer between 0 and 127.
`cos5-weight priority|no-priority`	
`cos6-weight priority|no-priority`	
`cos7-weight priority|no-priority`	
`cos8-weight priority|no-priority`	
`cos9-weight priority|no-priority`	

For example:

```
CLI (network-admin@switch) > port-cos-weight-modify cos5-weight 64
```

To display the port queue weight configuration, use the command:

```
CLI (network-admin@switch) > port-cos-weight-show
cos0-weight: 16
cos1-weight: 32
cos2-weight: 32
cos3-weight: 32
cos4-weight: 32
cos5-weight: 64
cos6-weight: 64
cos7-weight: 127
```

You can use the command `port-cos-bw-modify` to change the queue weight of ports on the front panel. This command provides granular control over bandwidth and weight settings for port queues.

```
CLI (network-admin@switch) > port-cos-bw-modify
```

`port-cos-bw-modify`	Modify port CoS bandwidth settings.
`cos` *`integer`*	Specify the CoS priority between 0 and 9. **Note:** queues 8 and 9 are platform specific.
`port` *`port-list`*	Specify the physical port(s).
`min-bw-guarantee` *`min-bw-guarantee-string`*	Specify the minimum bandwidth as a percentage.
`max-bw-limit` *`max-bw-limit-string`*	Specify the maximum bandwidth as a percentage.
`weight priority|no-priority`	Specify to enable or disable weight scheduling in proportion to the minimum bandwidth guarantee.

	Or, specify a scheduling weight

For example:

```
CLI (network-admin@switch) > port-cos-bw-modify cos 5 port 10 weight 64
```

Additionally, you can configure strict priority scheduling for any of the queues. If strict priority is configured on a queue, NetVisor OS gives this queue a higher priority over the other queues.

To configure strict priority for queue 6, use the command:

```
CLI (network-admin@switch) > port-cos-bw-modify cos 6 weight priority
```

Displaying and Configuring Port Queue Statistics

You can view the port queue statistics by using the command `port-cos-stats-show`. This command displays the number of bits of transmitted and dropped packets on a per queue basis.

```
CLI (network-admin@switch) > port-cos-stats-show
```

`port-cos-stats-show`	Display per port CoS queue statistics.
`time date/time: yyyy-mm-ddThh:mm:ss`	Time to start statistics collection.
`start-time date/time: yyyy-mm-ddThh:mm:ss`	Start time for statistics collection.
`end-time date/time: yyyy-mm-ddThh:mm:ss`	End time for statistics collection.
`duration: #d#h#m#s`	Duration of statistics collection.
`interval duration: #d#h#m#s`	Interval between statistics collection.
`since-start`	Displays statistics from the start of collection.
`older-than duration: #d#h#m#s`	Displays statistics older than a specified duration.
`within-last duration: #d#h#m#s`	Displays statistics within the last specified duration.

For example:

```
CLI (network-admin@switch) > port-cos-stats-show layout vertical
switch:          switch
time:            21:36:59
port:            21
cos0-obits:      0
cos0-dropbits:   0
cos1-obits:      0
cos1-dropbits:   0
cos2-obits:      0
cos2-dropbits:   0
cos3-obits:      0
cos3-dropbits:   0
cos4-obits:      0
cos4-dropbits:   0
cos5-obits:      0
cos5-dropbits:   0
cos6-obits:      0
cos6-dropbits:   0
cos7-obits:      0
cos7-dropbits:   0
```

To modify the port queue statistics collection settings, use the command:

```
CLI (network-admin@switch) > port-cos-stats-settings-modify
```

`port-cos-stats-settings-modify`	Modify port CoS statistics collection settings.
`enable|disable`	Specify if you want to enable or disable statistics collection.
`interval` *`duration: #d#h#m#s`*	Specify the interval to collect statistics.
`disk-space` *`disk-space-number`*	Specify the amount of memory allocated for statistics (including rotated log files). The default value is 50MB.

For example:

```
CLI (network-admin@switch) > port-cos-stats-settings-modify enable
```

To view the port CoS statistics collection settings, use the command:

```
CLI (network-admin@switch) > port-cos-stats-settings-show
switch:      Spine-1
enable:      yes
interval:    30m
disk-space:  50M
```

You can clear port queue statistics by using the command `port-cos-stats-clear`.

Configuring Forward Error Correction

Note: This feature is supported only on the following platforms. The support is for 25 GbE and 100 GbE unless stated otherwise.

Ericsson: NRU02 (100 GbE only), NRU03, NRU-S0301

Dell: S5232F-ON, S5248F-ON, S5224F-ON, Z9264

Edgecore: AS7726-32X, AS7326-56X, AS5835-54X, AS7816-64X

Freedom: F9432-C, F9476-V, F9460-X, F9664-C

Forward Error Correction (FEC) or channel coding is an error correction technique that helps to detect and correct a limited number of errors in transmitted data without the need for re-transmission. In this method, the sender sends a redundant Error Correcting Code (ECC) along with the data frames using a fixed, higher forward channel bandwidth.

The IEEE 802.3-2018 standard for Ethernet introduced the support for FEC on 25 GbE or 25 Gbps port link. For details, refer to the clauses 74, 91 and 108 in the IEEE 802.3-2018 standard.

From NetVisor OS version 5.1.4 onward, you can enable or disable FEC by using the NetVisor OS CLI or REST API commands on individual 25 GbE ports or 100 GbE ports on NRU03 and NRU-S0301 platforms and 100 GbE ports on NRU02 platform.

From NetVisor OS version 6.1.0 onward, the FEC support is extended to the AS7816-64X (F9664-C), AS7726-32X (F9432-C), AS7326-56X (F9476-V), AS5835-54X (F9460-X) and Z9264 platforms on 25 GbE and 100 GbE connections.

To implement FEC on 25 GbE and 100 GbE ports, NetVisor OS follows the IEEE 802.3-2018 standard rules. (Refer to Table 105-2 and Table 80-3 of the IEEE 802.3-2018 standard).

You can enable the FEC feature by using the `port-config-modify` command while ensuring that `eth-mode` and `auto-neg` configurations are consistent with the FEC configuration.

The FEC feature can be enabled on:

- DAC cables with auto-negotiation enabled
- Optical cables without auto-negotiation (`no-autoneg`)

1. To configure FEC on the above platforms, use the command:

```
CLI (network-admin@switch) > port-config-modify
```

`port-config-modify`	Use this command to modify the port configuration to enable FEC.
`port <port-list>`	Specify a physical port or the list of ports you want to configure for FEC support.

Specify one or more of the following options based on cable and transceivers in use:	
`eth-mode [sr4|lr4|cr4|caui4|sr|lr|cr|sys-default]`	Select one of the ethernet modes from the available options.
`autoneg|no-autoneg`	Specify one of the options to enable or disable auto-negotiation on the physical port .
`fec|no-fec`	Specify one of the options to enable or disable the port forward error correction (FEC) mode.

Note: For NRU03 and NRU-S0301 platforms, `eth-mode` should be set to `sys-default`.

2. On NRU03 and NRU03-S0301 platforms, in addition to the above configuration, use the `port-cable-type-modify` command to configure the port type as `sys-default` to ensure that the correct FEC setting is applied.

```
CLI (network-admin@switch) > port-cable-type-modify port <port-list> type sys-default
```

`port-cable-type-modify`	Use this command to associate a cable type with port (s).
Specify between 2 and 3 of the following options:	
`port <port-list>`	Specify the list of ports you want to associate with the cable type.
`type optical|dac|sys-default`	Specify the cable type: optical or dac associated with the specified ports.
`cable-sub-type default|ca-n|ca-l|ca-s`	Specify the port cable assembly type from one of the options. Note that this optional parameter is applicable only for DAC cable types. **Note**: The cable sub type (N, L, or S) is a hardware-specific parameter based on the material used by the cable manufacturer. Hence, you must consult the vendor specification to identify the cable assembly type before configuring this parameter.

You can verify the configurations by using the `port-fec-status-show` command.

Note: All lengths of Ericsson RPM 777 053/xx and 777 054/xx cables are of assembly type CA-L.

Note: All four lanes of QSFP port share the same type or sub-type.

When configuring the cable type, please follow the guidelines below for both 25 GbE and 100 GbE ports:

- The `cable-sub-type` option is not valid for optical cables.
- The `cable-sub-type` option is not required for DAC cables operating at 100Gbps (non-flexed) as

assembly type does not factor into the FEC variant per IEEE 802.3 2018.
- The `type` keyword is also used by NetVisor OS to determine the pre-emphasis setting used for this port. This capability was introduced in NetVisor OS release 5.1.3.

Note: The `type` and `cable-sub-type` parameters are not required to be configured for NRU02.

Usage Guidelines

- For NRU03 and NRU-S0301 platforms, the port cable type must be configured as `sys-default`.
- For DAC cables, physical port auto-negotiation should be configured for the peer links to come up.
- For optical cables, you can enable or disable FEC explicitly on both peers. Auto-negotiation should not be enabled.
- For DAC cables, the cable sub-type can be CA-S, CA-L, CA-N which is independent of the cable length. For example, you can have a 3 meter type S, 3 meter type N, or 3 meter type L cable.
- For optical cables, the `cable-sub-type` configuration is not allowed.
- For DAC cables, the `eth-mode` should be set to CR (25G) or CR4 (100G).
- The `eth-mode` should be configured by using the `port-config-modify` command to ensure the proper setting of the physical layer and also to address any link failure issues.

4. To clarify the FEC configuration further, an example configuration is provided below:

- Configure FEC on a 25 GbE port on a DAC cable connected on port 21 by using the commands:

```
CLI (network-admin@switch) > port-cable-type-modify port 21 type dac cable-sub-type ca-l

CLI (network-admin@switch) > port-config-modify port 21 eth-mode cr autoneg fec enable
```

- Configure FEC on a 25 GbE port (port 17) using a multi-mode optical cable by using the command:

```
CLI (network-admin@switch) > port-cable-type-modify port 17 type optical

CLI (network-admin@switch) > port-config-modify port 17 eth-mode sr no-autoneg fec enable
```

- Verify port status after FEC configuration:

```
CLI (network-admin@switch) > port-show port 1 layout vertical
switch:           switch
port:             1
bezel-port:       1
status:            up,host,LLDP,trunk,vlan-up
config:           fd,25g,fec
```

Note: It is not mandatory to configure the `no-autoneg` parameter for optical cables, but is specified here to ensure that the port is operating without *auto-negotiation*.

- View the configured cable type and assembly type configuration by using the `port-cable-type-`

`show` command on NRU03 or NRU-S0301:

```
CLI (network-admin@switch) > port-cable-type-show
switch port  type         cable-sub-type
------ ----  ----------- --------------
switch none  dac         ca-s
switch none  dac         ca-l
switch none  dac         ca-n
switch none  dac         default
switch none  optical     default
switch 1-125 sys-default default
```

- Configure FEC on a 100G port on NRU02 by using the command:

```
CLI (network-admin@switch) > port-config-modify port 25 fec enable
```

Note: For NRU02 platforms, the `port-cable-type` and `eth-mode` configuration is not required.

```
CLI (network-admin@switch) > port-show port 25 layout vertical
switch:            switch
port:              25
bezel-port:        7
ip:                192.170.20.110
mac:                66:0e:94:f6:37:f2
hostname:          nru02-alpha2
status:              up,PN-switch,PN-other,STP-BPDUs,LLDP,vlan-up
config:            fd,100g,fec
```

To view the details of the configuration use the `port-fec-stats-show` command. For example,

```
CLI (network-admin@switch) > port-fec-stats-show

switch   time     port BaseR-Fec-Corr BaseR-Fec-UnCorr Rs-Fec-Corr Rs-Fec-UnCorr
-------- -------- ---- -------------- ---------------- ----------- -------------
switch   12:34:00 5    0              0                0           0
switch   12:34:00 7    0              0                0           0
switch   12:34:00 8    0              0                0           0
switch   12:34:00 9    0              0                0           0
switch   12:34:00 10   0              0                0           0
switch   12:34:00 17   0              0                0           0
```

Caution: If you enable FEC between Cisco Nexus switches and the Arista switches AS7816-64X, F9664-C, or Z9264, the link may not come up. This is not an issue of NetVisor OS or Arista switches.

Configuring Static Pre-Emphasis (Signal Integrity) Port Settings

The SerDes (serializer/deserializer) component in switch ports has several associated Signal Integrity (SI) parameters that help in shaping the output into a clean and well-defined signal based on the transmission line characteristics of the signal traces. This output shaping is used to cancel-out factors such as reflection and signal attenuation and enables the receiver to get a clean and well-defined signal. Collectively, these signal integrity parameters are also known as pre-emphasis settings.

If the pre-emphasis settings are incorrect, you may encounter unacceptable bit error rate (BER), especially when the port speed scales to higher values. The pre-emphasis settings vary per port based on the circuit board layout, the operational speed of the port, and the transmission medium (cable type and length).

> **Note**: Refer to the cable manufacturer documentation where the required pre-emphasis values are defined.

This feature is not available on the following platforms:

• F9372-X (AS5812-54X)	• F9372-T (AS5812-54T)	• F9532L-C (AS7712-32X)
• F9572L-V (AS7312-54XS)		

In earlier releases, NetVisor OS had limitations to dynamically change the pre-emphasis settings based on the configured port number and port speed. However, from NetVisor OS version 6.0.0 onward, you can override the default settings for a specified set of ports by using the `port-cable-type-modify port <port-list> type [optical|dac]` command. For the `cable-type` parameter, select either *`dac`* cable setting (for copper DAC cable, which is the default setting) or *`optical`* cable setting (for optical transceivers).

The port cable type database saves the details of the cable type you had configured by using the above CLI. However, if the cable type is not specified, then a default DAC cable length value of 3m is assumed for the port. These settings are then saved to the hardware database (registers).

When you modify the port speed or the cable type, the signal integrity or pre-emphasis settings are written to the hardware. If the port is already enabled, then the port gets disabled and then re-enabled while the settings are being written to the hardware. You are prompted to proceed with the changes prior to the port being disabled and re-enabled. Additionally note that, when a port that you want to modify is part of a port group, then all the ports in that port group are modified with the new signal integrity settings.

The *DAC* settings in NetVisor OS is mapped to the 3m DAC settings, which is the current default on all Arista supported switches, except Z9100-ON.

> **Note -** Guidelines for Z9100-ON platforms:
>
> - During a fresh installation of NetVisor OS 6.0.0, by default, all the ports are automatically set to o*ptical* cable type.
> - While modifying the cable type, out of all the SFP ports, you cannot modify ports 129 and 130 (bezel ports 33 and 34) to `dac` cable type due to a hardware limitation on the Z9100-ON platform.

To change the cable type, and consequently, the pre-emphasis (Signal Integrity) settings for a specified port on any platform, use the command:

```
CLI (network-admin@switch) > port-cable-type-modify port <port-list> type optical|dac
```

`port-cable-type-modify`	Use this command to associate a cable type with port (s)
`port <port-list>`	Specify the list of ports you want to associate with the cable type.
`type optical|dac|sys-default`	Select the cable type: `optical` or `dac` be to associated with the specified ports.
`cable-sub-type default|ca-n|ca-l|ca-s`	Specify the port cable sub-type. The `cable-sub-type` parameter is applicable only for NRU03 and NRU-S03 platforms with 25G FEC support. Ignore the `cable-sub-type` parameter for all other platforms.

To view the currently applied cable type settings for all ports, use the command:

```
CLI (network-admin@switch) > port-cable-type-show
```

Example: Pre-Emphasis Settings Configuration

Let us consider an example to further explain the changes implemented by this feature.

First verify the current port and cable type settings on a switch by using the `port-cable-type-show` command. From NetVisor OS version 6.0.0, only *optical* and *dac* cable types are supported.

```
CLI (network-admin@switch*) > port-cable-type-show
port type         cable-sub-type
---- ---------- --------------
none dac         ca-s
none dac         ca-l
none dac         ca-n
none dac         default
none optical     default
1-24 sys-default default
```

The `default` string in the `cable-sub-type` column is used to display which settings have not been changed by the user.

Now, modify the settings for port 1 by using the below command below (See the warning message regarding the application of the settings to all member ports in the port group) and confirm to proceed:

```
CLI (network-admin@switch*) > port-cable-type-modify port 1-4 type optical
Warning: updating SI settings will apply to all ports in a port group and
will flap enabled ports
```

```
Please confirm y/n (Default: n):y
Updated SI settings for port(s) 1-4 to optical
```

The system log file and nvOSd log file are automatically updated to reflect the Signal Integrity (SI) setting changes.

View the updated settings by using the command:

```
CLI (network-admin@switch*) > port-cable-type-show
port type         cable-sub-type
---- ----------- --------------
none dac         ca-s
none dac         ca-l
none dac         ca-n
none dac         default
1-4  optical     default
5-24 sys-default default
```

To reset all the ports to `dac` cable type, use this command with the `port all` parameter:

```
CLI (network-admin@switch*) > port-cable-type-modify port all type dac

Warning: updating SI settings will apply to all ports in a port group and
will flap enabled ports
Please confirm y/n (Default: n):y
Updated SI settings for port(s) 1-24 to dac
```

View the changed settings again by using the command:

```
CLI (network-admin@switch*) > port-cable-type-show
port type         cable-sub-type
---- ----------- --------------
none dac         ca-s
none dac         ca-l
none dac         ca-n
1-24 dac         default
none optical     default
none sys-default default
```

> **Note**: For platforms with copper RJ-45 ports such as S4148T-ON, an *invalid* message is displayed while modifying the port cable type to `optical` cable type for the copper ports. For example,

```
CLI (network-admin@S4148T) > port-cable-type-modify port all type optical

Warning: updating SI settings will apply to all ports in a port group and
will flap enabled ports
Please confirm y/n (Default: n):y
SI settings invalid for port(s) 1-48 for optical
Updated SI settings for port(s) 49-56,65-72 to optical
```

View the details using the show command:

```
CLI (network-admin@S4148T) > port-cable-type-show port all

switch port    type          cable-sub-type
------ ------ -------       --------------
S4148T none   dac          ca-s
S4148T none   dac          ca-l
S4148T none   dac          ca-n
S4148T 1-48   dac          default
S4148T 49-72  optical      default
S4148T none    sys-default default
```

Note: This feature's settings are applied when you modify either the port speed or the cable type for a specified port. That is, the pre-emphasis (SI) settings in NetVisor OS are applied during the initial port configuration when the port speed is modified by using the `port-config-modify` command, and also when the cable type is modified using the `port-cable-type-modify` command.

Caution: This feature enables you to modify the pre-emphasis settings by modifying the *cable-type* and *port speed*. However, the link quality could be impacted if incorrect settings are chosen and caution must be taken to ensure correct settings are configured.

Guidelines and Limitations

The following guidelines and limitations should be considered while configuring the pre-emphasis settings on a switch port:

- The pre-emphasis settings are written to hardware whenever you change the cable type using the `port-cable-type-modify` command.
- The pre-emphasis settings are written to hardware whenever you change the port speed using the `port-config-modify` command.
- For Z9100-ON platforms with copper SFP ports, the cable type, `dac` is not accepted by NetVisor OS. If you try to change the cable type for a single SFP copper port by using the `port-cable-type-modify` command, then an error is displayed as follows:

```
CLI (network-admin@switch) > port-cable-type-modify port 129 type dac
Warning: updating SI settings will apply to all ports in a port group and
will flap enabled ports
Please confirm y/n (Default: n):y
port-cable-type-modify: Invalid cable type setting
```

- However, if both `port all` and `type optical` parameters are used on a switch with copper ports, then the settings are selectively applied only to the optical ports, and no error is displayed.
- For those platforms that have hidden ports, such as S4148T-ON with 48 x 10GBaseT and 4 x 100 GbE ports, the cable type changes are skipped when `port all` parameter is specified in the `port-cable-type-modify` command.
- For platforms on which no pre-emphasis settings are configured on the ports, a minimum port speed of 10 Gbps is assumed.
- While this feature allows you to select the pre-emphasis settings, the link quality may get impacted if incorrect settings are applied. Ensure to apply the correct pre-emphasis settings during initial configuration.

- Currently the 3m DAC setting is implemented for all DAC cable lengths.

Related Commands

```
CLI (network-admin@nru03-switch) > port-config-modify
```

`port-config-modify`	Use this command to modify a port configuration.
`port` *`port-list`*	Specify the port or list of ports that you want to modify.
`speed disable|10g|25g|40g|100g`	Specify the physical port speed.

Configuring Link Scan Mode

The link scan feature in NetVisor OS detects physical link state changes. NetVisor OS version 6.0.0 (and later) supports interrupt-based link state change detection, in addition to the software-based link scan mode available in the previous versions. The software-based link scan mode performs state polling to detect changes at regular configurable intervals. When the link scan mode is set to `hardware`, on the other hand, the software uses the interrupt infrastructure to provide an improved detection mechanism for link state changes.

You can configure the link scan mode and interval by using the `system-settings-modify` command:

`system-settings-modify`	Use this command to modify system settings.
`linkscan-interval` *`10000..1000000`*	Specify the linkscan interval as a value between 10000 µs and 1000000 µs. The default value is 150000 µs. **Note:** This parameter is not applicable when the link scan mode is hardware.
`linkscan-mode software|hardware`	Specify the linkscan mode as hardware or software. Software linkscan mode is enabled by default. **Note:** Arista recommends software linkscan mode.

For example, to enable interrupt-based link state change detection, use the command:

```
CLI (network-admin@switch) > system-settings-modify linkscan-mode hardware
```

To view the current configuration, use the command:

```
CLI (network-admin@switch) > system-settings-show format linkscan-mode
linkscan-mode: hardware
```

To modify the link scan interval used in software-based mode to 100,000 µs, enter the command:

```
CLI (network-admin@switch) > system-settings-modify linkscan-interval
100000
```

To view the details, enter the command:

```
CLI (network-admin@switch) > system-settings-show
switch:                          Leaf1
optimize-arps:                   off
lldp:                            on
policy-based-routing:            off
optimize-nd:                     off
reactivate-mac:                  on
reactivate-vxlan-tunnel-mac:     on
manage-unknown-unicast:          off
manage-broadcast:                off
```

```
auto-trunk:                       on
auto-host-bundle:                 off
cluster-active-active-routing:    on
fast-route-download:              off
fast-interface-lookup:            on
routing-over-vlags:               off
source-mac-miss:                  copy-to-cpu
igmp-snoop:                       use-l3
vle-tracking-timeout:             3
pfc-buffer-limit:                 40%
cosq-weight-auto:                 off
port-cos-drop-stats-interval(s): disable
lossless-mode:                    off
snoop-query-stagger:              no-stagger-queries
host-refresh:                     off
proxy-conn-retry:                 on
proxy-conn-max-retry:             3
proxy-conn-retry-interval:        500
manage-l2-uuc-drop:               on
xcvr-link-debug:                  disable
fastpath-bfd:                     off
linkscan-interval:                100000
linkscan-mode:                    hardware
single-pass-flood:                off
```

Note: To avoid flaps, the link scan mode must match on both the switch ports connected by a link.

Configuring Maintenance Mode

NetVisor OS version 6.1.0 introduces the Maintenance Mode feature in which the software prevents user traffic from entering or leaving a switch and gracefully steers the traffic to network peers. This mode is useful during hardware or software maintenance of the switch including Return Merchandise Authorization (RMA), correction of out of sync fabric transactions, cluster re-peer, switch power down or reboot, and software upgrade. This functionality is supported on cluster nodes, non-cluster spine nodes, and standalone switches. In a use case where you need to enable maintenance mode on a standalone switch, you must ensure that an alternate path exists if the switch goes down.

> **Note:** You can enable maintenance mode only when `control-network` and `fabric-network` are configured as `mgmt`. You cannot change the fabric and control networks or the fabric VLAN while in maintenance mode.

A switch does not leave the fabric while entering maintenance mode. When you issue the command to enable maintenance mode, NetVisor OS follows the normal port bring down sequence and disables all the user ports. Any cluster bring down configuration is also applicable. The sequence of actions performed by the software while entering maintenance mode (for a cluster switch) is as follows:

- Bring down orphan ports
- Bring down vLAG ports
- Disable VRRP service and BGP graceful shutdown
- Bring down Layer 3 ports
- Bring down ports with `defer-bringdown` configured
- Bring down cluster ports

While in maintenance mode, you can execute all CLI or REST API commands except port enable commands. The software defers all port enable actions and enables the ports only after exiting maintenance mode. If you reboot or power cycle the switch while in maintenance mode, the switch comes back up and stays in maintenance mode and does not enable any user ports.

A switch (cluster member) follows the below sequence of actions while leaving maintenance mode:

- Bring up cluster ports
- Enable VRRP and BGP services
- Bring up Layer 3 ports
- Bring up vLAG ports
- Bring up orphan ports or ports with `defer-bringup` configured

The switch assumes data forwarding responsibilities upon leaving maintenance mode. All pre-configured settings for cluster bring up are applicable during the exit and therefore, NetVisor OS enforces staggered or delayed port bring based on the existing configuration. For more information, see the '*Restoring Ports for Cluster Configurations*' section of the '*Configuring High Availability*' chapter.

Use the command `system-state-modify` to enable or disable maintenance mode. For example, to enter maintenance mode, use the command:

```
CLI (network-admin@Leaf1) > system-state-modify maintenance-enable
Warning: This configuration can have traffic impact. If required, collect
```

```
system snapshot via save-diags prior to this command
Please confirm y/n (Default: n):y
CLI (network-admin@Leaf1) >
```

Note: Arista recommends collecting the output of the `save-diags` command before entering maintenance mode. This helps in recording the state of the switch.

To view the status of maintenance mode, use the command:

```
CLI (network-admin@Leaf1) > system-state-show
system-state:      Maintenance mode, Ports disabled
```

To leave maintenance mode, use the command:

```
CLI (network-admin@Leaf1) > system-state-modify maintenance-disable
```

Note: You must ensure that transactions (fabric and cluster TIDs) are in sync with the rest of the fabric before executing the command to disable maintenance mode.

Use the `system-state-show` command to view the status:

```
CLI (network-admin@Leaf1) > system-state-show
system-state:      Operational, Ports enabled
```

The `system-state-show` command also displays the current state of port bring up or port bring down:

```
CLI (network-admin@Leaf1) > system-state-show
system-state:      coming up, l3 to vlag wait
system-state:      coming up, vlag ports being enabled
system-state:      coming up, defer bringup ports wait
```

When a switch enters or leaves maintenance mode, an event log messages is logged, as seen from the `log-event-show` output:

```
CLI (network-admin@Leaf1) > log-event-show
event maintenance_enabled(11529) : level=note event-type=system :   : System is in Maintenance mode
event maintenance_disabled(11530) : level=note event-type=system :   : System is in Operational mode
```

You can enforce maintenance mode on a switch as soon as the cluster re-peer process is complete by using the sample command:

```
CLI (network-admin@Leaf1) > fabric-join repeer-to-cluster-node Leaf1
maintenance-enable
```

This operation is useful when replacing a cluster node and if you want to be in maintenance mode after the re-peer process.

Configuring Forced Port Link-up

The Ethernet standard requires a port to have an active RX connection to a peer device to be able to negotiate link parameters (for example, for auto-negotiation purposes) before the port can be brought into *up* state.

In some special cases, though, (for example for security purposes) using both RX/TX wire connections/fiber strands in a port is not required when only unidirectional connectivity is being used: in such cases, in fact, only the RX wire connections/fiber strands are expected to receive traffic from the TX of the peer device (whose RX is unused). This unidirectionality is expected when one peer port is supposed to only receive traffic and the other one is supposed to only transmit it. However, in normal circumstances, connecting only the RX on one port to the TX on the other would not generate a link-up on the latter.

In other words, for security purposes, to prevent traffic from going back into the production network, only one half of the cable (TX to RX) is connected, where the traffic only goes out of the production network and not back into the network.

Starting from NetVisor OS version 7.0.0, the forced port link-up feature is available on the following platforms:

- Dell: S5248F-ON, S5232F-ON, S5224F-ON, S5048F-ON, Z9264F-ON
- Edgecore: AS7312-54XS, AS7712-32X, AS7816-64X, AS7726-32X, AS7326-56X, AS5835-54X/AS5835-54T, AS5812-54T/AS5812-54X
- Freedom: F9572L-V, F9532L-C, F9664-C, F9432-C, F9480-V, F9460-X/F9460-T, F9372-T/F9372-X

> **Note**: The forced port link-up feature can be enabled both in VirtualWire mode as well as in switch mode by using virtual Port Groups (vPGs) for Network Packet Broker deployments.

When a port is enabled for forced link-up:

- Port with only TX connection can transmit packets.
- When configuration changes are done on the TX port, RX port should also come up. That is, both ports (port with TX connection and the peer switch with only RX connection) should be in *Up* state.

You can configure the forced port link-up feature by using the `port-force-linkup-add` command.

> **Note**: Run the above command only on the switch with the TX connection port. That disables *fault detection* on the TX connection port as well as disables the RX connection on the TX port. That is, the forced port link-up feature cannot be enabled if the RX and TX cables are connected between ports. Running the above command also ensures disabling of TX connection and fault detection on the peer switch, enabling the RX port to come up.

For the forced port link-up feature, Arista recommends the following:

- Port speed of 1, 10, 25, 40, or 100 Gbps.
- *Optical LC cable type*
- *Any optical transceiver (*QSFP *and* SFP*) with an LC connector*

You can configure a port(s) to be forced link-up even when its RX connector/fiber is not connected to any

peer device by using the `port-force-linkup-add` command:

```
CLI (network-admin@switch) > port-force-linkup-add
```

`port-id` *`port-id-number`*	Specify the port numbers allowed for force linkup.
`mode tx|rx`	Specify the RX or TX side of the port. The default value is TX. **Note**: If two Arista switches are used, then there is no need to set RX only mode on the peer side.

The ports that are forced link-up remain persistent on switch reboot and so to disable the forced link-up ports, use the `port-force-linkup-remove port-id port-id-number` command.

To view the ports that are forced link-up with no RX connection, use the `port-force-linkup-show` command.

Below is an example to configure force link-up on port 6 by using the command:

```
CLI (network-admin@switch) > port-force-linkup-add ports 6 mode tx

CLI (network-admin@switch) > port-force-linkup-show

switch ports mode
------ ----- ----
switch 6     tx

CLI (network-admin@switch) > port-show

switch port bezel-port ip         mac               hostname          status
config
------ ---- ---------- ---------- ----------------- --------- ----------------------------------------
-------
switch 0    0         169.254.2.1 66:0e:94:d6:8e:2e switch   up,PN-internal,stp-edge-port
switch 2    2         169.254.2.1 66:0e:94:7c:a8:33 switch1  up,PN-switch,PN-other,STP-BPDUs,LLDP,vlan-up
fd,10g
switch 6    6                                               up,vlan-up
fd,10g
switch 42   42        169.254.2.1 66:0e:94:dc:f4:59 switch2  up,PN-switch,PN-other,STP-BPDUs,LLDP,vlan-up
fd,10g
switch 55   52        169.254.2.1 66:0e:94:7c:a8:33 switch1  up,PN-switch,PN-other,STP-BPDUs,LLDP,vlan-up
fd,100g
```

Port 6 above has no RX connection, but with this configuration the software is able to force the port in up state.

Use the following command to disable this configuration:

```
CLI (network-admin@leaf1) > port-force-linkup-remove ports 6
```

To view the transceiver details for port 6, use the command:q

```
CLI (network-admin@switch) > port-xcvr-show port 6
```

```
port bezel-port vendor-name   part-number serial-number temp[C] vcc33[V] tx-bias[mA] rx-pwr[dBm]
---- ---------- ------------ ----------- ------------- ------- -------- ----------- -----------
6    29         FINISAR CORP. FTL4C3QE1C  UTC135L        35.38  3.32     37.16        -inf
```

Guidelines while Configuring Forced Port Link-Up

Before configuring forced port link-up, ensure the following on the TX port:

- *Auto-negotiation* is disabled.
- The *link-learning* feature is disabled.
- Port flexing (breaking a high speed port to lower speed ports) is not allowed after enabling the force link-up feature.
- *Fault detection* is disabled to implement the forced port link-up feature. Hence if the TX cable gets disabled for any reason, to identify the issue, you must use the port-xcvr-show command on the TX connected port.

> **Note**: The forced port link-up feature can be used in VirtualWire setups and Network Packet Broker deployments.

For details about these features, see the:

- *Configuring Network Packet Broker* section in the *Configuring and Using Network Management and Monitoring* chapter of *this Guide*.

Limitations

- The forced port link-up feature is not supported on ports that are flexed (that is, you cannot enable force link-up if the port is flexed).
- The forced port link-up feature cannot be enabled if a duplex cable is used to connect ports.
- Link training is disabled for this feature and so the ports shows as down in `lldp-show` output.

Infrastructure Support for Transceiver Online Insertion and Removal (OIR)

The Transceiver OIR infrastructure supports the following three operations:

- Detecting the insertion or removal of optical devices and cables into or from the front-panel ports, thereby allowing subsequent actions to take place.
- Initializing the optical devices and cables at system start-up, and also as they are inserted into the chassis during run-time.
- Configuring key parameters into the hardware based on the required settings for the optical devices and cables as they are detected.

> **Note**: The support for the Transceiver OIR infrastructure was provided on NRU03 and NRU-S0301 platforms in NetVisor OS version 6.0.1. Starting with NetVisor OS version 7.0.0, the OIR infrastructure is also supported on the following platforms:
>
> - Dell: S5232F-ON, S5248F-ON, S5224F-ON, Z9264
> - Edgecore: AS7816-64X, AS7726-32X, AS5835-54X, AS7326-56X, AS5835-54T
> - Freedom: F9664-C, F9432-C, F9460-X, F9480-V, F9460-T

The support for transceiver OIR infrastructure enhances NetVisor OS to process the transceiver OIR events and apply appropriate configuration and startup sequencing based on the type of module inserted or removed. This infrastructure implementation improves the hardware monitor service to detect OIR events and notify NetVisor OS.

Prior to version 6.0.1, NetVisor supported initialization of optical devices on switches only at power-on. Starting with NetVisor OS version 6.0.1, with the Transceiver OIR infrastructure support, you can also facilitate run-time initialization of optical devices (hot-swappable) while the switch is powered-on. This provides scalable infrastructure for device-specific initialization requirements upon OIR of the transceiver on NRU03 and NRU-S0301 platforms. From NetVisor OS version 7.0.0 onward, this capability is extended to other platforms listed in the Note above.

That is, you can remove, insert, swap, or replace the transceivers when the system is functioning. Each transceiver has unique characteristics, parameters, timing requirement, initialization sequence, and configuration impact to the switch port and NetVisor OS adheres to these characteristics for the transceiver to operate correctly.

With transceiver OIR infrastructure support, NetVisor OS automatically detects the type of device installed in the physical port and the OIR determines the port settings if the `eth-mode` or pre-emphasis (`port-cable-type`) settings are not configured already.

An important distinction between pre-OIR support release and releases with OIR support is the use of sys-default configuration option, as explained in the table:

Table 3-1: Pre-OIR Release Vs OIR Supported Releases

Behavioral difference	Eth-mode set to sys-default	Cable-type set to sys-default
Pre-OIR support behavior	Eth-mode setting is determined by port speed; all ports of that speed will have same eth-mode setting.	Cable-type set to platform default (3m DAC) for all ports, regardless of which cable is plugged in.
Releases with OIR support behavior (only OIR support platforms)	Eth-mode setting is applied per physical port based on the device type that is plugged in.	Cable-type setting is applied per physical port based on the device type that is plugged in.

Note: Ensure to set the `eth-mode` settings in accordance with the switch vendor recommendation.

For example, releases prior to having support for OIR, a 100G DAC cable and 100G SR4 optic cable have the same eth-mode configuration if both ports are configured for `sys-default`. However, with OIR support, these ports have `eth-mode` configured as CR4 and SR4, respectively.

When you are upgrading NetVisor OS from an earlier version which does not have support for OIR to a version which supports OIR, the original settings are retained unless you revert to the system default settings by using the `port-config-modify port <port-num> eth-mode sys-default` command or the `port-cable-type-modify port <port-list> type sys-default` command upon upgrade. The `sys-default` parameter ensures that the hardware settings are set in accordance to the specific device type installed after upgrading to a version which supports OIR.

Note: It is highly recommended to configure `eth-mode` and `cable-type` to `sys-default` for newly deployed ports.

Below is an sample configuration after upgrading to a NetVisor version that features platform support for OIR:

```
CLI (network-admin@nru03-switch) > port-cable-type-modify port all type
sys-default
Warning: updating SI settings will apply to all ports in a port group and will flap enabled
ports
Please confirm y/n (Default: n):y
Updated SI settings for port(s) 1-125 to sys-default

CLI (network-admin@nru03-switch) > port-cable-type-show

switch        port   type         cable-sub-type
------------  -----  -----------  --------------
nru03-switch none   dac          ca-s
nru03-switch none   dac          ca-l
nru03-switch none   dac          ca-n
nru03-switch none   dac          default
nru03-switch none   optical      default
nru03-switch 1-125 sys-default default
```

The auto negotiation and the FEC are user-configurable features, that is, you can enable or disable these features; Transceiver OIR will not override those settings.

Note: When you upgrade from an earlier version which does not have support for OIR to a version which supports OIR, all existing *port configuration settings* are preserved.

Usage Guidelines

- At system startup with a fully-populated switch, an initial insertion event is generated for each transceiver, but this has a minimal impact on NetVisor startup time.
- The `eth-mode`, *ER4* is not supported by Broadcom SDK. Per Broadcom recommendation, the OIR infrastructure programs the hardware as *SR4* when *ER4* device is detected.
- For RDH 72/16 (PSM4) the recommended system default eth-mode is *CAUI4* at 100G speed and *LR4* for 4x25G speed as determined by the OIR.
- The transceiver OIR implementation in NetVisor OS version, which supports OIR has minimal error handling capabilities. Failures in the execution of OIR state machine events may cause port to enter into error state. If you encounter an error, you must remove and re-insert the transceiver to recover.

  > **Note**: If for any reason you are unable to recover the transceiver, you must contact Arista TAC for support.

- The transceiver OIR infrastructure introduces a new SNMP event when entering and exiting the error state. Below is a sample SNMP event generated during an enter/exit error state as a reference:

```
2020-09-23 21:40:39 localhost [UDP: [127.0.0.1]:37190->[127.0.0.1]:162]:
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (17946935) 2 days, 1:51:09.35
SNMPv2-MIB::snmpTrapOID.0 = OID: PN-LOG-MIB::pnLogMatchNotification
PN-LOG-MIB::pnLogMatchName.0 = STRING: PortOIRErr PN-LOG-MIB::pnLogFileName.0 =
STRING: /nvOS/log/event.log PN-LOG-MIB::pnLogMatchCount.0 = Gauge32: 3
PN-LOG-MIB::pnLogMatchData.0 = STRING: 2020-09-23,21:40:01.352234-07:00
nru02-alpha2 nvOSd(3830) event port_oir_error_enter(11527) : level=warn
event-type=port : port=1 : Port=1 enter xcvr oir error state

2020-09-23 21:41:39 localhost [UDP: [127.0.0.1]:37190->[127.0.0.1]:162]:
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (17952935) 2 days, 1:52:09.35
SNMPv2-MIB::snmpTrapOID.0 = OID: PN-LOG-MIB::pnLogMatchNotification
PN-LOG-MIB::pnLogMatchName.0 = STRING: PortOIRErr PN-LOG-MIB::pnLogFileName.0 =
STRING: /nvOS/log/event.log PN-LOG-MIB::pnLogMatchCount.0 = Gauge32: 4
PN-LOG-MIB::pnLogMatchData.0 = STRING: 2020-09-23,21:40:46.408637-07:00
nru02-alpha2 nvOSd(3830) event port_oir_error_exit(11528) : level=warn
event-type=port : port=1 : Port=1 exit xcvr oir error state
```

- It is possible to miss an insertion or removal event if a transceiver is inserted and removed rapidly.

  > **Note**: You must restrain from removing and inserting a transceiver rapidly because that could cause the port to enter error state in the OIR state machine, especially if a different type of transceiver is inserted.

- Transceiver OIR leverages the device characteristic information maintained in the device EEPROM.

  > **Note:** The vendors that do not comply with MSA standard may not operate properly when used with OIR. For best results, use only those devices that are listed in the Transceiver Compatibility Information, available on the Technical Documentation.

- With OIR infrastructure:
  - The pre-emphasis and the `eth-mode` settings are set in accordance with the device type
  - The DAC pre-emphasis setting is set in accordance with the DAC cable length

Note: Currently the OIR infrastructure uses the 3 meter DAC setting when programming the hardware, regardless of the actual DAC cable length.

CLI Settings

> **Note**: You should never modify the OIR settings.

Use the following CLI commands to verify the OIR details on your switch.

To display the XCVRs present on port in the switch, use:

```
CLI (network-admin@switch) > port-xcvr-show port <port number>
```

For example, to see the details:

```
CLI (network-admin@S5232F) > port-xcvr-show format switch,port,vendor-
name,part-number,serial-number


switch   port  vendor-name       part-number    serial-number
-------  ----  ---------------  -------------  --------------
S5232F   49    Amphenol         603020002      APF16010026PM3
S5232F   51    FINISAR CORP     FCBN510QE2C03  WXG0EXM
S5232F   55    Arista Networks  CAB-Q-Q-3M     XHC1315DG10H
S5232F   56    DELL             GMFC5          MY013602746004F
```

To check the port phy status, use:

```
CLI (network-admin@switch) > port-phy-show port <port number>
```

For example,

```
CLI (network-admin@S5232F) > port-phy-show

switch port state speed  eth-mode max-frame def-vlan
------ ---- ----- ------ -------- --------- --------
S5232F 1    down  100000 caui4    1540      1
S5232F 5    down  100000 caui4    1540      1
S5232F 57   up    100000 cr4      1540      1
S5232F 61   down  100000 caui4    1540      1
S5232F 117  up    100000 cr4      1540      1
S5232F 126  down  10000  xfi      1540      1
S5232F 127  down  10000  xfi      1540      1
```

Enabling Jumbo Frame Support

Jumbo frames are (oversized) Ethernet frames with a size greater than 1518 bytes (including the Ethernet header and FCS) or 1522 bytes (when a VLAN tag is present).

Different vendors may support different maximum frame sizes for jumbos, as the IEEE standard does not cover jumbo frames (except for baby giants). Typically jumbo frames are described with the maximum payload length they support, namely, the Maximum Transmission Unit (MTU). It is common for platforms (including servers) to support jumbo frames with (at least) an MTU of 9K bytes. 1 K usually corresponds to 1000 (but may also equate to 1024 on some platforms), so that means that the MTU is 9000 (or 9216) bytes.

When you enable the jumbo frame feature on a port on Arista switches, the port can accept and forward jumbo frames. This feature is meant to optimize server-to-server performance (by minimizing the CPU processing overhead for large block transfers).

By default, jumbo frames are disabled on NetVisor OS and so the default Ethernet MTU for all switch ports is 1500 bytes. When you enable the jumbo frame feature on a port, the MTU size is increased to accept 9K byte frames on that port.

The larger MTU supported on Arista switches helps on transport links to be able to deal with the transport overhead (for example, with the VXLAN and vLE functionalities described in the *Configuring VXLAN* and *Configuring Advanced Layer 2 Transport Services* sections).

To enable jumbo frame support, add the `jumbo` parameter to the `port-config-modify` command:

```
CLI (network-admin@switch) > port-config-modify port 1 jumbo
```

To enable jumbo frame support on trunk ports, add the jumbo parameter to the `trunk-modify` or `trunk-create` commands:

```
CLI (network-admin@switch) > trunk-modify name trunk1 jumbo
```

Configuring Uplink Groups

Certain features require that a switch monitors the total available aggregate uplink bandwidth compared to the amount of downlink traffic being generated by the attached devices. The requirement is to prevent oversubscription of the uplinks by traffic sourced from the downlinks (host-facing ports).

For that purpose, starting from NetVisor OS release 7.0.0, the software supports the configuration of *uplink groups*, which can be used with features like vLE and bidirectional vPG for uplink oversubscription avoidance. (Refer to the *Configuring Virtual Link Extension* and *Configuring Network Packet Broker* sections of the Configuration Guide for more details on those features.)

Uplink oversubscription is defined as: total *downlink capacity* > *total uplink capacity* (for a given switch).

> **Note**: The operational status of switch ports is not considered to calculate the uplink and downlink aggregated bandwidth capacities. Only the configuration/administrative state is considered.

> **Note**: Uplink groups support only bidirectional vPG-driven connections and not unidirectional vPG-driven connections.

The configuration of an uplink group allows users to track the available aggregate bandwidth of the physical ports added to a group. (Trunks cannot be added.) Then, a system configuration allows the users to select which action to apply in case of aggregate bandwidth mismatch between the uplink ports and the downlink ports:

- No warning or error is generated (default behavior)
- Allow the creation of a connection across the fabric, but:
    - Display an appropriate message to warn about oversubscription
    - Log the event in system.log
- Refuse the creation of a connection across the fabric if its creation leads to oversubscription.

> **Note**: The software currently does not support separate uplink port groups per feature connection to monitor oversubscription more granularly. Hence this could lead to inaccuracies in reporting (i.e., to false positives).

The software tracks port additions/modifications/deletions as well as port speed changes to calculate the aggregate bandwidth. Whenever a feature-driven connection across the fabric is created/deleted/disabled, the downlink port capacity is also modified (when applicable) and checked.

> **Note**: Uplink/downlink port speed changes when enabling/disabling auto-negotiation are not reflected in the respective group's aggregated bandwidth. So, after auto-negotiation, if a port speed changes and leads to oversubscription, the event is not reported.

Release 7.0.0 introduces a new set of commands to configure an uplink group, as shown below:

```
CLI  (network-admin@switch)  >  uplink-group-port-show
switch:  switch
```

```
ports: none

CLI (network-admin@switch) > uplink-group-port-add ports 12-14

CLI (network-admin@switch) > uplink-group-port-show
switch: switch
ports: 12-14
```

Note: Only physical ports can be added to an uplink group. Configuring trunks will produce an error.

You can remove ports from an uplink group like so:

```
CLI (network-admin@switch) > uplink-group-port-show
switch: switch
ports: 12-14

CLI (network-admin@switch) > uplink-group-port-remove ports 12
CLI (network-admin@switch) > uplink-group-port-show
switch: switch
ports: 13-14
```

You can configure the action to take when uplink oversubscription is detected with the following command:

```
CLI (network-admin@switch) > system-settings-modify uplink-
oversubscription-action default | warn | error
```

`default` means no warning or error is generated.

`warn` allows the creation of a connection across the fabric, but a warning message is produced if the creation causes uplink oversubscription.

If the switch is already in oversubscribed state, the system generates a specific warning message right when the option is configured:

```
CLI (network-admin@switch) > system-settings-modify uplink-
oversubscription-action warn
Warning: The switch is already in uplink oversubscription state. Configure
ports to uplink group
```

You can configure the error option so that the software refuses the creation of a connection across the fabric when causing uplink oversubscription:

```
CLI (network-admin@switch) > system-settings-modify uplink-
oversubscription-action error

CLI (network-admin@switch) > system-settings-show format uplink-
oversubscription-action
uplink-oversubscription: error
```

If the switch is already in oversubscribed state, the command will fail right away with an error when the option is configured:

```
CLI (network-admin@switch) > system-settings-modify uplink-
oversubscription-action error
system-setting-modify: failed to change to error action as the switch is
already in uplink oversubscription state. Add ports to uplink group or
remove existing VLE/VPG(bidir) connections.
```

If you remove ports from an uplink group, oversubscription may occur due to the capacity change. The warn and error options trigger, respectively, the following messages (and action):

```
CLI (network-admin@switch) > uplink-group-port-remove port 25
Warning: removing ports from uplink group is causing oversubscription of
uplink ports.

CLI (network-admin@switch) > uplink-group-port-remove port 25
uplink-group-port-remove: removing these ports from uplink group failed as
it is leading to oversubscription of uplink ports.
```

If you change uplink/downlink port speeds, that may lead to oversubscription. The `warn` and `error` options trigger, respectively, the following messages (and action):

```
CLI (network-admin@switch) > port-config-modify port 45 speed 40g
Warning: Change of speed for the port(s) 45 has caused uplink
oversubscription.

CLI (network-admin@switch) > port-config-modify port 45 speed 40g
port-config-modify: change port speed failed for given port(s) 45 as
leading to oversubscription of uplink ports.
```

In case of the `warn` option, this is the warning message generated for a vLE configuration with uplink oversubscription:

```
CLI (network-admin@switch) > vle-create node-1 switch node-1-port 12 node-2
switch2 node-2-port 13 name vle1
Warning: The creation/enabling of vle vle1 is causing uplink
oversubscription by 1000 mbps.
```

In this scenario, the calculated difference between the downlink and uplink capacity is 1000 Mbps.

In case of the `error` option, when oversubscription occurs locally to a vLE switch, this message is displayed:

```
CLI (network-admin@switch) > vle-create node-1 switch node-1-port 12 node-2
switch2 node-2-port 13 name vle1
vle-create: creation/enabling of VLE vle1 failed because it is causing
uplink oversubscription by 1000 mbps. Remove associated transparent VLAN
and VXLAN mapping for the VLE.
```

Similarly, when oversubscription happens on a remote switch due to the `vle-create` command, this message is generated:

```
CLI (network-admin@switch) > vle-create node-1 switch node-1-port 12 node-2
switch2 node-2-port 13 name vle1
```

```
vle-create: error from switch2: creation/enabling of VLE vle1 failed
because it is causing uplink oversubscription by 1000 mbps. Remove
associated transparent VLAN and VXLAN mapping for the VLE.
```

In case of the `warn` option, this is the warning message generated for a vPG configuration with uplink oversubscription:

```
CLI (network-admin@switch) > vflow-create name v1 scope fabric bidir-vpg-1
vpg1 bidir-vpg-2 vpg2
Warning: The creation/enabling of (bidir vpg)vflow v1 is causing uplink
oversubscription by 1000 mbps.
```

In case of the `error` option, when oversubscription occurs locally to a vPG switch, this message is displayed:

```
CLI (network-admin@switch) > vflow-create name v1 scope fabric bidir-vpg-1
vpg1 bidir-vpg-2 vpg3
vflow-create: The creation/enabling of (bidir vpg)vflow v1 is causing
uplink  oversubscription by 1000 mbps.
```

Similarly, when oversubscription happens on a remote switch due to the `vflow-create` command, this message is generated:

```
CLI (network-admin@switch) > vflow-create name v1 scope fabric bidir-vpg-1
vpg1 bidir-vpg-2 vpg3
vflow-create: error from switch2: The creation/enabling of (bidir vpg)vflow
v1 is causing uplink oversubscription by 1000 mbps.
```

With the `warn` option, when a vLE is created with endpoints that are local to the switch and have a speed mismatch, when oversubscription occurs this warning message is generated:

```
CLI (network-admin@switch) > port-show port 9,49 format port,config,
port config
---- ------
9 fd,10g
49 fd,40g

CLI (network-admin@switch) > vle-create name v1 node-1 switch node-1-port 9
node-2 switch node-2-port 49
Warning: creation/enabling of VLE is causing endpt 9 being oversubscribed.
```

In the same scenario, but with the `error` option, this action is generated in case of oversubscription:

```
CLI (network-admin@switch) > vle-create name v1 node-1 switch node-1-port 9
node-2 switch node-2-port 49
vle-create: creation/enabling of local VLE failed because endpoint 9 is
oversubscribed.
Remove associated transparent VLAN and VXLAN mapping for the VLE.
```

With the `warn` option, when vPGs are created with endpoints that are local to the switch (`port 17,45` in the example below) and have a speed mismatch, this warning message is generated:

```
CLI (network-admin@switch) > vpg-show
```

```
scope name type ports vni vlan
------ ---- ------------ ----- -------- ----
fabric vpg3 bidirectional 17 12500000 2750
fabric vpg4 bidirectional 22 12500001 2751
fabric vpg5 bidirectional 45 12500002 2752

CLI (network-admin@switch) > port-show port 17,45 format port,config,
port config
---- ------
17 fd,10g
45 fd,40g

CLI (network-admin@switch) > vflow-create name v3 scope local bidir-vpg-1
vpg3
bidir-vpg-2 vpg5
Warning: creation/enabling of local vflow is causing endpoint 17 to be
oversubscribed.
```

In the same scenario, but with the `error` option, this action is generated:

```
CLI (network-admin@switch) > vflow-create name v3 scope local bidir-vpg-1
vpg3
bidir-vpg-2 vpg5
vflow-create: creation/enabling of local vflow failed because endpoint 17
is
oversubscribed.
```

With the `warn` option, when adding ports to a vPG with an active vFlow connection, the following warning message is generated when oversubscription occurs:

```
CLI (network-admin@switch) > vflow-show format name,bidir-vpg-1,bidir-vpg-
2,inport,
name bidir-vpg-1 bidir-vpg-2 in-port
---- ----------- ----------- -------
v6 vpg3 vpg2 12
```

If port 12 (10g) is removed and trunk port 273 (20g) is added to `vpg3` of vFlow `v6`:

```
CLI (network-admin@switch) > vpg-port-remove vpg-name vpg3 ports 12
CLI (network-admin@switch) > vpg-port-add vpg-name vpg3 ports 273
warning: creation/enabling of Vflow v6 is causing uplink oversubscription
by
10000 mbps
```

In the same scenario, but with the `error` option, this action is generated in case of oversubscription:

```
CLI (network-admin@switch) > vpg-port-remove vpg-name vpg3 ports 12
CLI (network-admin@switch) > vpg-port-add vpg-name vpg3 ports 273
vpg-port-add: creation/enabling of Vflow v6 failed because it is causing
uplink
oversubscription by 1000 mbps. Consider adding ports to uplink group.
```

With the `warn` option, when trunk ports of a bidirectional vPG belonging to active vFlow connections are

modified, the following warning message is generated when oversubscription occurs:

```
CLI (network-admin@switch) > vflow-show format name,bidir-vpg-1,bidir-vpg-
2,inport,
name bidir-vpg-1 bidir-vpg-2 in-port
---- ----------- ----------- -------
v6 vpg3 vpg2 27

CLI (network-admin@switch) > trunk-show name t1 format name,trunk-id,ports
name trunk-id ports
---- -------- -----
t1 273 21
```

If the uplink bandwidth is 10 Gbps and the trunk `t1` is expanded to double its capacity, say, with ports 21-22 (20g), then oversubscription would occur:

```
CLI (network-admin@switch) > trunk-modify ports 21-22 name t1
Warning: modifying trunk parameters in use by bidir vpg is leading to
uplink oversubscription.
```

In the same scenario, but with the `error` option, this action is generated instead:

```
CLI (network-admin@switch) > trunk-modify ports 21-22 name t1
trunk-modify: modifying trunk parameters in use by bidir vpg failed because
it is leading to uplink oversubscription.
```

You can show the uplink oversubscription state with the following command:

```
CLI (network-admin@switch) > uplink-group-port-oversubscription-state-show
uplink-bw:               20000
downlink-bw:             50000
uplink-oversubscription: yes
```

Configuring Port Bandwidth Monitoring

NetVisor OS version 7.0.0 enables you to configure alerts in the form of SNMP traps and syslog messages when the data throughput across a port over an interval exceeds a configured bandwidth threshold. This feature helps in detecting network congestion and monitoring port bandwidth usage.

When you enable this feature, NetVisor OS logs separate syslog messages when the threshold is exceeded by ingress or egress traffic and also when the bandwidth usage decreases from a value above the threshold to a value below the threshold on a port. This creates four possible combinations of log messages for when the bandwidth usage exceeds or falls below the threshold by ingress or egress traffic. You can also enable SNMP traps for bandwidth monitoring through the CLI.

For more information on system log messages, refer *NetVisor OS Log Messages Guide*.

Note:

- NetVisor OS calculates the throughput for a port as the total bandwidth used by all types of network traffic. In other words, the bandwidth usage calculation is not traffic-specific.
- If you issue the `port-stats-clear` command during a port bandwidth monitoring interval, the throughput calculation for that interval may not be accurate.
- You cannot configure this feature on trunk ports. But if you configure bandwidth usage monitoring on a port and add it to a trunk, the port continues to be monitored.
- You cannot configure this feature on the CPU port (port 0).

To configure port bandwidth monitoring, use the command:

```
CLI (network-admin@switch) > port-bw-threshold-alert-modify
```

`port-bw-threshold-alert-modify`	Modify port bandwidth monitoring.
`port` *`port-list`*	Specify the list of ports that you want to monitor.
`bw-threshold` *`bw-threshold-string`*	Specify the bandwidth threshold as a percentage of the total bandwidth of the ports. The default value is 0, which means that bandwidth monitoring is disabled by default.

For example, to configure the bandwidth threshold as 25 percent for ports 24 to 26, use the command:

```
CLI (network-admin@switch) > port-bw-threshold-alert-modify port 24-26 bw-
threshold 25
```

Use the `port-bw-threshold-alert-show` command to display the configuration:

```
CLI (network-admin@switch) > port-bw-threshold-alert-show
switch port bw-threshold
------ ---- ------------
```

```
switch 24    25%
switch 25    25%
switch 26    25%
```

To configure a port bandwidth monitoring interval, use the command:

```
CLI  (network-admin@switch)  >  port-bw-threshold-mon-int-modify
```

`port-bw-threshold-mon-int-modify`	Modify port bandwidth monitoring interval.
`monitor-interval` *`10..300`*	Specify a port bandwidth monitoring interval. The allowed range of values is between 10 and 300 seconds. The default value is 30 seconds.

For example, to set a port bandwidth monitoring interval of 45 seconds, issue the command:

```
CLI  (network-admin@switch)  >  port-bw-threshold-mon-int-modify  monitor-
interval 45
```

Use the `port-bw-threshold-mon-int-show` command to display the monitoring interval configuration.

```
CLI  (network-admin@switch)  >  port-bw-threshold-mon-int-show
monitor-interval:  45
```

You can view the syslog message for events related to port bandwidth monitoring by issuing the `log-system-show` command:

```
CLI (network-admin@switch) > log-system-show layout vertical count 2
category:          system
time:               2021-09-27,06:57:54.743025-07:00
name:              tx_port_bw_th_exceed
code:             11551
level:            note
port:             9
message:           Egress port bandwidth threshold exceeded at port=9
category:          system
time:               2021-09-27,06:59:34.747463-07:00
name:              tx_port_bw_th_not_exceed
code:             11552
level:            note
port:             9
message:           Egress port bandwidth threshold no longer exceeding at
port=9
```

To receive SNMP alerts when the bandwidth usage exceeds the configured threshold, you must configure an SNMP trap for port bandwidth monitoring. Use the `snmp-trap-enable-modify` command to enable or disable this SNMP trap.

To enable the SNMP trap for port bandwidth monitoring, use the command:

```
CLI (network-admin@switch) > snmp-trap-enable-modify   port-bw-threshold-
exceed-event
```

To disable the SNMP trap for port bandwidth monitoring, use the command:

```
CLI (network-admin@switch) > snmp-trap-enable-modify  no-port-bw-threshold-exceed-event
```

The SNMP trap message for the event where port bandwidth usage exceeds the threshold is:

```
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (317194) 0:52:51.94
SNMPv2-MIB::snmpTrapOID.0 = OID: SNMPv2-SMI::enterprises.47269.4.7.3.1
SNMPv2-SMI::enterprises.47269.4.7.1.1.2.2 = STRING: "egressPortBwThresholdExceeded"
SNMPv2-SMI::enterprises.47269.4.7.1.1.3.2 = STRING: "/nvOS/log/system.log"

SNMPv2-SMI::enterprises.47269.4.7.1.1.4.2 = Gauge32: 5
SNMPv2-SMI::enterprises.47269.4.7.1.1.5.2 = STRING: "2021-09-27,06:57:54.743025-07:00 aries-ext-01 nvOSd(3775) system tx_port_bw_th_exceed(11551) : level=note : port=9 : Egress port bandwidth threshold exceeded at port=9
```

You can also view the event where the port bandwidth usage exceeds the configured threshold by using the `port-show` command. For example, in the case where the bandwidth threshold is exceeded by the ingress traffic on port 23, the output of the `port-show` command is:

```
CLI (network-admin@switch) > port-show port 23
switch port status                                                  config
------  ----  ------------------------------------------------------  ------
switch  23    defer-bringup-wait,rx-bw-threshold-exceeded  fd,10g
```

You can also view the counters for the port bandwidth exceed event by issuing the `port-stats-show` command:

```
CLI (network-admin@switch) > port-stats-show format port,Rx-bw-th-exceed-count,Tx-bw-th-exceed-count,ibits,obits, port 41-42 show-interval 1
port  Rx-bw-th-exceed-count  Tx-bw-th-exceed-count  ibits  obits
----  ---------------------  ---------------------  -----  -----
41    0                      0                      131K   754K
42    0                      0                      164M   749K
port  Rx-bw-th-exceed-count  Tx-bw-th-exceed-count  ibits  obits
----  ---------------------  ---------------------  -----  -----
41    0                      0                      131K   754K
42    0                      0                      171M   749K
port  Rx-bw-th-exceed-count  Tx-bw-th-exceed-count  ibits  obits
----  ---------------------  ---------------------  -----  -----
41    0                      0                      132K   754K
42    1                      0                      178M   749K
```

Configuration Using Port Description instead of Port Number

With NetVisor OS version 7.0.0, you can use the port description as a port selector for configuring various features and functionalities. You can assign port descriptions and trunk descriptions to ports and use these descriptions to create VLAN, vLE, or vPG configurations.

While creating configurations using port or trunk descriptions, various scenarios that arise are discussed below in the context of VLAN creation.

As a basic configuration to illustrate different scenarios, create trunks `trunk1` and `trunk2` with different trunk descriptions:

```
CLI (network-admin@switch) > trunk-create name trunk1 ports 15 description
vlan_test_trunk1

CLI (network-admin@switch) > trunk-create name trunk2 ports 18,20
description  vlan_test_trunk1
```

Case 1: You cannot use a description that is shared by multiple trunks for VLAN configuration.

Display the trunk descriptions:

```
CLI (network-admin@switch) > trunk-show format trunk-id,ports,description
trunk-id ports        description
--------  -----------  ----------------
275      15           vlan_test_trunk1
276       18,20        vlan_test_trunk1
```

Display the port descriptions:

```
CLI (network-admin@switch) > port-show port 15,18,20
switch port bezel-port status config        description      trunk
------ ---- ---------- ------ ------------- ---------------- ------
switch 15   15         trunk  fd,10g,autoneg vlan_test_trunk1 trunk1
switch 18   18         trunk  fd,10g         vlan_test_trunk1 trunk2
switch 20   20         trunk  fd,10g,autoneg vlan_test_trunk1 trunk2
```

Try creating a VLAN by using the trunk description shared by two trunks:

```
CLI (network-admin@switch) > vlan-create id 2 scope local port-desc
vlan_test_trunk1
vlan-create: Multiple ports found with description 'vlan_test_trunk1'
```

Case 2: You cannot use a description that is shared by a trunk port and a trunk member port for VLAN configuration.

Modify the configuration such that a trunk port and a member port from the other trunk have the same description:

```
CLI (network-admin@switch) > trunk-modify name trunk2 ports 18,20
description  vlan_test_trunk2
CLI (network-admin@switch) > port-config-modify port 18 description
vlan_test_trunk1
```

Display the trunk descriptions:

```
CLI (network-admin@switch) > trunk-show format trunk-id,ports,description,
trunk-id ports        description
--------  -----------  ----------------
275      15           vlan_test_trunk1
276       18,20        vlan_test_trunk2
```

Display the port descriptions:

```
CLI (network-admin@switch) > port-show port 15,18,20
switch port bezel-port status config        description      trunk
------ ---- ---------- ------ ------------- ---------------- ------
switch 15   15         trunk  fd,10g,autoneg vlan_test_trunk1 trunk1
switch 18   18         trunk  fd,10g         vlan_test_trunk1 trunk2
switch 20   20         trunk  fd,10g,autoneg vlan_test_trunk2 trunk2
```

Try creating a VLAN with a description that is shared by a trunk and a member port:

```
CLI (network-admin@switch) > vlan-create id 2 scope local port-desc
vlan_test_trunk1
vlan-create: Multiple ports found with description 'vlan_test_trunk1'
```

Case 3: You cannot use a description that is shared by a trunk port and a physical port for VLAN configuration.

Modify the current configuration such that a trunk port and a physical port has the same description:

```
CLI (network-admin@switch) > trunk-modify name trunk2 ports 18 description
vlan_test_trunk2
CLI (network-admin@switch) > port-config-modify port 20 description
vlan_test_trunk1
```

Display the trunk descriptions:

```
CLI (network-admin@switch) > trunk-show format trunk-id,ports,description,
trunk-id ports        description
--------  -----------  ----------------
275      15           vlan_test_trunk1
276      18           vlan_test_trunk2
```

Display the port descriptions:

```
CLI (network-admin@switch) > port-show port 15,18,20
switch port bezel-port status config        description      trunk
------ ---- ---------- ------ ------------- ---------------- ------
switch 15   15         trunk  fd,10g,autoneg vlan_test_trunk1 trunk1
switch 18   18         trunk  fd,10g         vlan_test_trunk2 trunk2
switch 20   20                fd,10g,autoneg vlan_test_trunk1
```

Try creating a VLAN using the port description shared by a trunk and a physical port:

```
CLI (network-admin@switch) > vlan-create id 2 scope local port-desc
vlan_test_trunk1
```

```
vlan-create: Multiple ports found with description 'vlan_test_trunk1'
```

Case 4: You can use a description shared by a trunk and all its member ports for configuring a VLAN.

Modify the current configuration such that a trunk port and its member ports have the same description:

```
CLI (network-admin@switch) > trunk-modify name trunk2 ports 18,20
description  vlan_test_trunk2
```

Display the trunk descriptions:

```
CLI (network-admin@switch) > trunk-show format trunk-id,ports,description
trunk-id ports        description
--------  -----------  ----------------
275      15           vlan_test_trunk1
276       18,20        vlan_test_trunk2
```

Display the port descriptions:

```
CLI (network-admin@switch) > port-show port 15,18,20
switch port bezel-port status config        description      trunk
------ ---- ---------- ------ -------------- ---------------- ------
switch 15   15         trunk  fd,10g,autoneg vlan_test_trunk1 trunk1
switch 18   18         trunk  fd,10g         vlan_test_trunk2 trunk2
switch 20   20         trunk  fd,10g,autoneg vlan_test_trunk2 trunk2
```

Create a VLAN using the port description of the trunk member ports:

```
CLI (network-admin@switch) > vlan-create id 2 scope local port-desc
vlan_test_trunk2
Vlans 2 created
```

Display the VLAN configuration for the ports:

```
CLI (network-admin@switch) > port-vlan-show vlans 2 ports 18,20
switch  bezel-port  port  vlans  untagged-vlan  description        active-vlans
------  ----------  ----  -----  -------------  ---------------  ------------
switch 18         18   1-2   1            vlan_test_trunk2 none
switch 20         20   1-2   1            vlan_test_trunk2 none
```

Case 5: You cannot use a port description shared by member ports of different trunks (with no trunk description) to configure a VLAN.

Modify the current configuration such that the port descriptions of member ports of two different trunks are the same:

```
CLI (network-admin@switch) > trunk-modify name trunk1 ports 15 description
""
CLI (network-admin@switch) > trunk-modify name trunk2 ports 18,20
description  ""
CLI (network-admin@switch) > port-config-modify port 15 description
vlan_test_trunk_member
CLI (network-admin@switch) > port-config-modify port 18 description
vlan_test_trunk_member
```

Display the trunk descriptions:

```
CLI (network-admin@switch) > trunk-show format trunk-id,ports,description
trunk-id  ports
--------  ------
275       15
276       18,20
```

Display the port descriptions:

```
CLI (network-admin@switch) > port-show port 15,18,20
switch port bezel-port status config          description              trunk
------ ---- ---------- ------ -------------  ---------------------  ------
switch 15   15           trunk  fd,10g,autoneg vlan_test_trunk_member trunk1
switch 18   18          trunk  fd,10g          vlan_test_trunk_member trunk2
switch 20   20         trunk  fd,10g,autoneg                        trunk2
```

```
CLI (network-admin@switch) > vlan-create id 2 scope local port-desc
vlan_test_trunk_member
vlan-create: Multiple ports found with description 'vlan_test_trunk_member'
```

Case 6: You can use a unique trunk member port description (with no trunk description) to configure a VLAN.

Modify the current configuration such that there is a unique port description for a trunk member port:

```
CLI (network-admin@switch) > port-config-modify port 18 description
vlan_test_trunk2
```

Display the trunk descriptions:

```
CLI (network-admin@switch) > trunk-show format trunk-id,ports,description
trunk-id  ports
--------  ------
275       15
276       18,20
```

Display the port descriptions:

```
CLI (network-admin@switch) > port-show port 15,18,20
switch        port bezel-port status config          description             trunk
------------ ---- ---------- ------ -------------  ---------------------  ------
switch 15    15           trunk  fd,10g,autoneg vlan_test_trunk_member trunk1
switch 18   18         trunk  fd,10g         vlan_test_trunk2       trunk2
switch 20   20         trunk  fd,10g,autoneg                        trunk2
```

Create a VLAN using a unique trunk member description:

```
CLI (network-admin@switch) > vlan-create id 2 scope local port-desc
vlan_test_trunk_member
Vlans 2 created
```

Display the VLAN configuration for the ports:

```
CLI (network-admin@switch) > port-vlan-show vlans 2 ports 15
switch        bezel-port port vlans untagged-vlan description             active-vlans
```

```
-----------  ----------  ----  -----  -------------  ----------------------  ------------
switch       15          15    1-2    1              vlan_test_trunk_member none
```

Configuring Layer 2 Features

This chapter provides information about the protocols and configurations supported on Layer 2 network by using the NetVisor OS command line interface (CLI) on a NetVisor OS switch.

Understanding the Supported Layer 2 Protocols in NetVisor OS

Layer 2 (the Data Link layer) enables the transfer of data between adjacent nodes in a network segment, such as local or wide area networks. Layer 2 frames do not cross the boundaries of the local network. Services provided by Layer 2 include, but are not limited to: frame encapsulation, device addressing, error detection, packet forwarding, loop prevention, flow control, frame queuing, Quality of Service (QoS), and traffic segmentation through Virtual LANs (VLANs).

NetVisor OS supports the following Layer 2 protocols and functionalities:

- Spanning Tree Protocol (STP)
- Rapid Spanning Tree Protocol (RSTP)
- Multiple Spanning Tree Protocol (MSTP)
- Link Aggregation (LAG)
- virtual Link Aggregation (vLAG)
- Link Aggregation Control Protocol (LACP)
- Link Layer Discovery Protocol (LLDP)
- Root Guard
- Fabric Guard

Configuring LLDP

The Link Layer Discovery Protocol (LLDP) is an open, vendor-independent protocol that advertises a device's identity, abilities, and neighboring devices connected within the Local Area Network. This protocol is based on IEEE 802.1ab standard. An LLDP device sends information as Ethernet frames at regular intervals and each frame contains one LLDP Data Unit (LLDPDU) which is a sequence of different Type-Length-Value (TLV) structures. These frames start with the mandatory TLVs that include the Chassis ID, Port ID, and Time-To-Live (TTL) followed by a number of optional TLVs.

NetVisor OS provides a generic LLDP ON/OFF toggle function set at the system level. LLDP is enabled on a switch by default and without the generic function, you must disable LLDP configuration on all ports to disable LLDP on a switch. This resets all related configurations of LLDP protocol settings at the port level and LLDP vFlows. Use the following CLI command to enable or disable the protocol at the system level:

```
CLI (network-admin@leaf1) > system-settings-modify [lldp|no-lldp]
```

LLDP packets are processed on the CPU after being directed there by specific vFlow policies. To clear all LLDP protocol system redirects, use the parameter `no-lldp`. To add all LLDP protocol system redirects, use the parameter `lldp`. This approach ensures that port LLDP configurations are not disturbed.

To verify the LLDP status, use the command:

```
CLI (network-admin@leaf1) > system-settings-show
switch:                      leaf1
optimize-arps:               off
lldp:                        off
policy-based-routing:        off
optimize-nd:                 off
reactivate-mac:              on
```

To enable/disable LLDP on one or more ports, use the command `port-lldp-modify.`

```
CLI (network-admin@Leaf1) > port-lldp-modify
```

`port-lldp-modify`	Modify the LLDP setting on ports.
Specify the following options:	
`port` *`port-list`*	Specify the ports on which LLDP has to be enabled or disabled as a list separated by commas.
`lldp|no-lldp`	Specify the option to enable or disable LLDP.

```
CLI (network-admin@switch) > port-lldp-modify port <port-list> [lldp|no-lldp]
```

`lldp-show` command displays the LLDP information of the neighbors in the network.

```
CLI (network-admin@Leaf1) > lldp-show
```

`lldp-show`	This command displays LLDP information.
Specify any of the following options to view the LLDP information specific to that parameter.	

`local-port` *`local-port-number`*	Specify a local port number.
`bezel-port` *`bezel-port-string`*	Specify the bezel port number.
`chassis-id` *`chassis-id-string`*	Specify the chassis ID of the neighboring device.
`port-id` *`port-id-string`*	Specify the port ID of the neighboring device.
`sys-name` *`sys-name-string`*	Specify the system name of the neighboring device.

For example:

```
CLI (network-admin@SW1) > lldp-show
switch        local-port bezel-port chassis-id port-id port-desc           sys-name
------------- ---------- ---------- ---------- ------- ------------------ --------
SW1           17         17         090009ef   17      PN Switch Port(17)  proto-1
SW1           121        121        090009ee   121     PN Switch Port(121) proto-2
SW2           13         13         090009ef   13      PN Switch Port(13)  proto-1
SW2           117        117        090009ee   117     PN Switch Port(117) proto-2
```

`port-lldp-show` command displays the LLDP status on a port basis.

`CLI (network-admin@Leaf1) > port-lldp-show`

`port-lldp-show`	Display LLDP configuration on the ports. This command when executed without any parameters displays the LLDP status (on/off) of all the ports on the switch.
Specify any of the following parameters to view the LLDP information specific to that parameter.	
`port <port-list>`	Specify the list of ports separated by commas.
`lldp|no-lldp`	Use the option `lldp` to view all the ports on which LLDP is enabled. Use the option `no-lldp` to view all the ports on which LLDP is disabled.

For example:

```
CLI (network-admin@Leaf1) > port-lldp-show port 12,13,14,15
switch        port lldp
------------- ---- ----
Leaf1         12   on
Leaf1         13   on
Leaf1         14   on
Leaf1         15   on
```

Understanding and Configuring VLANs

A Virtual Local Area Network (VLAN) enables devices to be segmented into logically separate broadcast domains within the same LAN. VLANs improve network performance by directing network traffic only to the parts of the network that need to receive it. Network segments so created keep traffic isolated based on the respective VLAN IDs associated to the transmitted frames. Applying targeted security features to specific network areas is also made simpler through the use of VLANs.

As per the standards, NetVisor OS uses the 12-bit field in the header of each packet as a VLAN identifier or VLAN tag. The maximum number of VLANs that can be defined is 4092. VLANs 4093, 4094, and 4095 are reserved for internal use while VLAN 1 is the default fabric VLAN for untagged traffic. Untagged packets can be mapped to any VLAN, but NetVisor OS maps this traffic to VLAN 1 by default.

Configuring an untagged VLAN is necessary while connecting a switch to devices that do not support IEEE802.1Q VLAN tags. The ports on a switch can be configured to automatically map untagged packets to a specific VLAN. NetVisor OS also allows you to block untagged traffic on a port basis, that is, the untagged VLAN on a port can be removed or deleted.

About VLAN 1

- VLAN 1 is enabled on all ports by default. However, VLAN 1 can be removed from any port on which it is the untagged VLAN. Now, the port has no untagged VLANs and all untagged traffic is dropped on that port.
- To generalize the point above, if VLAN x is the untagged VLAN for a port and if VLAN x is removed from that port, then the port has no untagged VLAN and all untagged traffic is dropped on that port.
- VLAN 1 can also function as a tagged VLAN for a port. This happens automatically in cases where VLAN 1 is the default untagged VLAN, and then another VLAN is configured as an untagged VLAN on the port.
- VLAN 1 cannot be created or deleted. VLAN 1 configuration is stored in persistent storage.

The default fabric VLAN can be changed from VLAN 1 to another VLAN ID using the command `fabric-local-modify`. For example, to set VLAN 20 as the default fabric VLAN, use the command:

```
CLI (network-admin@Leaf1) > fabric-local-modify vlan 20
```

Warning: If you create a VLAN with scope fabric and configure it as the untagged VLAN on all ports, it can disrupt the fabric communication.

Note: The untagged VLAN feature is not the same as the default VLAN using the IEEE 802.1Q tag 1.

The `vlan-create` command creates VLANs on the current switch.

```
CLI(network-admin@Leaf1) > vlan-create
```

`vlan-create`	Creates a VLAN. You can create a VLAN either by specifying a VLAN ID or by specifying a range of VLAN IDs.
`id 2...4092`	Specify the VLAN ID between 2 and 4092. Note: VLAN 0 and 1 represents all untagged or non-VLAN traffic, VLANs 4093, 4094, and 4095

	are reserved for internal use.
`range` *`vlan-list`*	Specify the range of VLAN IDs. Use this parameter if you want to specify a VLAN range instead of a VLAN ID.
`scope [local|cluster|fabric]`	Specify the VLAN scope as local, cluster, or fabric.
Specify any of the following options:	
`vnet` *`vnet-name`*	Specify the vNET name for this VLAN. Note: A vNET segregates a physical fabric into many logical networks, each with separate resources, network services, and Quality of Service (QoS) guarantees.
`vxlan 1..`*`16777215`*	Specify the VXLAN identifier for the tunnel.
`auto-vxlan|no-auto-vxlan`	Specify if you want to enable or disable `auto-vxlan`. Enabling this option automatically assigns a user-defined VLAN/VNI mapping to all VTEP connections in the fabric. If the vxlan option is not specified, the software assigns a VNI automatically.
`vxlan-mode [standard|transparent|qinq-access]`	Specify the VXLAN encapsulation mode as standard, transparent, or Q-in-Q.
`replicators [vtep-group name|none]`	Specify the replicator group. Provide a VTEP group name to add a replicator. Specify `none` to not add a replicator or remove a configured replicator.
`public-vlan` *`2..4092`*	Specify the Public VLAN for vNET VLAN.
`description` *`description-string`*	Provide a VLAN description.
`stats|no-stats`	Use the options to enable or disable statistics collection for the VLANs being created.
`ports` *`port-list`*	Specify the ports assigned to the VLAN as list separated by commas.
`port-desc` *`port-desc-string`*	Specify a port description.
`untagged-ports` *`port-list`*	Specify the untagged ports assigned to the VLAN as a list separated by commas.
`untagged-port-desc` *`untagged-port-desc-string`*	Specify a description for untagged ports.

Note: NetVisor OS allows you to create a large number of VLANs by using the `vlan-create` command and the `range` keyword. However, in large network topologies with several nodes with heavy CPU traffic, the CLI may timeout if you create large number of VLANs. In such scenarios, try creating smaller number of VLANs.

By default, all ports are tagged on a newly created VLAN. However, if you want to specify select ports that

should be trunked, then use the optional parameter `ports` with a comma separated list of ports, or specify a range of ports.

In some cases, you may not want a VLAN to be created on all ports. You can specify the port parameter as `none` to apply the VLAN only to the internal ports. For example:

```
CLI (network-admin@Leaf1) > vlan-create id 35 scope fabric ports none
```

To delete an existing VLAN, use the command:

```
CLI (network-admin@Leaf1) > vlan-delete
```

`vlan-delete`	Deletes a VLAN either by ID or by a range of IDs.
`id 2...4092`	Specify the VLAN ID that you want to delete.
`range vlan-list`	Specify the range of VLAN IDs that you want to delete. Use this parameter instead of `id`, if you want to specify a VLAN range.
Specify the following option:	
`vnet vnet-name`	Specify the name of the vNET from which the VLANs are to be deleted.

Configuration of an existing VLAN can be modified using the `vlan-modify` command.

```
CLI (network-admin@Leaf1) >vlan-modify
```

`vlan-modify`	Modify a VLAN by specifying the VLAN ID.
`id 2...4092`	Specify the VLAN ID that you intend to modify.
between 1 and 4 of the following options:	
`description description-string`	Provide a VLAN description.
`vxlan 1..16777215`	Specify the VXLAN identifier for the tunnel.
`replicators [vtep-group name | none]`	Specify the replicator group. Provide a VTEP group name to add a replicator. Specify `none` to not add a replicator or remove a configured replicator.
`vnet vnet name`	Specify the vNET name for this VLAN.
`public-vlan 2..4092`	Specify the public VLAN ID for vNET VLAN. Note: Public VLAN ID can only be specified for private VLANs.

For example, to modify VLAN25 description from blue to red:

```
CLI (network-admin@Leaf1) > vlan-modify id 25 description red
```

This description can be removed from VLAN25 using the command:

```
CLI (network-admin@Leaf1) > vlan-modify id 25 description ""
```

NetVisor OS allows the addition of ports to a VLAN through the vlan-port-add command.

```
CLI (network-admin@Leaf1) > vlan-port-add
```

`vlan-port-add`	Add ports to VLANs.
Specify one of the following VLAN parameters:	
`vlan-id` *2..4092*	Specify the VLAN ID to which ports are to be added.
`vlan-range` *vlan-list*	Specify the range of VLAN IDs to which ports are to be added.
`vlan-vnet` *vnet-name*	Specify the vNET for the VLANs to which the ports are to be added.
Provide the following port arguments:	
`switch` *switch-name*	Specify the name of the switch on which the ports are located.
`ports` *port-list*	Specify the ports that need to be added to the VLANs as a list separated by commas.
`port-desc` *port-desc-string*	Specify a previously configured port description.
`[untagged | tagged]`	Specify either of the options to configure the ports as untagged or tagged ports.

For example, to configure ports 17 and 18 to accept untagged packets and map them to VLAN 595, use the following command:

```
CLI (network-admin@Leaf1) > vlan-port-add vlan-id 595 ports 17,18 untagged
```

To map ports on different switches into the `scope fabric` VLAN, use the following command:

```
CLI (network-admin@Leaf1) > vlan-port-add vlan-id 1-4095 switch switch-
name ports port-list
```

Ports can be removed from a VLAN through the `vlan-port-remove` command.

```
CLI (network-admin@Leaf1) > vlan-port-remove
```

`vlan-port-remove`	Remove ports from VLANs.
Specify one of the following VLAN sectors:	
`vlan-id` *2..4092*	Specify the VLAN ID from which ports are to be removed.
`vlan-range` *vlan-list*	Specify the range of VLAN IDs from which ports

	are to be removed
`vlan-vnet` *`vnet name`*	Specify the vNET for the VLANs from which ports are to be removed.
Provide the following port arguments:	
`switch` *`switch name`*	Specify the name of the switch on which the ports are located.
`port` *`port list`*	Specify the ports that need to be removed from the VLANs as a list separated by commas.
`port-desc` *`port-desc-string`*	Specify the port description.

The `vlan-show` command displays the VLAN information.

```
CLI (network-admin@Leaf1) > vlan-show
```

`vlan-show`	Display VLAN information.
`Specify one of the following VLAN sectors:`	
`id` *`2..4092`*	Specify the VLAN ID for which the information has to be displayed.
`range` *`vlan-list`*	Specify the range of VLAN IDs for which the information has to be displayed.
`vnet` *`vnet name`*	Specify the vNET for which VLAN information has to be displayed.
`type [public | private]`	Specify either of the type options to display information for public VLANs or private VLANs.
`vxlan` *`1..16777215`*	Specify the VXLAN identifier for the tunnel.
`vxlan-mode [standard | transparent | qinq-access]`	Specify any of the VXLAN modes to display information for standard, transparent, or q-in-q modes.
`hw-vpn` *`hw-vpn-number`*	Specify the hardware VPN number to display the related information.
`hw-mcast-group` *`hw-mcast-group-number`*	Specify the hardware multi-cast group number to display the related information.
`replicators [vtep-group name | none]`	Provide a VTEP group name to view the VLAN information for that replicator group. Specify `none` to view the information on VLANs that do not involve replicator groups.
`repl-vtep` *`ip-address`*	Specify the IP address of the replicator VTEP to view the related information.
`public-vlan` *`2..4092`*	Specify the public VLAN ID for vNET VLAN to view

	the related information.
`scope [local | cluster | fabric]`	Provide any of the scope options to view the information on VLANs with that specified scope.
`description` *`description-string`*	Specify a description to view the information on VLANs with that specific description.
`active [yes | no]`	Specify `yes` to view information on active VLANs. Specify `no` to view information on inactive VLANs.
`stats|no-stats`	Specify VLAN statistics.
`vrg` *`vrg-name`*	Specify the VRG assigned to VLAN.
`ports` *`port-list`*	Specify the ports assigned to VLAN.
`port-desc` *`port-desc-string`*	Description for the port.
`untagged-ports` *`port-list`*	Specify the untagged ports assigned to VLAN.
`untagged-port-desc` *`untagged-port-desc-string`*	The untagged ports assigned to VLAN.
`active-edge-ports` *`port-list`*	Specify the active edge ports assigned to VLAN.

For example: `CLI (network-admin@Leaf1) > vlan-show layout vertical`

```
switch:            leaf1
id:                1
type:              public
auto-vxlan:        no
replicators:       none
scope:             local
description:       default-1
active:            yes
stats:             yes
ports:             2-72
untagged-ports:    2-69
active-edge-ports: 69-70

CLI (network-admin@Leaf1) > vlan-show format all layout vertical

switch:            leaf1
id:                1
type:              public
auto-vxlan:        no
hw-vpn:            0
hw-mcast-group:    0
replicators:       none
repl-vtep:         ::
scope:             local
description:       default-1
active:            yes
stats:             yes
```

```
vrg:                0:0
ports:              2-72
untagged-ports:     2-69
active-edge-ports:  69
```

Network traffic statistics per VLAN can be displayed using the `vlan-stats-show` command. This command may be useful when troubleshooting network issues.

```
CLI (network-admin@Leaf1) > vlan-stats-show format all layout vertical
switch: Leaf2
time: 10:51:02
vlan: 1
vnet:
ibytes: 36.2T
ipkts: 89.0G
idrops-bytes: 119M
idrops-pkts: 313K
obytes: 0
opkts: 0
odrops-bytes: 0
odrops-pkts: 0
switch: Leaf2
time: 10:51:02
vlan: 35
vnet:
ibytes: 10.8K
ipkts: 154
idrops-bytes: 0
idrops-pkts: 0
obytes: 0
opkts: 0
odrops-bytes: 0
odrops-pkts: 0
switch: Leaf1
time: 10:51:02
vlan:               1
vnet:
ibytes: 34.9T
ipkts: 84.6G
idrops-bytes: 3.03M
idrops-pkts: 5.69K
obytes: 0
opkts: 0
odrops-bytes: 0
odrops-pkts: 0
```

The output displays the following information:

- `switch` — switch name
- `time` — when the output was generated
- `VLAN ID` — ID assigned to the VLAN

- `vnet` — the vNET assigned to the VLAN
- `incoming and outgoing bytes` — in K (Kilobytes), M (Megabytes), or G (Gigabytes)
- `incoming and outgoing packets` — number of packets incoming and outgoing
- `incoming and outgoing dropped bytes` — in K (Kilobytes), M (Megabytes), or G (Gigabytes)
- `incoming and outgoing dropped packets` — number of dropped packets incoming and outgoing

Understanding and Configuring VLANs

A Virtual Local Area Network (VLAN) enables devices to be segmented into logically separate broadcast domains within the same LAN. VLANs improve network performance by directing network traffic only to the parts of the network that need to receive it. Network segments so created keep traffic isolated based on the respective VLAN IDs associated to the transmitted frames. Applying targeted security features to specific network areas is also made simpler through the use of VLANs.

As per the standards, NetVisor OS uses the 12-bit field in the header of each packet as a VLAN identifier or VLAN tag. The maximum number of VLANs that can be defined is 4092. VLANs 4093, 4094, and 4095 are reserved for internal use while VLAN 1 is the default fabric VLAN for untagged traffic. Untagged packets can be mapped to any VLAN, but NetVisor OS maps this traffic to VLAN 1 by default.

Configuring an untagged VLAN is necessary while connecting a switch to devices that do not support IEEE802.1Q VLAN tags. The ports on a switch can be configured to automatically map untagged packets to a specific VLAN. NetVisor OS also allows you to block untagged traffic on a port basis, that is, the untagged VLAN on a port can be removed or deleted.

About VLAN 1

- VLAN 1 is enabled on all ports by default. However, VLAN 1 can be removed from any port on which it is the untagged VLAN. Now, the port has no untagged VLANs and all untagged traffic is dropped on that port.
- To generalize the point above, if VLAN x is the untagged VLAN for a port and if VLAN x is removed from that port, then the port has no untagged VLAN and all untagged traffic is dropped on that port.
- VLAN 1 can also function as a tagged VLAN for a port. This happens automatically in cases where VLAN 1 is the default untagged VLAN, and then another VLAN is configured as an untagged VLAN on the port.
- VLAN 1 cannot be created or deleted. VLAN 1 configuration is stored in persistent storage.

The default fabric VLAN can be changed from VLAN 1 to another VLAN ID using the command `fabric-local-modify`. For example, to set VLAN 20 as the default fabric VLAN, use the command:

```
CLI (network-admin@Leaf1) > fabric-local-modify vlan 20
```

Warning: If you create a VLAN with scope fabric and configure it as the untagged VLAN on all ports, it can disrupt the fabric communication.

Note: The untagged VLAN feature is not the same as the default VLAN using the IEEE 802.1Q tag 1.

The `vlan-create` command creates VLANs on the current switch.

```
CLI(network-admin@Leaf1)  >  vlan-create
```

`vlan-create`	Creates a VLAN. You can create a VLAN either by specifying a VLAN ID or by specifying a range of VLAN IDs.
`id 2...4092`	Specify the VLAN ID between 2 and 4092. Note: VLAN 0 and 1 represents all untagged or non-VLAN traffic, VLANs 4093, 4094, and 4095 are reserved for internal use.
`range vlan-list`	Specify the range of VLAN IDs. Use this parameter if you want to specify a VLAN range instead of a VLAN ID.
`scope [local|cluster|fabric]`	Specify the VLAN scope as local, cluster, or fabric.
Specify any of the following options:	
`vnet vnet-name`	Specify the vNET name for this VLAN. Note: A vNET segregates a physical fabric into many logical networks, each with separate resources, network services, and Quality of Service (QoS) guarantees.
`vxlan 1..16777215`	Specify the VXLAN identifier for the tunnel.
`auto-vxlan|no-auto-vxlan`	Specify if you want to enable or disable `auto-vxlan`. Enabling this option automatically assigns a user-defined VLAN/VNI mapping to all VTEP connections in the fabric. If the vxlan option is not specified, the software assigns a VNI automatically.
`vxlan-mode [standard|transparent|qinq-access]`	Specify the VXLAN encapsulation mode as standard, transparent, or Q-in-Q.
`replicators [vtep-group name|none]`	Specify the replicator group. Provide a VTEP group name to add a replicator. Specify `none` to not add a replicator or remove a configured replicator.
`public-vlan 2..4092`	Specify the Public VLAN for vNET VLAN.
`description description-string`	Provide a VLAN description.
`stats|no-stats`	Use the options to enable or disable statistics collection for the VLANs being created.
`ports port-list`	Specify the ports assigned to the VLAN as list separated by commas.
`port-desc port-desc-string`	Specify a port description to be assigned to the VLAN.
`untagged-ports port-list`	Specify the untagged ports assigned to the VLAN as a list separated by commas.

`untagged-port-desc` *`untagged-port-desc-string`*	Specify a description for untagged ports.

Note: NetVisor OS allows you to create a large number of VLANs by using the `vlan-create` command and the `range` keyword. However, in large network topologies with several nodes with heavy CPU traffic, the CLI may timeout if you create large number of VLANs. In such scenarios, try creating smaller number of VLANs.

By default, all ports are tagged on a newly created VLAN. However, if you want to specify select ports that should be trunked, then use the optional parameter `ports` with a comma separated list of ports, or specify a range of ports.

In some cases, you may not want a VLAN to be created on all ports. You can specify the port parameter as `none` to apply the VLAN only to the internal ports. For example:

```
CLI (network-admin@Leaf1) > vlan-create id 35 scope fabric ports none
```

To delete an existing VLAN, use the command:

```
CLI (network-admin@Leaf1) > vlan-delete
```

`vlan-delete`	Deletes a VLAN either by ID or by a range of IDs.
`id` *2...4092*	Specify the VLAN ID that you want to delete.
`range` *`vlan-list`*	Specify the range of VLAN IDs that you want to delete. Use this parameter instead of `id`, if you want to specify a VLAN range.
Specify the following option:	
`vnet` *`vnet-name`*	Specify the name of the vNET from which the VLANs are to be deleted.

Configuration of an existing VLAN can be modified using the `vlan-modify` command.

```
CLI (network-admin@Leaf1) >vlan-modify
```

`vlan-modify`	Modify a VLAN by specifying the VLAN ID.
`id` *2...4092*	Specify the VLAN ID that you intend to modify.
between 1 and 4 of the following options:	
`description` *`description-string`*	Provide a VLAN description.
`vxlan` *1..16777215*	Specify the VXLAN identifier for the tunnel.
`replicators [vtep-group name | none]`	Specify the replicator group. Provide a VTEP group name to add a replicator. Specify `none` to not add a replicator or remove a configured replicator.

`vnet` *`vnet name`*	Specify the vNET name for this VLAN.
`public-vlan` *`2..4092`*	Specify the public VLAN ID for vNET VLAN. Note: Public VLAN ID can only be specified for private VLANs.

For example, to modify VLAN 25 description from blue to red:

```
CLI (network-admin@Leaf1) > vlan-modify id 25 description red
```

This description can be removed from VLAN25 using the command:

```
CLI (network-admin@Leaf1) > vlan-modify id 25 description ""
```

NetVisor OS allows the addition of ports to a VLAN through the vlan-port-add command.

```
CLI (network-admin@Leaf1) > vlan-port-add
```

`vlan-port-add`	Add ports to VLANs.
Specify one of the following VLAN parameters:	
`vlan-id` *`2..4092`*	Specify the VLAN ID to which ports are to be added.
`vlan-range` *`vlan-list`*	Specify the range of VLAN IDs to which ports are to be added.
`vlan-vnet` *`vnet-name`*	Specify the vNET for the VLANs to which the ports are to be added.
Provide the following port arguments:	
`switch` *`switch-name`*	Specify the name of the switch on which the ports are located.
`ports` *`port-list`*	Specify the ports that you want to add to the VLAN as a list separated by commas.
`port-desc` *`port-desc-string`*	Specify the port description of the port that you want to add to the VLAN.
`[untagged | tagged]`	Specify either of the options to configure the ports as untagged or tagged ports.

For example, to configure ports 17 and 18 to accept untagged packets and map them to VLAN 595, use the following command:

```
CLI (network-admin@Leaf1) > vlan-port-add vlan-id 595 ports 17,18 untagged
```

To map ports on different switches into the `scope fabric` VLAN, use the following command:

```
CLI (network-admin@Leaf1) > vlan-port-add vlan-id 1-4095 switch switch-
name ports port-list
```

To add ports with the description `red` to VLAN 70, use the command:

```
CLI (network-admin@switch) > vlan-port-add vlan-id 70 port-desc red
```

Ports can be removed from a VLAN through the `vlan-port-remove` command.

```
CLI (network-admin@Leaf1) > vlan-port-remove
```

`vlan-port-remove`	Remove ports from VLANs.
Specify one of the following VLAN selectors:	
`vlan-id` *2..4092*	Specify the VLAN ID from which ports are to be removed.
`vlan-range` *vlan-list*	Specify the range of VLAN IDs from which ports are to be removed
`vlan-vnet` *vnet name*	Specify the vNET for the VLANs from which ports are to be removed.
Provide the following port arguments:	
`switch` *switch name*	Specify the name of the switch on which the ports are located.
`port` *port list*	Specify the ports that you want to remove from the VLAN as a list separated by commas.
`port-desc` *port-desc-string*	Specify the port description of the port that you want to remove from the VLAN.

The `vlan-show` command displays the VLAN information.

```
CLI (network-admin@Leaf1) > vlan-show
```

`vlan-show`	Display VLAN information.
`Specify one of the following VLAN sectors:`	
`id` *2..4092*	Specify the VLAN ID for which the information has to be displayed.
`range` *vlan-list*	Specify the range of VLAN IDs for which the information has to be displayed.
`vnet` *vnet name*	Specify the vNET for which VLAN information has to be displayed.
`type [public | private]`	Specify either of the type options to display information for public VLANs or private VLANs.
`vxlan` *1..16777215*	Specify the VXLAN identifier for the tunnel.

`vxlan-mode [standard | transparent | qinq-access]`	Specify any of the VXLAN modes to display information for standard, transparent, or q-in-q modes.
`hw-vpn` *`hw-vpn-number`*	Specify the hardware VPN number to display the related information.
`hw-mcast-group` *`hw-mcast-group-number`*	Specify the hardware multi-cast group number to display the related information.
`replicators [vtep-group name | none]`	Provide a VTEP group name to view the VLAN information for that replicator group. Specify `none` to view the information on VLANs that do not involve replicator groups.
`repl-vtep` *`ip-address`*	Specify the IP address of the replicator VTEP to view the related information.
`public-vlan` *`2..4092`*	Specify the public VLAN ID for vNET VLAN to view the related information.
`scope [local | cluster | fabric]`	Provide any of the scope options to view the information on VLANs with that specified scope.
`description` *`description-string`*	Specify a description to view the information on VLANs with that specific description.
`active [yes | no]`	Specify `yes` to view information on active VLANs. Specify `no` to view information on inactive VLANs.
`stats|no-stats`	Specify VLAN statistics.
`vrg` *`vrg-name`*	Specify the VRG assigned to VLAN.
`ports` *`port-list`*	Specify the ports assigned to VLAN.
`port-desc` *`port-desc-string`*	Specify the port description.
`untagged-ports` *`port-list`*	Specify the untagged ports assigned to VLAN.
`untagged-port-desc` *`untagged-port-desc-string`*	The untagged ports assigned to VLAN.
`active-edge-ports` *`port-list`*	Specify the active edge ports assigned to VLAN.

For example:

```
CLI (network-admin@Leaf1) > vlan-show layout vertical
switch:            leaf1
id:                1
type:              public
auto-vxlan:        no
replicators:       none
scope:             local
description:       default-1
active:            yes
stats:             yes
```

```
ports:              2-72
untagged-ports:     2-69
active-edge-ports:  69-70

CLI (network-admin@Leaf1) > vlan-show format all layout vertical
switch:             leaf1
id:                 1
type:               public
auto-vxlan:         no
hw-vpn:             0
hw-mcast-group:     0
replicators:        none
repl-vtep:          ::
scope:              local
description:        default-1
active:             yes
stats:              yes
vrg:                0:0
ports:              2-72
untagged-ports:     2-69
active-edge-ports:  69
```

To associate multiple VLANs with a port, use the `port-vlan-add` command:

`ports` *`port-number`*	Specify the port number.
`port-desc` *`port-desc-string`*	Specify the description to add the port with that description to the VLAN.
Specify one or both of the following options:	
`vnet` *`vnet-name`*	Specify the name of the VNET for the port.
`vlans` *`vlans-list`*	Specify the list of VLANs to associate with the port.
`untagged-vlan` *`vlan-id`*	Specify an untagged VLAN to apply to the port. The value can be from 0-4095.

For example, to associate the VLANs 10 and 12 with port 25, use the command:

```
CLI (network-admin@switch) > port-vlan-add port 25 vlans 10,12
```

Use the `port-vlan-show` command to display this configuration.

```
CLI (network-admin@switch) > port-vlan-show
```

`port-vlan-show`	Display ports and the associated VLANs.
`vnet` *`vnet-name`*	The name of the vNET.
`bezel-port` *`bezel-port-string`*	The bezel port number.
`ports` *`port-list`*	The list of ports.
`vlans` *`vlans-list`*	The list of VLANs associated with the port.

`untagged-vlan` *`vlan-id`*	The untagged VLANs associated with the port.
`description` *`description-string`*	The VLAN description.
`port-desc` *`port-desc-string`*	The port description.
`active-vlans` *`vlan-list`*	The active VLAN list.

```
CLI (network-admin@switch) > port-vlan-show
switch bezel-port port vlans untagged-vlan description active-vlans
------ ---------- ---- ----- ------------ ----------- ------------
switch 25         25   10,12 1                       none
```

Use the `port-vlan-remove` command to remove a port from the VLANs.

`port-vlan-remove`	Removes a port from VLANs.
`vlans` *`vlans-list`*	Specify the list of VLANs.
`vnet` *`vnet-name`*	Specify the name of the vNET.
`port` *`port-number`*	Specify the port to remove.
`port-desc` *`port-desc-string`*	Specify the port description to remove the port with that description.

For example, you can remove the ports with the description `green` from VLANs 45 and 46 by using the command:

```
CLI (network-admin@switch) > port-vlan-remove vlans 45,46 port-desc green
```

Network traffic statistics per VLAN can be displayed using the `vlan-stats-show` command. This command may be useful when troubleshooting network issues.

```
CLI (network-admin@Leaf1) > vlan-stats-show format all layout vertical
switch: Leaf2
time: 10:51:02
vlan: 1
vnet:
ibytes: 36.2T
ipkts: 89.0G
idrops-bytes: 119M
idrops-pkts: 313K
obytes: 0
opkts: 0
odrops-bytes: 0
odrops-pkts: 0
switch: Leaf2
time: 10:51:02
vlan: 35
vnet:
ibytes: 10.8K
ipkts: 154
idrops-bytes: 0
idrops-pkts: 0
```

```
obytes:  0
opkts:  0
odrops-bytes:  0
odrops-pkts:  0
switch:  Leaf1
time:  10:51:02
vlan:              1
vnet:
ibytes:  34.9T
ipkts:  84.6G
idrops-bytes:  3.03M
idrops-pkts:  5.69K
obytes:  0
opkts:  0
odrops-bytes:  0
odrops-pkts:  0
```

The output displays the following information:

- `switch` — switch name
- `time` — when the output was generated
- `VLAN ID` — ID assigned to the VLAN
- `vnet` — the vNET assigned to the VLAN
- `incoming and outgoing bytes` — in K (Kilobytes), M (Megabytes), or G (Gigabytes)
- `incoming and outgoing packets` — number of packets incoming and outgoing
- `incoming and outgoing dropped bytes` — in K (Kilobytes), M (Megabytes), or G (Gigabytes)
- `incoming and outgoing dropped packets` — number of dropped packets incoming and outgoing

Configuring Rapid Spanning Tree Protocol (RSTP)

Rapid Spanning Tree Protocol (RSTP), a standard inter-switch protocol, ensures a loop-free forwarding network topology at Layer 2. This protocol was defined by the IEEE 802.1w standard and is an extension of the 802.1D Spanning Tree Protocol (STP). RSTP is an improvement over STP as it provides faster convergence after a network topology change or failure. RSTP introduces new port roles, and the original five port states of STP are reduced to three.

To build a loop-free topology, switches (bridges) determine the root bridge and compute the port roles. To do this, the bridges use special data frames called Bridge Protocol Data Units (BPDUs) that exchange bridge IDs and root path cost information. BPDUs are exchanged regularly, typically at two second intervals, and enable switches to keep track of network topology changes and to start and stop forwarding on ports as required. Hosts should not send BPDUs to the switch ports and to avoid malfunctioning/malicious hosts from doing so, the switch can filter or block BPDUs. If you enable BPDU filtering on a port, BPDUs received on that port are dropped but other network traffic is forwarded as usual. If you enable BPDU blocking on a port, BPDUs received on that port are dropped and the port is shut down.

Port Roles in RSTP

Root Port (one per bridge): The forwarding port on each bridge which is on the best path to reach the root bridge.
Designated Port: The forwarding port for each LAN segment that leads away from the root bridge.
Alternate Port: An alternative path to the root bridge on a particular LAN segment, which is part of a bridge other than the one that has a designed port for the LAN segment. Alternate port is the second best root port.
Backup port: A backup/redundant port for the segment that already has one designated port. This port leads away from the root port.
Disabled: A port which is manually disabled and is not a part of STP.

Port States in RSTP

Discarding: No data is exchanged over the port.
Learning: Frames are not forwarded, but the MAC address table is populated.
Forwarding: Fully functional.

Switches in RSTP expect a BPDU every 2 seconds (hello time) and if they do not receive a BPDU for 6 seconds (3 hello time intervals), it is considered to be a link failure. This is significantly faster than the STP link failure detection time of 20 seconds, dictated by the max age timer. RSTP can actively confirm if a port can safely be transitioned to the forwarding state without having to rely on the timer mechanism. Ports can be configured as edge ports if they are attached to a LAN that has no other bridges connected to it. Such a port can transition directly to the forwarding state, but it loses the edge port status as soon as it receives a BPDU. RSTP achieves rapid transition to the forwarding state on edge ports and point-to-point links (operating in full-duplex mode) but not on shared links (i.e., ports connected to a shared medium, hence operating in. half-duplex mode)

If network connections form loops and STP is disabled, packets are forwarded indefinitely across the switches, causing degradation of network performance. STP supports limited Layer 2 multipathing and can result in sub-optimal utilization of available network links. Therefore, a fabric of switches does not rely only on RSTP within the boundaries of the network. Arista Networks recommends the use of RSTP for ad hoc networks that inter-operate in a heterogeneous, multi-vendor switch environment.

> **Note:** RSTP is enabled on the switch by default.

Before you begin configuring RSTP, view the status of the protocol on the switch by using the command `stp-show`

```
CLI (network-admin@Leaf1) > stp-show
switch: Leaf1
enable: yes
stp-mode: rstp
bpdus-bridge-ports: yes
bridge-id: 3a:7f:b1:43:8a:0f
bridge-priority: 32768
hello-time: 2
forwarding-delay: 15
max-age: 20
cluster-mode: master
```

The `cluster-mode` of a switch in an STP cluster could be `master` or `slave`. The master in an STP cluster is elected on the basis of which node has been up longer. The other node is the slave.

To display the STP state, use the following command:

```
CLI (network-admin@Leaf1) > stp-state-show
```

`stp-state-show`	Displays the STP state information.
Specify one or more of the following options to view the information specific to those options. Specifying no parameter will display all the information.	
`vlan` *`vlan-list`*	Specify the VLANs as a list separated by commas.
`port` *`port-list`*	Specify the ports as a list separated by commas.
`instance-id` *`instance-id-number`*	Specify the STP instance ID.
`name` *`name-string`*	Specify the name of the STP instance.
`bridge-id` *`mac-address`*	Specify the bridge ID for which the information has to be displayed.
`bridge-priority` *`bridge-priority-number`*	Specify the bridge priority number.
`root-id` *`mac-address`*	Specify the root ID.
`root-priority` *`root-priority-number`*	Specify the STP root priority.
`root-port` *`root-port-number`*	Specify the STP root port.
`root-port(peer)` *`root-port(peer)-number`*	Specify the root port of the peer.
`hello-time` *`hello-time-number`*	Specify the STP hello time between 1s and 10s. The hello time is the time between each bridge protocol data unit (BPDU) that is sent on a port.

	The default hello time is 2s.
`forwarding-delay` *`forwarding-delay-number`*	Specify the STP forwarding delay between 4s and 30s. This is the time interval that is spent in the listening and learning states. Default forwarding delay timer is 15s.
`max-age` *`max-age-number`*	Specify the maximum age between 6s and 40s. This is the maximum length of time interval that an STP switch port saves its configuration BPDU information. The default max-age timer is 20s.
`internal|no-internal`	Specify if the STP state is internal or not.
`peer|no-peer`	Specify if the STP state is peer state or not.

```
For example: CLI (network-admin@Leaf1) > stp-state-show layout vertical

switch:                     Leaf1
vlan:                       1
ports:                      none
instance-id:           1
name:                            stg-default
bridge-id:                  66:0e:94:65:e1:ef
bridge-priority:    8193
root-id:                       64:0e:94:c0:06:4b
root-priority:         4097
root-port:                128
hello-time:              2
forwarding-delay:  15
max-age:                   20
disabled:                  none
learning:                  none
forwarding:                25-28,128-129
discarding:              none
edge:                         25-28
designated:               25-28,129
alternate:                none
backup:                      none
```

The STP information pertaining to the ports can be displayed by using the command `stp-port-show`.

```
CLI (network-admin@Leaf1) > stp-port-show
```

`stp-port-show`	Displays the STP port information.
Specify one or more of the following options to view the information specific to those options. Specifying no parameter will display all the information.	
`port` *`port-list`*	Specify the ports as a list separated by commas.
`block|no-block`	Specify if BPDU blocking is enabled on the ports or

	not.
`filter|no-filter`	Specify if BPDU filtering is enabled on the port or not.
`edge|no-edge`	Specify if the ports are edge ports or non-edge ports.
`bpdu-guard|no-bpdu-guard`	Specify if BPDU guard is configured on the ports or not.
`root-guard|no-root-guard`	Specify if root guard is configured on the ports or not.
`priority` *`0..240`*	Specify the priority as a value between 0 and 240.
`cost` *`1..200000000`*	Specify the port cost as a value between 1 and 200000000.

The STP state at the port level can be viewed using the command:

```
CLI (network-admin@Leaf1) > stp-port-state-show
```

`stp-port-state-show`	Display STP information at the port level.
Specify one or more of the following options to view the information specific to those options. Specifying none will display the information for all the parameters below.	
`vlan` *`vlan-list`*	Specify the VLANs as a list separated by commas.
`port` *`port-list`*	Specify the ports as a list separated by commas.
`stp-state` *`Disabled|Discarding|Learning|Forwarding`*	Specify one among the options as the STP state.
`role` *`Disabled|Root|Designated|Alternate|Backup`*	Specify one among the options as the port role.
`selected-role` *`Disabled|Root|Designated|Alternate|Backup`*	Specify one among the options as the selected role.
`state` *`new-info|proposing|proposed|agreed|agreed|learn|learning|forward|forwarding|reselect|selected|reroot|rcvd-bpdu|rcvd-msg|rcvd-tc|rcvd-tc-ack|send-rstp|tc-prop|tc-ack|update-info|sync|synced|disputed|fdb-flush|online|looping|manual-online|edge|vlag-local-up|vlag-remote-up|requested-online|first-sync|p-is-d|p-is-m|root-guard-active`*	Specify one among the options as the port state machine state.

`designated-priority` *`designated-priority-string`*	Specify the designated priority vector.
`port-priority` *`port-priority-string`*	Specify the port priority vector.
`message-priority` *`message-priority-string`*	Specify the message priority vector.
`info-is` *`disabled`*|*`received`*|*`mine`*|*`aged`*	Specify the origin of port information.
`designated-times` *`designated-times-string`*	Specify the designated times: age, max age, hello, and forward delay
`port-times` *`port-times-string`*	Specify the port times: age, max age, hello, and forward delay
`message-times` *`message-times-string`*	Specify the message times: age, max age, hello, and forward delay
`hello-timer` *`hello-timer-number`*	Specify the STP hello time between 1s and 10s. The hello time is the time between each Bridge Protocol Data Unit (BPDU) that is sent on a port. The default hello time is 2s.
`topology-timer` *`topology-timer-number`*	Specify the topology change timer value.
`forward-timer` *`forward-timer-number`*	Specify the STP forwarding delay between 4s and 30s. This is the time interval that is spent in the listening and learning states. The default forwarding delay time is 15s.
`rcvd-info-timer` *`rcvd-info-timer-number`*	Specify the received info timer value.
`recent-root-timer` *`recent-root-timer-number`*	Specify the recent root timer value.
`recent-backup-timer` *`recent-backup-timer-number`*	Specify the recent backup timer value.
`edge-delay-timer` *`edge-delay-timer-number`*	Specify the edge delay timer value.
`migration-timer` *`migration-timer-number`*	Specify the migration delay timer value.
`root-guard-timer` *`root-guard-timer-number`*	Specify the root guard BPDU delay timer value.
`sm-table-bits` *`sm-table-bits-number`*	Specify the state machine table state.
`sm-table` *`sm-table-string`*	Specify the state machine table description.
`vlag-peer-port` *`vlag-peer-port-number`*	Specify the VLAG peer port if active-active.

`peer|no-peer`	Specify the STP peer state.

RSTP can be configured using the command `stp-modify.`

```
CLI (network-admin@switch1) > stp-modify
```

`stp-modify`	Modify the Spanning Tree Protocol parameters.
Specify one or more of the following options:	
`enable|disable`	Specify to enable or disable STP
`stp-mode rstp|mstp`	**Specify the STP mode as RSTP or MSTP.**
`bpdus-bridge-ports|bpdus-all-ports`	Specify to send BPDUs only on switch ports or on all ports.
`bridge-id` *`mac-address`*	Specify the STP bridge ID. The first part of the bridge ID is a 2-byte bridge priority field (which can be configured) while the second part is the 6-byte MAC address of the switch.
`bridge-priority` *`0..61440`*	Specify the STP bridge priority in multiples of 4096. The default value is 32768.
`hello-time` *`1..10`*	Specify the STP hello time between 1s and 10s. The hello time is the time between each BPDU that is sent on a port. The default value is 2s.
`forwarding-delay` *`4..30`*	Specify the STP forwarding delay between 4s and 30s. The forwarding delay is the time that is spent in the listening and learning states. The default forwarding delay is 15s.
`max-age` *`6..40`*	Specify the max age time between 6s and 40s. The max age timer defines the maximum time for which a switchport stores config BPDU information. The default value is 20s. If a config BPDU does not arrive at a port for 20s (default), the switch detects a link failure and takes action to restore connectivity through the backup links.
`mst-max-hops` *`1..32`*	Specify the maximum hop count for MSTP BPDU. The default value is 20.
`mst-config-name` *`mst-config-name-string`*	Specify the name for MST configuration instance.
`mst-config-revision` *`mst-config-revision-number`*	Specify the MST configuration revision number. Enter a value between 0 and 65535.
`root-guard-wait-time` *`0..300`*	Specify the root guard wait time between 0s and 300s. The default value is 20. Specify the value as 0 to disable wait.

Note: Hello time, forwarding delay, and max age timers are not used by RSTP but are relevant to STP.

NetVisor OS optimizes RSTP by not sending BPDUs on any ports except on inter-switch link-ports by default. However, if you do not configure Link Layer Discovery Protocol (LLDP), NetVisor does not detect host ports (i.e., ports directly connected to end devices) or send BPDU packets. As a result, both ports are in

Forwarding state.

When you add the parameter `bpdus-all-ports` to the `stp-modify` command, it allows sending BPDUs on all ports even if hosts are not detected, unless the port is configured as an edge port. On a switch with a port connected to itself with this configuration, one of the ports goes into discarding state.

For example, to send BPDUs only on switch ports, use the command:
```
CLI (network-admin@switch1) > stp-modify bpdus-bridge-ports
```

To send BPDUs on all ports, use the command:
```
CLI (network-admin@switch1) > stp-modify bpdus-all-ports
```

STP ports can be configured using the command:

```
CLI (network-admin@Leaf1) > stp-port-modify
```

`stp-port-modify`	Displays the STP port information.
`port` *`port-list`*	Specify the ports as a list separated by commas.
Specify one or more of the following options:	
`block|no-block`	Specify if BPDU blocking is to be enabled on the ports or not.
`filter|no-filter`	Specify if BPDU filtering is to be enabled on the port or not.
`edge|no-edge`	Specify if the ports are to be configured as edge ports or non-edge ports.
`bpdu-guard|no-bpdu-guard`	Specify if BPDU guard is to be configured on the ports or not.
`root-guard|no-root-guard`	Specify if root guard is to be configured on the ports or not.
`priority` *`0..240`*	Specify the priority as a value between 0 and 240.
`cost` *`1..200000000`*	Specify the port cost as a value between 1 and 200000000.

For example: To filter BPDUs on port 17, use the following command:

```
CLI (network-admin@Leaf1) > stp-port-modify port 17 filter
```

To block BPDUs on port 17 and shut down the port if BPDUs are received on the port, use the following command:

```
CLI (network-admin@Leaf1) > stp-port-modify port 17 block
```

To stop blocking BPDUs on port 17, use the following command:

```
CLI (network-admin@Leaf1) > stp-port-modify port 17 no-block
```

Edge ports are the ports on a switch that connect to workstations or computers. An edge port does not take part in spanning tree calculations and therefore, port flapping on edge ports does cause topology changes. BPDUs are not sent on edge ports and they can quickly transition from disabled mode to forwarding mode.

To configure a port as an edge port, use the command:

```
CLI (network-admin@Leaf1) > stp-port-modify port 17 edge
```

> **Note:** You can disable STP on a port or a group of ports. If the devices connected to the switch ports are hosts and not downstream switches, or you know that a loop is not possible, disable STP to enable the port much faster when the switch restarts.

To view STP events on a switch, the command `stp-port-event-show` command is used. This command displays the port states as specified by the timing parameters.

```
CLI (network-admin@Leaf1) > stp-port-event-show
```

`stp-port-event-show`	Displays information about STP port events.
`port` *`port-list`*	Specify the ports as a list separated by commas.
Specify one or more of the following options:	
`time` *`date/time: yyyy-mm-ddTHH:mm:ss`*	Specify the time to start statistics collections.
`start-time` *`date/time: yyyy-mm-ddTHH:mm:ss`*	Specify the start time of statistics collection.
`end-time` *`date/time: yyyy-mm-ddTHH:mm:ss`*	Specify the end time of statistics collection.
`duration` *`duration: #d#h#m#s`*	Specify the duration of statistics collection.
`interval` *`duration: #d#h#m#s`*	Specify the interval between statistics collection.
`older-than` *`duration: #d#h#m#s`*	Specify the time older than which the statistics has to be displayed.
`within-last` *`duration: #d#h#m#s`*	Display statistics within last specified duration.
`port` *`port-number`*	Specify the port number.
`vlan` *`vlan-list`*	Specify the list of VLANs.
`instance` *`instance-number`*	Specify the STP instance number.
`count` *`count-number`*	Specify the number of STP port events.
`initial-state Disabled|Discarding|Learning|Forwarding`	Specify the initial state as one among the options.
`other-state Disabled|Discarding|Learning|Forwarding`	Specify the other state as one among the options.
`final-state Disabled|Discarding|Learning|Forwarding`	Specify the final state as one among the options.

For example:

```
CLI  (network-admin@Leaf1)  >  stp-port-event-show

switch        time     port vlan instance count initial-state other-state final-state
------------- -------- ---- ---- -------- ----- ------------- ----------- -----------
Leaf1         20:36:39 121  1    0        1     Forwarding    Disabled    Disabled
Leaf1         20:38:05 17   1    0        4     Disabled      Disabled    Forwarding
Leaf1         20:40:04 17   1    0        1     Forwarding    Disabled    Disabled
```

Configuring Multiple Spanning Tree Protocol (MSTP)

Multiple Spanning Tree Protocol as defined in IEEE802.1s or IEEE802.1Q-2005 provides the ability to manage multiple VLANs using Multiple Spanning Tree Instances (MSTIs). MSTP allows the formation of MST regions that can run multiple MSTIs. MSTP regions and other STP bridges are interconnected using the Common and Internal Spanning Tree (CIST).

MSTP regions are defined to be a collection of switches that have the same VLANs configured on all of them. All switches in a region must have the same configuration name, revision level, VLANs, and VLAN to MSTI associations. Each MST region may have multiple MSTIs operating within it but an MSTI cannot span multiple regions. Each MSTI must have a regional root which it may or may not share with another MSTI.

The CIST is the default spanning tree instance in MSTP. CIST forms a larger spanning tree for the entire bridged network by combining MSTP regions and single-instance spanning trees. The CIST instance has an MSTI ID of 0 which cannot be deleted or changed. When a new port-based or tagged VLAN is created, it is associated with the CIST by default and is automatically given an MSTI ID of 0. The default VLAN on a switch is also associated with the CIST. When a VLAN is assigned to an MSTI, it partially remains a member of the CIST. This is because CIST is used by MSTP to enable communication with MSTP regions and RSTP/STP single-instances in the network. CIST also has regional roots.

The switch with the lowest CIST priority value functions as the root bridge for all the MSTP regions and STP/ RSTP single-instance spanning trees in the network.

The following commands support the configuration of MST instances on a local switch:

```
CLI  (network-admin@Leaf1)  >  mst-config-create
```

`instance-id` *`0..64`*	Specify the ID as a number between 1 and 64 for MST configuration. MST ID 0 corresponds to the CIST instance.
`vlans` *`vlan-list`*	Specify the list of VLANs associated with the MST configuration
`bridge-priority` *`0..61440`*	Specify the bridge priority as a number between 0 and 61440 which is a multiple of 4096. For example, the values can be 0, 4096, 8192, up to 61440. The default value is 32768.

Prior to NetVisor OS version 6.0.1, VLAN 1 could not be configured on any MST instance ID except 0. NetVisor OS 6.0.1 release eliminates this constraint. For example:

```
CLI  (network-admin@Leaf1)  >  mst-config-create  instance-id  10  vlans  1-20
```

Use the `mst-config-show` command to view the current configuration:

```
CLI (network-admin@Leaf1) > mst-config-show
switch  instance-id  vlans          bridge-priority
Leaf1   0            21-4093,4095  32768
Leaf1   10           1-20          32768
Leaf1   64           4094          32768
```

In the above example, VLAN 1 is configured on instance ID 10.

You can modify the MST instance by using the command:

```
CLI (network-admin@Leaf1) > mst-config-modify
```

`instance-id` *`0..64`*	Specify the ID as a number between 1 and 64 for MST configuration. MST ID 0 corresponds to the CIST instance.
Specify one or more of the following options:	
`vlans` *`vlan-list`*	Specify the list of VLANs associated with the MST configuration.
`bridge-priority` *`0..61440`*	Specify the bridge priority as a number between 0 and 61440 which is a multiple of 4096. For example, the values can be 0, 4096, 8192, up to 61440. The default value is 32768.

Use the `mst-config-delete` command to delete the configuration.

```
CLI (network-admin@Leaf1) > mst-config-delete
```

`instance-id` *`0..64`*	Specify the MST instance ID that needs to be deleted.

Achieving a Loop-Free Layer 2 Topology

> Note: This feature can be configured only in a full mesh topology.

Rapid Spanning Tree Protocol (RSTP) and Multiple Spanning Tree Protocol (MSTP) ensure a loop-free topology in the Layer 2 as far as the networking equipment is concerned. Though RSTP prevents loops in the network caused by mis-cabled networking equipment, the protocol does not address mis-configured hosts. NetVisor OS Loop Detection operates in conjunction with RSTP and MSTP to detect, log, and mitigate misbehaving and misconfigured hosts to prevent looping layer 2 traffic.

NetVisor OS Control Plane — The NetVisor OS control plane includes information about every MAC address in the Layer 2 network in a vPort database. This database is distributed throughout the fabric so that each NetVisor OS switch has a copy of it for the entire fabric.

A MAC address is stored in a vPort, which includes the following information:

- MAC address, VLAN ID, and VXLAN ID
- Owner-port and local-port
- Migration history including owner, time, and port
- vPort state as active, static, moving, or loop-probe

Based on control plane data structures including the vPort database, NetVisor OS decides if endpoints are to be allowed to access the network.

Detecting Loops

NetVisor OS Loop Detection is implemented as part of NetVisor OS source MAC address miss handling. NetVisor OS disables hardware learning of MAC addresses, so when a packet arrives with an unknown source MAC address, the switch sends the packet to NetVisor OS rather than switching the packet normally. NetVisor OS examines the vPort table to determine if a packet with an unknown source MAC indicates a loop.

NetVisor OS uses two criteria to detect a loop in the network:

- A MAC address associated with an in-band NIC of a node in the fabric appears as the source MAC on a packet that ingresses on a host port. NetVisor OS detects this situation by noting the `PN-internal` status of a vPort that would otherwise migrate to a host port. NetVisor does not allow the migration to take place and starts loop mitigation.

For the purposes of NetVisor OS Loop Detection, a host port is defined as a port not connected to another Arista switch, not an internal port, and does not participate in STP with NetVisor OS which means that NetVisor OS is not configured for STP or the device connected on the port is not configured for STP.

- Packets with the same source MAC address arrive on multiple host ports in the fabric at approximately the same time. In order to support VM and host migration, some rapid movement of MAC addresses through the fabric is tolerated. When the same MAC address moves rapidly back and forth between two ports, a loop is assumed and loop mitigation starts.

VRRP MAC addresses are not subject to loop detection and mitigation, and can migrate freely.

Loops are detected on a port by port basis. A single loop typically involves two ports, either on the same switch or on two different switches. When multiple loops occur with more than two ports then NetVisor OS responds to each port separately.

Loop Mitigation

When NetVisor OS detects a loop, a message appears in the system log indicating the host port and VLAN involved in the loop. In addition the host port involved in the loop has the "*loop*" status added and NetVisor OS adds the VLAN to the host port loop-vlans VLAN map. Looping ports and VLANs are displayed in the `port-show` output.

At the start of loop mitigation, NetVisor OS creates vPorts to send loop probe packets. The vPorts use the port MAC address for the in-band NIC port, status of PN-internal, and a state of loop-probe. NetVisor OS propagates Loop-probe vPorts throughout the fabric. NetVisor OS creates a loop-probe vPort for each looping VLAN.

NetVisor OS deletes all vPorts from the looping host port and VLAN at the start of loop mitigation. This prevents the hardware from sending unicast packets to the looping port, and causes every packet arriving on the looping port to appear in the software as a source MAC miss. During loop mitigation, NetVisor OS drops all packets arriving on the looping port.

During loop mitigation, NetVisor OS sends loop probe packets on the looping VLANs every 3 seconds. As long as the loop persists, NetVisor OS receives the probe packets as source MAC miss notification on the looping ports, so NetVisor OS can determine if the loop is still present. If 9 seconds elapse with no received probe packets, NetVisor OS detects the loop is resolved and ends loop mitigation.

At the end of loop mitigation, log messages are added to the system log, loop-probe vPorts are removed, and loop stats and loop VLANS are removed from the looping port.

To view affected ports, use the `port-show` command and add the parameter, `status loop`:

```
CLI (network-admin@switch-31) > port-show status loop

switch       port hostname status                    config
---------- ---- -------- --------------------   ------
switch-31   9               up,stp-edge-port,loop   fd,10g
switch-32   9               up,stp-edge-port,loop   fd,10g
```

> **Note:** the new status, `loop`, in the `status` column. When the loops are removed from the port, the *loop* flag is removed from the `port-show status` command output and log message is added regarding the removal of *loop*.

During loop mitigation, the MAC addresses for loop probes are displayed in the vPort table:

```
CLI (network-admin@switch-31) > vport-show state loop-probe

owner          mac               vlan   ports state       hostname    status
---------- ----------------- ----   ----- ---------- ---------- -----------
switch-32 06:c0:00:16:f0:45    42    69     loop-probe leo-ext-32 PN-internal
```

```
switch-31 06:c0:00:19:c0:45    42    69     loop-probe leo-ext-31 PN-internal
```

Note the `loop-probe` state as well as the PN-internal state. The loop probes use the port MAC address format, and use the internal port for the in-band NIC.

> **Note**: The `state` and the `status` columns are different in the above `vport-show stats loop-probe` command output. The `status` column refers to the vPort peer owner state in the fabric (the *PN-internal* parameter indicates that the MAC belongs to the PN fabric). The `state` column displays the vPort state.

If you notice a disruption in the network, use the `port-show` command to find the looping ports, and fix the loop. Fixing the loop typically involves correcting cabling issues, configuring virtual switches, or as a stop-gap measure, using the `port-config-modify` command to change port properties for the looping host ports. Once the loop is resolved, NetVisor OS no longer detects probes and leaves the loop mitigation state, while logging a message:

```
2016-01-12,12:18:41.911799-07:00 leo-ext-31 nvOSd(25695) system
host_port_loop_resolved(11381) : level=note : port=9 :
Traffic has stopped looping on host-port=9
```

At this point the `loop` status is removed from the `port-show` output for port 9 and the loop-probe vPorts are removed.

NetVisor OS Loop Detection exposes loops using system log messages, `port-show` output, and `vport-show` output.

When NetVisor OS detects an internal port MAC address on a host port, NetVisor OS prints a log message as below:

```
system  2016-01-19,15:36:40.570184-07:00  mac_move_denied
         11379 note   MOVE DENIED mac=64:0e:94:c0:03:b3 vlan=1 vxlan=0
          from switch=leo-ext-31 port=69 to deny-switch=leo-ext-31 deny-port=9
        reason=internal MAC of local switch not allowed to change ports
```

NetVisor OS starts Loop Mitigation by logging a message:

```
system  2016-01-19,15:36:40.570334-07:00  host_port_loop_detected
         11380 warn   Looping traffic detected on host-port=9
        vlan=1. Traffic on this port/VLAN will be ignored until loop
resolved
```

During Loop Mitigation, NetVisor OS sends loop probes. When these probes, as well as any other packets, are received on a looping host port, NetVisor OS logs a message:

```
system  2016-01-19,15:59:54.734277-07:00  mac_move_denied
         11379 note   MOVE DENIED mac=06:c0:00:19:c0:45 vlan=1 vxlan=0
```

```
 from switch=leo-ext-31 port=69 to deny-switch=leo-ext-31
deny-port=9 reason=port is looping
```

NetVisor OS limits `mac_move_denied` messages are limited to one every 5 seconds for each vPort. This prevents the system log from filling up with `mac_move_denied` messages during loop mitigation.

During loop mitigation, you can use the `port-show` command to see which ports are involved in the loop:

```
CLI (network-admin@Leaf1) > port-show status loop

switch     port  hostname  status                          loop-vlans  config
------   ----  --------  ---------------------   ----------  ------
leaf1    9                up,stp-edge-port,loop    1             fd,10g
leaf1    9                up,stp-edge-port,loop    1             fd,10g
```

Note the `loop` status in the status column and the `loop-vlans` column.

During loop mitigation the MAC addresses for loop probes are displayed in the vPort table:

```
CLI (network-admin@Leaf1) > vport-show state loop-probe

owner        mac               vlan    ports  state       hostname      status
------  -----------------   ----    ----- ----------  --------     ---------
leaf1    06:c0:00:16:f0:45   42     69      loop-probe  leo-ext-32  PN-internal
leaf1    06:c0:00:19:c0:45   42     69      loop-probe  leo-ext-31  PN-internal
```

Fast Failover for STP and Cluster

Previously, cluster STP operation did not support fast failover because NetVisor OS did not share STP state between the two nodes. As a result, when the master failed and when it came back online, the slave had to recompute the STP state from scratch. This resulted in topology changes twice, causing traffic loss until STP converged.

Currently, NetVisor OS supports fast failover by default. In the cluster-STP mode, the node that has been up longer is elected as the master. Cluster syncs (keep-alives) are used to detect the peer node being online, and negotiate the initial cluster state. Cluster syncs determine which node has been up longer based on exchanged uptime values.

The master runs the state machine for both nodes and sends the STP and port states to the slave. The slave in turn maintains the state as informed by the master. The slave generates its own BPDUs based on the synchronized state and forwards the BPDUs that it receives to the master. When the cluster goes offline, the slave/master uses the same bridge ID and priority, uses consistent port IDs in BPDUs, and continues from the existing synchronized state. The STP state machine state is thus never lost.

Internal state synchronization using consistent bridge ID/priority and port IDs regardless of whether the cluster is online or offline, and active-active vLAG handling ensure that an end node detects no topology change when the cluster nodes go offline/online.

When a cluster is created, the STP configuration between the two cluster nodes is checked and is synchronized. The following guidelines are true regardless of whether the cluster is online or offline, and whether the peer node is online or offline:

- Both nodes use the same bridge ID or priority.
- When node1 sends a BPDU, the port ID inside the packet is 1-256, except for active-active vLAGs.
- When node2 sends a BPDU, the port ID inside the packet is 257-512, except for active-active vLAGs.
- When either node sends a BPDU on an active-active vLAG, the port ID inside the packet is node1's port number.
- Configuration changes (STP mode, MST instances, bridge ID, etc.) are mirrored on both nodes through cluster transactions.

Due to the above guidelines, a BPDU sent on an active-active vLAG appears exactly the same to a third party receiver regardless of whether that packet came from cluster node1 or node2.

NetVisor OS provides two show commands to view the details of this functionality: stp-state-show and stp-port-state-show.

For example:

```
CLI (network-admin@Leaf1) > stp-state-show
switch:                  Leaf-1
vlan:                    1
ports:                   none
instance-id:          1
name:                       stg-default
bridge-id:                66:0e:94:d5:b0:cc
bridge-priority:    32769
root-id:                    66:0e:94:35:c2:ce
root-priority:       32769
```

```
root-port:             128
hello-time:            2
forwarding-delay:  15
max-age:                20
disabled:              none
learning:              none
forwarding:            none
discarding:            none
edge:                     none
designated:            none
alternate:             none
backup:                  none

CLI (network-admin@Switch2) > stp-port-state-show port 17
switch:                Switch2
vlan:                  1
port:                  17
stp-state:             Forwarding
role:                  Designated
selected-role:         Designated
state:               agreed,learn,learning,forward,forwarding,selected,send-rstp,synced,online,requested-
online
designated-priority: 32769-66:0e:94:38:39:80,100,32769-66:0e:94:b7:65:91,32785
port-priority:       32769-66:0e:94:38:39:80,100,32769-66:0e:94:b7:65:91,32785
message-priority:    0-00:00:00:00:00:00,0,0-00:00:00:00:00:00,0
info-is:               mine
hello-timer:           2
root-guard-timer:      0
sm-table-bits:         0xfaedee
sm-table:               prx=discard*,bdm=not-edge*,ptx=idle*,pim=current*,prt-
disabled=disable*,prt-root=root*,prt-desg=designated*,prt-alt-
bk=block*,pst=forwarding*,tcm=active*
```

Configuring Auto-Recovery of a Disabled Port

NetVisor OS automatically disables physical ports due to certain violations. For example, if a port receives BPDU messages on an edge port, NetVisor OS disables the port because receiving BPDUs on a edge port becomes a security violation. This happens because the edge port is configured for BPDU guard for the violation to take effect.

The port may be disabled due to the following errors:

- BPDU Guard Messages: The port is set to `err-disabled` when a BPDU is received on an edge port with BPDU guard enabled.
- Link Flaps: There are too many link flaps for a configured interval of time.
- MAC address Security Violation: The number of MAC addresses on an interface is greater than the configured limit.

To clear the counters for err-disable caused by BPDU guard and MAC security, use the following command:

```
CLI (network-admin@Leaf1) > err-disable-clear-counters
```

`err-disable-clear-counters`	Resets err-disable counters for all the ports on the switch.
Specify any of the following options:	
`bpduguard|no-bpduguard`	Specify if BPDU guard counters are to be cleared.
`macsecurity|no-macsecurity`	Specify if MAC security counters are to be cleared.
`recovery-timer` *`duration: #d#h#m#s`*	Specify the recovery timer value. The default timer value is 5 minutes. Example: 20s or 1d or 10d20m3h15s

To configure BPDU guard or MAC security on a switch, use the command:

```
CLI (network-admin@Leaf1) > err-disable-modify
```

`err-disable-modify`	Modifies the `err-disable` settings for the switch.
Specify any of the following options:	
`bpduguard|no-bpduguard`	Specify either of the options to enable or disable BPDU guard.
`macsecurity|no-macsecurity`	Specify either of the options to enable or disable MAC security.
`recovery-timer` *`duration: #d#h#m#s`*	Specify the recovery time value. The default timer value is 5 minutes. Example: 20s or 1d or 10d20m3h15s

To view the error recovery settings on a switch, use the `err-disable-show` command. For example:

```
CLI (network-admin@Leaf1) > err-disable-show
switch: Leaf1
bpduguard: off
macsecurity: off
recovery-timer: 5m
```

Configuring STP Root Guard

The Root Guard feature is used to enforce the positioning/placement of root bridge in a network. In STP, there is no provision to have full control on the selection of the root bridge or switch. Any switch can be selected as the root bridge. If the bridge priority is set to 0, that switch is likely to become the root bridge. However, even with this configuration, there is no guarantee since there can be another switch with priority 0 and a lower MAC address that gets selected as root bridge.

The Root Guard feature forces a port to be a designated port (and does not allow it to become root port). This prevents any one of the neighboring switches from becoming the root switch. Thus, the Root Guard feature provides a way to enforce the placement or positioning of the root bridge in the network.

If a port on which the Root Guard feature is enabled receives a superior BPDU, it moves the port into a root-inconsistent state (similar to a listening state). In this state, no traffic is forwarded across this port. Root Guard must be enabled on all ports where the root bridge should not appear.

To configure root guard, use the command:

```
CLI (network-admin@Leaf1) > stp-port-modify port port-list root-guard
```

`stp-port-modify`	Modify the Spanning Tree Protocol Parameters.
`port` *`port-list`*	Specify the port or port list.
`Specify one or more of the following options:`	
`block|no-block`	Specify if a STP port blocks BPDUs.
`bpdu-guard|no-bpdu-guard`	Enable or disable STP port BPDU guard.
`root-guard|no-root-guard`	Enable or disable STP port Root guard.

To view the root guard configuration details, use the command:

```
CLI (network-admin@Leaf1) > stp-port-show
```

Configuring Fabric Guard

Currently, NetVisor OS detects a Layer 2 loop using STP, LLDP, or loop detect code. However if a third party device connected to a Arista Networks switch consumes LLDP such as a hypervisor vSwitch, and you configure the port as an edge port, NetVisor OS cannot detect loops in the network.

If you configure a port as fabric-guard port, NetVisor OS triggers sending global discovery multicast packets on this port after the port is physically up and in an adjacency wait state. If a port with fabric-guard configuration receives a global discovery packet, NetVisor OS disables the port in the same way LLDP disables the port when receiving messages from the same switch.

To enable fabric guard, use the following syntax:

```
CLI (network-admin@Leaf1) > port-config-modify port port-number fabric-guard
```

To disable fabric guard, use the following syntax:

```
CLI (network-admin@Leaf1) > port-config-modify port port-number no-fabric-guard
```

In order to re-enable the port once you fix the loop, you must manually enable the port using the command, `port-config-modify port port-number enable`.

Configuring Layer 2 Static Multicast Groups

The traffic addressed to a multicast group is confined to the ports in the group, which prevents flooding of traffic on all the ports. Hosts join multicast groups either by sending an unsolicited IGMP join message or by sending an IGMP join message in response to a general query from a multicast router (the switch forwards general queries from multicast routers to all ports in a VLAN). When you specify group membership for a multicast group address statically, the static setting supersedes any IGMP snooping learning. Hence, static multicast groups can prevent L2 multicast flooding, if the hosts do not implement the desired IGMP mechanism to restrict the traffic to certain ports, or if IGMP snooping is disabled.

To create an L2 static multicast group, use the command:

```
CLI (network-admin@leaf1) > l2-static-multicast-group-create
```

`l2-static-multicast-group-create`	Create a Layer 2 static multicast group.
`group-mac` *`mac-address`*	Specify a MAC address for the multicast group.
`vlan` *`vlan-id`*	Specify a VLAN ID for the multicast group.
`ports` *`port-list`*	Specify a list of ports for the multicast group.

For example, to create an L2 static multicast group for the MAC address 01:00:5e:7f:00:12, VLAN 25, and ports 30-35, use the following syntax:

```
CLI (network-admin@leaf1) > l2-static-multicast-group-create group-mac
01:00:5e:7f:00:12 vlan 25 ports 30-35
```

To add ports to a preconfigured multicast group, use the command:

```
CLI (network-admin@leaf1) > l2-static-multicast-group-port-add
```

`l2-static-multicast-group-port-add`	Add ports to a L2 static multicast group.
`group-mac` *`mac-address`*	Specify a MAC address for the multicast group.
`vnet vnet-name`	Specify a vNET for the multicast group.
`bd` *`bridge-domain-name`*	Specify a bridge domain for the multicast group.
`vlan` *`vlan-id`*	Specify a VLAN ID for the multicast group.
`switch` *`switch-name`*	Specify the name of the switch.

`ports port-list`	Specify a list of ports.

For example, to add port 36 to the multicast group created above, use the command:

```
CLI (network-admin@leaf1) > l2-static-multicast-group-port-add group-mac
01:00:5e:7f:00:12 vlan 25 ports 36
```

To view the configuration, use the command:

```
CLI (network-admin@leaf1) > l2-static-multicast-group-show
switch group-mac          vnet bd vlan ports
------ ----------------- ---- -- ---- --------
leaf1  01:00:5e:7f:00:12          25   30-36
```

To remove port 36 from the multicast group, use the command:

```
CLI (network-admin@leaf1) > l2-static-multicast-group-port-remove group-mac
01:00:5e:7f:00:12 vlan 25 ports 36
```

To view the updated configuration, use the command:

```
CLI (network-admin@leaf1) > l2-static-multicast-group-show
switch group-mac          vnet bd vlan ports
------ ----------------- ---- -- ---- --------
leaf1  01:00:5e:7f:00:12          25   30-35
```

To delete the L2 static multicast group, use the command:

```
CLI (network-admin@leaf1) > l2-static-multicast-group-delete group-mac
01:00:5e:7f:00:12 vlan 25
```

Configuring Hardware Batch Move in Layer 2 Table

Note: This feature is useful in deployments with scaled-up regular VLANs, where the configuration has a lot of VLANs or MACs on a given vLAG. This feature is disabled by default on Netvsior ONE.

The Netvsior ONE version 6.0.1 release provides support for hardware batch move mechanism to accelerate the MAC programming in Layer 2 table in a switch.

You can enable the hardware batch move feature by using the `batch-move-mac-hw-group-for-vlan-only` option in the `system-settings-modify` command. If enabled, this feature uses an internal switch specific construct called *portgroups* to identify and move the MACs between a vLAG and Cluster communication ports when the vLAG comes up or goes down in a VLAN network.

Caution: You must enable this feature only for scaled-up VLAN or MAC environments.

When disabled (this feature is disabled by default), the MAC move mechanism uses the conventional software batch move process, which has an acceptable level of traffic loss during the MAC move process in a VLAN or VLAN-VXLAN network. The software batch move feature supports hybrid mechanism (that is, software batch move in VLAN network, VXLAN network, and VLAN-VXLAN network) that can accelerate the MAC programming in Layer 2 table of an underlying switch.

To elaborate, when the hardware batch move feature is enabled or disabled:

- For VLAN only networks - uses the Hardware batch move feature if enabled.
- For regular VXLAN only configurations - uses the Software batch feature.
- For a combination of regular VLAN and VXLAN- VLAN network - uses hybrid method. That is, regular VLAN network uses hardware batch move if the feature is enabled. The VXLAN-VLAN network uses software batch move.
- When the hardware batch move feature is disabled, both regular VLAN and VXLAN-VLAN network uses the software batch move as the default option.

To configure the hardware batch move, use the `system-settings-modify` command with the `batch-move-mac-hw-group-for-vlan-only` command option.

Note: When you enable the feature , you must reboot the switch for the changes to take effect.

To enable the hardware batch move feature:

```
CLI (network-admin@switch1) > system-settings-modify batch-move-mac-hw-group-for-vlan-only|no-batch-move-mac-hw-group-for-vlan-only
```

`system-settings-modify`	Use this command to modify the system settings on a switch.
`batch-move-mac-hw-group-for-vlan-only|no-batch-move-mac-hw-group-for-vlan-only`	Specify to enable or disable the HW batch move for MAC by group for VLAN network only.

To verify if the hardware batch move is enabled or not on a switch, use the command:

```
CLI (network-admin@switch1) > system-settings-show
```

An example to enable the hardware batch move feature and view the details is provided below:

```
CLI (network-admin@switch1*) > system-settings-show format batch-move-mac-hw-group-for-vlan-only,
batch-move-mac-hw-group-for-vlan-only:   off

CLI (network-admin@switch1*) > system-settings-modify batch-move-mac-hw-group-for-vlan-only
Modified settings require a REBOOT to take effect.

CLI (network-admin@cham-colo-1*) > switch-reboot

CLI (network-admin@cham-colo-1*) > system-settings-show format batch-move-mac-hw-group-for-vlan-only,

batch-move-mac-hw-group-for-vlan-only:   on
```

Note: This feature is available on all platforms except AS7816-64X (F9664-C) and Z9264 platforms.

Configuring Excessive MAC or IP Move Protection

Excessive MAC or IP moves necessitate numerous updates to vPort and Layer 3 tables over a short interval, which result in high CPU and disk utilization among other network problems. A MAC move is detected when two devices send the same MAC address on different interfaces on the same switch, or on different switches in a fabric with the same VLAN. An IP move is observed when an IP address oscillates between two MAC addresses.

NetVisor OS version 6.0.1 offers protection against excessive MAC or IP moves by quarantining vPort and L3 entries, that is, by not updating the entries until MAC or IP move condition is resolved. When you enable the protection feature, if more than five IP moves or MAC moves are detected within an interval of 5s, NetVisor OS performs the following:

- Updates the `excess-mac-move-detected` or `excess-ip-move-detected` flags
- Logs `excess_mac_move` or `excess_ip_move` message.

While sending the vPort or Layer 3 updates to other fabric nodes, the software skips the entries that have an excess move flag set, and thereby avoids sending a large number of updates.

The software then monitors the quarantined entries and if no MAC or IP moves are detected for a duration of 15s, NetVisor OS performs the following:

- Clears the `excess-mac-move-detected` or `excess-ip-move-detected` flags.
- Logs `clear_excess_mac_move` or `clear_excess_ip_move` message.

NetVisor OS can also protect the CPU from excessive traffic related to MAC or IP moves by regulating the punt rate of associated CoS (Class of Service) queues. MAC moves and IP moves are punted to the CPU from the *smac-miss* queue and the *arp* queue respectively. When excessive MAC or IP moves are detected, and if CPU utilization is above 70 percent, the software can limit the punt rate from *smac-miss* or *arp* queues by 50 percent.

To configure CoS queue protection, you must first enable extended queue setting by using the command:

```
CLI (network-admin@switch1) > system-settings-modify cpu-class-enable
```

Use the `vport-settings-modify` command to configure MAC and IP move protection. These protection schemes are disabled by default.

```
CLI (network-admin@switch1) > vport-settings-modify
```

`vport-settings-modify`	Modify vPort settings.
Specify one or more of the following options:	
`vport-disk-space` *`vport-disk-space-number`*	Specify the amount of disk space for vPorts. The default is 500M.
`stats-max-memory` *`stats-max-memory-number`*	Specify the maximum memory for collecting vPort information. The default memory is 50M.
`stats-log-enable|` `stats-log-disable`	Specify if you want to enable or disable logs for vPort statistics. Enabled by default.
`stats-log-interval duration:` *`#d#h#m#s`*	Specify the interval between logging events. The

	default is one minute.
`stats-log-disk-space` *`disk-space-number`*	Specify the amount of disk space for vPort logs. The default is 50M.
`system-stats-log-enable|system-stats-log-disable`	Specify if you want to enable or disable logging for the system. Enabled by default.
`system-stats-log-interval duration:` *`#d#h#m#s`*	Specify the interval between logging events. The default is one minute.
`system-stats-log-disk-space` *`disk-space-number`*	Specify the disk space for system statistics. The default is 50M.
`excess-mac-move-protection-enable| no-excess-mac-move-protection-enable`	Enable or disable excess MAC move protection.
`excess-mac-move-queue-protect|no-excess-mac-move-queue-protect`	Enable or disable excess MAC move queue protection.
`excess-ip-move-protection-enable| no-excess-ip-move-protection-enable`	Enable or disable excess IP move protection.
`excess-ip-move-queue-protect|no-excess-ip-move-queue-protect`	Enable or disable excess IP move queue protection.

For example, to configure excess MAC move protection, use the command:

```
CLI (network-admin@switch1) > vport-settings-modify excess-mac-move-
    protection-enable
```

To enable excess MAC move CoS queue protection in order to limit the punt rate from the *smac-miss* queue to the CPU by 50 percent, use the command:

```
CLI (network-admin@switch1) > vport-settings-modify excess-mac-move-queue-
    protect
```

To configure excess IP move protection, use the command:

```
CLI (network-admin@switch1) > vport-settings-modify excess-ip-move-
    protection-enable
```

To enable excess IP move CoS queue protection in order to limit the punt rate from the *arp* queue to the CPU by 50 percent, use the command:

```
CLI (network-admin@switch1) > vport-settings-modify excess-ip-move-queue-
    protect
```

Use the `vport-settings-show` command to view the current status of various protection schemes:

```
CLI (network-admin@switch1) > vport-settings-show format all
switch:                         switch1
```

```
vport-disk-space:                500M
stats-max-memory:                50M
stats-log-enable:                yes
stats-log-interval:              1m
stats-log-disk-space:            50M
system-stats-max-memory:         50M
system-stats-log-enable:         yes
system-stats-log-interval:       1m
system-stats-log-disk-space:     50M
loop-prevent:                    enabled
excess-mac-move-protect-enable:  yes
excess-mac-move-queue-protect:   yes
excess-mac-move-queue-state:     active
excess-ip-move-protect-enable:   yes
excess-ip-move-queue-protect:    yes
excess-ip-move-queue-state:      active
```

If queue protection is enabled, the fields `excess-mac-move-queue-state` and `excess-ip-move-queue-state` are set to `active` when MAC or IP moves are detected and CPU utilization is above 70 percent.

If excess MAC move is detected, the `vport-show` and `l2-table-show` outputs display the state of the corresponding entries with an `excess-mac-move-detected` flag. For example:

```
CLI (network-admin@switch) > vport-show vlan 100

owner   mac         vlan ip       num-ips ports state                          hostname migrate
------- ----------- ---- -------- ------- ----- ------------------------------ -------- -------
switch 00:x:x:x:x:x 100  100.0.0.1  2      126   active,excess-mac-move-detected  host    52840

CLI  (network-admin@serpens-vle-1*)  >  l2-table-show  vlan  100
mac               vlan ports state                            migrate
----------------  ----  -----  -------------------------------  -------
00:11:22:33:44:55 100   33     active,excess-mac-move-detected 56
```

If an excess IP move situation is detected, the `l3-table-show` output displays the state of the corresponding entry with an `excess-ip-move-detected` flag. For example:

```
CLI (network-admin@switch) > l3-table-show vlan 200

switch   mac         ip       vlan state
------ ----------- --------- ---- ------------------------------
switch 00:x:x:x:x:x 200.0.0.1 200  active,excess-ip-move-detected
```

You can view the log messages for excess MAC and IP move detection and resolution by using the command, `log-system-show.` For example:

```
CLI (network-admin@switch1) > log-system-show name
excess_mac_move,excess_ip_move,clear_excess_mac_move,clear_excess_ip_move

category:          system
time:               2020-08-12,00:55:16.738453-07:00
name:              clear_excess_mac_move
code:              11525
level:             note
message:            Excess MAC move condition cleared for mac=00:01:02:03:04:05, vnet= vlan=100 vxlan=0
category:          system
time:               2020-08-12,00:55:26.463901-07:00
```

```
name:               clear_excess_ip_move
code:              11526
level:             note
message:             Excess IP move condition cleared for ip=200.0.0.1 vnet= vlan=200 vxlan=0
category:          system
time:                2020-08-12,00:57:33.577668-07:00
name:              excess_mac_move
code:              11523
level:             note
message:             Excess MAC moves detected for mac=00:01:02:03:04:05, vnet= vlan=100
category:          system
time:                2020-08-12,00:57:33.717490-07:00
name:              excess_ip_move
code:              11524
level:             note
message:             Excess IP moves detected for ip=200.0.0.1 vnet= vlan=200 vxlan=0
```

About Layer 2 Hardware Hashing

A hardware hashing operation comprises two parts that can be performed at extremely high speed: first, packet field selection and extraction, then a mathematical bitwise computation.

The first step therefore is for the hardware parsing logic to select a number of packet fields 'to be hashed' depending on the traffic type.

The ***Table 4-1*** describes the traffic types and related packet fields that NetVisor OS currently supports by default to perform hashing on:

Table Caption 4-1: Hardware Hashing Field Selection

Application	Traffic Type[2]	Hashed Fields[3]
Non-IP packets[1]	Ethernet (a.k.a. 'Layer 2')	Source MAC address, Destination MAC address, Ethertype, VLAN number, source physical port
Plain unicast and multicast IPv4 packets	IP	Source IPv4 address, Destination IPv4 address, VLAN number, Destination Layer 4 port, Source Layer 4 port, IP protocol number, source physical port
Plain unicast and multicast IPv6 packets	IP	Source IPv6 address, Destination IPv6 address, VLAN number, Destination Layer 4 port, Source Layer 4 port, IP protocol number, source physical port
VXLAN packets (UDP encapsulated)	IP	Source IP address, Destination IP address, VLAN number, Destination Layer 4 port, *Source Layer 4 port*, IP protocol number, source physical port
L2GRE	IP	Source IP address, Destination IP address, VLAN number, GRE Key, IP protocol number, source physical port
IPSEC	IP	Source IP address, Destination IP address, VLAN number, SPI field, IP protocol number, source physical port

Note:

- [1] Certain traffic types such as *Mac-in-Mac*, *FCoE* and *SCTP* are currently not supported for field selection; hence they are not load balanced as opposed to the supported traffic types (see also the next section).
- [2] Traffic types are based on IANA's Ethertype definitions (e.g., 0x0800 for IPv4 and 0x86DD for IPv6). Packet fields are based on standard protocol definitions (IEEE 802.3 for Ethernet, RFC 791 and RFC 8200 for IPv4 and IPv6 respectively).
- [3] (v)LAG hashing applies to bridged as well as routed traffic without distinction.

Once the proper field values are extracted based on the traffic type (as per the table above), the second step performed by the hardware is the *hash function calculation*. NetVisor OS uses the standard *CRC16 CCITT* cyclic redundancy check algorithm for the hash function calculation.

However, before calculating the CRC, 128-bit IPv6 source and destination addresses get 'folded' into 32-bit shortened versions with the *FoldAndXOR* method:

address_bits(127:96) ^ address_bits(95:64) ^ address_bits(63:32) ^ address_bits(31:0)

where ^ means 32-bit XOR.

This preserves the variability over the entire 128-bit address length when fed in 32-bit chunks to the CRC16 CCITT calculation.

Starting from NetVisor OS release 6.1.0, the default hashing mode is called `enhanced` hashing. It reflects the hashing field selection described in ***Table 4-1*** above.

As an alternate configuration for special cases, it is possible to select only a subset of the fields in Table 4-1 by explicitly configuring the hash mode to one of the following hashing options:

- *source MAC address based*
- *destination MAC address based*
- *source and destination MAC address based*
- *source IP address based*
- *destination IP address based*
- *source and destination IP address based*

These modes are less granular than the default enhanced mode, therefore they should be used only when the purpose is to have a less granular load balancing behavior (which in most cases means a less optimal traffic distribution).

About Resilient Hashing

Starting from NetVisor OS release 6.1.0, a new resilient hashing mode is introduced. It is called `resilient` because it helps prevent unnecessary traffic disruption when the number of trunk member ports changes. It takes advantage of regular hashing field selection as shown in ***Table 4-1*** and employs an advanced table-based mechanism to provide better port selection resiliency. This feature is applicable to unicast flows only.

Resilient hashing can be configured when a trunk is created. Its configuration is not modifiable on the fly.

In case of auto trunks, they will come up using the default hashing mode, that is, enhanced. In order to apply resilient mode to these trunks, they need to be converted into regular (manual) trunks. In other words, for each of them you need to disable all the required auto-trunk ports, delete the trunk and re-create it as a regular trunk with resilient hashing configured. However, in such case you lose the convenience of the automatic trunk creation.

About Symmetric Hashing

In addition to regular (asymmetric) hashing, NetVisor OS supports symmetric hashing for IPv4 and IPv6 traffic. When this mode is enabled, IP traffic is *normalized* before the fields are hashed. That means that the source and destination IP addresses and Layer 4 ports are swapped so that both directions of a bidirectional IP connection are hashed to the same output port in a LAG.

Symmetric hashing's normalization algorithm has to ignore certain fields that can vary on a per direction basis, such as the VLAN number, protocol value, and the ingress physical interface.

Moreover, as shown in ***Table 4-1,*** for the Hardware Hashing Field Selection, certain fields such as the GRE key or the IPSec SPI are intrinsically unidirectional. If any GRE or IPSec traffic is expected in the network, then NetVisor OS provides an additional option for symmetric hashing to ignore these unidirectional GRE and IPSec fields so that the normalization only uses source and destination IP addresses and L4 ports for the hash computation.

Symmetric hashing is useful when traffic in both directions needs to be hashed to the same member port in a LAG. This is required, for instance, for analysis and correlation purposes, when a monitoring device connected to a member port of a trunk needs to receive a copy of all the packets of a connection in both directions (that is, both requests and replies).

Configuring Layer 3 Features

This chapter provides information about the protocols and configurations supported on Layer 3 network (aka Network layer protocols) by using the NetVisor OS command line interface (CLI) on a NetVisor OS switch.

Understanding the Layer 3 Protocols in NetVisor OS

Layer 3 is the Network layer protocol that behaves as a transmission medium from the source to the destination by performing the routing and addressing aspects of a packet and ensuring data integrity without having any packet loss.

NetVisor OS supports the following Layer 3 or Network protocols and functionalities:

- Open Shortest Path First (OSPF)
- OSPFv3
- Border Gateway Protocol (BGP)
- MP-BGP
- RIP
- VRRP for IPv4 and IPv6
- Dual-stack support (IPv4/IPv6)
- Equal-cost multi-path routing (ECMP)
- Policy Based Routing (PBR)
- Bidirectional Forwarding Detection (BFD) (IPv4 and IPv6), OSPF, BGP and static routes
- Static routes
- Loopback interface
- DHCP relay
- Multicast Listener Discovery (MLD)

Configuring Packet Relay for DHCP Servers

In general, routers do not forward broadcast packets from one subnet to another. However, there are cases in which the broadcast packets need to be passed onto other subnets. One typical example is DHCP (Dynamic Host Configuration Protocol) in which a new computer needs to acquire an available IP address by broadcasting the requests to DHCP server. DHCP protocols work seamlessly if both DHCP client and DHCP server are located in the same broadcast domain. However, that does not work across broadcast domains without supporting features such as DHCP relay. That is, when the DHCP server is not in the same subnet as the clients, use DHCP relay for broadcasting traffic.

Packet Relay is the router functionality that helps forward broadcast packets between broadcast domains. In NetVisor OS architect, the packet relay functionality is implemented as a service running within a particular vRouter.

You can configure a vRouter to relay DHCP requests from local clients to a centralized DHCP server. Since the initial DHCP request arrives from a client that typically does not have an IP address, the client must find the DHCP server using a Layer 2 broadcast.

The DHCP server needs information before the server can allocate an IP address to the client. It must know the subnet and the MAC address of the client. The DHCP server needs the subnet information to ensure that the IP address that the client receives can work on the client's subnet. The MAC address is necessary so that the DHCP server can find any information that is unique to the client.

When you configure DHCP relay on a vRouter, the vRouter converts the local broadcast packet from the client to a unicast packet and forwards it to the server.

Since the DHCP client does not have an IP address when it sends the DHCP request packet, the client uses the IP address, 0.0.0.0, as the source IP address and the general broadcast address 255.255.255.255 for the destination.

The vRouter replaces the source address with the IP address assigned to the interface where the request is received, and replaces the destination IP address with the address you specify in the vRouter packet-relay command.

To configure packet-relay for a DHCP server with the IP address `172.16.21.34` and vRouter interface (on the network where the client is connected to) - `eth11.100`, use the following syntax:

```
CLI (network-admin@switch) > vrouter-packet-relay add vrouter-name vrouter-dhcp forward-proto dhcp forward-ip 172.16.21.34 nic eth11.100
```

Once you add the configuration, you cannot modify it. If you make a mistake or want to add a new configuration, you must use the `vrouter-packet-relay-remove` command.

Configuring DHCP Packet Relay over VXLAN

NetVisor OS version 6.1.0 supports DHCP packet relay configuration over VXLAN. To demonstrate this functionality, consider the topology below:

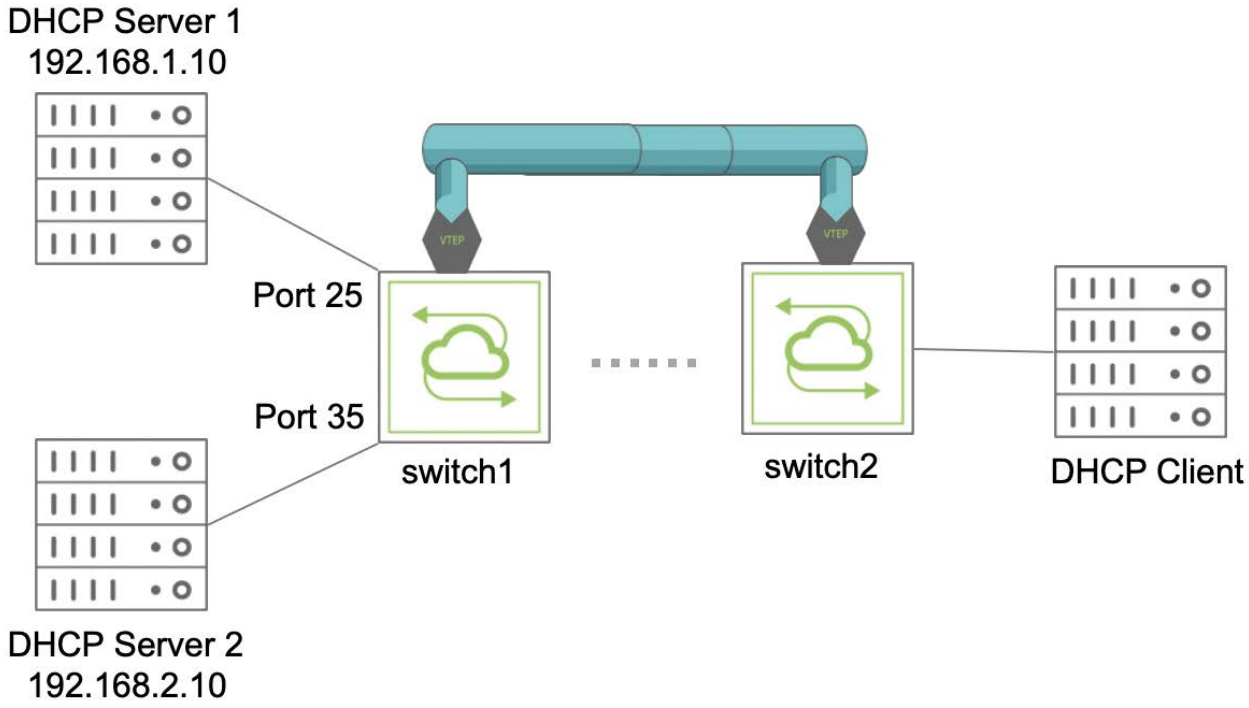


Figure 5-1 - DHCP Relay Configuration over VXLAN

The nodes switch1 and switch2 belong to two separate subnets and have VXLAN connectivity between them. For more information, see the *Configuring VXLAN* chapter. The DHCP servers 1 and 2 are connected to switch1 while the DHCP client is connected to switch2. The DHCP servers and the DHCP client belong to the VRF `vrf1`.

From NetVisor OS version 6.1.1, DHCP relays are VRF-aware. To create a VRF, use the command:

```
CLI (network-admin@switch1) > vrf-create name vrf1 scope fabric
```

Follow the steps below to configure DHCP relay between the DHCP servers and the client.

Create a VLAN for the DHCP client and associate a VXLAN with the VLAN:

```
CLI (network-admin@switch2) > vlan-create id 101 scope fabric vxlan 100101
```

Create VLANs for the DHCP servers and associate the VLANs with VXLANs:

```
CLI (network-admin@switch1) > vlan-create id 103 scope fabric ports 25
vxlan 100102

CLI (network-admin@switch1) > vlan-create id 104 scope fabric ports 35
vxlan 100103
```

Create a vRouter and vRouter interface for VLAN 101:

```
CLI (network-admin@switch2) > vrouter-create name vr1 vnet vnet1 router-
type hardware

CLI (network-admin@switch2) > vrouter-interface-add vrouter-name vr1 ip
```

```
192.168.102.1/24 vlan 101 vlan-type public if data if-nat-realm internal
vrf vrf1
```

Note: You also need to configure VRRP interfaces if the DHCP client is attached to a cluster.

View the configuration using the `vrouter-interface-show` command:

```
CLI (network-admin@switch1) > vrouter-interface-show vlan 101 format
vrouter-name,nic,ip,mac,vlan,vlan-type,nic-state

vrouter-name nic      ip              mac               vlan vlan-type nic-state
------------ -------- --------------- ----------------- ---- --------- ---------
vr1          eth2.102 192.168.102.1/24 66:0e:94:79:34:fe 101  public    up
```

Associate the vRouter to the VRF by using the command:

```
CLI (network-admin@switch1) > vrouter-vrf-add vrouter-name vr1 vrf vrf1
bgp-as 65100 router-id 110.0.1.1 bgp-redistribute connected
```

Use the `vrouter-vrf-show` command to view the vRouter VRF configuration:

```
CLI (network-admin@switch1) > vrouter-vrf-show format switch,vrouter-name,vrf,hw-vrid,bgp-as,router-id,bgp-
redistribute
switch  vrouter-name vrf      hw-vrid bgp-as router-id bgp-redistribute
------- ------------ -------- ------- ------ --------- ----------------
switch1 vr1          vrf1     1       65100  110.0.1.1 connected
```

Configure DHCP packet relay for both DHCP servers:

```
CLI (network-admin@switch2) > vrouter-packet-relay-add vrouter-name vr1
forward-ip 192.168.1.10 forward-proto dhcp nic eth2.102

CLI (network-admin@switch2) > vrouter-packet-relay-add vrouter-name vr1
forward-ip 192.168.2.10 forward-proto dhcp nic eth2.102
```

Here, `forward-ip` is the IP address of the DHCP server, `forward-proto` is the protocol supported by packet relay which is DHCP, and `nic` is the ingress interface of the vRouter on which DHCP broadcast packets are expected. The VRF ID is derived from the NIC value provided to the `vrouter-packet-relay-add` command.

View the packet relay configuration by using the command:

```
CLI (network-admin@switch2) > vrouter-packet-relay-show

vrouter-name forward-proto forward-ip    nic       vrf
------------ ------------- ------------  --------  ----
vr1          dhcp          192.168.1.10 eth2.102 vrf1
vr1          dhcp          192.168.2.10 eth2.102 vrf1
```

Note:

- If the DHCP client is connected through a vLAG to a cluster pair, you must configure vRouters, vRouter

interfaces, and DHCP packet relay for both the cluster switches.

- If you configure the DHCP relay and filter on different nodes, the uplink port (from the DHCP filter node to the relay) and the downlink port (from the DHCP filter node to the server interface) must be configured as trusted ports. For more information, see *Support for DHCP Snooping* section of the *Configuring Network Security* chapter.
- You cannot specify VRRP VIP NICs in the `vrouter-packet-relay` command. You can specify VRRP primary NICs instead.

Configuring vRouter Services

A Virtual Router (vRouter) is an important part of fabric functionality. For example, for a VLAN to communicate with other VLANs, or networks external to the fabric, it may need a vRouter that spans the internal and the external network.

Fabric-wide vRouter commands can only be executed at the fabric level by the fabric administrator.

> **Note:** For switches with ONVL, the only available vNET is a global vNET created when a fabric is created for the first time. Use tab complete in the CLI to display the vNET and continue the configuration. However, some switches support multiple vNETs based on the platforms. Please refer to the Release Notes for the supported platforms.

Routing protocols essentially work the same way on virtual routers as physical routers.

Configuring vRouter Interfaces

You can now add IPv4 and IPv6 addresses to a vRouter interface. If you enable or disable the interface, both IPv4 and IPv6 are affected. Routing protocols can be enabled or configured independently using the IP address.

When you add a vRouter interface, you can configure it with two IP addresses, IPv4 and IPv6 and can have multiple IP addresses (primary and secondary).

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name name-string vlan vlan-id ip ip-address netmask netmask assignment none|dhcp ip2 ip-address netmask2 netmask
```

You can display the IP addresses:

```
CLI (network-admin@switch) > vrouter-interface-show vrouter-name name-string vlan vlan-id ip ip-address netmask netmask assignment none|dhcp| dchpv6 ip2 ip-address netmask2 netmask
```

To migrate the interface from a IPv4 address to a IPv6 address, the following commands are used:

```
CLI (network-admin@switch) > vrouter-interface-ip-add
```

`vrouter-name` *`name-string`*	Specify the name of the vRouter.
`nic` *`nic-string`*	Specify the virtual NIC assigned to the interface.
`ip` *`ip-address`*	Specify the IP address assigned to the interface.
`netmask` *`netmask`*	Specify the netmask of the IP address.

Specify any of the following options:

`vnet` *`vnet-name`*	Specify the vNET assigned to the vRouter.

`CLI (network-admin@switch) > vrouter-interface-ip-remove`

`vrouter-name` *`name-string`*	Specify the name of the vRouter.
`nic` *`nic-string`*	Specify the virtual NIC assigned to the interface.
`ip` *`ip-address`*	Specify the IP address assigned to the interface.

`CLI (network-admin@switch) > vrouter-interface-ip-show`

`vrouter-name` *`name-string`*	Specify the name of the vRouter.
`nic` *`nic-string`*	Specify the virtual NIC assigned to the interface.
`ip` *`ip-address`*	Specify the IP address assigned to the interface.
`netmask` *`netmask`*	Specify the netmask of the IP address.
Specify any of the following options	
`vnet` *`vnet-name`*	Specify the vNET assigned to the vRouter.

Configuring MTU Parameters for vRouter Interfaces

Support for IPv4 and IPv6 addresses requires configuring the MTU parameter differently for each type of IP address. By default MTU is set at 1500 for the vNICs. For IPv4 addresses, the Maximum Transmission Unit (MTU) the range is to 68 to 9216, but the range for IPv6 addresses is 1280-9216. However, NetVisor OS displays the MTU range as 68-9216 which is supported by IPv4 addresses. But if you configure an IPv6 interface with an MTU value outside of the 1280-9216 for IPv6 addresses, NetVisor OS returns an error.

In the case of interfaces configured with IPv4 and IPv6 addresses (dual-stack), NetVisor OS limits the MTU range to 1280-9216. The following special rules apply to dual-stack interfaces:

- If you configure the MTU option for a vRouter interface with an IPv6 address using the command, `vrouter-interface-create`, NetVisor OS performs a range check to ensure the MTU value is within 1280-9348.
- If you configure the MTU option for a vRouter interface using the `vrouter-interface-modify` command, NetVisor OS performs a check for IPv6 addresses, and then performs a range check for 1280-9348.
- If you add an IPv6 address with the command, `vrouter-interface-ip-add`, NetVisor OS performs a range check to ensure the MTU of the VNIC interface is within 1280-9348.

- If you configure a hardware vRouter, NetVisor OS propagates the MTU value to the switch interface, so the MTU values are in the switch ASIC.

> **Informational Note**: Starting from NetVisor OS release 5.1.1 the maximum MTU size has been increased to 9348.

Configuring 802.1p-based Prioritization and Marking on Egress Interfaces

In Layer 3 networks, traffic priority is typically indicated in the Type of Service (ToS) or Diff Serv Code Point (DSCP) fields in the IPv4 or IPv6 headers of the packets.

Switches can be programmed to map these Quality of Service (QoS) or Class of Service (CoS) values to one of 8 traffic classes and then to use such class assignment to place packets into one of 8 egress queues of a port. This important capability (also known as class-based traffic queuing) helps to deal with traffic congestion in the network's choke-points.

However, in certain network designs such as, multi-tenant DC designs, it may not be always possible to use the ToS or DSCP fields for class-based prioritization.

An alternative solution is offered by the IEEE 802.1p standard classification, which allows any VLAN-tagged Ethernet traffic to be explicitly classified based on a Class of Service (CoS) 3-bit field, (also known as Priority Code Point (PCP).

Therefore, since packets forwarded on a Layer 3 interface do not usually carry an IEEE VLAN tag (and associated CoS field) by default, this enhancement introduces the capability of explicitly adding a CoS field in the traffic egressing a vRouter interface.

NetVisor OS now supports a new parameter for the configuration of an IEEE 802.1p priority tag (i.e., CoS) on an interface:

```
CLI (network-admin@switch) > vrouter-interface-add priority-tag|no-priority-tag

CLI (network-admin@switch) > vrouter-interface-modify priority-tag|no-priority-tag
```

When priority-tag is selected, NetVisor OS adds a CoS value based on the following ingress scenarios and configuration:

- Case A: On a vRouter interface when the ingress traffic contains an IEEE 802.1p priority tag (CoS), the default behavior is to trust such classification. Hence NetVisor OS uses the same CoS value received in ingress for the egress traffic.

- Case B: When the ingress traffic is untagged, NetVisor OS uses the default CoS value (0) for the egress traffic on the vRouter interface.

- Case C: When the ingress traffic contains a DSCP value and if the network admin adds a DSCP map to the port, then NetVisor OS will set the egress CoS value according to the configured DSCP map. In other words, when this specific configuration is chosen by the network admin, the cases A and B is overridden and the egress traffic's CoS value will instead be decided by the ingress DSCP map (while the ingress CoS value is ignored).

Configuring IPv6 for vRouter Loopback Addresses

NetVisor OS now supports IPv4 and IPv6 addresses on the same loopback interface.

When you configure a loopback interface, you can optionally associate a vRouter interface to the configuration. If you do not associate a vRouter interface, NetVisor OS selects the first non-VRRP vRouter interface. NetVisor OS uses the loopback interface for a vRouter as a host route in the Forwarding Information Database (FIB) for packets received from any port and routes them to the correct vRouter.

If there is no vRouter interface configured, the loopback interface is unreachable on the network. When you add the first vRouter interface, the loopback interface is reachable.

If you remove the associated vRouter interface, NetVisor OS selects the next available vRouter interface.

NetVisor OS does not support multiple vRouter loopback interfaces for IPv6 addresses.

To add an IPv4 address to a loopback interface for vRouter, vr1, use the following syntax:

```
CLI (network-admin@switch) > vrouter-loopback-interface-add vrouter-name
vr1 ip 99.1.1.1
```

To add an IPv6 vRouter loopback interface to vRouter, vr1, use the following syntax:

```
CLI (network-admin@switch) > vrouter-loopback-interface-add vrouter-name
vr1 ip 2999:1000::1
```

To display the vRouter loopback interface, use the `vrouter-loopback-interface-show` command:

```
CLI (network-admin@switch) > vrouter-loopback-interface-show

vrouter-name       index ip               router-if
174 ------------ ----- ------------ ---------
175 vr1             1      99.1.1.1      eth0.100
176 vr1             2      2999:1000::1 eth0.100
```

```
CLI (network-admin@switch) > vrouter-routes-show

 vrouter-name  network            type          interface    next-hop                    distance  metric
 ------------  ---------          ---------     ----------   --------------              --------  ------
 vr1           10.1.1.0/24        connected     eth0.100
 vr1           10.1.1.0/24        connected     eth0.100
 vr1           20.1.1.0/24        ospf          eth0.121     12.1.1.2                    110       20
 vr1           88.1.1.1/32        ospf          eth0.121     12.1.1.2                    110       20
 vr1           99.1.1.1/32        connected     lo
 vr1           2100:100::/64      connected     eth0.100
 vr1           2121:100::/64      connected     eth0.121
 vr1           2200:100::/64      ospf6         eth0.121     fe80::640e:94ff:fe4d:d0c8   110       10
 vr1           2888:1000::1/128   ospf6         eth0.121     fe80::640e:94ff:fe4d:d0c8   110       10
 vr1           2999:1000::1/128   connected     lo
```

```
vr1             fe80::/64           connected     eth0.121
```

Displaying FRR Routing and Debug Information for vRouters

This feature provides a new vRouter command to display the output of an IP Stack FRR (Free Range Routing) and debug information from the NetVisor OS command line interface(CLI).

```
CLI (network-admin@Spine1) > vrouter-vtysh-cmd
```

`vrouter-vtysh-cmd`	Specify the vRouter FRR command.
`name` *`name-string`*	name of service config.
`cmd` *`cmd-string`*	any FRR show/debug/undebug/no debug/clear ip/clear ipv6 command.

Show Output Examples

```
CLI (network-admin@Spine1) > vrouter-vtysh-cmd name vr1 cmd "show ip route"

Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, A - Babel,
         > - selected route, * - FIB route

C>* 100.1.1.0/24 is directly connected, eth1.100

C>* 127.0.0.0/8 is directly connected, lo0

CLI (network-admin@Spine1) > vrouter-vtysh-cmd name vr1 cmd "show running-
config"

Building configuration...

Current configuration:
!
frr version 7.2-pluribus
frr defaults traditional
hostname switch
log file zebra.log
log timestamp precision 6
log file bgpd.log
!
enable password zebra
password zebra
!
router-id 1.1.1.7
!
router bgp 650033
 bgp router-id 1.1.1.7
 coalesce-time 1000
 bgp bestpath as-path multipath-relax
 bgp prefer nh-global
 neighbor 18.18.18.2 remote-as 650033
 neighbor 10.10.50.129 remote-as 65001
```

```
 neighbor 10.10.60.129 remote-as 65002
 !
 address-family ipv4 unicast
  redistribute connected
  neighbor 18.18.18.2 next-hop-self
  neighbor 10.10.50.129 allowas-in
  neighbor 10.10.50.129 weight 200
  neighbor 10.10.60.129 allowas-in
  neighbor 10.10.60.129 weight 200
  maximum-paths 16
 exit-address-family
 !
 address-family ipv6 unicast
  redistribute connected
  maximum-paths 16
 exit-address-family
!
line vty
!
end
```

For this feature, only `show` commands are allowed as legal values in the CLI. For example,

```
CLI (network-admin@switch) > vrouter-vtysh-cmd name vr1 cmd "config t"

vrouter-vtysh-cmd: Illegal value for "cmd"
vrouter-vtysh-cmd: Legal values: commands starting with show, in quotes
```

Viewing FRR (Free Range Routing) Logs

FRR logs files, such as Zebra, OSPF, OSPF6, BGP, BFD and RIP can be viewed directly from the console using the NetVisor OS CLI.

Since FRR log files accumulate on the switch, you may want to clear a particular log file so they do not create space issues.

Use this command to view FRR logs files directly from your console:

```
CLI (network-admin@switch) > vrouter-log-show vrouter-name vrouter-name
protocol
```

`vrouter-log-show`	Displays vrouter protocol logs
`vrouter-name` *`vrouter-name`*	Specify name of the vRouter.
Specify any of the following options:	
`protocol` `zebra|ospf|ospf6|bgp|bfd|`	Specify the name of the protocol files to view.

`rip`

Use this command to clear FRR files from a specific protocol log file:

```
CLI (network-admin@switch) > vrouter-log-clear
```

`vrouter-log-clear`	Clears vrouter protocol logs from a protocol log file
`vrouter-name` *`vrouter name`*	Specify the name of the vRouter service.
Specify any of the following options:	
`protocol zebra|ospf|ospf6|bgp|bfd|rip`	Specify the name of the log file to clear.

Configuring Hardware-based Routing

Configuring Hardware Routing for a vRouter

Create interfaces on hardware routers and map them to virtual NICs in the vRouter zone. The number of hardware vRouters you can create depends on the platform. You can create only one vRouter on low-end platforms and can range up to 32 vRouters on high-end platforms.

The supported protocols are as follows:

- **OSPF** — OSPF does not use a TCP/IP transport protocol such as UDP or TCP, but is encapsulated in the IP datagram with protocol number 89. OSPF uses multicast addressing for route flooding on a broadcast domain. For non-broadcast network, special provisions in the configuration facilitate neighbor discovery. OSPF reserves the multicast addresses 224.0.0.5/6 for IPv4 or FF02::5/6 for IPv6.
- **BGP** — BGP uses TCP and port number 179.
- **RIP** — uses the following parameters:
  - RIPv1 — IPv4 uses UDP and port 520, and advertise address - broadcasting
  - RIPv2 — IPv4 uses UDP and port 520, and advertise address - 224.0.0.9
  - RIPng — IPv6 uses UDP and port 521, and advertise address - FF02::9
- **PIM** — IPv4 uses protocol 103 with multicast address 224.0.0.13

To create a hardware-based vRouter called `hwtest`, use the following command:

```
CLI (network-admin@switch) > vrouter-create hwtest router-type hardware
```

Use the same commands as software routing to add protocols and interfaces.

IPv6 Hardware Routing

NetVisor OS supports both IPv4 and IPv6 natively, and both can be configured on the same interface.

In addition to static routing, NetVisor OS supports the OSPFv3 and Multi-protocol BGP protocols for IPv6 unicast routing. vRouter interfaces can be configured with globally scoped IPv6 addresses.

```
CLI (network-admin@Spine-1) vrouter-interface-add vrouter-name vrouter-1 ip
2200:1111:2222:3333::1/64 vlan 100

CLI (network-admin@Spine-1) vrouter-interface-add vrouter-name vr0 ip
2200::1/64 vlan 200

CLI (network-admin@Spine-1) vrouter-interface-add vrouter-name vr0 ip
2211::1/64 vlan 211

CLI (network-admin@Spine-1) vrouter-ospf6-add vrouter-name vr0 nic eth0.200
ospf6-area 0.0.0.0

CLI (network-admin@Spine-1) vrouter-ospf6-add vrouter-name vr0 nic eth0.211
ospf6-area 0.0.0.0
```

BGP with IPv6 Unicast Peers Distributing IPv6 Prefixes

```
CLI (network-admin@Spine1) >  vlan-create id 200 scope fabric ports 9

CLI (network-admin@Spine1) >  vlan-create id 211 scope fabric

CLI (network-admin@Spine1) >  vnet-create name vnet0 scope fabric vlans 200

CLI (network-admin@Spine1) >  vrouter-create name vr0 vnet vnet0 router-
type hardware router-id 1.1.1.1 hw-vrrp-id 1

CLI (network-admin@Spine1) >  vrouter-interface-add vrouter-name vr0 ip
2200::1/16 vlan 200

CLI (network-admin@Spine1) >  vrouter-interface-add vrouter-name vr0 ip
2211::1/24 vlan 211

CLI (network-admin@Spine1) >  vrouter-modify name vr0 bgp-as 200 no-bgp-
ipv4-unicast

CLI (network-admin@Spine1) >  vrouter-modify name vr0 bgp-redistribute
static,connected

CLI (network-admin@Spine1) >  vrouter-bgp-add vrouter-name vr0 neighbor
2211::2 remote-as 222 multi-protocol ipv6-unicast

CLI (network-admin@Spine1) >  vrouter-bgp-add vrouter-name vr0 neighbor
2211::3 remote-as 233 multi-protocol ipv6-unicast
```

```
CLI (network-admin@Spine1) >  vrouter-bgp-network-add vrouter-name vr0
network 2200::/64

CLI (network-admin@Spine1) >  vrouter-bgp-network-add vrouter-name vr0
network 2211::/64
```

IPv6 and Prefix Lists

```
CLI (network-admin@Spine1) >  vrouter-prefix-list-add vrouter-name vr0 name
block200 action deny prefix 2200::/16 seq 95

CLI (network-admin@Spine1) >  vrouter-bgp-add vrouter-name vr0 neighbor
2211::2 remote-as 222 multi-protocol ipv6-unicast prefix-list-out block200
```

Static Route

```
CLI (network-admin@Spine1) >  vrouter-static-route-add vrouter-name vr0
network 7777::/64 gateway-ip 2200::9 distance 20
```

Layer 3 Table Validation

Layer 3 entries can become unsynchronized between the software table and the hardware table when you modify routes while the routes update in the network.

Use the `l3-setting-modify` command to automatically check Layer 3 entries:

```
CLI (network-admin@spine1) > l3-setting-modify
```

One or more of the following options:	
`aging-time` *`seconds`*	Specify the aging-time for Layer 3 entries, use 0 to disable aging.
`convergence-time` *`seconds`*	Specify the unicast convergence time on bootup (seconds)
`l3-checker|no-l3-checker`	Specify if you want to check Layer 3 consistency
`l3-checker-interval duration: #d#h#m#`	Specify the interval for Layer 3 consistency checker
`l3-checker-fix|no-l3-checker-fix`	Enable fixing of inconsistent entries

NetVisor OS uses two commands to manually check and fix Layer 3 inconsistencies:

```
CLI (network-admin@spine1) > l3-check-show

CLI (network-admin@spine1) > l3-check-fix
```

Displaying Hardware Routes History

You can display the history of hardware routes in the RIB table. This is useful when troubleshooting hardware routing in the network.

```
CLI (network-admin@spine1) > vrouter-rib-history-show time 15:30

time       caller      reason      ip            prelen   nexthop       flags                    vlan  intf_ip       i
-------- --------- ------ ----------- ------ ----------- ------------------   ---- -----------     ---
15:29:41  router-if   add       10.16.111.0   24        10.16.111.1  in-hw,local-subnet  111   10.16.111.1
15:29:43  router-if   add      10.16.100.0   24       10.16.100.1  in-hw,local-subnet  100   10.16.100.1    1
```

You can also modify the settings for collecting the history:

```
CLI (network-admin@spine1) > vrouter-rib-history-settings-modify
```

Understanding ECMP Path Selection and Load Balancing

NetVisor supports a number of technologies for traffic load balancing and path redundancy, such as LAG and vLAG.

For Layer 3 designs, NetVisor OS supports standard routing protocols such as OSPF and BGP. When multiple next hops exist, NetVisor OS employs ECMP (Equal-Cost Multi-Path) routing to choose amongst the available paths for traffic load balancing and redundancy.

In order to perform path selection in hardware, ECMP uses a high-performance technology called *packet field hashing*.

What that means is that the hardware extracts a number of packet fields and with them performs a special calculation to generate a *hardware index*. Such index is then used to select a next hop for an ECMP routing decision. Up to 16 next hops are supported for load balancing by the hardware-based hashing function.

About Layer 3 Hardware Hashing

A hardware hashing operation comprises two parts that can be performed at extremely high speed: first, packet field selection and extraction, then a mathematical bitwise computation.

The first step therefore is for the hardware parsing logic to select a number of packet fields 'to be hashed' depending on the traffic type. These are the traffic types and related packet fields that NetVisor OS currently supports *by default* to perform hashing on:

Table Caption 5-1: Hardware Hashing Field Selection

Application	Traffic Type[1]	Hashed Fields
Plain unicast and multicast IPv4 packets	IP	Source IPv4 address, Destination IPv4 address, VLAN number, Destination Layer 4 port, Source Layer 4 port, IP protocol number, source physical port
Plain unicast and multicast IPv6 packets	IP	Source IPv6 address, Destination IPv6 address, VLAN number, Destination Layer 4 port, Source Layer 4 port, IP protocol number, source physical port
VXLAN packets (UDP encapsulated)	IP	Source IP address, Destination IP

		address, VLAN number, Destination Layer 4 port, *Source Layer 4 port*, IP protocol number, source physical port

Note: [1] Traffic types are based on IANA's Ethertype definitions (e.g., 0x0800 for IPv4 and 0x86DD for IPv6). Packet fields are based on standard protocol definitions (RFC 791 and RFC 8200 for IPv4 and IPv6 respectively).

Once the proper field values are extracted based on the traffic type (as per the table above), the second step performed by the hardware is the *hash function calculation*. NetVisor OS uses the standard *CRC16 CCITT* cyclic redundancy check algorithm for the hash function calculation.

However, before calculating the CRC, 128-bit IPv6 source and destination addresses get 'folded' into 32-bit shortened versions with the *FoldAndXOR* method:

address_bits(127:96) ^ *address_bits(95:64)* ^ *address_bits(63:32)* ^ *address_bits(31:0)*

where ^ means 32-bit XOR.

This preserves the variability over the entire 128-bit address length when fed in 32-bit chunks to the CRC16 CCITT calculation.

Configuring Static Routes

vRouters forward packets using either routing information from manually configured route tables or routing information calculated using dynamic routing algorithms.

Static routes define explicit paths between two vRouters and are not automatically updated. When network changes occur, you have to reconfigure static routes. Static routes use less bandwidth than dynamic routes.

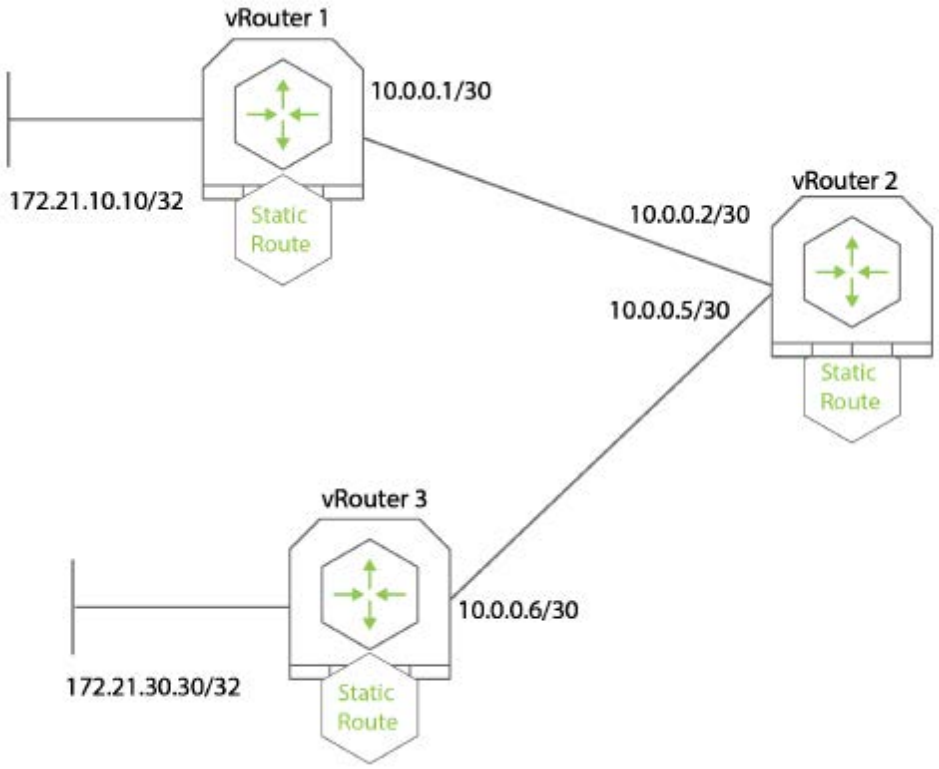


Figure 5-2 Configuring a Static Route

In this example, you configure a static route on vRouter1 by using the command `vrouter-static-route-add`.

```
CLI (network-admin@switch) > vrouter-static-route-add
```

`vrouter-name` *`name-string`*	Specify the name of the vRouter service.
`network` *`ip-address`*	Specify the IP subnet of the network that you want to add a static route.
`netmask` *`netmask`*	Specify the netmask of the IP subnet.
`gateway-ip` *`ip-address`*	Specify the IP address of the gateway that you want to route packets destined for the network IP address to.

`drop`	Specify to drop traffic matching this route. **Note:** This option is mutually exclusive with `gateway-ip` parameter.
`bfd-dst-ip` *`ip-address`*	Specify the destination IP address for BFD monitoring.
`distance` *`number`*	Specify the administrative distance as a number between 0 and 255. • 0 – Connected interface • 1 – Static route • 110 – OSPF • 120 – RIP • 200 – Internal BGP
`interface` *`vrouter-interface-nic`*	Specify the vRouter interface for the static route, if the `gateway-ip` is a link-local IPv6 address.
`vrf` *`vrf`*	Specify the name of the VRF.

To create a static route on vRouter1 for the network 172.16.10.10/24 with a gateway IP address of 172.16.20.1, use the command:

```
CLI (network-admin@switch) > vrouter-static-route-add vrouter-name vRouter1
network 172.16.10.10/24 gateway-ip 172.16.20.1
```

To modify the configuration, use the command `vrouter-static-route-modify`.

```
CLI (network-admin@switch) > vrouter-static-route-modify vrouter-name
vRouter1 network 172.16.10.10/24 gateway-ip 172.16.20.1 distance 20
```

Note: The administrative distance is the only parameter that can be modified using the `vrouter-static-route-modify` command. You must supply other parameters including `network` and `gateway-ip` correctly to make the modification.

To view the details of the configuration, use the command:

```
CLI (network-admin@switch) > vrouter-static-route-show

switch    vrouter-name network        gateway-ip  bfd-dst-ip distance interface vrf
--------- ------------ -------------- ----------- ---------- -------- --------- ---
switch1   vRouter1     172.16.10.0/24 172.16.20.1 ::         20
```

To remove the static route, use the command `vrouter-static-route-remove`. For example:

```
CLI (network-admin@switch) > vrouter-static-route-remove vrouter-name
vRouter1 network 172.16.10.10/24
```

Adding IPv6 Link-Local Addresses for Static Routing

IPv6 link-local addresses have a specific prefix `fe80::/10` and are relevant only on the local link. This makes it possible to have the same link-local address on multiple IPv6 interfaces. If you add a static route reachable through a link-local address, you must specify the outgoing interface.

For example, use the command below to specify the interface for a static route with an IPv6 link-local address as `gateway-ip`:

```
CLI (network-admin@switch) > vrouter-static-route-add vrouter-name vr1
network 6666::/64 gateway-ip fe80::640e:94ff:fe6f:8521 interface eth0.4092
```

To view the details, use the command:

```
CLI (network-admin@switch) > vrouter-static-route-show

switch vrouter-name network   gateway-ip               bfd-dst-ip distance interface
------ ------------ --------- ------------------------ ---------- -------- ---------
Leaf1   vr1         6666::/64 fe80::640e:94ff:fe6f:8521   ::         1       eth0.4092
```

Configuring Static Null Routing

You add a static route to a Null 0 interface when an access server with many clients is in your network. This causes NetVisor OS to install host routes in the access server routing table. To ensure reach-ability to these clients, while not flooding the entire network with host routes, other routers in the network typically have a summary route pointing to the access server. In this type of configuration, the access server has the same summary route pointing to the access server Null 0 interface. If not, routing loops may occur when outside hosts attempt to reach IP addresses not currently assigned to a client but are part of the summary route. The access server bounces the packets back over the access server default route into the core network, and because the access server lacks a specific host route for the destination.

This feature introduces a new parameter for the command, `vrouter-static-route-add`:

```
CLI (network-admin@switch) > vrouter-static-route-add vrouter-name vr1
network 192.168.2.0/24 gateway-ip ip-address
drop          Drop packets matching this route
```

NetVisor OS designates the drop keyword as mutually exclusive with keywords such as gateway or interface. Either drop or gateway must be used for valid syntax. Static routes with gateways cannot co-exist with drop routes for the same prefix.

```
CLI (network-admin@switch) > vrouter-static-route-show

vrouter-name network         gateway-ip    bfd-dst-ip distance  drop
------------ --------------- ------------- ---------- --------  ----
vr1          192.168.20.0/24 ::             ::         1        yes
vr1          192.168.30.0/24 192.168.100.2  ::         1        no
vr1          192.168.22.0/23 ::             ::         1        yes

CLI (network-admin@switch) > vrouter-routes-show format network, type,
interface, next-hop, distance

vrouter-name network               type      interface next-hop      distance
------------ --------------------- --------- --------- ------------- --------
vr1          192.168.20.0/24       static     Null0                    1
vr1          192.168.22.0/23       static     Null0                    1
vr1          192.168.30.0/24       static     eth2.4092 192.168.100.2  1
vr1          192.168.100.0/24      connected  eth2.4092
vr1          2018:192:168:100::/96 connected  eth2.4092
vr1          fe80::/64             connected  eth2.4092
```

Configuring Static ARP for Unicast Traffic

A static Address Resolution Protocol (ARP) entry is a permanent entry in your ARP cache. You can create mappings between IP addresses and MAC addresses (called static ARP entry), which can remain active without aging out for the specified time. NetVisor OS enables you to create static ARP entries for security and troubleshooting purposes.

One of the use cases in configuring static ARP table entries is to keep the Layer 2 and Layer 3 entries active for select hosts (IP addresses) even if those hosts do not send traffic very often. Another reason to add a static ARP entry is if you want two hosts to communicate regularly and can have a binding of the MAC address and the IP address of the hosts in general.

Note: You must create a vRouter interface on the switch to send ARP requests and enable ARP binding.

The static ARP binding is established only after the first ARP packet is communicated (received and replied) between the vRouter and the host. Thereafter, the binding remains permanent by sending and receiving the ARP requests and replies as in the case of *host refresh* requests.

Note: You can enable a static ARP on a Layer 3 entry only if the Layer 3 entry is active (see `l3-table-show` output below).

To create a static ARP entry using MAC address, use the command:

```
CLI (network-admin@switch) > static-arp-create scope [local|cluster|fabric]
mac mac-address ip ip-address
```

`static-arp-create`	Creates a unique static ARP entry
`scope [local|cluster|fabric]`	Specify the scope. A static entry with scope, `cluster` or `fabric` creates a static ARP entry configuration on one switch and enables the static flag for Layer 3 entry in the L3 table for all the other switches in the fabric or cluster.
`mac mac-address`	Unicast MAC address to bind with the IP address of the host
`ip ip-address`	IP address of the host to bind with the MAC address.
`vlan/vxlan (optional keywords)`	Specify the VLAN ID or VXLAN name for the static ARP entry configuration. You cannot specify both VLAN and VXLAN on the same configuration at the same time. You can specify either VLAN or VXLAN (both are optional parameters in establishing a static ARP entry.

To have the static ARP entries functional, you must have an active Layer 2 entry and a corresponding Layer 3 entry in the system.

After configuring the static ARP entry using the above CLI command, follow the tasks to ensure a working configuration:

- Check the matching Layer 3 entry (i.e, the host interface IP address and MAC address) on the switch
  - If there is no information available, no action is required. You can view the details using the `static-arp-show` command.
  - If a matching Layer 3 entry is found, that entry is marked with a *static* flag and you can verify this using the `l3-table-show ip <ip-address> mac <mac-address>` command.
- **When the Layer 3 entry age-out timer expires**, The static Layer 3 entries are kept alive by forced *arp-refresh*. That is, the vRouter sends an ARP request to the host and when a ARP reply is received. the L3 entry gets refreshed and remains active. This ARP reply ensures that the Layer 2 entry is refreshed, thereby, keeping the Layer 3 entry active and hence keeping the static ARP binding intact.
- If no ARP reply is received (when the port is down), then the Layer 2 entry ages out and the corresponding Layer 3 entry also ages out, keeping the *static* flag on the Layer 3 entry.
- If the ARP replies are received on a different port for the same IP address or MAC address, that triggers a modification of the previous Layer 2 entry with the new port details, which reactivates the previous Layer 3 entry and send out the ARP refresh messages. (MAC move happens)
- When the Layer 2 entry age-out timer expires, it deactivates the Layer 2 entry and the corresponding Layer 3 entry.

> **Note**: While configuring static ARP, ensure that:
>
> - the IP address is not 0.0.0.0 or :: for IPv6 addresses.
> - the IP address is not multicast or broadcast
> - the MAC is unicast only.

To delete a static ARP entry, use the command:

```
CLI (network-admin@switch) > static-arp-delete ip ip-address
```

When you delete the static ARP configuration, the corresponding Layer 3 entry is cleared off the *static* flag. In cases where there are multiple static ARP entries within the same VLAN, you must use specific parameters to delete the static ARP entry. For example, in such cases, include the required parameters in the command:

```
CLI (network-admin@switch) > static-arp-delete ip ip-address mac mac-address vlan/vxlan
```

To view the details, use the command:

```
CLI (network-admin@switch) > static-arp-show scope [local|cluster|fabric] mac mac-address ip ip-address
```

Below is a sample configuration for creating static ARP using the commands described earlier in this section. To create a static ARP binding between the host IP address 172.179.1.120 with the host MAC address 00:12:c0:88:0c:1d, use the command:

```
CLI (network-admin@switch) >  static-arp-create ip 172.148.0.0 mac
00:12:00:88:0c:00

CLI (network-admin@switch) >  static-arp-show scope local ip 172.148.0.0

switch  scope     ip             mac
------- ------ ------------  -------------------
switch  local  172.148.0.0  00:12:00:88:0c:00
```

When the host is actively sending ARP replies, the show command displays:

```
CLI (network-admin@switch) > l3-table-show ip 172.148.0.0 format all show-
interval 1 layout vertical

mac:                    00:12:00:88:0c:00
ip:                     172.148.0.0
vlan:                   4092
intf:                   29
hw-intf:                29
rt-if:                  eth1.4092
state:                  active,static
egress-id:              100008
hit:                    1
```

When the static ARP entry is removed, the show output displays:

```
CLI (network-admin@switch) > l3-table-show ip 172.148.0.0 format all show-
interval 1 layout vertical

mac:                    00:12:00:88:0c:00
ip:                     172.148.0.0
vlan:                   4092
intf:                   29
hw-intf:                29
rt-if:                  eth1.4092
state:                  active
egress-id:              100008
hit:                    23
```

Configuring VRF Aware Static ARP

NetVisor OS offers static ARP support for VRF subnets configured with VXLAN. To have the static ARP entries functional, you must have an active VRF with subnets and a corresponding Layer 3 entry in the system. You must also create a subnet anycast gateway IP on the switch to send ARP requests. As in the case of static ARP on vRouters where ARP-reply is sent to the vRouter interface alone, the ARP-reply in this case is sent only to the subnet anycast gateway IP. For VRF configuration details, refer to the VRF section in the VXLAN chapter.

To create a static ARP entry on VXLAN enabled VRF, use the command `static-arp-create.` For example:

```
CLI (network-admin@switch) > static-arp-create scope cluster mac
00:2d:01:00:00:02 ip 30.1.2.200 vxlan 3012
```

To view the configuration details for static ARP and VRF, use the following commands:

```
CLI (network-admin@switch) > static-arp-show

scope   ip          vxlan vlan mac
------- ----------- ----- ---- -----------------
cluster 30.1.2.200 3012  0    00:2d:01:00:00:02

CLI (network-admin@switch) > vrf-show layout vertical

name:                     vrf1
vnet:                     0:0
scope:                    local
anycast-mac:              64:0e:94:40:00:02
vrf-gw:                   ::
vrf-gw2:                  ::
vrf-gw-ipv6:              ::
vrf-gw2-ipv6:             ::
active:                   yes
hw-router-mac             hw-vrid
enable                    yes
description:

CLI (network-admin@switch) > subnet-show layout vertical

name:                     sub2
scope:                    local
vlan:                     3012
vxlan:                    3012
vrf:                      vrf1
network:                  30.1.2.0/24
anycast-gw-ip             30.1.2.1
packet-realy:             disable
forward-proto:            dhcp
state:                    ok
enable:                   yes
description:
```

To view the L3 table when the host is actively sending ARP replies, use the command `l3-table-show.`

```
CLI (network-admin@switch) > l3-table-show vxlan 3012 state active layout
vertical

switch:                   switch
mac:                      00:2d:01:00:00:02
ip:                       30.1.2.200
vlan:                     3012
vxlan:                    3012
rt-if:
state:                    active, static
egress-id:
tunnel:
switch:                   switch2
mac:                      00:2d:01:00:00:02
```

```
ip:                     30.1.2.200
vlan:                   3012
vxlan:                  3012
rt-if:                  sub2
state:                  active, static, vxlan-loopback
egress-id:              100064
tunnel:                 auto-tunnel-10.30.0.1_10.30.1.1
```

After the static entry is removed, the L3 table output does not show the '`static`' state. This can be seen from the example below:

```
CLI (network-admin@switch) > l3-table-show vxlan 3012 state active layout
vertical

switch:                 switch
mac:                    00:2d:01:00:00:02
ip:                     30.1.2.200
vlan:                   3012
vxlan:                  3012
rt-if:
state:                  active
egress-id:
tunnel:
switch:                 switch2
mac:                    00:2d:01:00:00:02
ip:                     30.1.2.200
vlan:                   3012
vxlan:                  3012
rt-if:                  sub2
state:                  active, vxlan-loopback
egress-id:              100064
tunnel:                 auto-tunnel-10.30.0.1_10.30.1.1
```

Guidelines and Limitations While Configuring Static ARP

- The host refresh time is twice the Layer 3 aging time (2*L3 age).
- You must create a vRouter interface on the switch to send ARP request. For VRF aware static ARP, a VRF subnet should be created.
- Host sends the ARP reply to the vRouter interface only. In the case of VRF aware static ARP, the ARP reply is sent to subnet anycast gateway IP.
- In cases where there are multiple static ARP entries within the same VLAN, use the `static-arp-delete ip` *`ip-address`* `mac` *`mac-address`* `vlan/vxlan` command to delete the specific entry.
- In the beginning, the Layer 3 entry should be activated with ping for the static ARP entry to be functional.
- Static ARP entry can be created with VLAN or VXLAN options. You cannot have both VLAN and VXLAN at the same time while creating the static ARP entry. For VRF-aware static ARP, the static ARP entry should be created only with the VXLAN option.
- When you create a static ARP entry with scope, `cluster` or `fabric`, NetVisor creates a static ARP entry configuration on one switch and enables the static flag for Layer 3 entry in the L3 table for all the other switches in the fabric or cluster.
- The modify static ARP command is not supported.

Configuring IPv4 and IPv6 Neighbor Discovery Process and Optimization

The IPv6 Neighbor Discovery Process (NDP) uses ICMPv6 messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), verify the reach-ability of a neighbor, and keep track of neighboring routers. NDP provides the same functionality as ARP in an IPv4 network. NDP has additional features such as auto-configuration of IPv6 addresses and duplicate address detection (DAD).

In an IPv6 Layer 3 network, a NetVisor OS vRouter can be configured as a First Hop Router and send Router Advertisements to announce the presence, host configuration parameters, routes, and on-link prefixes. In a Layer 2 network, NetVisor OS can enable NDP optimization to prevent flooding of neighbor solicitation messages.

Supported NDP Messages

- Router Solicitation (ICMPv6 type 133)
- Router Advertisement (ICMPv6 type 134)
- Neighbor Solicitation (ICMPv6 type 135)
- Neighbor Advertisement (ICMPv6 type 136)
- Redirect (ICMPv6 type 137)

NetVisor OS sends Neighbor Solicitation messages (ICMPv6 Type 135) on the local link by nodes attempting to discover the link-layer addresses of other nodes on the local link. NetVisor OS sends the Neighbor Solicitation message to the solicited-node multicast address. The source address in the neighbor solicitation message is the IPv6 address of the node sending the Neighbor Solicitation message. The Neighbor Solicitation message also includes the link-layer address of the source node.

After receiving a Neighbor Solicitation message, the destination node replies by sending a Neighbor Advertisement message (ICPMv6 Type 136) on the local link. The source address in the Neighbor Advertisement message reflects the IPv6 address of the node sending the Neighbor Advertisement message. The destination address reflects the IPv6 address of the node sending the Neighbor Solicitation message. The data portion of the Neighbor Advertisement message includes the link-layer address of the node sending the Neighbor Advertisement message.

After the source node receives the Neighbor Advertisement, the source node and destination node communicate.

NetVisor OS uses Neighbor Solicitation messages to verify the reach-ability of a neighbor after identifying the link-layer address of a neighbor. When a node requires verification of the reachability of a neighbor, the destination address in a Neighbor Solicitation message includes the unicast address of the neighbor.

NetVisor OS sends Neighbor Advertisement messages when a change occurs in the link-layer address of a node on a local link. When there is such a change, the destination address for the Neighbor Advertisement includes the all-nodes multicast address.

NetVisor OS periodically sendsRouter Advertisement messages (ICMPv6 Type 134) to each IPv6 configured interface of security appliance. NetVisor OS also sends the Router Advertisement messages to the all-nodes multicast address.

Router Advertisement messages typically include the following information:

- One or more IPv6 prefix the nodes use on the local link to automatically configure the IPv6 addresses.
- Lifetime information for each prefix included in the advertisement.
- Sets of flags that indicate the type of auto-configuration (stateless or stateful) that can be completed.
- Default router information (whether the router sending the advertisement should be used as a default router and, if so, the amount of time (in seconds) the router should be used as a default router).
- Additional information for hosts, such as the hop limit and MTU a host should use in origination packets.
- The amount of time between neighbor solicitation message re-transmissions on a given link.
- The amount of time a node considers a neighbor reachable.

NetVisor OS sends Router Advertisements t in response to Router Solicitation messages (ICMPv6 Type 133). Hosts sends Router Solicitation messages at system startup so that the host can immediately auto-configure without waiting for the next scheduled router advertisement message. Router Solicitation messages usually sent by hosts at system startup, and the host does not have a configured unicast address, the source address in Router Solicitation messages includes the unspecified IPv6 address (0:0:0:0:0:0:0:0). If the host has a configured unicast address, the source address in the message uses the unicast address of the interface sending the Router Solicitation message. The destination address in Router Solicitation messages uses the all-routers multicast address with a scope of the link. When sending a Router Advertisement in response to a Router Solicitation message, the destination address in the Router Advertisement message uses the unicast address of the source of the Router Solicitation message.

Configure the following settings for router advertisement messages:

- The time interval between periodic Router Advertisement messages. NetVisor OS uses the default time interval of 200 seconds with a range of 3 to 1800 seconds or 500 to 1800000 milliseconds if you specify milliseconds.

- The router lifetime value, which indicates the amount of time IPv6 nodes should consider the switch to be the default router. Valid values range from 0 to 9000 seconds. NetVisor OS has a default value of 1800 seconds. Entering 0 indicates that the switch is not considered a default router on the selected interface.

- The IPv6 network prefixes in use on the link. In order for stateless auto-configuration to work properly, the advertised prefix length in Router Advertisement messages must always be 64 bits.

- Whether or not an interface transmits Router Advertisement messages. By default, NetVisor OS automatically sends Router Advertisement messages in response to Router Solicitation messages. If you suppress the Router Advertisement messages, the switch appear as a regular IPv6 neighbor on the link and not as an IPv6 router.

Unless otherwise noted, the interface has specific the Router Advertisement message settings.

To configure NDP, use the vrouter-interface-config-add command:

```
CLI  (network-admin@switch)  >  vrouter-interface-config-add
```

`vrouter-name name-string`	Specify the name of the service configuration.
Specify one of the options below:	Options:
`nic vrouter if-list nic`	Specify the vnic name.
`ospf-hello-interval <1..65535>`	Specify the OSPF hello interval from 1 to 65535. The default value is 10 (IPv4 or IPv6).
`ospf-dead-interval <2..65535>`	Specify the OSPF dead interval from 2 to 65535. The default value is 40 (IPv4 or IPv6).
`ospf-retransmit-interval <3..65535>`	Specify the OSPF retransmit interval from 3 to 65535. The default value is 5 (IPv4 or IPv6).
`ospf-priority <0..255>`	Specify the OSPF priority from 0 to 255. The default value is 1 (IPv4 or IPv6).
`ospf-auth-key ospf-auth-key-string`	Specify the OSPF authentication key (IPv4 only).
`ospf-cost <0..65535>`	Specify the OSPF Cost (IPv4 or IPv6).
`ospf-msg-digest-id <0..255>`	Specify the OSPF digest ID from 0 to 255 (IPv4 only).
`ospf-msg-digest-key ospf-msg-digest-key-string`	Specify the OSPF message digest key (IPv4 only).
`ospf-passive-if|no-ospf-passive-if`	Specify the OSPF passive interface (IPv4 or IPv6).
`ospf-network-type default|point-to-point`	Specify the OSPF network type (IPv4 or IPv6).
`ospf-bfd default|enable|disable`	Specify the BFD protocol support for OSPF fault detection.
`bfd-interval <200..3000>`	Specify the BFD desired transmit interval from 200 ms to 3000 ms. The default value is 750 ms.
`bfd-min-rx <200..3000>`	Specify the BFD required minimum receive interval from 200 ms to 3000 ms. The default value is 500 ms.
`bfd-multiplier <1..20>`	Specify the BFD detection multiplier from 1 to 20. The default value is 3.
`nd-suppress-ra|no-nd-suppress-ra`	Control the transmission of IPv6 Router Advertisements.
`ra-interval <1..1800>`	Specify the time interval between ipv6 router advertisements.
`ra-lifetime <0..9000>`	Specify the time for which router is considered as default router.

Configuring Routing Information Protocol (RIP)

Routing Information Protocol (RIP) remains the oldest routing protocol and provides networking information to routers. Routers need to know the available networks and the distance required to reach it.

RIP consists of a distance vector protocol, and uses hop counts to determine distance and destination. Every 30 seconds, RIP sends routing information to UDP port 520. If you designate the router as default gateway, the router advertises by sending 0.0.0.0 with a metric of 1.

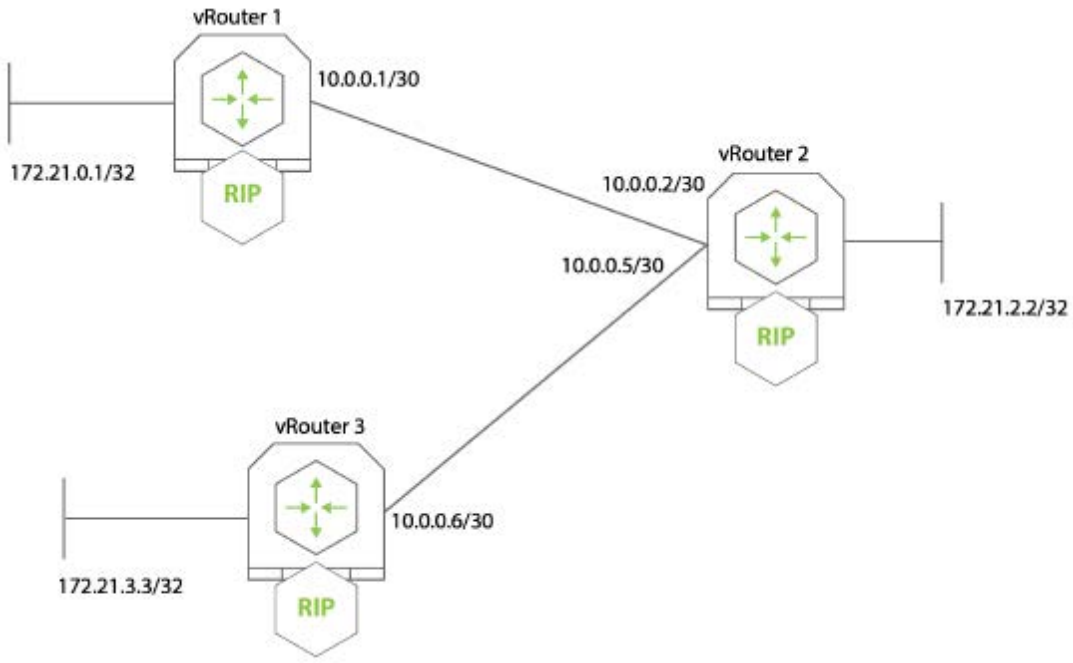


Figure 5-3: RIP Topology

1) Create vRouter1 on VLAN33:

```
CLI (network-admin@switch) > vrouter-create name vrouter1 fabricname-global router-type hardware
```

You can also specify how NetVisor OS distributes RIP routes using the parameter, `rip-redistribute static|connected|ospf|bgp`.

2) Add network `10.16.33.0/24` to vrouter1:

```
CLI (network-admin@switch) > vrouter-rip-add vrouter-name vrouter1 network 10.16.33.0/24 metric 2
```

3) Add network `10.16.35.0/24` to vrouter1:

```
CLI (network-admin@switch) > vrouter-rip-add vrouter-name vrouter1 network 10.16.55.0/24 metric 2
```

4) To view the configuration, use the `vrouter-rip-show` command. This displays all RIP routes

configured using the `vrouter-rip-add` command.

To view RIP routes not configured using the `vrouter-rip-add` command, use the `vrouter-rip-routes-show` command.

Configuring Open Shortest Path First (OSPF)

Open Shortest Path First (OSPF) is a robust link-state interior gateway protocol (IGP). It uses the concept of Areas which allows further segmentation on the network.

OSPF uses link-state information to make routing decisions, and make route calculations using the shortest path first (OSPF) algorithm. Each vRouter configured for OSPF floods link-state advertisements throughout the area that contains information about the interfaces attached to the router and routing metrics.

You can add more configuration options, such as hello intervals, for OSPF using the `vrouter-interface-config` commands. In addition, you can add stub or not-so-stubby areas to the OSPF configuration.

You can manually change the OSPF cost for the configuration. Cost is the metric used by OSPF to judge the feasibility of a path. If you specify 0 as the cost, the vRouter automatically calculates the cost based on the bandwidth of the interface. In NetVisor OS, the OSPF value does not change.

> **Note:** For switches with ONVL, the only available vNET is a global vNET created when a fabric is created for the first time. Use tab complete in the CLI to display the vNET and continue the configuration.

In this example, you configure OSPF for two vRouters with an area, zero. The network has the following configuration:

- VLAN 35 with IP addresses 10.0.3.0/24
- VLAN 55 with IP addresses 172.37.0.0/24

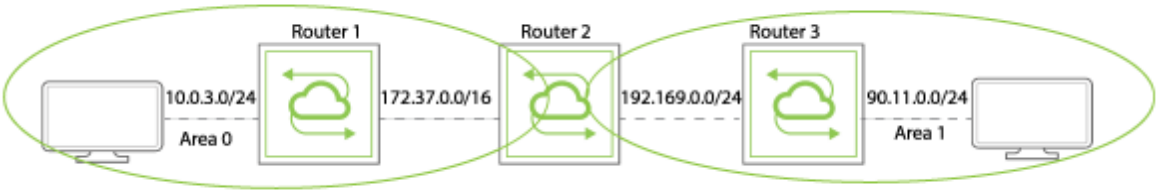


Figure 5-4 - OSPF

1) First, create the vRouter for Router1.

```
CLI (network-admin@switch) > vrouter-create name vrouter1 vnet vnet-name fabricname-global router-type hardware
```

2) Add vRouter interfaces to the vRouter:

```
CLI (network-admin@switch) > vrouter-interface-add vnet vnet-name vrouter-name vrouter1 ip 10.0.3.0 netmask 24 vlan 35 if data nic-enable


CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrouter1 ip 172.37.0.0 netmask 16 vlan 55 if data nic-enable
```

3) Add the subnets, 10.0.3.0/24 and 172.37.0.0/16, to VLAN35 with the area 0:

```
CLI (network-admin@switch) > vrouter-ospf-add vrouter-name vrouter1 network
10.0.3.0/24 ospf-area 0
```

4) Add the second IP address with the area 0.

```
CLI (network-admin@switch) > vrouter-ospf-add vrouter-name vrouter1 network
172.37.0.0/16 ospf-area 0
```

5) Add interfaces for OSPF hello intervals of 30 seconds:

```
CLI (network-admin@switch) > vrouter-interface-config-add name router1 nic
eth0.35 ospf-hello-interval 30 ospf-cost 0


CLI (network-admin@switch) > vrouter-interface-config-add name router1 nic
eth0.55 ospf-hello-interval 30 ospf-cost 0
```

If you specify 0 as the cost value, the vRouter calculates the OSPF cost automatically based on the bandwidth of the interface. When you modify the OSPF hello interval, the `ospf-dead-interval` is automatically reset to 4 times the hello interval.

6) Display the configuration by using the vrouter-ospf-show command:

```
CLI (network-admin@switch) > vrouter-ospf-show layout vertical

vrouter-name: vrouter1
network: 10.0.3.0
netmask: 24
ospf-area: 0
vrouter-name: vrouter1
network: 172.37.0.0
netmask: 16
ospf-area: 0
stub-area: 11
stub-type: stub
ospf-hello-interval: 30
metric: 34
```

The metric value can reflect the cost of routes advertised as OSPF routes. It may also reflect the cost of routes advertised with other protocols.

Displaying Default Timers for OSPF Configurations

NetVisor OS now allows you to display default timers for OSPF configurations. To add a timer or modify an existing timer, use the following commands:

```
CLI (network-admin@switch) > vrouter-interface-config-add
```

`ospf-retransmit-interval` *seconds*	Specify the OSPF retransmit interval from 3 to 65535 seconds. The default is 5 seconds. (IPv4

	orIPv6)

```
CLI (network-admin@switch) > vrouter-interface-config-modify
```

`ospf-retransmit-interval` *`seconds`*	Specify the OSPF retransmit interval from 3 to 65535 seconds. The default is 5 seconds. (IPv4 orIPv6)

A new command displays the OSPF configuration for an interface:

```
CLI (network-admin@switch) > vrouter-ospf-interface-show
```

`vrouter-name` *`name-string`*	Displays the name of the vRouter.
Specify any of the following optional OSPF parameters:	
`nic vrouter interface` *`nic`*	Displays the name of the vNIC.
`nic-state down|up`	Displays the NIC state.
`l3-port` *`l3-port-number`*	Displays the Layer 3 port numbers.
`ip` *`ip-address`*	Displays the IPv4 address of the interface.
`netmask` *`netmask`*	Displays the netmask of the IPv6 address.
`broadcast` *`ip-address`*	Displays the broadcast IP address.
`area ip-address`	Displays the area ID for the interface in IPv4 format.
`mtu` *`mtu-number`*	Displays MTU for the interface.
`mtu-mismatch-detection| no-mtu-mismatch-detection`	Displays if MTU mismatch detection is configured.
`router-id` *`ip-address`*	Displays the router ID as an IP address.
`network-type point-to-point| broadcast|loopback`	Displays the OSPF network type.
`state down|loopback|waiting|point-to-point|dr-other|backup|dr`	Displays OSPF interface state.
`dr-id` *`ip-address`*	Displays the designated router ID.
`dr-ip` *`ip-address`*	Displays the designated router IP address.
`bdr-id` *`ip-address`*	Displays the backup designated router ID.
`bdr-ip ip-address`	Displays the designated router IP address.
`priority` *`priority-number`*	Displays the priority.
`cost` *`cost-number`*	Displays the cost.
`hello` *`hello-number(s)`*	Displays the hello-interval in seconds.

`dead` *`dead-number(s)`*	Displays the dead time in seconds.
`retransmit` *`retransmit-number(s)`*	Displays the retransmit interval time in seconds.
`hello-due` *`hello-due-string`*	Displays the hello due in.
`neighbor` *`neighbor-number`*	Displays the neighbor count.
`adjacent` *`adjacent-number`*	Displays the adjacent number count.

```
CLI (network-admin@switch) > vrouter-ospf6-interface-show
```

`vrouter-name` *`name-string`*	Displays the name of the vRouter.
Specify any of the following optional OSPF parameters:	
`nic vrouter interface` *`nic`*	Displays the name of the vNIC.
`nic-state down|up`	Displays the vNIC state.
`l3-port` *`l3-port-number`*	Displays the Layer 3 port numbers.
`link-local` *`ip-address`*	Displays the IPv6 link-local IP address.
`ip6` *`ip-address`*	Displays the IPv6 address of the interface.
`netmask-ip6` *`netmask`*	Displays the netmask of the IP address.
`area ip-address`	Displays the area ID for the interface in IPv4 format.
`mtu` *`mtu-number`*	Displays MTU for the interface.
`mtu-mismatch-detection| no-mtu-mismatch-detection`	Displays if MTU mismatch detection is configured.
`state down|loopback|waiting|point-to-point|dr-other|backup|dr`	Displays OSPF interface state.
`dr-id` *`ip-address`*	Displays the designated router ID.
`bdr-id` *`ip-address`*	Displays the backup designated router ID.
`priority` *`priority-number`*	Displays the priority.
`cost` *`cost-number`*	Displays the cost.
`hello` *`hello-number(s)`*	Displays the hello-interval in seconds.
`dead` *`dead-number(s)`*	Displays the dead time in seconds.
`retransmit` *`retransmit-number(s)`*	Displays the retransmit interval time in seconds.
`if-scoped-lsa` *`if-scoped-lsa-number`*	Displays the number of interface LSAs scoped for the area.
`ls-update` *`ls-update-number`*	Displays the number of pending LSAs for LSUpdate.
`ls-ack` *`ls-ack-number`*	Displays the number of pending LSAs for LSAck.

Configuring Route Maps for OSPF Routes

Configure route maps to associate a redistributed metric or metric-type for OSPFv3. You can define a route map to prevent OSPF routes from getting added to the routing table. This filtering happens at the time when OSPF installs the route in the routing table.

Before you configure route maps, configure a list of prefixes using the following command:

```
CLI (network-admin@switch) > vrouter-prefix-list-add
```

Use the following set of commands to configure route maps for OSPF:

```
CLI (network-admin@switch) > vrouter-route-map-add
```

`vrouter-name` *`name-string`*	Specify the name of the vRouter service.
`name` *`name-string`*	Specify the name for the route map.
`seq` *`integer`*	Specify the sequence number as an integer between 1 and 4294967295.
`action permit|deny`	Specify the route map action as permit or deny.
`match-prefix vrouter prefix-list` *`name-string`*	Specify the name of the prefix list used for the route map.
`community-attribute unset|none|no-export|no-advertise| internet|local-AS`	Specify the community attribute for the route map.
`local-pref` *`integer`*	Specify the local preference as an integer between -1 and 4294967295.
`metric none`	Specify the metric as none.
`metric-type 1|2`	Specify the metric type as 1 or 2.

```
CLI (network-admin@switch) > vrouter-route-map-remove
```

`vrouter-name` *`name-string`*	Specify the name of the vRouter service.
`name` *`name-string`*	Specify the name for the route map.
`seq` *`integer`*	Specify the sequence number as an integer between 1 and 4294967295.

```
CLI (network-admin@switch) > vrouter-route-map-show
```

`vrouter-name` *`name-string`*	`Displays the name of the vRouter service.`
`name` *`name-string`*	`Displays the name for the route map.`

`seq` *`integer`*	`Displays the sequence number as an integer between 1 and 4294967295.`
`action permit|deny`	`Displays the route map action as permit or deny.`
`match-prefix vrouter prefix-list` *`name-string`*	`Displays the name of the prefix list used for the route map.`
`community-attribute unset|none|no-export|no-advertise| internet|local-AS`	`Displays the community attribute for the route map.`
`local-pref` *`integer`*	`Displays the local preference as an integer between -1 and 4294967295.`
`metric none`	`Displays the metric as none.`
`metric-type 1|2`	`Displays the metric type as 1 or 2.`

Enabling or Disabling OSPF SNMP MIBs

You now enable or disable SNMP MIBs for OSPF configurations by using the command, `vrouter-create`, or `vrouter-modify`:

```
CLI (network-admin@switch) > vrouter-create name-string ospf-snmp|no-ospf-snmp   ospf-snmp-notification|no-ospf-snmp-notification
```

Configuring Default Route Information Settings for OSPF Routing

An OSPF vRouter, by default, does not generate a default route into the OSPF domain. In order for OSPF to generate a default route, you must explicitly configure the vRouter with this information.

There are two ways to inject a default route into a normal area.

1) If the ASBR already has the default route in the routing table, you can advertise the existing 0.0.0.0/0 into the OSPF domain with the `ospf-default-information` parameter.

2) If the ASBR doesn't have a default route, you can add the keyword `always` to the `ospf-default-information` command. However, we recommend not to add the `always` keyword.

The OSPF vRouter advertises a default route into the OSPF domain, even if a route to 0.0.0.0 is configured.

When you configure the metric type, you can use the parameters described for the `vrouter-*` commands for configuring a route map. These parameters control default route generation for IPv4 and IPv6 default routes.

```
CLI (network-admin@switch) > vrouter-create
```

`ospf-default-information none| originate|always`	Specify if you want to use the default route information for OSPF:

	• `none` — no default route is generated. • `originate` — the default route is generated only if a default route is present in the routing table. • `always` — the default route is generated even if no default route is present in the routing table.
`ospf-default-info-originate-metric none`	Specify the metric for the default route.
`ospf-default-info-originate-metric-type 1|2`	Specify the metric type as 1 or 2.

`CLI (network-admin@switch) > vrouter-modify`

`ospf-default-information none|originate|always`	Specify if you want to use the default route information for OSPF: • `none` — no default route is generated. • `originate` — the default route is generated only if a default route is present in the routing table. • `always` — the default route is generated even if no default route is present in the routing table.
`ospf-default-info-originate-metric none`	Specify the metric for the default route.
`ospf-default-info-originate-metric-type 1|2`	Specify the metric type as 1 or 2.
`ospf-default-info-originate-route-map vrouter route-map` *`name`*	Specify the OSPF default information route map.

`CLI (network-admin@switch) > vrouter-show`

`ospf-default-information none|originate|always`	Displays the default route information for OSPF: • `none` — no default route is generated. • `originate` — the default route is generated

	only if a default route is present in the routing table. • `always` — the default route is generated even if no default route is present in the routing table.
`ospf-default-info-originate-metric none`	Displays the metric for the default route.
`ospf-default-info-originate-metric-type 1|2`	Displays the metric type as 1 or 2.
`ospf-default-info-originate-route-map vrouter route-map` *`name`*	Displays the OSPF default information route map.

Configuring Metric and Metric Type for Route Maps

Configure route maps to associate a redistribute metric and metric-type for OSPF and OSPFv3 and you cannot configure metrics using the current OSPF commands. Use route maps method to configure metric and metric-type on routers. Configuration of metrics and metric-type for OSPFv3 requires route maps.

```
CLI (network-admin@switch) > vrouter-route-map-add
```

`metric none`	Specify a metric for the route map.
`metric-type none|1|2`	Specify the metric type.

The new parameters also apply to the `vrouter-route-map-modify` and `vrouter-route-map-show` commands.

You can also specify metrics and route maps for vRouters configured as OSPF and BGP vRouters.

```
CLI (network-admin@switch) > vrouter-modify
```

`bgp-redist-static-route-map vrouter route-map` *`name`*	Specify the route map for BGP redistribution of static routes.
`bgp-redist-connected-route-map vrouter route-map name`	Specify the route map for BGP redistribution of connected routes.
`bgp-redist-ospf-route-map vrouter route-map` *`name`*	Specify the route map for BGP redistribution of OSPF routes.
`ospf-redist-static-route-map`	Specify the route map for OSPF redistribution of static routes.

`vrouter route-map` *`name`*	
`ospf-redist-connected-route-map vrouter route-map` *`name`*	Specify the route map for OSPF redistribution of connected routes.
`ospf-redist-bgp-route-map vrouter route-map` *`name`*	Specify the route map for OSPF redistribution of BGP routes.

Configuring BGP on a vRouter

Border Gateway Protocol (BGP) is a path-vector protocol and is the most commonly used routing protocol on the Internet. It advertises the paths required to reach a certain destination. BGP is also a protocol that sits on top of TCP, and is simpler than Open Shortest Path First (OSPF).
For example, in Figure 5-5, the network administrator wants to configure network traffic from the source host to reach the destination host. But when different VLANs are configured, the source host traffic is not aware of the route between the source host and the destination host. However, there is a VLAN that spans VLAN 33 and VLAN 55. You can solve this problem by configuring BGP in the same Autonomous System (AS) 100 that sends traffic over VLAN 35. This allows the source host to learn the route to the destination host.

Using a loopback address for peering is useful when there are multiple paths between the BGP peers which would otherwise tear down the BGP session if the physical interface used for establishing the session goes down. It also allows the vRouters running BGP with multiple links between vRouters to load balance over the available paths.

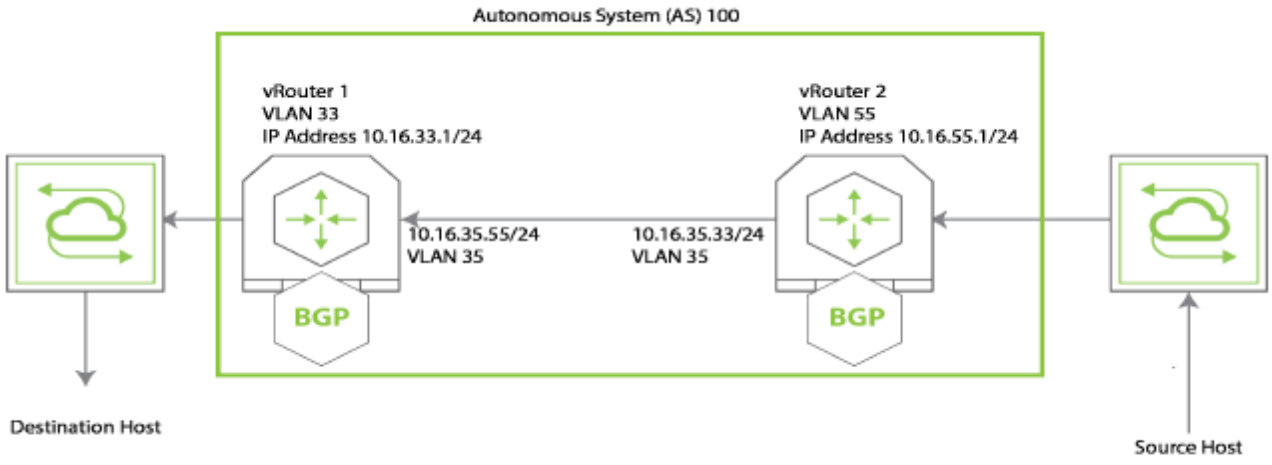


Figure 5-5: Configuring BGP for Two VLANs

This example assumes that you have two VLANs, VLAN33 and VLAN55. Also, that you have added ports to the configuration.

Begin by configuring vRouter1, a software vRouter, on VLAN 33 with the BGP information:

```
CLI (network-admin@switch) > vrouter-create name vrouter1  fabricname-
global router-type hardware bgp-as 100 bgp-redist-connected-metric none
```

Additional BGP parameters include the following:

- `bgp-redist-static-metric` — redistribute static BGP route metric number
- `bgp-redist-connected-metric` — redistribute connected BGP route metric
- `bgp-redist-rip-metric` — redistribute BGP into RIP process metric

- `bgp-redist-ospf-metric` — redistribute BGP into OSPF process metric
- `bgp-max-paths` — maximum number of BGP paths
- `bgp-ibgp-multipath` — allow the BGP vRouter to select multiple paths for load sharing.
- `bgp-bestpath-as-path` — allow BGP to use the best path for traffic forwarding.
- `bgp-dampening|no-bgp-dampening` — suppress flapping routes so they are not advertised.
- `bgp-stalepath-time` — how long a router waits before deleting stale routes after an end of record (EOR) message is received from the restarting router.
- `bgp-graceful-shutdown|no-bgp-graceful-shutdown`— how to configure BGP graceful shutdown (RFC8326)

Add the IP addresses and VLANs:

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrouter1 ip
10.16.35.33/24 vlan 35

CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrouter1 ip
10.16.33.1/24 vlan 33
```

Add the BGP information:

```
CLI (network-admin@switch) > vrouter-bgp-add vrouter-name vrouter1 neighbor
10.16.35.55 remote-as 100

CLI (network-admin@switch) > vrouter-bgp-network-add vrouter-name vrouter1
network 10.16.33.0/24
```

Display the interface information for vrouter1:

```
CLI (network-admin@switch) > vrouter-interface-show format all layout
vertical

vrouter-name: vrouter1
nic: eth1.33
ip: 10.9.100.100/16
assignment: static
mac: 66:0e:94:30:c6:92
vlan: 33
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: up
secondary-macs:
vrouter-name: vrouter1
nic: eth2.33
ip: 192.168.42.11/24
assignment: static
```

```
mac:  66:0e:94:30:25:5e
vlan:  33
vxlan:  0
if:  data
alias-on:
exclusive:  no
nic-config:  enable
nic-state:  up
secondary-macs:
```

If you want to filter IP hosts, you can add prefix lists to the BGP configuration. See Configuring Prefix Lists for BGP and OSPF.

Then, configure vRouter2 on VLAN 55:

```
CLI (network-admin@switch) > vrouter-create name vrouter2  fabricname-
global router-type hardware bgp-as 100 bgp-redist-connected-metric none
```

Add the IP addresses and VLANs:

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrouter2 ip
10.16.35.55/24 vlan 35

CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrouter2 ip
10.16.55.1/24 vlan 55
```

Then add the BGP information:

```
CLI (network-admin@switch) > vrouter-bgp-add vrouter-name vrouter2 neighbor
10.16.35.33 remote-as 100

CLI (network-admin@switch) > vrouter-bgp-network-add vrouter-name vrouter2
network 10.16.55.0/24
```

And finally, add the loopback address:

```
CLI (network-admin@switch) > vrouter-loopback-interface-add vrouter-name
vrouter1 index 5 ip 1.1.1.1
```

Display the vRouter BGP configuration:

```
CLI (network-admin@switch) > vrouter-bgp-show format all layout vertical

vrouter-name:  vrouter33
ip:  10.16.35.55
neighbor:  10.16.35.55
remote-as:  100
next-hop-self: no
route-reflector-client:  no
override-capability:  no
soft-reconfig-inbound:  no
```

```
max-prefix-warn-only:  no
vrouter-name:  vrouter33
ip:  10.16.33.0
network:  10.16.33.0/24
description:  none
vrouter-name:  vrouter55
ip:  10.16.35.33
neighbor:  10.16.35.33
remote-as:  100
next-hop-self:  no
route-reflector-client:  no
override-capability:  no
soft-reconfig-inbound:  no
max-prefix-warn-only:  no
vrouter-name:  vrouter55
ip:  10.16.55.0
network:  10.16.55.0/24
description:  none
```

To reset BGP neighbors, use the `vrouter-bgp-neighbor-reset` command.

To display BGP neighbors, use the `vrouter-bgp-neighbor-show` command.

```
CLI  (network-admin@switch)  >  vrouter-bgp-neighbor-show

vrouter-name:  vrouter1
neighbor:          10.9.100.201
ver:                4
remote-as:       100
msg_rcvd:        11
msg_sent:        19
tblver:           0
inQ:                0
outQ:              0
up/down:           00:54:04
state/pfxrcd:  Connect
remote-router:  vrouter2
description:    none
vrouter-name:  vrouter2
neighbor:          10.9.100.101
ver:                4
remote-as:       100
msg_rcvd:        12
msg_sent:        18
tblver:           0
inQ:                0
outQ:              0
up/down:           00:53:37
state/pfxrcd:  Connect
remote-router:  vrouter2
description:    none
```

Additional BGP Parameters

There are additional BGP parameters that you can use to optimize your BGP network. Add any of the following parameters:

- `ebgp-multihop` — a value for external BGP to accept or attempt BGP connections to external peers, not directly connected, on the network. This is a value between 1 and 255.
- `update-source vrouter` — the source IP address of BGP packets sent by the router. This parameter is required if you want BGP to perform peering over a loopback interface.
- `prefix-list-in` — specify a list of incoming prefixes for route redistribution.
- `prefix-list-out` — specify a list of outgoing prefixes for route redistribution.
- `override-capability` — override the result of capability negotiation with the local configuration. This parameter allows you to ignore a remote peer's capability value.
- `soft-reconfig-inbound` — defines the route refresh capability by allowing the local device to reset inbound routing tables dynamically by exchanging route refresh requests to supporting peers.
- `max-prefix` — allows you to specify the maximum number of IP prefixes to filter.
- `max-prefix-warn` — add a parameter to warn when the maximum number of prefixes is reached.

Enabling or Disabling BGP SNMP MIBs

You can enable or disable SNMP MIBs for BGP configurations by using the `vrouter-create` or `vrouter-modify` commands:

```
CLI (network-admin@switch) > vrouter-create name-string bgp-snmp|no-bgp-snmp bgp-snmp-notification|no-bgp-snmp-notification
```

Configuring AS and AS Prepending on BGP

You can prepend one or more autonomous system (AS) numbers at the beginning of an AS path. The AS numbers are added at the beginning of the path after the actual AS number from which the route originates has been added to the path. Prepending an AS path makes a shorter AS path look longer and therefore less preferable to BGP.

The BGP best path algorithm determines how the best path to an autonomous system (AS) is selected. The AS path length determines the best path when all of the following conditions are met:

There are multiple potential routes to an AS.

- BGP has the lowest preference value, sometimes referred to as the administrative distance, of the available routes.
- The local preferences of the available routes are equal.

When these conditions are met, the AS path length is used as the tie breaker in the best path algorithm. When two or more routes exist to reach a particular prefix, BGP prefers the route with the shortest AS Path length.

If you have multi-homing to one or more service providers, you may prefer for incoming traffic take a particular path to reach your network. Perhaps you have two connections, but one costs less than the other. Or you might have one fast connection and another, much slower connection that you only want to use as a backup if your primary connection is down. AS path prepending is an easy method that you can use to influence inbound routing to your AS.

NetVisor OS has new parameters for the `vrouter-route-map-*` commands:

```
CLI (network-admin@switch) > vrouter-route-map-add
```

`as-path-prepend` *`integer`*	Specify a value between 1 and 4,294,967,295.
`as-path-prepend-last-as` *`integer`*	Specify a value between 1 and 10.
`as-path-exclude` *`integer`*	Specify a value between 1 and 4,294,967,295.

```
CLI (network-admin@switch) > vrouter-route-map-show
```

`as-path-prepend` *`integer`*	Displays a value between 1 and 4,294,967,295.
`as-path-prepend-last-as` *`integer`*	Displays a value between 1 and 10.
`as-path-exclude` *`integer`*	Displays a value between 1 and 4,294,967,295.

Configuring Border Gateway Protocol (BGP) Communities

A BGP community is a group of prefixes that share some common property and can be configured with the BGP community attribute. The BGP Community attribute is an optional transitive attribute of variable length. The attribute consists of a set of four octet values that specify a community. The community attribute values are encoded with an Autonomous System (AS) number in the first two octets, with the remaining two octets defined by the AS. A prefix can have more than one community attribute. A BGP speaker that sees multiple community attributes in a prefix can act based on one, some or all the attributes. A router has the option to add or modify a community attribute before the router passes the attribute on to other peers.

The local preference attribute is an indication to the AS which path is preferred in order to reach a certain network. When there are multiple paths to the same destination, the path with the higher preference is preferred (the default value of the local preference attribute is 100).

Common Community Attributes

- **Standard** (well known) — These community attributes are 4 octets long, with well known values
  - Internet (0) — advertise these routes to all neighbors.
  - no-export (0xFFFFFF01) — do not advertise to outside a BGP confederation boundary.
  - no-advertise (0xFFFFFF02) — do not advertise to other BGP peers .
  - local-AS (0xFFFFFF03) — do not advertise to external BGP peers.

- **Standard** - generic (AS:value) — These community attributes are also 4 octet long, but values can be really generic. The first 16-bit number is normally the AS number of the network that sets the community or looks for it, and the second number is one that conveys the intended information, for example: 65001:100.

For example to set the community attribute, **no-export**, to all route prefixes matching prefix **subnet100**, use the following syntax:

```
CLI (network-admin@switch) > vrouter-route-map-add vrouter-name vr1 name
rmap1 seq 10 action permit match-prefix subnet100 community-attribute no-
export
```

To set the community attribute, **65002:200** to all route prefixes matching prefix subnet100, use the following syntax:

```
CLI (network-admin@switch) > vrouter-route-map-add vrouter-name vr1 name
peer vr2 action permit seq 20 match-prefix subnet99 community-attribute-
generic 65002:200
```

Community Lists

BGP community list is a user defined BGP communities attribute list. The BGP community list can be used for matching or manipulating BGP communities attribute in updates. This is used on the receive side of the BGP updates to match what is set in the received updates. Community lists can be used in route-map with match-community keyword to apply any policy on the receive side.

- **Standard** — Standard community list defines attribute which matches standard communities as defined above (well known or generic).

To set the community list permitting the community value **300** for AS **65002**, use the following syntax:

```
CLI (network-admin@switch) > vrouter-community-list-add vrouter-name vr2
style standard name clist300 action permit community-attribute 65002:300
```

The NetVisor OS commands for `vrouter-route-maps-*` support additional parameters for BGP communities:

```
CLI (network-admin@switch) > vrouter-route-map-add
```

`match-community` *`match-community-string`*	Specify the community string to match. (BGP only)
`exact-match|no-exact-match`	Specify if the community string is an exact match or not. (BGP only)
`community-attribute-generic` *`community-attribute-generic-string`*	Specify a generic community attribute such as AA:NN. (BGP only)
`additive|no-additive`	Specify if a given community is appended to existing communities value.

`comm-list-del vrouter` *`community-list name`*	Specify if you want to remove community values from BGP community attributes.

New commands support creating BGP Communities:

```
CLI (network-admin@switch) > vrouter-community-list-add
```

`vrouter-name` *`name-string`*	Specify a vRouter to add the community list.
Add the following community list parameters:	
`style standard`	Specify the style of the community list.
`name` *`name-string`*	Specify a name for the community list.
`action permit|deny`	Specify the action for the community list.
`community-attribute` *`community-attribute-string`*	Specify the community attribute.

```
CLI (network-admin@switch) > vrouter-community-list-remove
```

`vrouter-name` *`name-string`*	Specify a vRouter to remove the community list.
Add the following community list parameters:	
`style standard`	Specify the style of the community list.
`name` *`name-string`*	Specify a name for the community list.
`action permit|deny`	Specify the action for the community list.
`community-attribute` *`community-attribute-string`*	Specify the community attribute.

```
CLI (network-admin@switch) > vrouter-community-list-show
```

`vrouter-name` *`name-string`*	Displays the vRouter name.
Add the following community list parameters:	
`style standard`	Displays the style of the community list.
`name` *`name-string`*	Displays a name for the community list.

`action permit|deny`	Displays the action for the community list.
`community-attribute` *`community-attribute-string`*	Displays the community attribute.

Configuring BGP Route Maps for Origin

Use the `vrouter-route-map` commands to define the conditions for redistributing routes from one routing protocol to another. The `route-map` command contains a list of match and set commands associated with it.

The `match` commands specify the match criteria: the conditions under which NetVisor OS redistributes the current route-map command.

The set commands specify the set actions: the particular redistribution actions to perform if the criteria enforced by the match commands are met.

```
CLI (network-admin@switch) > vrouter-route-map-modify vrouter-name vr2 name
TEST6 seq 10 origin <tab>
```

`none`	Resets the origin to none.
`egp`	Specifies the route as a remote EGP route.
`igp`	Specifies the route as a local IGP route.
`incomplete`	Route is of unknown heritage.

Configuring BGP Graceful Shutdown

The BGP Graceful Shutdown functionality enables the network administrators to gracefully shutdown all BGP sessions per vrouter between BGP peer(s). This feature does not shutdown the sessions, but notifies the neighbor peers that a maintenance event is occurring and that the traffic should be re-routed to another peer to avoid traffic black-holing. The *graceful*, *no traffic impact* behaviour of the feature depends on the topology and BGP setup in a network.

When a BGP speaker is rebooted, upgraded, or shut down during a maintenance window, the admin re-sends all the BGP routes (route refresh) to the neighbor peers by attaching a BGP GRACEFUL_SHUTDOWN community to each prefix. Upon receiving the prefixes, the BGP neighbors should dynamically change the local preference associated to these subnets to a very low value, typically, LP=0. By changing the local preference to 0, the BGP neighbors select another path (i.e. next-hop) to reach the prefixes carrying the GRACEFUL_SHUTDOWN community attribute. When the traffic is completely re-routed to alternate path, the BGP speaker can administratively shutdown all BGP sessions without experiencing traffic black-holing issue.

You must configure the vrouter and BGP neighbors prior to enabling the bgp-graceful-shutdown knob on the vrouter.

Note: This feature do not support a per-session graceful shutdown, but supports a per-vRouter graceful

shutdown (all BGP sessions of a particular vRouter are affected).

To configure BGP graceful shutdown, use the following commands (this command enables you to modify several parameters of BGP and other protocols.):

```
CLI (network-admin@switch) >  vrouter-modify name name-string bgp-graceful-shutdown
```

`vrouter-create`	Allows you to create a hardware vRouter and specify the parameters for the vRouter.
`name name-string`	Specify a name for the service configuration.
`bgp-graceful-shutdown|no-bgp-graceful-shutdown`	Specify to enable or disable BGP graceful shutdown (RFC8326).

Below is an example of a vRouter configuration enabling BGP graceful shutdown on vr2:

```
CLI (network-admin@switch) >  vrouter-modify name vr2 router-type hardware bgp-as 65200 router-id 22.22.22.22  bgp-graceful-shutdown
```

To display the details, use the `vrouter-show` command (similar to other BGP parameters the new field is displayed only if the field is listed in the format statement or `format all` keyword is used) :

```
CLI (network-admin@switch) >  vrouter-show format name,scope,router-type,bgp-as,router-id,bgp-graceful-shutdown

name scope  router-type bgp-as router-id   bgp-graceful-shutdown
---- ------ ----------- ------ ----------- --------------------
vr1  fabric  hardware   65100  11.11.11.11    off
vr2  fabric  hardware   65200  22.22.22.22    on
vr3  local   hardware   65300  33.33.33.33    off
```

To disable BGP graceful shutdown knob, the configuration, use the command:

```
CLI (network-admin@switch) >  vrouter-modify name name-string no-bgp-graceful-shutdown
```

Configuring BGP Unnumbered

The Border Gateway Protocol (BGP) is a path vector protocol used to exchange routing and reachability information. Traditionally, to exchange IPv4 prefixes, you must configure explicit BGP sessions with the neighbor IP address and remote-AS information for each BGP peer, which can become cumbersome in large networks. This also occupies significant address space as each BGP peer must have an IPv4 address.

By using RFC5549 (Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop), NetVisor OS enables you to create BGP unnumbered sessions which are simpler to configure. Practically, you need not have an IPv4 address for every BGP-enabled interface and therefore it is called an unnumbered configuration. Instead of specifying an IPv4 neighbor address and the remote-AS number, you can provide the L3 port number and declare an eBGP session . BGP unnumbered uses the link-local IPv6

address as the next-hop IP address for both IPv4 and IPv6 prefixes.

In effect, the need for configuring IPv4 addresses on all BGP-enabled interfaces and the need for declaring an explicit ASN for the remote side is eliminated through BGP unnumbered.

Note: NetVisor OS does not support iBGP for unnumbered BGP sessions.

Note: Prior to NetVisor OS version 6.1.1 HF1 on Whitebox platforms, in cases where the next hop BGP neighbors on unnumbered interfaces having NetVisor OS releases with different FRR versions (for example, NetVisor OS version 5.2.x with FRR4 and version 6.1.1 with FRR 7.2), the route updates from FRR 7.2 gets dropped by FRR 4 based releases. See the configuration section below for additional details.

To configure a BGP unnumbered session, use the two following commands:

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name <name-
string> l3-port <l3-port-usable-port-number> ipv6-unnumbered
```

`vrouter-interface-add`	Use this command to add an interface to a vRouter.
`vrouter-name` *`name-string`*	Specify the name of the vRouter to which the interface is to be added.
`vlan` *`<0..4095>`*	Specify the VLAN ID to which the interface has to be added (optional). **Note:** This option is not supported while configuring BGP unnumbered.
`Specify any of the following options:`	
`ip ip-address`	Specify the IP address to be assigned to the interface.
`Netmask` *`netmask`*	Specify the network mask for the IP address.
`Assignment [none | dhcp | dhcpv6]`	Specify one among the options as the type of IP address assignment.
`ip2` *`ip-address`*	Specify the second IP address of the interface.
`netmask2` *`netmask`*	Specify the network mask for the second IP address.
`assignment2 [none | dhcp | dhcpv6]`	Specify the type of assignment for the second IP address.
`linklocal` *`ip-address`*	Specify the IPv6 link local address.
`ipv6-unnumbered|no-ipv6-unnumbered`	Specify either of the options to enable or disable IPv6 unnumbered on the interface.
`vnet` *`vnet name`*	Specify the VNET name assigned to the vRouter.

`bd bridge-domain name`	Specify the Interface bridge domain name.
`vlan-type [public | private]`	Specify the interface VLAN type.
`if [mgmt | data | span | span2 | span3]`	Specify one among the options as the interface type.
`alias-on alias-on-string`	Specify an alias for the interface.
`exclusive | no-exclusive`	Specify if the interface is exclusive to the configuration. Exclusive means that other configurations cannot use the interface. Exclusive is specified when you configure the interface as a span interface and allows higher throughput through the interface.
`nic-enable | nic-disable`	Specify one of the options to enable or disable the NIC.
`pim | no-pim`	Specify either of the options to state if the VNIC is a Protocol Independent Multicast (PIM) interface or not.
`pim-dr-priority <1..4294967295>`	Specify the designated router priority for the PIM interface. NetVisor selects the vRouter interface with higher DR priority as the designated router.
`pim-cluster | no-pim-cluster`	Specify if the VNIC is on exclusive transit VLAN.
`fabric-nic | no-fabric-nic`	Specify if the VNIC is on a VLAN used for fabric setup or not.
`vrrp-id <0..255>`	Specify the VRRP ID.
`vrrp-primary vrrp-primary-string`	Specify the primary interface for VRRP failover.
`vrrp-priority <0..254>`	Specify the VRRP priority for the interface. This is a value between 0 (lowest) and 254 (highest).
`vrrp-adv-int <300..40950>`	Specify the VRRP advertisement interval in ms. The range is from 30 to 40950 with a default value of 1000.
`l3-port l3port-usable-port name`	Specify the name of the Layer 3 port to be added to the vrouter.
`secondary-macs secondary-macs-string`	Specify a secondary MAC address.
`mtu <68..9398>`	Specify a MTU value in bytes between 68 and 9216.
`if-nat-realm [internal | external]`	Specify the NAT interface realm as internal or external.
`priority-tag | no-priority-tag`	Specify either of the options to add a VLAN 0 priority tag on forwarded traffic or remove it.

Now, issue the second command to add a BGP neighbor to the vrouter and finish the BGP unnumbered

configuration.

```
CLI (network-admin@switch) > vrouter-bgp-add vrouter-name <name-string> l3-port <l3port-usable-port name> [remote-as external]
```

`vrouter-bgp-add`	Use this command to add a BGP neighbor to a vrouter.
`vrouter-name` *`name-string`*	Specify the name of the vrouter to which the neighbor is being added.
Specify one of the two options:	
`neighbor` *`ip-address`*	Specify the IP address for BGP neighbor.
`l3-port` *`l3port-usable-port name`*	Specify the L3 port name for the BGP session.
Specify any of the following options:	
`remote-as <external|internal|-2..429496729>`	Supply this parameter to specify if the BGP session is external/internal or to provide an AS number **Note:** Specify `external` to configure a BGP unnumbered session. This parameter is redundant as an eBGP session is assumed for unnumbered BGP sessions.
`next-hop-self | no-next-hop-self`	Specify either of the options to set the next hop as self or to remove that configuration.
`password` *`password-string`*	Provide a password for MD5 BGP.
`ebgp-multihop <1..255>`	Specify the value for external BGP between 1 and 255.
`update-source` *`vrouter loopback-interface ip`*	Specify the source IP address of BGP packets sent by the router. This parameter is required if you want BGP to perform peering over a loopback interface.
`update-source-interface` *`update-source-interface-string`*	Specify the interface of BGP packets sent by the router. This parameter is required if you want BGP to perform peering over any interface.
`prefix-list-in` *`vrouter prefix-list name`*	Specify the prefix list to filter inbound packets.
`prefix-list-out` *`vrouter prefix-list name`*	Specify the prefix list to filter outbound packets.
`route-reflector-client|no-route-reflector-client`	Specify if a route reflector client is used.
`override-capability|no-override-capability`	Specify either of the options to enable or disable override capability.
`soft-reconfig-inbound|no-soft-`	Specify either of the options to enable or disable

`reconfig-inbound`	soft reset to reconfigure inbound traffic.
`max-prefix` *`max-prefix-number`*	Specify the maximum number of prefixes.
`max-prefix-warn-only|no-max-prefix-warn-only`	Specify either of the options to enable or disable the warning if the maximum number of prefixes is exceeded.
`bfd|no-bfd`	Specify either of the options to enable or disable BFD protocol support for fault detection.
`bfd-multihop|no-bfd-multihop`	Specify either of the options to enable or disable the use of BFD multi-hop port for fault detection.
`multi-protocol [ipv4-unicast|ipv6-unicast]`	Specify either of the options as a multi-protocol feature.
`weight [none | -1..65535]`	Specify the default weight value for the neighbor's routes either as `none` or as a value between 0 and 65535.
`default-originate|no-default-originate`	Specify either of the options to enable or disable the announcing of default routes to the neighbor.
`neighbor-keepalive-interval <0..65535>`	Specify the BGP keepalive interval in seconds. The keepalive interval stipulates how often the the keepalive messages are sent.
`neighbor-holdtime <0..65535>`	Specify the BGP holdtime in seconds. The hold time specifies how long a router will wait for incoming BGP messages before it assumes the neighbor is dead.
`connect-retry-interval <0..65535>`	Specify the BGP connect retry interval in seconds.
`send-community|no-send-community`	Specify either of the options to enable or disable the sending of any community attribute to the neighbor.
`route-map-in vrouter route-map name`	Specify the name of the route map for incoming routes.
`route-map-out vrouter route-map name`	Specify the name of the route map for outgoing routes.
`allowas-in|no-allowas-in`	Specify either of the options to allow/reject routes with local AS in AS_PATH.
`interface vrouter interface nic`	Specify the Interface to reach the neighbor.
`advertisement-interval <0..600>`	Specify the minimum interval between sending BGP routing updates.
`description` *`description-string`*	Add a vRouter BGP neighbor description.

These commands create an interface with a link-local IPv6 address and configures an eBGP session for the interface.

To view the configuration on the vrouter, use the command, `vrouter-interface-showcommand.` For example:

```
CLI (network-admin@switch) > vrouter-interface-show format nic,l3-port,vlan,ip,ipv6-unnumbered

vrouter-name nic       l3-port vlan ip                          ipv6-unnumbered
------------ --------- ------- ---- --------------------------- --------------
vr1          eth0.4092 17      4092 fe80::640e:94ff:feff:7bdc/64 yes
```

View the BGP configuration by using the command:

```
CLI (network-admin@switch) > vrouter-bgp-show format l3-port,nic,neighbor,remote-as

vrouter-name l3-port nic       remote-as
------------ ------- --------- ---------
vr1           17      eth0.4092 external
```

Use the following command to view the relevant neighbor information:

```
CLI (network-admin@switch) > vrouter-bgp-neighbor-show vrouter-name vr2 format neighbor,l3-
port,nic,remote-as,up/down

vrouter-name neighbor       l3-port nic        remote-as up/down
------------ ------------ ------- --------- --------- --------
vr2                       17      eth2.4092 65100     00:06:20
vr2          192.168.101.1                  65100     00:02:48
```

The output shows that the neighbor field for the unnumbered interface does not have an IP address, while the l3-port and nic fields have information. Note that a BGP unnumbered session can co-exist with a numbered BGP session, but not on the same interface.

The Routing Information Database (RIB) and Forwarding Information Database (FIB) confirms that both IPv4 and IPv6 prefixes learned by BGP use IPv6 next-hop addresses:

```
CLI (network-admin@switch) > vrouter-rib-routes-show format ip,prelen,nexthop,flags,vlan

ip            prelen nexthop                  flags vlan
------------- ------ ---------------------- ----- ----
192.168.111.0 24     fe80::640e:94ff:feff:7bdc in-hw 4092
5001:11:1::   48     fe80::640e:94ff:feff:7bdc in-hw 4092

CLI (network-admin@switch) > vrouter-fib-routes-show format ip,prelen,vlan,port,nexthop-
mac,egress-id

ip            prelen vlan port nexthop-mac       egress-id
------------- ------ ---- ---- ----------------- ---------
192.168.111.0 24     4092 17   66:0e:94:ff:7b:dc 100010
5001:11:1::   48     4092 17   66:0e:94:ff:7b:dc 100010
```

You can modify unnumbered BGP neighbor parameters by using the command `vrouter-bgp-modify.` For example:

```
CLI (network-admin@switch) >  vrouter-bgp-modify vrouter-name vr3 l3-port
19 multi-protocol ipv6-unicast
```

You can remove an L3 port from a BGP configuration by using the command vrouter-bgp-remove. For example:

```
CLI (network-admin@switch) > vrouter-bgp-remove vrouter-name vr3 l3-port 17
```

To remove an L3 interface from a vrouter, use the command `vrouter-interface-remove.` For example:

```
CLI (network-admin@switch) > vrouter-interface-remove vrouter-name vr3 l3-
port 20
```

To reset the Information regarding BGP neighbors, use the command vrouter-bgp-neighbor-reset. For example:

```
CLI (network-admin@switch) > vrouter-bgp-neighbor-reset vrouter-name vr3
l3-port port 11
```

You can shut down a BGP neighbor by using the command vrouter-bgp-neighbor-shutdown. For example:

```
CLI (network-admin@switch) > vrouter-bgp-neighbor-shutdown vrouter-name vr3
l3-port port 26
```

To bring a BGP neighbor back up after the shut down, use the command vrouter-bgp-neighbor-no-shutdown. For example:

```
CLI (network-admin@switch) > vrouter-bgp-neighbor-no-shutdown vrouter-name
vr3 l3-port 10
```

As mentioned in the beginning of this topic, prior to NetVisor OS version 6.1.1 HF1, in cases where the next hop BGP neighbors on unnumbered interfaces having NetVisor OS releases with different FRR versions (for example, NetVisor OS version 5.2.x with FRR4 and version 6.1.1 with FRR 7.2), the route updates from FRR 7.2 gets dropped by FRR 4 based releases.

For releases between versions 6.0.x and 6.1.1, the unnumbered neighbor interoperability with 5.2.x is not feasible. This issue is addressed in version 6.1.1 HF1 and beyond with the introduction of a new BGP config option, `unnumbered-interop` on a per unnumbered neighbor basis. For example, use the `unnumbered-interop` parameter in the `vrouter-bgp-add` or `vrouter-bgp-modify` commands as:

```
CLI(network-admin@switch)> vrouter-bgp-modify vrouter-name <name-string>
l3-port <l3port-usable-port name> unnumbered-interop
OR
CLI(network-admin@switch)> vrouter-bgp-add vrouter-name <name-string> l3-
port <l3port-usable-port name> unnumbered-interop
```

The `no-unnumbered-interop` option restores the configuration to default values. In the newer releases when configured with the new knob, `unnumbered-interop`, Netvsior sends both *link-local* and *global next-hop* in the BGP Update. This option is specific to interop with unnumbered neighbors.

Below is an output of a sample configuration:

```
CLI(network-admin@switch)> vrouter-bgp-show format l3-port,neighbor,unnum-
interop,
```

```
vrouter-name         l3-port              unnum-interop
-------------------  -------------------  -------------
antlia-dc-1-vrouter  antlia-dc-1-eq-dc-1  no
antlia-dc-1-vrouter 53                    yes
```

Configuring BGP ASN Migration Mechanisms

NetVisor OS release 6.1.0 supports BGP mechanisms for Autonomous System Number (ASN) migration as specified in RFC7705. These mechanisms are leveraged in the scenario of an Internet Service Provider (ISP) merger, acquisition, or divestiture to ensure that internal and external BGP speakers are migrated seamlessly from one ASN to another.

Consider the use case where there are two ISPs, ISP M (AS 65300) and ISP N (AS 65400) directly attached to customer X (AS 65200) and customer Y (AS 65100) respectively. In a scenario where ISP M merges the ASNs of both ISP M and ISP N, AS 65300 becomes the permanently retained ASN used across the consolidated set of both ISPs' network equipment, while AS 65400 is retired. After the ASN migration, there will be only ISP M and all internal BGP speakers are configured to use AS 65300. This is illustrated by the figures below.

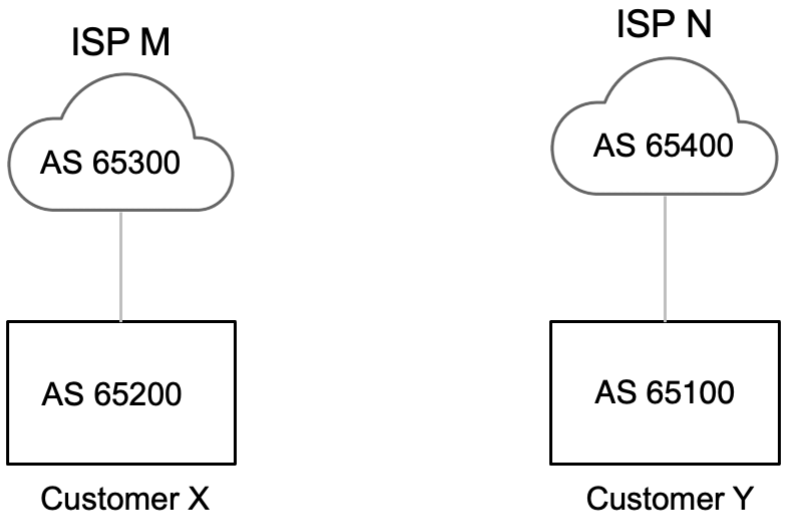


Figure 5-6: Before Migration

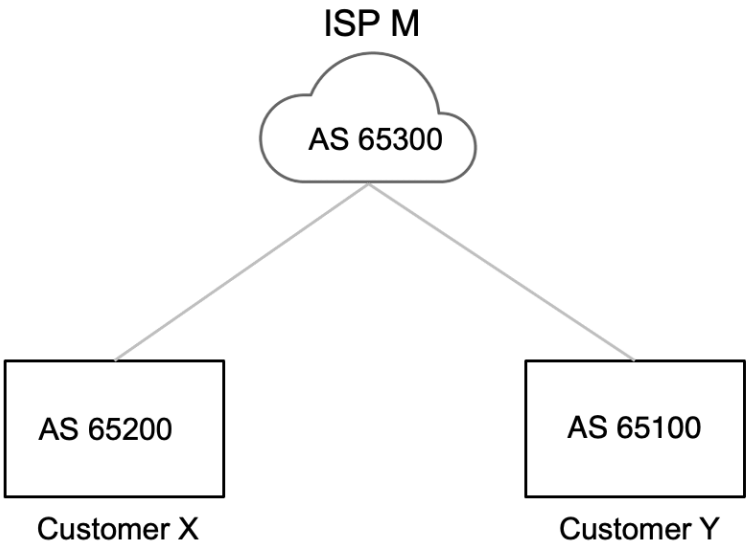


Figure 5-7: After Migration

During the migration, ISP N changes the global BGP ASN used by a Provider Edge (PE) router, from ASN 65400 to 65300. Immediately after this change, the router is no longer able to establish External BGP (eBGP) sessions toward the existing Customer Edge (CE) routers that are connected to it and still using AS 65400. Here, we make use of the migration mechanisms to enable the router to establish BGP neighbors using the legacy ASN and to modify the AS_PATH advertisement when transmitted toward CE devices to achieve the desired effect of not increasing the length of the AS_PATH.

Configuring "Local AS" Mechanism

The "Local AS" mechanism allows the PE router undergoing the ASN migration to establish eBGP sessions with existing CE devices that are using old ASN. This result is achieved by superceding the globally configured ASN with a locally defined ASN for a BGP neighbor or a group of neighbors. When this feature is used, the local router prepends the old or local ASN to the AS_PATH while installing or advertising routes received from a CE to iBGP neighbors inside the Autonomous System.

In the example illustrated below, when Local AS is configured on PE-Y, CE-X sees an AS_PATH of 65300 65400 65100, with an increased (and hence not desirable) AS_PATH length. The "No Prepend Inbound" mechanism described below solves this issue.

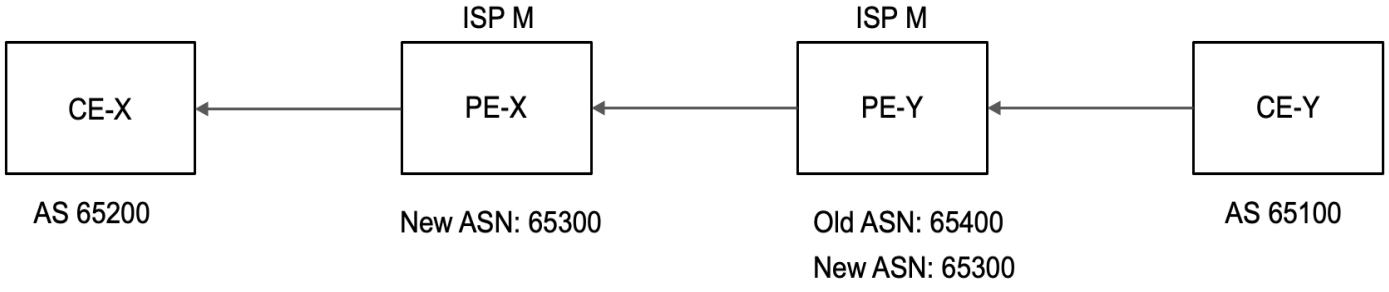


Figure 5-8: Local AS and No Prepend BGP UPDATE Diagram

To configure Local AS mechanism on the PE router which migrates to the new ASN, use the command:

```
CLI (network-admin@switch) > vrouter-bgp-modify vrouter-name vr1 neighbor-
ip 192.168.10.1 local-as 65400
```

Configuring the "No Prepend Inbound (of Local AS)" Mechanism

The "No Prepend Inbound " mechanism is used in conjunction with the Local AS mechanism. When `no-prepend` option is configured, the local BGP routers do not prepend the old or local ASN value to the AS_PATH while installing or advertising routes received from the CE. As a result, for the illustrated case above, CE-X sees an AS_PATH of 65100 65300, with a reduced AS_PATH length.

In this case, the `no-prepend` option has to be configured in the inbound direction on PE-Y, that is, in the direction of reception of routes. Use the command below to enable this option.

```
CLI (network-admin@switch) > vrouter-bgp-modify vrouter-name vr1 neighbor-
ip 192.168.10.1 local-as 65400 no-prepend
```

If you do not configure the `no-prepend` option , PE-X may drop the route it receives from PE-Y as the presence of the old ASN in the AS_PATH is perceived as a routing loop.

Table 5-1 - No Prepend Inbound Configuration

Configuration	AS_PATH as seen by CE-X
Without `no-prepend`	65100 65400 65300
With `no-prepend`	65100 65300

Configuring the "Replace Old AS" Mechanism

The "Local AS" and "No Prepend Inbound" configurations do not modify the AS_PATH attribute for BGP UPDATEs that are transmitted by the ISP's PEs to CE devices in the outbound direction. The "Replace Old AS" capability allows ISP M to prevent routers from appending the global or new ASN in outbound BGP UPDATEs toward customer networks that are using the "Local AS" mechanism. Instead, only the old or local AS is prepended in outbound BGP UPDATEs.

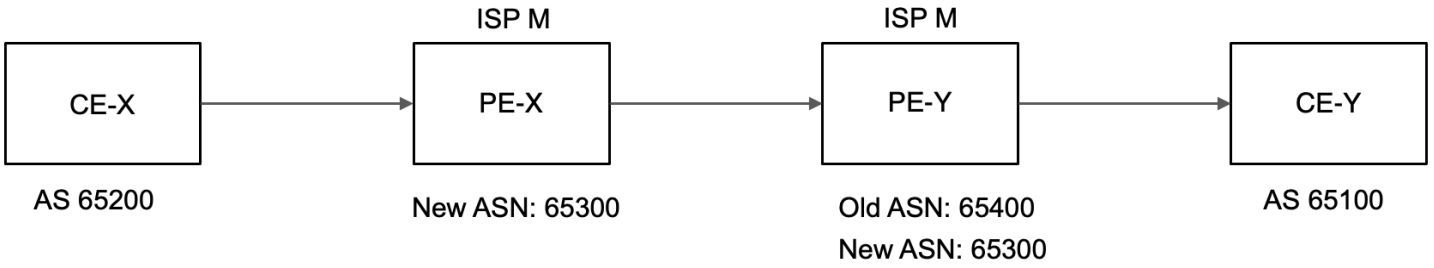


Figure 5-9: Replace AS BGP UPDATE Diagram

For example, without the use of "Replace Old AS", CE-Y would see an AS_PATH of 65400 65300 65200, with an unacceptable increase in AS_PATH length. After you configure PE-Y to use "Replace Old AS", CE-Y receives an AS_PATH of 65400 65200, which is the same AS_PATH length prior to AS migration. Therefore, the Replace AS configuration helps retain the same AS_PATH length before and after ISP migration.

To configure this mechanism, use the `replace-as` option as in this example:

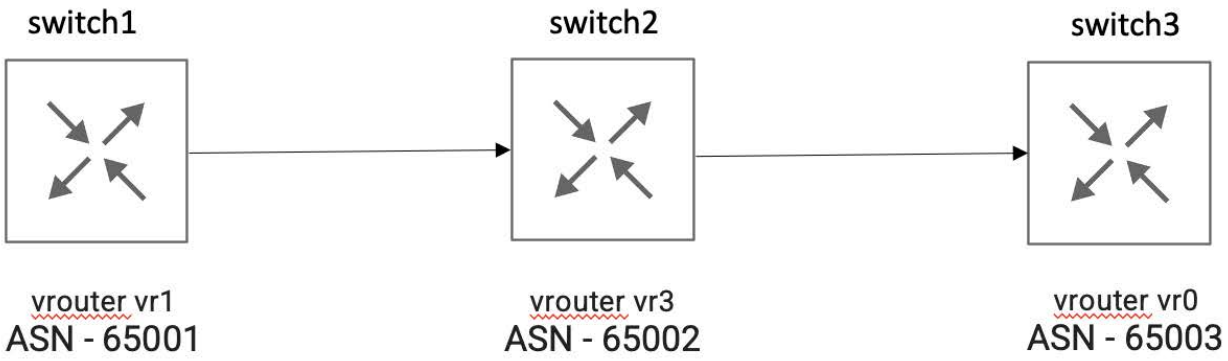
```
CLI (network-admin@switch) > vrouter-bgp-modify vrouter-name vr1 neighbor-
ip 192.168.10.1 local-as 65400 no-prepend replace-as
```

Table 5-2 - Replace Old AS Configuration

Configuration	AS_PATH as seen by CE-Y
Without `replace-as`	65400 65300 65200
With `replace-as`	65400 65200

Configuring BGP Route Summarization

BGP Route Aggregation also known as BGP Route Summarization is a method used to minimize the size of the routing table as well as increase network stability.



switch1 - advertises prefixes 10.0.54.0/24, 10.0.55.0/24, 10.0.56.0/24, 10.0.57.0/24, 10.0.58.0/24 and 10.0.59.0/24.

switch2 - advertises the received and/or summary address to switch3 depends on the configuration.

switch3 - Receives the summary and/or more specific prefixes from switch2.

Figure 5-10: Three node topology for BGP Route Aggregation

Configuring BGP Route Summarization using aggregate-address commands:

To add a BGP aggregate address to a vRouter, use the following command:

```
CLI (network-admin@switch) > vrouter-bgp-aggregate-address-add vrouter-
name <NAME> network <A.B.C.D> netmask <M> [vrf <NAME>][as-set <true|false>]
[summary-only <true|false>][matching-MED-only <true|false>][origin <egp|
igp|incomplete>][route-map <NAME>][suppress-map <NAME>]
```

`vrouter-name` *`name-string`*	Specify the name of the service configuration.

Specify the following `bgp-aggregate-address` arguments:	
`network` *`ip-address`*	Specify an IP address for BGP aggregate address.
`netmask` *`netmask`*	Specify the netmask for BGP aggregate address.
Specify one or more of the following options:	
`[ vrf vrouter vrf vrf | default ]`	Specify the name of the VRF.
`[ as-set true|false ]`	Specifies to generate AS set path information.
`[ summary-only true|false ]`	Specifies to filter more specific routes from updates.
`[ matching-MED-only true|false ]`	Specifies the only aggregate routes with matching MED.
`[ origin none|egp|igp|incomplete ]`	Specifies the origin code of BGP prefix.
`[ route-map` *`vrouter route-map name ]`*	Specifies the route map to aggregate network.
`[ suppress-map` *`vrouter route-map name ]`*	Specifies to suppress the selected more specific routes.

To modify a BGP aggregate address on a vRouter, use the following command:

```
CLI (network-admin@switch) > vrouter-bgp-aggregate-address-modify vrouter-
name <NAME> network <A.B.C.D> netmask <M>
```

To remove a BGP aggregate address from a vRouter, use the following command:

```
CLI (network-admin@switch) > vrouter-bgp-aggregate-address-remove vrouter-
name <NAME> network <A.B.C.D> netmask <M>
```

To display a BGP aggregate address on a vRouter, use the following command:

```
CLI (network-admin@switch) > vrouter-bgp-aggregate-address-show
```

Sample Configuration

Below is a sample configuration for BGP Route Summarization using the aggregate commands described earlier in this section.

```
i. Prefixes received from switch1 without route summarization:

 CLI (network-admin@switch3*) > vrouter-fib-routes-show intf-id 5
vrid ip        prelen intf-id bd vlan port nexthop-mac       flags egress-id
---- --------- ------ ------- -- ---- ---- ----------------- ----- ---------
0    10.0.54.0 24     5          62   19   66:0e:94:90:c3:bb       100023
0    10.0.55.0 24     5          62   19   66:0e:94:90:c3:bb       100023
0    10.0.56.0 24     5          62   19   66:0e:94:90:c3:bb       100023
0    10.0.57.0 24     5          62   19   66:0e:94:90:c3:bb       100023
0    10.0.58.0 24     5          62   19   66:0e:94:90:c3:bb       100023
```

```
 0    10.0.59.0 24     5          62   19   66:0e:94:90:c3:bb      100023
```

ii. vrouter-bgp-aggregate-address-add vrouter-name <NAME> network <A.B.C.D> netmask <M>

```
 CLI (network-admin@switch2*) > vrouter-bgp-aggregate-address-add vrouter-
name vr3 network 10.0.48.0 netmask 21
 CLI (network-admin@switch2*) > vrouter-bgp-aggregate-address-add vrouter-
name vr3 network 10.0.56.0 netmask 21

 CLI (network-admin@switch2*) > vrouter-bgp-aggregate-address-show
 vrouter-name network      vrf     as-set summary-only matching-MED-only origin route-map
suppress-map
 ------------ ------------ ------- ------ ------------ ---------------- ------ ---------
------------
 vr3          10.0.48.0/21 default false  true         false            none
 vr3          10.0.56.0/21 default false  true         false            none

 CLI (network-admin@switch3*) > vrouter-fib-routes-show intf-id 5
 vrid ip        prelen intf-id bd vlan port nexthop-mac       flags egress-id
 ---- --------- ------ ------- -- ---- ---- ----------------- ----- ---------
 0    10.0.48.0 21     5          62   273  66:0e:94:90:68:9d Trunk 100020
 0    10.0.56.0 21     5          62   273  66:0e:94:90:68:9d Trunk 100020
```

iii. vrouter-bgp-aggregate-address-add vrouter-name <NAME> network <A.B.C.D> netmask <M> as-set true

```
 CLI (network-admin@switch2*) > vrouter-bgp-aggregate-address-add vrouter-
name vr3 network 10.0.48.0 netmask 21
 CLI (network-admin@switch2*) > vrouter-bgp-aggregate-address-add vrouter-
name vr3 network 10.0.56.0 netmask 21 as-set true

 CLI (network-admin@switch2*) > vrouter-bgp-aggregate-address-show
 vrouter-name network      vrf     as-set summary-only matching-MED-only origin route-map
suppress-map
 ------------ ------------ ------- ------ ------------ ---------------- ------ ---------
------------
 vr3          10.0.48.0/21 default false  true         false            none
 vr3          10.0.56.0/21 default true   true         false            none

 CLI (network-admin@switch3*) > vrouter-fib-routes-show intf-id 5
 vrid ip        prelen intf-id bd vlan port nexthop-mac       flags egress-id
 ---- --------- ------ ------- -- ---- ---- ----------------- ----- ---------
 0    10.0.48.0 21     5          62   273  66:0e:94:90:68:9d Trunk 100020
 0    10.0.56.0 21     5          62   273  66:0e:94:90:68:9d Trunk 100020

 CLI (network-admin@switch3*) > vtysh-cmd vrouter name vr0 cmd "show ip bgp
10.0.48.0/21"
 BGP routing table entry for 10.0.48.0/21
 Paths: (1 available, best #1, table default)
 Advertised to non peer-group peers:
 10.0.62.2
 65002, (aggregated by 65002 30.13.13.13)       =======> without 'as-set'
AS_PATH information is not included.
```

```
 10.0.62.2 from 10.0.62.2 (30.13.13.13)
 Origin IGP, valid, external, atomic-aggregate, best (First path received)
 Last update: Fri Jul 15 14:38:11 2022

 CLI (network-admin@switch3*) > vtysh-cmd vrouter name vr0 cmd "show ip bgp
10.0.56.0/21"
 BGP routing table entry for 10.0.56.0/21
 Paths: (1 available, best #1, table default)
 Advertised to non peer-group peers:
 10.0.62.2
 65002 65001, (aggregated by 65002 30.13.13.13) =======> with 'as-set'
AS_PATH information is included.
 10.0.62.2 from 10.0.62.2 (30.13.13.13)
 Origin IGP, valid, external, best (First path received)
 Last update: Fri Jul 15 15:14:08 2022
```

iv. vrouter-bgp-aggregate-address-add vrouter-name <NAME> network <A.B.C.D> netmask <M> with match-MED-only

```
 CLI (network-admin@switch1*) > vrouter-prefix-list-add vrouter-name vr1
name PREFIX58 action permit seq 5 prefix 10.0.58.0 netmask
 CLI (network-admin@switch1*) > vrouter-route-map-add vrouter-name vr1 name
SET_COMM_METRIC seq 10 action permit match-prefix PREFIX58 metric 300
 CLI (network-admin@switch1*) > vrouter-route-map-add vrouter-name vr1 name
SET_COMM_METRIC seq 15 action permit

 CLI (network-admin@switch1*) > vtysh-cmd vrouter name vr1 cmd "show ip bgp
neighbor 10.0.55.2 advertised-routes" BGP table version is 33, local router
ID is 10.0.57.1, vrf id 0
 Default local pref 100, local AS 65001
 Status codes:  s suppressed, d damped, h history, * valid, > best, =
multipath,i internal, r RIB-failure, S Stale, R Removed
 Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
 Origin codes:  i - IGP, e - EGP, ? - incomplete
    Network          Next Hop            Metric LocPrf Weight Path
 *> 10.0.48.0/21     10.0.55.2                0         65002 i
 *> 10.0.54.0/24     0.0.0.0                  0         32768 i
 *> 10.0.55.0/24     0.0.0.0                  0         32768 i
 *> 10.0.56.0/24     0.0.0.0                  0         32768 i
 *> 10.0.57.0/24     0.0.0.0                  0         32768 i
 *> 10.0.58.0/24     0.0.0.0                300         32768 i
 *> 10.0.59.0/24     0.0.0.0                  0         32768 i

 Total number of prefixes 7

 CLI (network-admin@switch2*) > vrouter-bgp-aggregate-address-show
 vrouter-name network       vrf      as-set summary-only matching-MED-only origin route-map suppress-map
 ------------ ------------ ------- ------ ------------ ----------------- ------ --------- ------------
vr3          10.0.48.0/21 default false  true         false             none
vr3          10.0.56.0/21 default false  true         false             none


 CLI (network-admin@switch3*) > vrouter-fib-routes-show intf-id 5
```

```
 vrid ip        prelen intf-id bd vlan port nexthop-mac       flags egress-id
 ---- --------- ------ ------- -- ---- ---- ----------------- ----- ---------
 0    10.0.48.0 21     5          62   273  66:0e:94:90:68:9d Trunk 100020
 0    10.0.56.0 21     5          62   273  66:0e:94:90:68:9d Trunk 100020
```

```
 CLI (network-admin@switch2*) > vrouter-bgp-aggregate-address-add vrouter-
name vr3 network 10.0.56.0 netmask 21 matching-MED-only true
 CLI (network-admin@switch2*) > vrouter-bgp-aggregate-address-show
 vrouter-name network      vrf     as-set summary-only matching-MED-only origin route-map
suppress-map
 ------------ ------------ ------- ------ ------------ ---------------- ------ ---------
------------
 vr3          10.0.48.0/21 default false  true         false            none
 vr3          10.0.56.0/21 default false  true         true             none
```

```
 CLI (network-admin@switch3*) > vrouter-fib-routes-show intf-id 5
 vrid ip        prelen intf-id bd vlan port nexthop-mac       flags egress-id
 ---- --------- ------ ------- -- ---- ---- ----------------- ----- ---------
 0    10.0.56.0 24     5          62   273  66:0e:94:90:68:9d Trunk 100020
 0    10.0.59.0 24     5          62   273  66:0e:94:90:68:9d Trunk 100020
 0    10.0.57.0 24     5          62   273  66:0e:94:90:68:9d Trunk 100020
 0    10.0.58.0 24     5          62   273  66:0e:94:90:68:9d Trunk 100020
 0    10.0.48.0 21     5          62   273  66:0e:94:90:68:9d Trunk 100020
```

Setting Metric of all specific routes to 300

```
 CLI (network-admin@switch1*) > vrouter-route-map-modify vrouter-name vr1
name SET_COMM_METRIC action permit seq 15 metric
 CLI (network-admin@switch1*) > vtysh-cmd vrouter name vr1 cmd "show ip bgp
neighbor 10.0.55.2 advertised-routes"
 BGP table version is 44, local router ID is 10.0.57.1, vrf id 0
 Default local pref 100, local AS 65001
 Status codes:  s suppressed, d damped, h history, * valid, > best, =
multipath,
                      i internal, r RIB-failure, S Stale, R Removed
 Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
 Origin codes:  i - IGP, e - EGP, ? - incomplete

    Network          Next Hop            Metric LocPrf Weight Path
 *> 10.0.48.0/21     10.0.55.2              300             0 65002 i
 *> 10.0.54.0/24     0.0.0.0                300         32768 i
 *> 10.0.55.0/24     0.0.0.0                300         32768 i
 *> 10.0.56.0/21     10.0.55.2              300             0 65002 i
 *> 10.0.56.0/24     0.0.0.0                300         32768 i
 *> 10.0.57.0/24     0.0.0.0                300         32768 i
 *> 10.0.58.0/24     0.0.0.0                300         32768 i
 *> 10.0.59.0/24     0.0.0.0                300         32768 i

 Total number of prefixes 8
```

When all the metrics are matched, only summary address is advertised and specific routes are suppressed.

```
CLI (network-admin@switch3*) > vrouter-fib-routes-show intf-id 5
 vrid ip        prelen intf-id bd vlan port nexthop-mac       flags egress-id
 ---- --------- ------ ------- -- ---- ---- ----------------- ----- ---------
 0    10.0.48.0 21     5          62   273  66:0e:94:90:68:9d Trunk 100020
 0    10.0.56.0 21     5          62   273  66:0e:94:90:68:9d Trunk 100020
```

Guidelines and Limitations

The following guidelines and limitations apply while configuring the BGP Route Aggregation:

- Making `summary-only` as the default option for aggregation reduces routing table & increases network stability. Without `summary-only` option, the `aggregate-address` command sends additional aggregated route in addition to the original networks.
- `summary-only` and `suppress-map` cannot be configured at the same time.

Configuring Prefix Lists for BGP and OSPF

Prefix lists allow you to permit or deny host IP addresses from route distribution in BGP and OSPF configurations. To configure prefix lists for BGP, this example assumes that you have a vRouter configured for BGP, vrouter-bgp, and you want to deny the IP address, 172.26.0.0 with the netmask 255.255.0.0, sequence number 5, and minimum prefix length 17 bits:

```
CLI (network-admin@switch) > vrouter-prefix-list-add vrouter-name vrouter-bgp name deny-bits action deny prefix 172.26.0.0 netmask 255.255.0.0 seq 5 min-prefix-len 17
```

This prefix list rejects any subnets of 172.26.0.0/16 with prefixes 17 bits or longer. For example, the subnets 172.26.16.9/30 and 172.26.101.0/24 are rejected from route distribution.

The sequence number allows you to insert or remove new lines in a prefix list as well as at the beginning or end. It is recommended that you increment the sequence numbers by 10 so you can easily add or subtract lists from the configuration. See also:

- Configuring Open Shortest Path First (OSPF)
- Configuring BGP on a vRouter

Configuring Bidirectional Forwarding Detection for IPv4 and IPv6

This feature adds bidirectional forwarding detection for IPv4 and IPv6 BGP neighbor and provides support for IPv6 BGP NBR reach-ability detection by using BFD protocol. When a BFD session goes from UP to DOWN, BFD informs BGP to bring the neighbor (NBR) down, until BFD returns to an UP state.

You create the BFD session by adding the bfd parameter to the vRouter configuration using the bfd parameter for the command, vrouter-bgp-modify. IPv6 BFD sessions for BGP NBRs are hosted in NetVisor OS. The BFD session is started when you add the bfd parameter the BGP vRouter configuration.

```
CLI (network-admin@switch) > vrouter-bgp-neighbor-show layout vertical

 vrouter-name:                    vr10
 neighbor:                        2002:100::2
 ver:                             4
 remote-as:                       51000
 msg_rcvd:                        189
 msg_sent:                        193
 up/down:
 state/pfxrcd:                    00:00:49 0
 multi-protocol:                  ipv6-unicast
 description:                     none
```

NetVisor OS supports IPv6 static route reach-ability detection using BFD protocol. Add IPv6 BFD session by specifying two end point IPv6 addresses, a source IPv6 address and a destination IPv6 address. The source IPv6 address must be a known local IPv 6 address. When a BFD session is up, NetVisor OS assesses all the IPv6 static routes configured with a gateway or a BFD destination IPv6 address matching the destination IPv6 address of the BFD session. When a match is found, this static route is installed in Routing Information Database (RIB) and Forwarding Information Database (FIB).

```
CLI (network-admin@switch) > vrouter-static-bfd-show layout vertical

 vrouter-name:                    vr10
 src-ip:                          2006:100::2
 dst-ip:                          2002:100::2
 type:                            multi-hop
```

Configuring BFD for OSPF Fault Detection

Bidirectional Forwarding Detection (BFD) can be used for OSPF fault detection. This feature provides fast failure detection when there is an intermediate device between two non-adjacent OSPF neighbors. That is, BFD provides fast failure detection between two nodes and notifies any protocol (OSPF) of this event to make it converge much faster than with default timers in protocols. Since OSPF hello timers may not be fast enough for detecting neighbor loss, you can use BFD to establish a BFD connection with the OSPF neighbor and bring an OSPF neighbor down as soon as BFD detects an issue.

Note that BFD is used for down detection only. This means that regular OSPF neighbor discovery and state machine transitions are not affected by enabling BFD.

You can enable BFD on all OSPF interfaces at the global level or on a specific interface. By default, BFD is disabled globally and on all interfaces. Individual interface configuration takes precedence over the global configuration.

> **Note:** BFD is not supported for OSPFv6.

Setting the vrouter global OSPF level:

To enable OSPF BFD per vRouter on global OSPF level:

```
CLI (network-admin@switch) > vrouter-modify name vrouter-name
```

`vrouter-modify`	Modify a vrouter
One or more of the following options	
`ospf-bfd-all-if|no-ospf-bfd-all-if`	Enables or disables BFD protocol for fault detection on all OSPF interface. Default is disabled

Setting the Interface (nic) OSPF level:

To enable OSPF BFD per interface :

```
CLI (network-admin@switch) > vrouter-interface-config-add vrouter-name nic
```

`vrouter-interface-config-add`	Add an interface configuration to a vRouter VNIC name.
One or more of the following options	
`nic vrouter interface nic`	Specify the name of the vNIC.
`[ospf-bfd] default|enable|disable`	Enable BFD protocol support for OSPF fault detection

To modify OSPF BFD per interface:

```
CLI (network-admin@switch) > vrouter-interface-config-modify vrouter-name
name nic nic ospf-bfd enable|disable|default
```

`vrouter-interface`	Modify an interface configuration to a vRouter
One or more of the following options:	Specify the name of the VNIC.
`ospf-bfd enable|disable|default`	Enables or disables the BFD protocol for fault detection on all OSPF interface. Default is disabled.

Displaying the OSPF BFD Configuration State

To display the configuration state of OSPF BFD, use these `show` commands:

```
CLI (network-admin@switch) > vrouter-show

CLI (network-admin@switch) > vrouter-interface-show
```

Note: There are no changes to the commands: `vrouter-ospf-neighbor-show` and `vrouter-bfd-neighbor-show`

Configuring Optimized BFD Path

To address BFD flaps, NetVisor OS offers an optimized path for BFD control packets to reach a destination vRouter. BFD packets are relayed by the NetVisor OS OS from the switch to the destination vRouter using a PCIe link which connects one of the switch ports to the host CPU. The BFD fastpath functionality alleviates BFD timeout/delay issues when other high bandwidth traffic is coming to the host CPU.

The optimized path option can be enabled or disabled using the command:

```
CLI  (network-admin@switch)  >  system-settings-modify
```

`system-settings-modify`	Use this command to modify system settings.
Specify one or more of the following options:	
`optimize-arps|no-optimize-arps`	Enable or disable ARP optimization.
`lldp|no-lldp`	Enable or disable LLDP.
`policy-based-routing|no-policy-based-routing`	Enable or disable Policy-Based Routing (PBR). This enables flexible packet forwarding and routing through user-defined policies.
`optimize-nd|no-optimize-nd`	Enable or disable ND optimization.
`reactivate-mac|no-reactivate-mac`	Enable or disable reactivation of aged out MAC entries.
`reactivate-vxlan-tunnel-mac|no-reactivate-vxlan-tunnel-mac`	Enable or disable reactivation of MAC entries over VXLAN tunnels.
`manage-unknown-unicast|no-manage-unknown-unicast`	Enable or disable unknown unicast management.
`manage-broadcast|no-manage-broadcast`	Enable or disable broadcast management
`block-loops|no-block-loops`	Enable or disable loop blocking. **Note**: This parameter is available only on NSU, NRU-02, NRU-03, and NRU-S0301 platforms.
`auto-trunk|no-auto-trunk`	Enable or disable auto trunking.
`auto-host-bundle|no-auto-host-bundle`	Enable or disable auto host bundling.
`cluster-active-active-routing|no-cluster-active-active-routing`	Enable or disable active-active routing on a cluster.
`fast-route-download|no-fast-route-download`	Enable or disable fast route download from routesnoop.
`fast-interface-lookup|no-fast-interface-lookup`	Enable or disable fast router interface lookup.

`routing-over-vlags|no-routing-over-vlags`	Enable or disable routing to vLAGs from cluster links.
`source-mac-miss to-cpu|copy-to-cpu`	Specify either of the options as the unknown source MAC learn behavior.
`use-igmp-snoop-l2|use-igmp-snoop-l3`	Specify whether L2 or L3 tables are to be used for IGMP snooping.
`vle-tracking-timeout <3..30>`	Set a vLE tracking timeout as a value between 3 and 30s. The default timeout is 3s.
`pfc-buffer-limit` *`pfc-buffer-limit-string`*	Specify the percent of global system buffer space allowed for PFC.
`cosq-weight-auto|no-cosq-weight-auto`	Specify either of the options to enable or disable automatic weight assignment for CoS (Class of Service) queues based on min-guarantee configuration.
`lossless-mode|no-lossless-mode`	Enable or disable lossless mode.
`stagger-queries|no-stagger-queries`	Stagger igmp/mld snooping queries.
`host-refresh|no-host-refresh`	Enable or disable refreshing host ARP entries to keep L2 entries active.
`proxy-conn-retry|no-proxy-conn-retry`	Enable or disable proxy connection retry.
`proxy-conn-max-retry 0..10`	Set the maximum number of proxy connection retry attempts as a value between 0 and 10.
`proxy-conn-retry-interval 100..2000`	Set the number of milliseconds to wait between proxy connection retry attempts. This is a value between 100 and 2000.
`nvos-debug-logging|no-nvos-debug-logging`	Logging mode selection (Direct OR viz. nvlog demon)
`manage-l2-uuc-drop|no-manage-l2-uuc-drop`	Enable or disable L2 UUC (Unknown Unicast Drop) towards data port.
`xcvr-link-debug|no-xcvr-link-debug`	Enable or disable system debug to capture link information.
`fastpath-bfd|no-fastpath-bfd`	Enable or disable BFD fastpath. This feature is disabled by default.
`linkscan-interval 10000..1000000`	Specify the linkscan interval as a value between 10000µ s and 1000000µ s. The default value is 150000µ s.
`linkscan-mode software|hardware`	Specify the linkscan mode as hardware or software. Software linkscan mode is enabled by default.
`single-pass-flood|no-single-pass-flood`	Enable or disable single-pass flood.

For example, use the command below to enable BFD fastpath:

```
CLI (network-admin@switch) > system-settings-modify fastpath-bfd
```

This command creates a new vFlow which redirects all BFD packets to CPU through the PCIe link.

To see the status of BFD fastpath on a switch, use the command `system-settings-show`. For example, to view the BFD fastpath status exclusively, use the command:

```
CLI (network-admin@switch) > system-settings-show format fastpath-bfd

fastpath-bfd: off
```

NetVisor OS has the transmission queue `tx-class-inband-bfd` for sending out BFD packets. For reception, the `rx-bfd` queue is used.

To view the details of the transmission queue, use the command:

```
CLI (network-admin@switch) > nv-queue-stats-show name tx-class-inband-bfd
```

To view the receiver queue details, use the command:

```
CLI (network-admin@switch) > nv-queue-stats-show name rx-bfd
```

Configuring Policy-Based Routing

Policy-Based Routing (PBR) enables flexible packet forwarding and routing through user defined policies. Unlike traditional routing based on destination IP address only, PBR allows you to define flexible routing policies based on other parameters such as source and destination IP addresses, IP protocol type, or source and destination L4 port numbers.

PBR policies are implemented with vFlow entries, which NetVisor OS allocates in a dedicated (hardware) vFlow table, called *System-L3-L4-PBR*.

In addition, the PBR policy configuration process leverages the vFlow command syntax as explained later in this section (refer also to the *Configuring and Using vFlows* section for further details on the feature).

PBR routing policies are higher priority than static and dynamic routes. They can match packets based on all Layer 4 and Layer 3 packet fields, as supported by the vFlow configuration syntax.

> **Note:** If a PBR policy clause is matched but the next-hop is not resolved, then the matching traffic is dropped until the next-hop gets resolved.

To enable PBR, use the following command:

```
CLI (network-admin@switch) > system-settings-modify policy-based-routing

Note: nvOSd must be restarted for this setting to take effect
```

To disable PBR, use the following command:

```
CLI (network-admin@switch) > system-settings-modify no-policy-based-routing
```

> **Note:** Restart the switch for this setting to take effect.

Use the following vflow command to configure a PBR policy. For details on configuring vFlows, see the *Configuring and Using vFlows* chapter.

```
CLI (network-admin@switch) > vflow-create name <policy-name> vrouter-name
<vr-name> scope local [<match qualifiers>] action to-next-hop-ip action-to-
next-hop-ip-value <ip-address> table-name System-L3-L4-PBR-1-0
```

> **Note:** You can only specify the scope as `local`.

To modify the PBR policy, use the command:

```
CLI (network-admin@switch) > vflow-modify name <policy-name> vrouter-name
<vr-name> [<match qualifiers>] action to-next-hop-ip action-to-next-hop-ip-
value <ip-address>
```

Use the following command to delete the policy:

```
CLI (network-admin@switch) > vflow-delete name <policy-name>
```

Below is an example of PBR policy creation:

```
CLI (network-admin@switch) > vflow-create name test_pbr scope local in-port
10 src-ip 192.168.1.1 src-ip-mask 255.255.255.0 vrouter-name vr1 action to-
next-hop-ip  action-to-next-hop-ip-value  192.168.10.10
```

To view the configure policy, use the following command:

```
CLI  (network-admin@switch)>  vflow-show

switch:   spine1
name:     test_pbr
scope:    local
type:     pbr
in-port:  10
src-ip:   192.168.1.1/255.255.255.0
burst-size:    auto
vrouter-name:  vr1
precedence:    default
action:   to-next-hop-ip
action-to-next-hop-ip-value: 192.168.10.10
enable:   enable
table-name:    System-L3-L4-PBR-1-0
```

To modify this policy, the *vrouter name* and *action to-next-hop-ip* parameters are required in `vflow-modify` command to identify it is a *PBR vFlow entry* that is getting modified. For example, this command modifies the in-port value:

```
CLI (network-admin@switch) > vflow-modify name test_pbr in-port 20 vrouter-
name  vr1  action  to-next-hop-ip  action-to-next-hop-ip-value  192.168.10.10
```

To display the vFlow table's usage and a specific PBR policy, use the following command sequence:

```
CLI (network-admin@switch) > vflow-table-show layout vertical

name:     Egress-Table-1-0
flow-max-per-group:512
flow-used:     0
flow-tbl-slices:    2
capability:    match-metadata
flow-profile:  system
name:     System-L1-L4-Tun-1-0
flow-max-per-group:2048
flow-used:     54
flow-tbl-slices:    2
capability:    set-metadata
flow-profile:  system
name:     System-VCAP-table-1-0
flow-max-per-group:512
flow-used:     0
```

```
flow-tbl-slices:   1
capability:   none
flow-profile:  system
name:    System-L3-L4-PBR-1-0
flow-max-per-group:
flow-used:
flow-tbl-slices:
capability:   set-metadata
flow-profile:  system


CLI (network-admin@switch) > vflow-show name pbr_test layout vertical

name:    pbr_test
scope:   local
type:    pbr
src-ip:  10.10.10.1/255.255.255.0
burst-size:   auto
vrouter-name:  vr1
precedence:   default
action:  to-next-hop-ip
action-to-next-hop-ip-value: 30.30.30.1
enable:  enable
table-name:   System-L3-L4-PBR-1-0
```

Sending Network Traffic to an ECMP Group with PBR

When it is required to specify multiple next hops for redundancy purposes in Policy-Based Routing policies, it is possible to use *static ECMP groups*. They can be created with the `static-ecmp-group-create` command and then used in a vFlow PBR configuration to identify all the next hops.

You can add up to 16 next hops (NH) to an ECMP group.

Static ECMP groups can be defined with any of the three scopes: `local`, `cluster` or `fabric`. They can become active only if they are associated with a vRouter in the configuration. In other words, only if a static ECMP group is associated with a vRouter or a VRF (with an active sub-net), does NetVisor OS create an ECMP group entry in the hardware.

A static ECMP group can be linked to a vFlow PBR policy using the `action to-ecmp-group` parameter and the group's name as the action value for `action-to-ecmp-group-value`. For example:

```
CLI (network-admin@switch) > vflow-create name PBR_ECMP scope local src-ip
3.3.3.0/24 vlan 300 action to-ecmp-group action-to-ecmp-group-value
group_name vrouter-name vr-s2 table-name System-L3-L4-PBR-1-0
vflow-create: ecmp group group_name not created in hw
```

In the above case the vRouter did not exist hence the group was not programmed in hardware.

In addition, only if a Layer 3 entry is resolved and therefore is active as a given next hop, the associated egress ID is added to the ECMP group. Then, if a vFlow policy using the ECMP group is matched by some traffic, the hardware hashes (i.e., distributes) the traffic over the corresponding active next hops based on the Layer 3 and Layer 4 fields in the packets.

To create a static ECMP group associated with a vRouter, use the command:

```
CLI (network-admin@switch) > static-ecmp-group-create
```

`group-name` *`group-name-string`*	Specify an ECMP group name.
`scope local|cluster|fabric`	Specify the scope of the group.
`vrouter-name` *`vrouter-name`*	Specify the vRouter name.
`vrf` *`vrf-name`*	Specify the name of the VRF.
`vnet` *`vnet-name`*	Specify the vNET for the static ECMP group. **Note:** `vnet` is an optional parameter that you can specify along with `vrf`.
`hash-type static-fixed|resilient`	Specify the ECMP hash type.

For example:

```
CLI (network-admin@switch) > static-ecmp-group-create group-name gr1 scope
local vrf vrf1 vnet vnet1 hash-type static-fixed
```

To display a static ECMP group's information you can use the command:

```
CLI (network-admin@switch) > static-ecmp-group-show
```

`group-name group-name-string`	Displays an ECMP group name.
`scope local|cluster|fabric`	Displays the scope of the group.
`vrouter-name vrouter-name`	Displays the vRouter name.
`vrf` *`vrf-name`*	Displays the name of the VRF.
`vnet` *`vnet-name`*	Displays the vNET for the static ECMP group.
`vrid vrid-number`	Displays the vRouter ID.
`hw-ecmp-id hw-ecmp-id-number`	Displays the hardware ID.
`hash-type static-fixed|resilient`	Displays the ECMP hash type.

For example, to view the information for the static ECMP group `gr1` configured above, use the command:

```
CLI (network-admin@switch) > static-ecmp-group-show group-name gr1

switch    group-name scope vrf  vnet  vrid hw-ecmp-id hash-type
------- ---------- ----- ---- ----  ---- ---------- ------------
switch1   gr1       local vrf1 vnet1  1    200256    static-fixed
```

To delete a static ECMP group, use the command:

```
CLI (network-admin@switch) > static-ecmp-group-delete group-name group-
```

name-string

Informational note: You cannot delete a static ECMP group while it is in use by any vFlow configuration.

To modify a static ECMP group, use the command:

```
CLI (network-admin@switch) > static-ecmp-group-modify group-name <group-name-string> vrouter-name <vrouter name> hash-type static-fixed|resilient
```

To add or remove a next hop to an ECMP group you can use:

```
CLI (network-admin@switch) > static-ecmp-group-nh-add
```

`group-name group-name-string`	Specify the name of the ECMP group.
`ip ip-address`	Specify the IP address for the next hop.

```
CLI (network-admin@switch) > static-ecmp-group-nh-remove
```

`group-name group-name-string`	Specify the name of the ECMP group.
`ip ip-address`	Specify the IP address for the next hop.

To show the next hop information you can use:

```
CLI (network-admin@switch) > static-ecmp-group-nh-show
```

`group-name group-name-string`	Displays the name of the ECMP group.
`ip ip-address`	Displays the IP address for the next hop.
`vlan vlan-id`	Displays the VLAN of the next hop.
`egress-id egress-id-number`	Displays the hardware egress ID.

By default ECMP groups use a fixed hashing algorithm to distribute the traffic across multiple next hops. The advantage of this choice is that such algorithm is simple to implement in hardware and hence is widely available on all switch models.

However, when a link associated with a next hop goes down, the traffic is automatically re-distributed to adapt to the change in the number of paths: this action requires a complete remapping of the hash values thus resulting in unnecessary traffic disruption for certain flows.

Therefore, starting from NetVisor OS release 5.1.1, on certain models only, a new more flexible hashing algorithm is supported. It is called *resilient hashing*, because it helps prevent unnecessary traffic disruption when the number of next hops changes.

The hash type can be specified as a parameter when a static ECMP group is created like so:

```
CLI (network-admin@switch) > static-ecmp-group-create group-name <name> [hash-type static-fixed|resilient]
```

The default hash type is static-fixed. For example, two groups with two different hash types can be created with the following commands:

```
CLI (network-admin@switch) > static-ecmp-group-create group-name gr1 scope fabric

CLI (network-admin@switch) > static-ecmp-group-nh-add group-name gr1 ip 2.2.2.2

CLI (network-admin@switch) > static-ecmp-group-create group-name gr2 scope fabric hash-type resilient

CLI (network-admin@switch) > static-ecmp-group-nh-add group-name gr2 ip 3.3.3.3

CLI (network-admin@switch) > static-ecmp-group-show

group-name scope  vrouter-name vrid hw-ecmp-id hash-type
---------- ------ ------------ ---- ---------- -------------
gr1        fabric   vr1          1    200001    static-fixed
gr2        fabric   vr1          1    200000    resilient
```

Informational note: Resilient hashing is not supported in the following switch models:

- Dell Z9100, Freedom F9532-C
- Dell S5048, Freedom F9572L-V

Configuring vRouter-based VRF (Virtual Routing and Forwarding)

Virtual Routing and Forwarding (VRF) is a technology used to partition the routing table into virtual instances (called VRF instances, or simply VRFs) that segregate routing entries in the control plane as well as in the data plane. Because the routing instances are independent, the same or overlapping IP addresses can be used without conflicting with each other.

Before NetVisor OS release 6.1.1, in order to support multiple VRF instances you needed to configure multiple vRouters to create isolated Layer 3 routing contexts in software. By using this strategy, you could create a 'dumb' 1:1 association between a vRouter and a VRF instance used to isolate Layer 3 domains in the data plane. *Unicast and Multicast Fabric VRFs* (see the *Configuring VXLAN* section for more details) were introduced as hardware entities for high-performance distributed routing and traffic segmentation. There was no corresponding VRF-aware vRouter entity associated to them and therefore no routing protocols could be run to exchange routing information (only static routing was supported).

NetVisor OS release 6.1.1 introduces '*native*' multi-VRF support to vRouters by making use of new advanced control plane capabilities.

The new implementation is more scalable compared to previous releases as it is much less demanding in terms of memory usage and CPU load (note that the previous vRouter-per-VRF approach is basically limited by the maximum number of supported vRouters, which is switch model-dependent). In addition, since a vRouter can now run routing protocols that are VRF-aware, per VRF peering is supported with BGP.

The new implementation also allows the same interface IP address to be used across different VRF instances. (However, note that reusing the same VLAN number for two interfaces of the same vRouter is not supported.) From a multi-tenant management perspective, VRF instances can be allocated to different vNETs so that each vNET administrator can independently add or remove the VRFs from the vRouters.

The configuration of vRouter-based VRF instances leverages the existing vRouter CLI, which is augmented to include the `vrf` parameter in many of the existing commands (see the list below).

In addition, a key new command, `vrouter-vrf-add`, is added to associate VRF instances to vRouters. Corresponding `vrouter-vrf-modify`, `vrouter-vrf-remove` and `vrouter-vrf-show` commands are introduced too. (Note that `vrouter-vrf-remove` removes a VRF instance as well as all vRouter interfaces belonging to it.)

For example, you can create a vRouter and associate two VRFs to it by using the following commands:

```
CLI (network-admin@switch) > vrf-create name VRF1 scope fabric
CLI (network-admin@switch) > vrf-create name VRF2 scope fabric
```

> **Note:** The maximum VRF name length is 15 characters in NetVisor OS.

```
CLI (network-admin@switch) > vrouter-create name vRouter1 vnet vNET1
router-type hardware bgp-as 100

CLI (network-admin@switch) > vrouter-vrf-add vrouter-name vRouter1 vrf VRF1
CLI (network-admin@switch) > vrouter-vrf-add vrouter-name vRouter1 vrf VRF2
```

Note : The same VRF cannot be added to more than one vRouter on the same node.

The next step is to configure Layer 3 interfaces and add them to the specific VRFs, for example:

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vRouter1
vrf VRF1 ip 100.1.1.1/24 vlan 100
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vRouter1
vrf VRF2 ip 100.1.1.1/24 vlan 101
```

Then you can specify the BGP neighbors on a per VRF basis with these commands:

```
CLI (network-admin@switch) > vrouter-bgp-add vrouter-name vRouter1 vrf VRF1
neighbor 100.1.1.2 remote-as 100
CLI (network-admin@switch) > vrouter-bgp-add vrouter-name vRouter1 vrf VRF2
neighbor 100.1.1.2 remote-as 101
```

Per-VRF BGP Parameters

Starting from NetVisor OS release 6.1.1 the addition of the `vrf` keyword makes the routing parameters VRF-aware. Some BGP parameters (such as `bgp-as`, `bgp-redistribute`, etc.) can now be specified during the `vrouter-vrf-add` configuration like so:

```
CLI (network-admin@switch) > vrouter-vrf-add vrouter-name <name> vrf <vrf-
name>  [bgp-as|bgp-redistribute|...]
```

(If a parameter is not specified, its value is inherited from the vRouter configuration.)
Subsequently, a parameter can be modified using the `vrouter-vrf-modify` command:

```
CLI (network-admin@switch) > vrouter-vrf-modify vrouter-name <name> vrf
<vrf-name>  [bgp-as|bgp-redistribute|...]
```

For example, it is possible to configure and modify a router ID on a per-VRF basis with the `vrouter-vrf-add` and `vrouter-vrf-modify` commands like so:

```
CLI (network-admin@switch) > vrouter-vrf-add vrouter-name vRouter1 vrf VRF1
router-id 11.1.1.1

CLI (network-admin@switch) > vrouter-vrf-show format vrf,hw-vrid,bgp-
as,router-id,

vrouter-name vrf   hw-vrid bgp-as router-id
------------ ---- ------- ------ ---------
vRouter1     VRF1 3       100    11.1.1.1

CLI (network-admin@switch) > vrouter-vrf-modify vrouter-name vRouter1 vrf
VRF1 router-id 11.1.1.10

CLI (network-admin@switch) > vrouter-vrf-show format vrf,hw-vrid,bgp-
as,router-id,

vrouter-name vrf   hw-vrid bgp-as router-id
```

```
------------  ----  -------  ------  ---------
vRouter1      VRF1  3        100     11.1.1.10
```

BGP uses the per-VRF `router-id` parameter value when it is configured. If it's not, the operational per-VRF router ID value is inherited from the corresponding vRouter's (global) router ID.

This is the full list of supported per-VRF parameters:

`bgp-as`	BGP Autonomous System number from 1 to 4294967295
`router-id`	BGP router id
`bgp-redistribute`	BGP route redistribution
`bgp-redist-static-metric`	BGP route redistribution static metric
`bgp-redist-static-route-map`	Route map for BGP redistribution of static routes
`bgp-redist-connected-metric`	Metric for redistributing BGP connected routes
`bgp-redist-connected-route-map`	Route map for BGP redistribution of connected routes
`bgp-redist-rip-metric`	Metric for redistributing RIP connected routes
`bgp-redist-ospf-metric`	Metric for BGP to redistribute OSPF connected routes
`bgp-redist-ospf-route-map`	Route map for BGP redistribution of OSPF routes
`bgp-cluster-id`	IP address for BGP cluster ID
`no-bgp-dampening`	No dampening for BGP routes
`bgp-dampening`	dampening is active for BGP routes
`bgp-keepalive-interval`	BGP Keepalive interval (seconds) - default 60
`bgp-holdtime`	BGP Holdtime (seconds) - default 180
`bgp-distance-external`	BGP distance for routes external to AS
`bgp-distance-internal`	BGP distance for routes internal to AS
`bgp-distance-local`	BGP distance for local routes
`no-bgp-default-shutdown`	Disabled
`bgp-default-shutdown`	Enabled
`no-bgp-redist-static-route-map`	Remove BGP static redist route-map
`no-bgp-redist-connected-route-map`	Remove BGP connect redist route-map
`no-bgp-redist-ospf-route-map`	Remove BGP OSPF redist route-map
`no-bgp-graceful-shutdown`	BGP graceful shutdown RFC 8326
`bgp-graceful-shutdown`	BGP graceful shutdown RFC 8326

The BGP parameters below, instead, are inherited from the vRouter (in other words, for them per VRF configuration is not supported):

```
bgp-delayed-startup
bgp-update-delayed-strict
bgp-max-paths
bgp-ibgp-multipath
bgp-bestpat-as-path
bgp-global-nh-preference
```

VRF-aware BFD Support

VRF support was added to BFD for both static routing and dynamic routing with BGP. The following vRouter commands are VRF aware. For static routing:

```
CLI (network-admin@switch) > vrouter-static-bfd-show
vrouter-name src-ip    dst-ip    type       vrf
------------ --------- --------- ---------- ----
vRouter1     100.1.1.1 100.1.1.2 single-hop VRF1
vRouter2     100.1.1.2 100.1.1.1 single-hop VRF2

CLI (network-admin@switch) > vrouter-bfd-neighbor-show format out-
addr,neighbor,holdown,multiplier,state,interface,vrf,flap-count,remote-router

vrouter-name  out-addr   neighbor   holdown(ms)  multiplier  state  interface  vrf   flap-count  remote-router
------------  ---------  ---------  -----------  ----------  -----  ---------  ----  ----------  -------------
vRouter1     100.1.1.1 100.1.1.2 2191        3          up    eth0.100  VRF1 0          vr1
vRouter2     100.1.1.2 100.1.1.1 2230        3          up    eth1.100  VRF2 0          vr2
```

For BGP

```
CLI (network-admin@switch) > vrouter-bgp-neighbor-show format neighbor,l3-port,nic,ver,remote-
as,up/down,state/pfxrcd,remote-router,description,vrf

vrouter-name neighbor  l3-port nic ver remote-as up/down  state/pfxrcd remote-router
description  vrf
------------ --------- ------- --- --- --------- -------- ------------ -------------
-----------  ----
vRouter1     100.1.1.2 0           4   200       00:00:18 Established  vr1
   VRF1
vRouter2     100.1.1.1 0           4   100       00:00:19 Established  vr2
   VRF2

CLI (network-admin@switch) > vrouter-bfd-neighbor-show format out-
addr,neighbor,holdown,multiplier,state,interface,vrf,flap-count,remote-router

vrouter-name out-addr  neighbor  holdown(ms) multiplier state interface vrf  flap-count
remote-router
------------ --------- --------- ----------- ---------- ----- --------- ---- ----------
-------------
vRouter1     100.1.1.1 100.1.1.2 2118        3          up    eth0.100  VRF1 0          vr1
vRouter2     100.1.1.2 100.1.1.1 2024        3          up    eth1.100  VRF2 0          vr2
```

VRF-aware Commands

The full list of vRouter commands with the new vrf keyword is:

- vrouter-bfd-neighbor-show
- vrouter-bgp-add

- vrouter-bgp-modify
- vrouter-bgp-neighbor-detail-show
- vrouter-bgp-neighbor-no-shutdown
- vrouter-bgp-neighbor-reset
- vrouter-bgp-neighbor-show
- vrouter-bgp-neighbor-shutdown
- vrouter-bgp-network-add
- vrouter-bgp-network-remove
- vrouter-bgp-network-show
- vrouter-bgp-remove
- vrouter-bgp-show
- vrouter-cached-routes-show
- vrouter-ecmp-cached-routes-show
- vrouter-ecmp-group-show
- vrouter-fib-arps-show
- vrouter-fib-routes-show
- vrouter-interface-add
- vrouter-interface-modify
- vrouter-interface-show
- vrouter-ping
- vrouter-routes-show
- vrouter-routes-stats-show
- vrouter-static-bfd-add
- vrouter-static-bfd-remove
- vrouter-static-bfd-show
- vrouter-static-route-add
- vrouter-static-route-remove
- vrouter-static-route-show
- vrouter-traceroute

Below are a few examples of VRF-aware commands:

```
CLI (network-admin@switch) > vrf-show format switch,name,scope,anycast-mac,active,hw-vrid,flags,enable

switch name  scope   anycast-mac       active hw-vrid flags   enable
------ ----- ------- ----------------- ------ ------- ------- ------
switch vrf1  cluster 64:0e:94:40:00:02 no     1       vrouter yes
switch vrf10 cluster 64:0e:94:40:00:02 no     10      vrouter yes
switch vrf11 cluster 64:0e:94:40:00:02 no     11      vrouter yes
switch vrf12 cluster 64:0e:94:40:00:02 no     12      vrouter yes
switch vrf13 cluster 64:0e:94:40:00:02 no     13      vrouter yes
switch vrf14 cluster 64:0e:94:40:00:02 no     14      vrouter yes
switch vrf15 cluster 64:0e:94:40:00:02 no     15      vrouter yes
switch vrf2  cluster 64:0e:94:40:00:02 no     2       vrouter yes
switch vrf3  cluster 64:0e:94:40:00:02 no     3       vrouter yes
switch vrf4  cluster 64:0e:94:40:00:02 no     4       vrouter yes
switch vrf5  cluster 64:0e:94:40:00:02 no     5       vrouter yes
switch vrf6  cluster 64:0e:94:40:00:02 no     6       vrouter yes
switch vrf7  cluster 64:0e:94:40:00:02 no     7       vrouter yes
switch vrf8  cluster 64:0e:94:40:00:02 no     8       vrouter yes
switch vrf9  cluster 64:0e:94:40:00:02 no     9       vrouter yes

CLI (network-admin@switch) > vrouter-vrf-show vrouter-name leaf1_vr1 format vrf,hw-vrid,bgp-as,bgp-max-
```

```
paths,bgp-bestpath-as-path,bgp-ibgp-multipath

vrouter-name  vrf     hw-vrid  bgp-as  bgp-max-paths  bgp-bestpath-as-path  bgp-ibgp-multipath
------------  -----  -------  ------  -------------  --------------------  ------------------
leaf1_vr1     vrf1   1        12101   16             multipath-relax       16
leaf1_vr1     vrf2   2        12102   16             multipath-relax       16
leaf1_vr1     vrf3   3        12103   16             multipath-relax       16
leaf1_vr1     vrf4   4        12104   16             multipath-relax       16
leaf1_vr1     vrf5   5        12100   16             multipath-relax       16
leaf1_vr1     vrf6   6        12106   16             multipath-relax       16
leaf1_vr1     vrf7   7        12107   16             multipath-relax       16
leaf1_vr1     vrf8   8        12108   16             multipath-relax       16
leaf1_vr1     vrf9   9        12109   16             multipath-relax       16
leaf1_vr1     vrf10  10       12110   16             multipath-relax       16
leaf1_vr1     vrf11  11       12111   16             multipath-relax       16
leaf1_vr1     vrf12  12       12112   16             multipath-relax       16
leaf1_vr1     vrf13  13       12113   16             multipath-relax       16
leaf1_vr1     vrf14  14       12114   16             multipath-relax       16
leaf1_vr1     vrf15  15       12115   16             multipath-relax       16

CLI (network-admin@switch) > vrouter-interface-show vrouter-name leaf1_vr1 vrf vrf1 format
nic,ip,mac,vlan,vlan-type,nic-state,vrf

vrouter-name  nic        ip             mac                vlan  vlan-type  nic-state  vrf
------------  ---------  -------------  -----------------  ----  ---------  ---------  -----
leaf1_vr1     eth0.110   11.1.0.2/24    66:0e:94:1b:c5:21  110   public     up         vrf1
leaf1_vr1     eth1.110   11.1.0.1/24    00:00:5e:00:01:79  110   public     down       vrf1
leaf1_vr1     eth0.111   11.1.1.2/24    66:0e:94:1b:c5:21  111   public     up         vrf1
leaf1_vr1     eth1.111   11.1.1.1/24    00:00:5e:00:01:79  111   public     down       vrf1
leaf1_vr1     eth0.112   11.1.2.2/24    66:0e:94:1b:c5:21  112   public     up         vrf1
leaf1_vr1     eth1.112   11.1.2.1/24    00:00:5e:00:01:79  112   public     down       vrf1
leaf1_vr1     eth0.113   11.1.3.2/24    66:0e:94:1b:c5:21  113   public     up         vrf1
leaf1_vr1     eth1.113   11.1.3.1/24    00:00:5e:00:01:79  113   public     down       vrf1
leaf1_vr1     eth0.114   11.1.4.2/24    66:0e:94:1b:c5:21  114   public     up         vrf1
leaf1_vr1     eth1.114   11.1.4.1/24    00:00:5e:00:01:79  114   public     down       vrf1
leaf1_vr1     eth2.3001  11.1.254.3/29  66:0e:94:1b:c5:21  3001  public     up         vrf1

CLI (network-admin@switch) > vrouter-bgp-neighbor-show vrouter-name leaf1_vr1 remote-router leaf2_vr1 vrf
vrf1 format neighbor,remote-as,msg_rcvd,msg_sent,up/down,remote-router,vrf

vrouter-name  neighbor     remote-as  msg_rcvd  msg_sent  up/down   remote-router  vrf
------------  ----------   ---------  --------  --------  --------  -------------  -----
leaf1_vr1     11.1.254.4   12101      7133      7147      4d22h42m  leaf2_vr1      vrf1

CLI (network-admin@switch) > vrouter-bfd-neighbor-show vrouter-name leaf1_vr1 remote-router leaf2_vr1 vrf
vrf1 format out-addr,neighbor,state,interface,vrf,flap-count,remote-router

vrouter-name  out-addr    neighbor    state  interface  vrf        flap-count  remote-router
------------  ----------  ----------  -----  ---------  --------   ----------  -------------
leaf1_vr1     11.1.254.3  11.1.254.4  up     eth2.3001  vrf1       0           leaf2_vr1
```

Configuring Multicast Listener Discovery (MLD)

In IPv4, Layer 2 switches can use IGMP snooping to limit the flooding of multicast traffic by dynamically configuring Layer 2 interfaces so that multicast traffic is forwarded to only those interfaces associated with IP multicast address. In IPv6, MLD snooping performs a similar function. With MLD snooping, IPv6 multicast data is selectively forwarded to the select list of ports, instead of being flooded to all ports in a VLAN. This list is constructed by snooping IPv6 multicast control packets.

MLD is a protocol used by IPv6 multicast routers to discover the presence of multicast listeners (nodes configured to receive IPv6 multicast packets) on its directly attached links and to discover which multicast packets are of interest to neighboring nodes. MLD is derived from the IGMP protocol. MLD version 1 (MLDv1) is equivalent to IGMPv2, and MLD version 2 (MLDv2) is equivalent to IGMPv3. MLD is a sub-protocol of Internet Control Message Protocol version 6 (ICMPv6), and MLD messages are a subset of ICMPv6 messages, identified in IPv6 packets by a preceding Next Header value of 58.

The switch can snoop on both MLDv1 and MLDv2 protocol packets and bridge IPv6 multicast data based on destination IPv6 multicast MAC addresses. You can configure the for MLD snooping and IGMP snooping simultaneously.

To display MLD routers on the network, use the `mld-router-show` command:

```
CLI (network-admin@switch) > mld-router-show

group-ip:
node-ip:
vlan:
port:
source:
node-type:
expires:
```

The show output displays the following information:

- Multicast group IP address in IPv6 format
- Host node IP address in IPv6 format
- Host VLAN ID
- Port number
- Multicast traffic source IP address in IPv6 format
- Node type as host or router
- Expires as the age-out time

To display MLD group membership on the network, use the `mld-show` command:

```
CLI (network-admin@switch) > mld-show

group-ip:
node-ip:
```

```
vlan:
port:
source:
node-type:
expires:
```

The show output displays the following information:

- Multicast group IP address in IPv6 format
- Host node IP address in IPv6 format
- Host VLAN ID
- Port number
- Multicast traffic source IP address in IPv6 format
- Node type as host or router
- Expires as the age-out time

To enable or disable MLD snooping or modify the scope, use the `mld-snooping-modify` command:

```
CLI (network-admin@switch) > mld-snooping-modify
```

To modify the scope from fabric to local, use the following syntax:

```
CLI (network-admin@switch) > mld-snooping-modify scope fabric
```

To display information about MLD snooping, use the `mld-snooping-show` command.

Configuring an IGMP Querier IP Address

You can configure an IGMP querier IP address for a VLAN or as a global IGMP querier. The IGMP querier sends IGMP General Query messages. on the network.

If you do not specify a querier IP address, then NetVisor OS uses 0.0.0.0 as the default value. There can be an unique querier IP for each VLAN, or you can configure the same Querier IP address for all the VLANs participating in IGMP snooping. The Querier IP address should have a local scope and every switch should have a unique Querier IP address.

With a valid source IP address on IGMP Query packets, NetVisor OS adds the VLAN receiving the Query to an IGMP Snoop switch list, and now reflects in the `igmp-switches-show` output and the IGMP queries sent to the peer switch as well. This solicits a report from the hosts listening on the peer switch.

Use these NetVisor OS commands to configure an IGMP querier IP address.

`igmp-querier-ip-modify`	Modify IGMP Querier IP configuration
`querier-ip` *`ip-address`*`s`	Specify the Snooping Querier IP address
`vlans-on-querier-ip` *`vlan-list`*	Specify the VLAN map.

`igmp-querier-ip-show`	Display IGMP Querier IP config
`querier-ip is`	Specify the Snooping Querier IP address
`vlans-on-querier-ip` *`vlan-list`*	VLAN MAP

Configuring Multicast Listener Discovery (MLD) Snooping per VLAN

MLD snooping provides support per individual VLANs. In addition to current global enable and disable support, you can do the following using the `mld-snooping-modify` command:

- Specify a range of VLANs on which MLDv1 and MLDv2 messages are used. These two VLAN ranges are mutually exclusive. When a MLDv1 query is received on a VLAN configuredforMDv2, the MLDv2 query is moved to a VLAN configured with MLDv1 if there is only one multi-cast router for this VLAN. If a MLDv2 message is later received for the same VLAN, existing entries are removed and the supported version is changed to MLDv2.
- Specify VLAN ranges for snooping link-local addresses and ND Solicited Node addresses. You must include these VLANs in one MLDv1 and MLDv2 VLAN list.

Use these command options for the `mld-snooping-modify` command:

```
CLI (network-admin@Spine1) > mld-snooping-modify
```

`mld-snooping-modify`	Enables or disables Multicast Listener Discovery (MLD) snooping
One or more of the following options:	
`scope local|fabric`	Specify the MLDsnooping scope - fabric or local
`enable|disable`	Enable or disable MLD snooping
`mldv1-vlans vlan-list`	Specify the VLANs to enable snooping and use MLDv1 protocol. Default: none
`mldv2-vlans vlan-list`	Specify the VLANs on which to enable snooping and use MLDv2 protocol. Default 1 - 4092
`snoop-linklocal-vlans vlan-list`	Allow snooping of link-local groups(ff02::/16) on these VLANs. Default 1 - 4092
`snoop-nd-vlans vlan-list`	Allow snooping of ND SN Multicast addresses (ff02::1:ff/104) on these VLANs. Default 1 - 4092

The `mld-snooping-modify` show format all command displays the following sample output:

```
CLI (network-admin@Spine1) > mld-snooping-show format all
```

switch:	Name of Switch
enable:	no
mldv1-vlans:	none
mldv2-vlans:	1-4092
snoop-linklocal-vlans:	1-4092
snoop-nd-vlans:	1-4092
nvOS-managed-vlans:	none
interop-v1-vlans:	none
vlans:	1-4092

Creating MLD Static Sources and Static Groups

To determine how to forward multicast traffic, a switch with MLD snooping enabled maintains information about the following interfaces in its multicast forwarding table:

- **Multicast-router interfaces** — These interfaces lead toward multicast routers or MLD queriers.
- **Group-member interfaces** — These interfaces lead toward hosts that are members of multicast groups.

The switch learns about these interfaces by monitoring MLD traffic. If an interface receives MLD queries, the switch adds the interface to the multicast forwarding table as a multicast-router interface. If an interface receives membership reports for a multicast group, the switch adds the interface to the multicast forwarding table as a group-member interface.

Table entries for interfaces that the switch learns about are subject to aging. For example, if a learned multicast-router interface does not receive MLD queries within a certain interval, the switch removes the entry for that interface from the multicast forwarding table.

You can create MLD static sources using IPv6 addresses and then create static groups using the static sources.

To create an MLD static source for IPv6 address, `ff02::1:ff11:111` as the group, and IPv6 `2001:db8::2:1` as the source on `VLAN 25`, `port 44-45`, use the following syntax:

```
CLI network-admin@switch > mld-static-source-create source-ip 2001:db8::2:1
group-ip ff02::1:ff11:111 vlan 25 ports 44-45
```

The parameter, `ports`, is an optional parameter. You can delete an MLD static source, but you cannot modify the parameters.

To display MLD static sources, use the `mld-static-source-show` command.

To create an MLD static group for IPv6 address, `ff02::1:ff11:1111`, on `VLAN 25, ports 44-45`, use the `mld-static-group-create` command:

```
CLI network-admin@switch > mld-static-group-create group-ip
ff02::1:ff11:1111 vlan 25 ports 44-45
```

You can delete an MLD static group, but you cannot modify the parameters. To display MLD static groups, use the `mld-static-group-show` command.

Displaying MLD Statistics for a VLAN

To display MLD statistics for a VLAN, use the `mld-stats-show` command:

```
CLI  network-admin@switch  >  mld-stats-show

switch:
vlan:
v1-queries:
v2-queries:
v1-member-reports:
v1-done-group:
v2-member-reports:
queries-sent:
drops:
ignored:
```

Configuring and Administering the Unified Cloud Fabric

This chapter provides information about Arista Unified Cloud Fabric and how to configure the fabric using the NetVisor OS CLI.

Arista Networks offers a unique and highly differentiated approach to software-defined networking (SDN), called Unified Cloud Fabric. The distributed architecture enables organizations to build scalable private and public clouds with improved ease of management, reliability, security and performance. Arista Networks innovative NetVisor® OS software virtualizes open networking hardware and builds a holistic, standard-based distributed network, referred to as a fabric, which provides improved management, automation, telemetry and resiliency.

Understanding the NetVisor® OS Unified Cloud Fabric™

Arista Networks offers a unique and highly differentiated approach to software-defined networking (SDN), called *Unified Cloud Fabric*. The distributed architecture enables organizations to build scalable private and public clouds that enjoy improved ease of management, reliability, security and performance.

Arista Networks' innovative NetVisor® OS software *virtualizes open networking hardware* to build a holistic, standard-based *distributed network*, referred to as '*a fabric*', which provides improved management, automation, telemetry and resiliency.

These enhancements are achieved thanks to the adoption of an advanced network virtualization paradigm.

In a unified cloud fabric every node behaves like a *'virtual module'* within a *logically-unified physically-distributed* '*virtual chassis*' which abstracts the *network topology*[1] . Each fabric node shares a common view of the network, including MAC and IP addresses, connections and application flows for management as well as redundancy purposes.

In addition, thanks to advanced embedded telemetry technology, the NetVisor OS software provides *fabric-wide traffic visibility* to reveal network congestion issues and application performance bottlenecks so as to speed up troubleshooting, improve operational efficiency and strengthen security.

Arista Unified Cloud Fabric is a distributed application on top of a standard Layer 2/Layer 3 network that unifies the management plane for all the switches of the fabric (see figure below) and at the same time enhances the network control plane with the ability to automate and simplify several network functions.

Note: [1]The Unified Cloud Fabric supports any network topology, including ring, leaf-spine and multi-site.

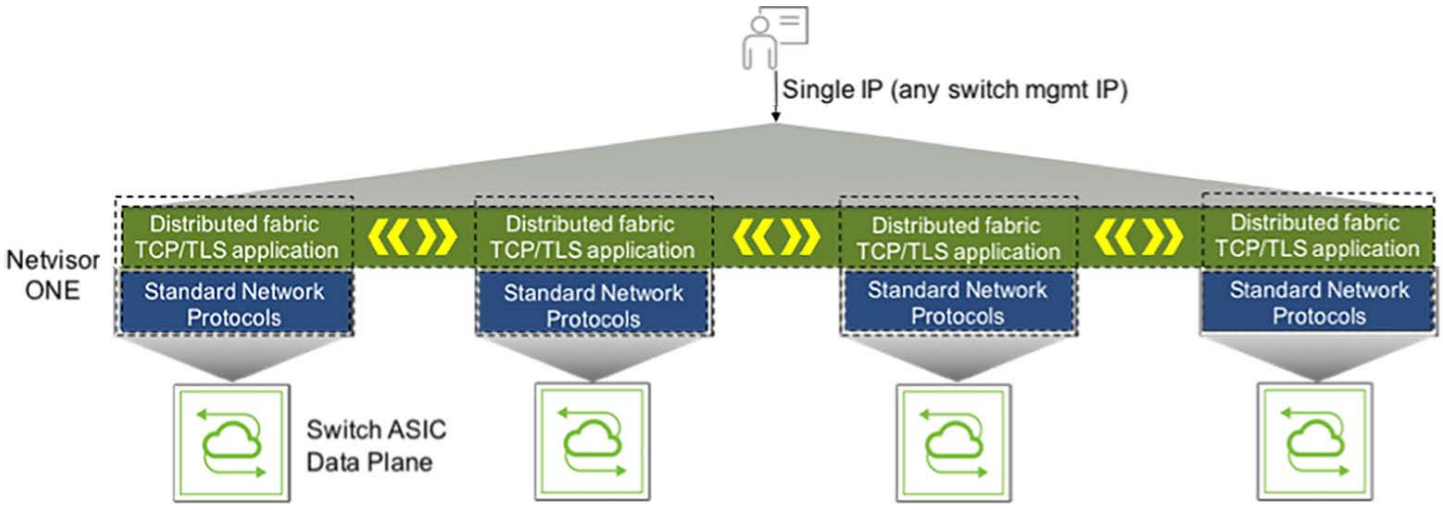


Figure 6 - 1: Unified Cloud Fabric Simplified Management with Single IP to Configure and Monitor the Fabric

This distributed control plane is *fully symmetrical* (or *peer-to-peer*) so that any node can act as a single point of management for the entire fabric. Therefore, *it does not require any controller entity* to guarantee the proper operation of the network and hence does not suffer from the intrinsic limits of a centralized control plane SDN architecture (such as *split brain* problems). At the same time, it is also able to inter-

operate with a centralized orchestration system or controller through RESTful APIs or an OVSDB interface. This enables additional powerful management strategies when leveraging popular platforms such as Ansible or OpenDaylight.

Unified Cloud Fabric's advanced transactional model guarantees that device configuration is maintained consistent across fabric nodes and supports also rollback capabilities. Therefore, *a single point of provisioning* provides 'atomic' fabric-wide configuration with commands that can operate on a list of dispersed fabric devices (instead of simply on individual ones).

Since Unified Cloud Fabric does not inherently require special controllers to operate (i.e., uses a *controller-less*, *de-centralized* model), and since it leverages standard protocols (instead of proprietary technologies) it is fully interoperable with devices from other vendors.

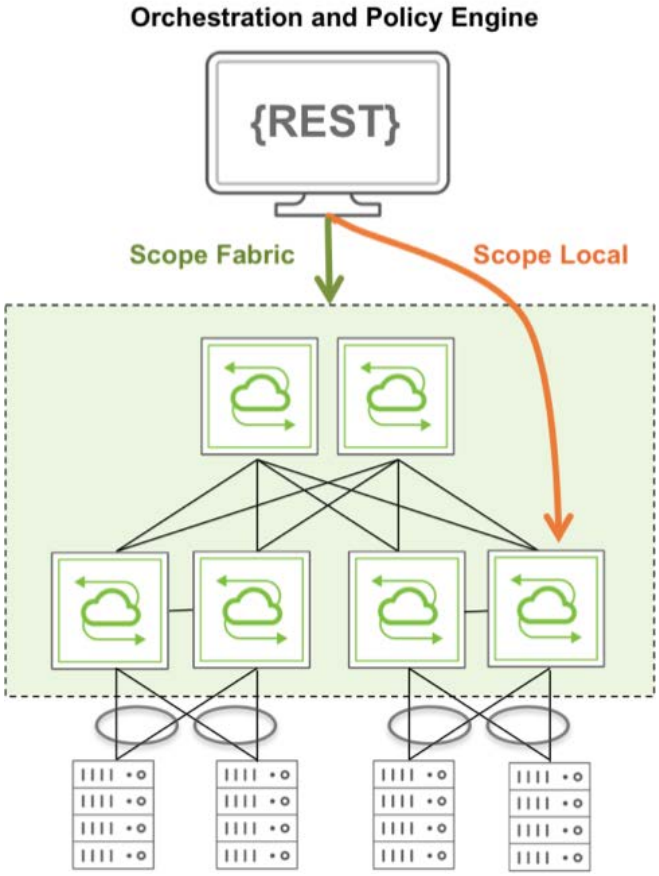


Figure 6 - 2: Network Orchestration with Unified Cloud Fabric

Unified Cloud Fabric's open architecture offers users a far superior degree of interoperability compared to centralized SDN architectures, while at the same time providing support for powerful market-leading network-wide analytics.

While each node can belong to only one fabric, multiple fabric domains can be part of the same network so as to implement more granular management domain segregation.

For example, in a typical leaf and spine data center configuration, it is often preferred to provision one management domain for all leaf switches and one separate domain for all spine switches, so as to preserve homogeneity as well as provisioning and monitoring simplicity within each functional domain.

The following picture (***Figure 6-3***) depicts an example of data center 'pod' topology comprising two

management domains corresponding to *two separate fabric* instances: one for all the leaf switches and one for the two spine switches.

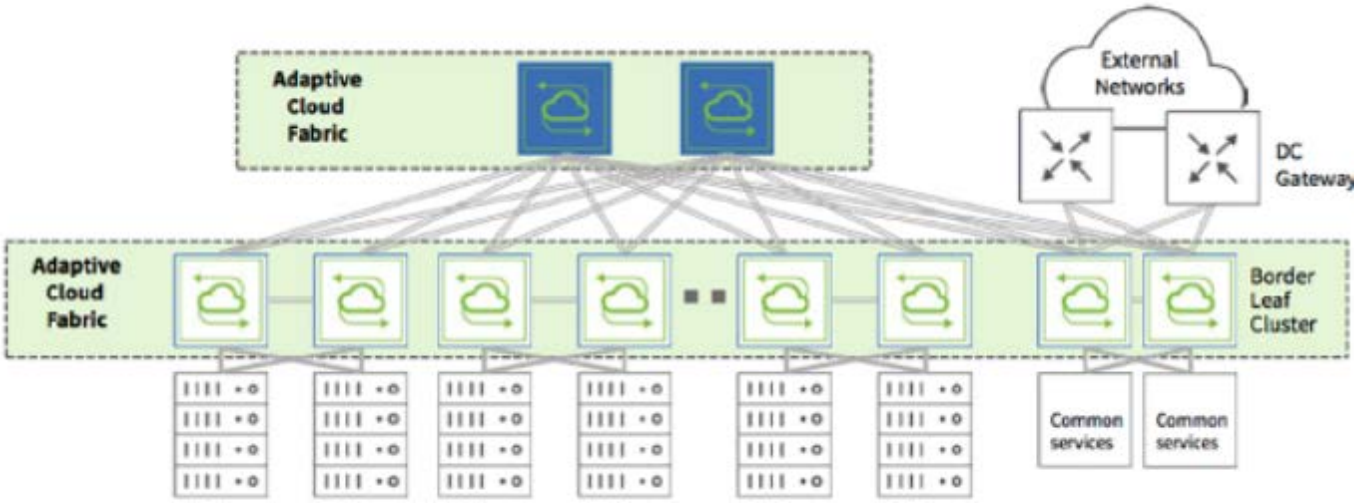


Figure 6 - 3: Two Fabric Instances to Achieve Two Management Domains

The Unified Cloud Fabric technology offers many additional benefits (described in more detail in the following chapters) that include more granular management segmentation for multi-tenancy, geographically dispersed data center interconnection (DCI), sophisticated security, etc.

Understanding Fabric Transactions

The Unified Cloud Fabric uses *transactions* to synchronize configuration changes across the nodes of the fabric. NetVisor OS records transactions as *atomic* operations that must either succeed and persist or fail and rollback, across the entire fabric. Transactions cannot be partially completed.

The fabric and NetVisor OS adheres to the four standard transaction requirements: *atomicity, consistency, isolation and durability* (also collectively known as *ACID*).

NetVisor OS does not require a fixed master node to coordinate all transactions across the fabric. Transactions start from the node where a command is run. This node is called the *originator node* or the *originator*, and coordinates the transactions with all the other fabric nodes.

NetVisor OS commands originate from clients such as a CLI user, a RESTful API user or an external orchestration system. The commands are executed on a chosen switch, which becomes the originator node.

The originator first applies the configuration change specified by the command on the local node. If that fails, NetVisor rolls back any partial changes and then returns a failure message to the user. Only after the local change succeeds does the originator start the transaction. NetVisor OS then atomically sends the configuration change such as create, delete, modify, add or remove commands to other fabric nodes.

NetVisor OS transmits fabric transactions over a dedicated TCP socket, does not retain it and closes after each phase of the transaction. Transactions are encrypted using the TLS protocol.

All transactions are logged in a log file on a per scope basis in this location: `/var/nvOS/etc/<scope>`, where <scope> is `Local`, `Cluster`, or `Fabric`.

Scope defines the set of nodes participating in a transaction:

- Local — only the local node participates in the transaction.
- Cluster — only two redundant nodes participate in the transaction.
- Fabric — all nodes in the same fabric instance participate in the transaction.

For several commands, you can specify the scope of the intended action and therefore the scope of the ensuing transaction.

If a failure occurs on the fabric, transactions on certain nodes in the fabric can become out of sync. Once transactions become out of sync, no further transactions can be executed across the scope of local, fabric, or cluster.

You can verify the fabric node states with the command, `fabric-node-show`, and review the fab-tid values for matching values.

```
CLI (network-admin@switch) > fabric-node-show  format  name,fab-name,fab-
tid,state,device-state,

name      fab-name   fab-tid state  device-state
--------- ---------- ------- ------ ------------
pnswitch2 pnfabric   2       online ok
```

```
pnswitch1 pnfabric   2       online ok
```

The `state` column represents the communication status between members of the fabric and the `device-state` column represents the overall health of each switch. Also note that the fabric transaction ID (2 in this example) is consistent across all members of the fabric.

Mismatching `fab-tid` values for one or more nodes within the same fabric instance represent a corner case that the fabric control logic typically prevents. For more details, see the *Rolling Back and Rolling Forward Transactions* section for more details.

Understanding Fabric Status Updates, vPorts and Keepalives

Instead of transactions, which are reserved for create, delete, modify, add or remove commands, NetVisor OS uses status update messages to distribute information across the fabric.

Unlike transactions, NetVisor OS does not strictly guarantee the status update data to be synchronized across the fabric.

NetVisor OS sends the following states from the local node to the other nodes in the fabric:

- Node State
- Port State
- VLAN State
- Owned vPort State
- Layer 3 Entry State

NetVisor OS uses vPort (or virtual port) as Layer 2 entries managed by the software and associated to ports where a node performs MAC address learning. vPorts contain information such as the MAC and IP address of a host, the VLAN and the connected port, the state, and other parameters.

In a fabric a node only sends status updates for vPorts that “is part of the fabric, which means that it sends updates only for the states of the hosts directly connected to that node. NetVisor OS uses the term '*owned vPorts*' to represent directly connected hosts.
You can display this information about directly connected hosts using the command, `vport-show`.

For example:

```
CLI (network-admin@switch) > vport-show

mac                vlan   ip           ports  state  migrate
-----------------  ----   --------     -----  -----  ------
00:00:4c:06:91:f8  514    10.81.114.21 61     active   9
```

In addition, NetVisor OS sends status updates whenever a state changes on the fabric. For example, if a port goes down, or you create a new VLAN, the node with the port or VLAN sends a status update about the specific change.
If sending the update message fails, NetVisor OS attempts to resend it every 250ms.

With multiple nodes configured in the same fabric instance, switches send out special messages, called *fabric keepalives*, typically every 10 seconds to allow fabric peers to keep track of the online state of other nodes in the same fabric instance.

If a fabric keep-alive message is not received from a specific node for a period of time (30 s) equal to 3 times the transmission interval (10 s), the suddenly-gone-silent node’s state is marked as `offline` by its neighbors. This inter-device connectivity/status check is performed with every other node within the same fabric instance.

Understanding the Different Fabric Deployment Models

When you create a fabric instance, by default, NetVisor OS uses Layer 2 communication over a configurable (user-selectable) VLAN for dedicated messages. The default behavior is the basic mode of operation and is referred to as *Fabric over L2* deployment model.

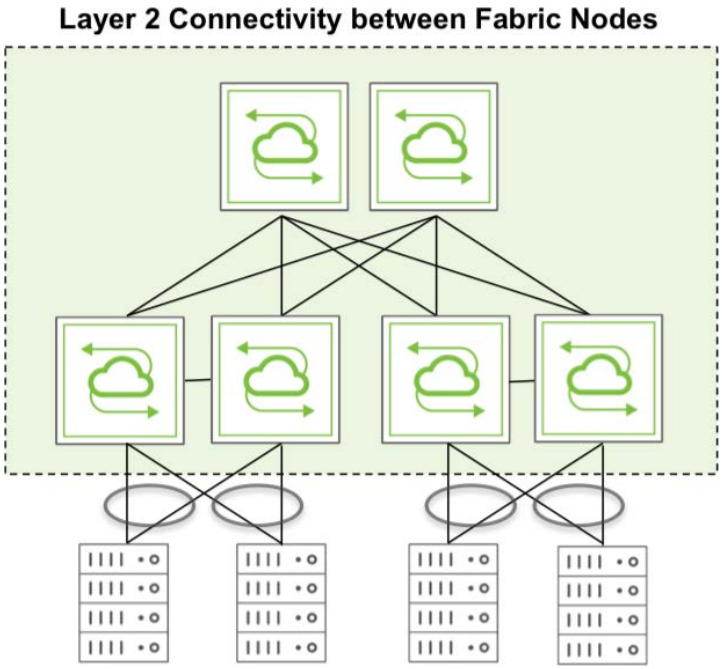


Figure 6 - 4 Fabric over Layer 2 Deployment Model

To perform fabric-related communication between nodes, NetVisor OS uses different message groups called:

- fabric-network
- control-network

The fabric-network group consists of a number of messages such as transactions, fabric notifications, and proxy I/O messages such as API calls.

The control-network group consists of status updates, cluster synchronization messages, proxy I/O messages and forwarded packets. They may (but don't have to) be configured separately for improved message forwarding flexibility.

In certain network designs, a *dedicated management network* is deployed *in parallel* to the regular network, also referred to as *in-band* network, to interconnect nodes through management interfaces (see ***Figure 6-5***).

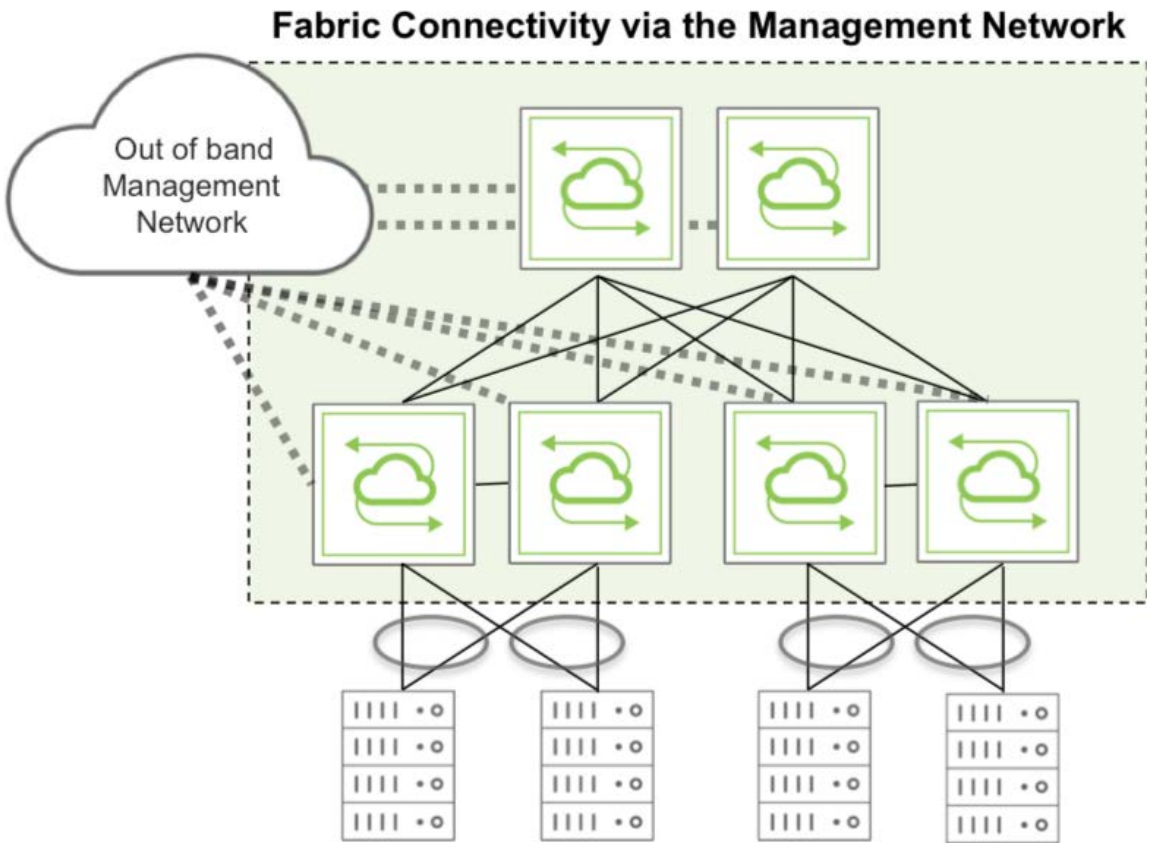


Figure 6-5: Fabric Connectivity over the Management Network

The *out-of-band* management network provides a robust alternate communication path between the nodes unaffected by user data traffic and user-related events.

In such designs a network administrator may leverage the dedicated management network for fabric communication. Therefore, NetVisor OS allows the network administrator, using the `fabric-create` command, to select if a communication group uses the node management interface or the in-band interface. The fabric deployment model leveraging the management network for communication is called *Fabric over Management Interface*.

In addition to the two described network configurations, a third deployment model exists, *Fabric over L3*, and leverages a Layer 3 network running BGP or OSPF routing protocol to establish point-to-point fabric communication between nodes.

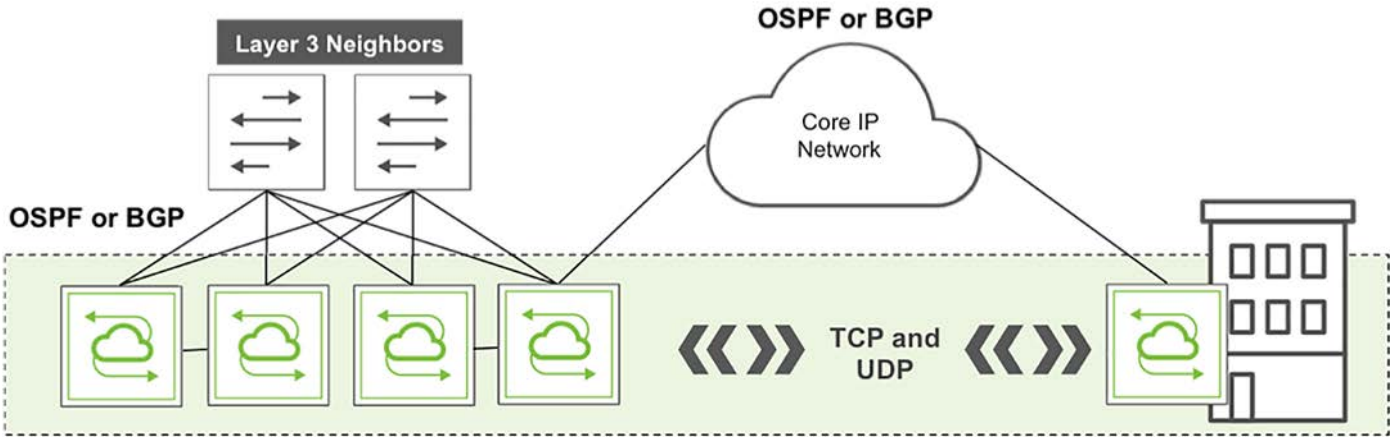


Figure 6-6: Fabric over Layer 3 Deployment

In the *Fabric over L3* deployment, multicast messages cannot be exchanged, as no Layer 2 adjacency exists between the nodes. Direct TCP or UDP communication must be established instead.

Therefore, you must first create the fabric on one node and then all the other nodes connected through Layer 3 can join the fabric one by one by specifying the IP address of the initial node using the command `fabric-join switch-ip`.

Creating an Initial Fabric

After completing the initial setup of a switch, you can create a new `fabric instance` to add the switch, or you can add the switch to an existing fabric.

When multiple switches join a fabric, the switches act as 'virtual modules' of a 'virtually distributed switch' with a single logical management plane. In this mode of operation, switches share status information and exchange commands based on the configured scope.

For example, any command with the scope, `fabric`, executes on each switch belonging to a shared fabric instance. This virtualized management paradigm significantly simplifies the network configuration and speeds up the deployment of complex networks.

A fabric instance can consist of just one individual switch, even though it is more common to have more than one switch, to ensure redundancy. For example, see Figure 6- 3 Two Fabric Instances to Achieve Two Management Domains, where two redundant spine switches belong to a dedicated fabric instance.

NetVisor OS continues to maintain the sharing of state and scope of a switch as long as the switch belongs to the fabric instance. When a switch leaves one fabric instance to join another one, the switch loses the synchronized fabric state and configuration of the first instance and learns the state and configuration of the second one.

To create a new fabric instance, use the following command:

```
CLI (network-admin@switch) > fabric-create name name-string
```

where *name-string* designates the name of the fabric.

Use the `password` option if you want to assign a password to the fabric instance creation process so that other switches are required to securely join the fabric only if the administrator knows the password.

```
CLI (network-admin@switch) > fabric-create name name-string
password<return>

fabric password: hidden-password-string <return>
confirm fabric password: hidden-password-string <return>
```

For modifying the fabric password, see the Modifying the Fabric Password section in the Installing NetVisor OS and Initial Configuration chapter.

About the Default Configuration

By default, NetVisor OS creates a new fabric instance on VLAN 1.

However, VLAN 1 is often the default VLAN in most networks and therefore security best practices recommend using a non-default VLAN, whenever possible, for maximum robustness and error prevention.

To assign a non-default VLAN, for example, a VLAN ID between 2 and 4093, use the command:

```
CLI (network-admin@switch) > fabric-create name name-string vlan vlan-id
```

To change the VLAN ID of an existing fabric, use the command, `fabric-local-modify`.

Use the same command to change the fabric administration network, control plane network, and the network to send fabric advertisements.

```
CLI (network-admin@switch) > fabric-local-modify vlan vlan-id
```

> **Note:** In order to change the VLAN number of an existing instance you must recreate it. Also, a switch can belong to only one fabric instance.

When you create a fabric, NetVisor OS uses the in-band network for fabric communication by default.

To create a simple fabric of two switches in the same subnet with in-band connectivity, follow the steps below:

Assign IP addresses to the in-band network on the switches by using the commands:

```
CLI (network-admin@switch1*) > switch-setup-modify in-band-ip
192.168.0.1/30

CLI (network-admin@switch2*) > switch-setup-modify in-band-ip
192.168.0.2/30
```

You can also specify an IPv6 addresses for in-band network by providing the `in-band-ip6` parameter.

Create the fabric:

```
CLI (network-admin@switch1) > fabric-create name fabric1
```

Display the fabric configuration:

```
CLI (network-admin@switch1) > fabric-show
name             id               vlan fabric-network control-network tid fabric-advertisement-network
---------------- ---------------- ---- -------------- --------------- --- ----------------------------
fabric1          b002698:6078060b 1    in-band        in-band         2   inband-mgmt
```

As seen from the output, the fabric and control networks are auto-configured to use the in-band network.

As the fabric uses VLAN 1 by default, verify the configuration using the `vlan-show` command:

```
CLI (network-admin@switch1) > vlan-show id 1
id type   auto-vxlan replicators scope description active stats ports    untagged-ports active-edge-ports
-- ------ ---------- ----------- ----- ----------- ------ ----- -------- -------------- ----------------
1  public no         none        local default-1   yes    yes   0-72,272 0-72,272       0,2,272
```

You can verify that the ports that connect the two switches are untagged and belong to VLAN 1.

Displaying Fabric Instances

To show all the fabric instances and their specific details, use the `fabric-show` command:

```
CLI  network-admin@switch  >  fabric-show

name         id                   vlan  fabric-network  control-network   tid   fabric-advertisement-network
------    ----------------  ----  --------------  ---------------    ---  ----------------------------
Fabric1  b000707:59b6a9ef   0       mgmt                mgmt               inband-mgmt
Fabric2  90004eb:59b7da05   0       mgmt                mgmt               inband-mgmt
```

NetVisor OS discovers all available fabric instances by sending out special multicast messages, called *global discoveries*, whenever a physical port becomes forwarding or on demand.

For example, when you execute a `fabric-show` command, NetVisor OS sends discovery messages over in-band as well as over the management interface.

After receiving a global discovery message the receiving device responds with a *global keep-alive* message containing the required fabric and node information.

This local multicast-based discovery mechanism implies that direct Layer 2 connectivity exist between the discoverer and the polled switches.

Adding Switches to an Existing Fabric

To add a switch to an available fabric instance, use the following command:

```
CLI (network-admin@switch) > fabric-join name name-string
```

For example:

```
CLI (network-admin@switch) > fabric-join name MyFabric
```

In case of directly connected switches as in ***Figure 6-7***, switch B learns about available fabrics from switch A and then sends a message to join the selected fabric.

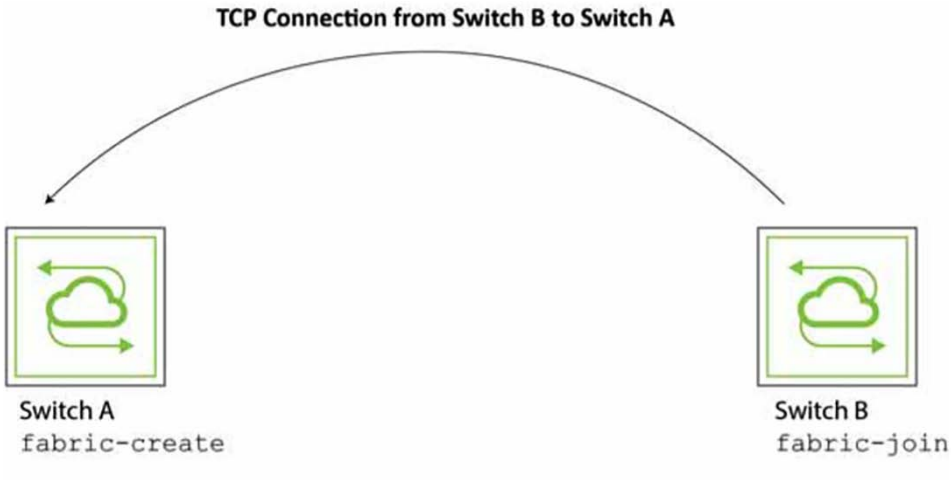


Figure 6 -7 Directly Connected Switch Joining a Fabric

Using the Tab key, NetVisor OS displays all fabrics configured in the local network as options.

A switch joins the fabric either by using a discovered fabric name or by using a switch IP address. To join a remote fabric, use the `fabric-join` command with a remote switch IP address:

```
CLI (network-admin@switch) > fabric-join switch-ip 192.168.2.2
Joined fabric MyFabric. Restarting nvOS...
```

In addition, the NetVisor OS software can use the password set up during the fabric creation process to encrypt communication between the nodes in the fabric.

In such cases, when the switch joins a fabric instance from a node, you must type in the password to join it.

> **Note:** Avoid creating fabrics with the same name to prevent conflicts.

Once a new switch joins an existing fabric, the new switch downloads all fabric configuration from the existing fabric switch and restarts the nvOSd (reboots).

After the switch is rebooted and is up and running, the new switch becomes part of the existing fabric.

Configuring the Fabric Over the Management Interface

Fabric configuration information can be exchanged over the network through in-band communication. However, occasional disruption to in-band traffic occurs due to factors such as network re-convergence, port flapping, power system transients, and other events. Therefore, an alternative method is to configure fabric communication over the management interface for a dedicated communication channel.

This can be achieved while creating the fabric, for example:

```
CLI (network-admin@switch) > fabric-create name MyFabric fabric-network
mgmt
```

When you create a fabric over the management interface, any other node joining the fabric inherits this setting. In other words, all nodes within the same fabric communicate through the same network type with fabric peers. You cannot have mixed fabric configurations using both management interfaces and in-band communication.

Therefore, NetVisor does not display fabrics over an incompatible networks when you execute the `fabric-join` command. This prevents a switch from joining an incompatible fabric.

When you configure the fabric communication over the management interface, all fabric communication stays on the management network, except the following types of packets:

- Cluster synchronization-related messages and cluster keep-alive packets sent over the in-band interface.
- The fabric advertisements such as fabric keep-alive packets and global-discovery packets are controlled by fabric-advertisement-network, which is configured while creating or modifying a fabric.

While fabric-related communication such as transactions, notifications, file system replication messages, and other communications can be configured to be sent over the management network, for consistency, it is recommended to use the same management network for other purposes or communication types such as network status updates and forwarded packets, collectively referred to as control network and fabric advertisements.

The `fabric-create` command allows you to select the transmission medium for other traffic types using specific parameters named `control-network` and `fabric-advertisement-network`:

```
CLI (network-admin@switch) > fabric-create name MyFabric [fabric-network
in-band|mgmt] [control-network in-band|mgmt] [fabric-advertisement-network
inband-mgmt|inband-only|mgmt-only]
```

Displaying Fabric Nodes

NetVisor OS uses fabric keepalive packets to determine the state of each fabric node. To display the state, use the fabric-node-show command with the syntax:

```
CLI network-admin@switch > fabric-node-show [state offline| online|in-band-only-online|mgmt-only-online|fabric-joined|eula-required|setup-required| fabric-required|fresh-install]
```

NetVisor OS supports monitoring and reporting on both management and in-band network, therefore the node state can be one of the following:

- `online` — reach-ability of node over both management and in-band interfaces
- `In-band-only-online` — reach-ability of node through in-band channel only
- `mgmt-only-online` — reach-ability of node through management network only
- `offline` — no reach-ability over either communication channel.

In this example, NetVisor OS displays the `online` node state in the command output:

```
CLI (network-admin@switch) > fabric-node-show layout vertical
id:                     167772208
name:                   switch
fab-name:               MyFabric
fab-id:                 a000030:5537b46c
cluster-id:              a000030:1
fab-mcast-ip:           ::
local-mac:               64:0e:94:28:00:8e
fabric-network:         in-band
mgmt-ip:                 10.9.100.100/16
mgmt-mac:                64:0e:94:28:00:8f
mgmt-l3-port:           0
mgmt-secondary-macs:
in-band-ip:              192.168.42.10/24
in-band-mac:             64:0e:94:28:00:8e
in-band-l3-port:        0
in-band-secondary-macs:
fab-tid:                8
cluster-tid:            1
out-port:               0
version:                 5.0.0-5000014540
state:                  online
firmware-upgrade:        not-required
device-state:           ok
ports:                  0
```

Also check the `fab-tid` value for consistency on each node. See the *Troubleshooting the Fabric* section for details.

Displaying Fabric Information and Statistics

To display information on the configured fabrics, use the fabric-show command:

```
CLI (network-admin@switch) > fabric-show

 name       id              vlan  fabric-network control-network  tid
------   ----------------   ----  -------------- ---------------  -----
Fabric1  a000030:5537b46c   3      in-band        in-band         365
Fabric2  6000210:566621ee   100    mgmt           in-band         5055
```

To display the information about the fabric instance of the local switch, use the `fabric-info` command:

```
CLI (network-admin@switch) > fabric-info format all layout vertical

name: Fabric1
id: a000030:5537b46c
vlan: 3
fabric-network: in-band
control-network: in-band
tid: 365
fabric-advertisement-network: inband-only
```

To display fabric statistics use the `fabric-stats-show` command:

```
CLI (network-admin@switch) > fabric-stats-show

 switch    id  server   storage  VM  vlan  vxlan  tcp-syn  tcp-est  tcp-completed  tcp-bytes  udp-bytes  arp
 -------   --  -------  -------  --  ----  -----  -------  -------  -------------  ---------  ---------  ----
 pubdev02  0   0        0        0   0     0      14.0k    5        40             125K       0          0
 pubdev03  0   0        0        0   0     0      3.85K    3        24             110M       0          0
```

To display fabric statistics in vertical format, use the following command:

```
CLI (network-admin@switch) > fabric-stats-show format all layout vertical
switch: sw45
id: 0
servers: 0
storage: 0
VM: 0
vlan: 0
vxlan: 0
tcp-syn: 0
tcp-est: 0
tcp-completed: 0
tcp-bytes: 0
udp-bytes: 0
arp: 0
```

Guidelines and Limitations

The following guidelines and limitations apply while configuring the fabric:

- Arista recommends, for security reasons, to change the VLAN ID used to create an in-band fabric over a Layer 2 or Layer 3 network as NetVisor OS uses a default VLAN ID, VLAN 1.
- A fabric over the in-band network (Layer 2 or Layer 3) does not require a special MTU configuration.
- A fabric over management uses Layer 2 multicast, so be sure you do not filter multicast traffic on your management switches.
- Changing the IP address of a node joined to the fabric may cause the node to appear as offline for other fabric members. Therefore, before changing the IP address, unjoin the fabric, and then rejoin the fabric to avoid any inconsistencies.
- An Arista switch can only be part of one single fabric at a time.
- NetVisor OS supports the following currently validated, tested number of fabric nodes in the same fabric:
    - 24 nodes for non-Arista switches
    - 32 nodes for Arista Networks switches
- When setting up a fabric, switches can only join the fabric one at a time. If you have a large fabric, you should consider this delay during the provisioning of the fabric.

Configuring Layer 4 Ports for Fabric Communication

NetVisor OS receives fabric communication messages by using system vFlow entries to match traffic flows based on the IP packet protocol, TCP or UDP, and Layer 4 destination port.

Any existing TCP or UDP traffic across the network using the same Layer 4 destination port may match the vFlows and may potentially cause conflicts.

Table 6-1: Default Port Values for NetVisor OS

Message Type	Layer 4 Port
fabric	23399
notify	23398
proxy	23397
fabric-keepalive	23394
filesystem-replication	23392
cluster-traffic-forwarding	23391
vport-statistics	23390
l2-encap	23389
igmp-encap	23388
icmpv6-encap	23387
arp-encap	23386
cluster-analytics	23385

To avoid conflicts with generic TCP or UDP traffic, configure an alternate Layer 4 port range for special messages using the command:

```
CLI  (network-admin@switch)  >  fabric-comm-ports-modify
```

You can specify the value for the starting point of the range:

```
range-start  1024..65435  port  range  start
```

To display the current starting point and range values, use the following command:

```
CLI  (network-admin@switch)  >  fabric-comm-ports-show

switch:                                pnswitch2
range-start:                           23300
fabric-port:                           23399
notify-port:                           23398
proxy-port:                            23397
fabric-keepalive-port:                 23394
filesystem-replication-port:           23392
cluster-traffic-forwarding-port:       23391
vport-statistics-port:                 23390
```

```
l2-encap-port:                  23389
igmp-encap-port:                23388
icmpv6-encap-port:              23387
arp-encap-port:                 23386
cluster-analytics-port:         23385
```

When you modify the port range, you must configure each node in the fabric individually.

This change temporarily interrupts fabric communication until you have completed the configuration of each node with the same port range.

There is no loss of switched traffic if the interruption is brief.

Because application of this command prevents communication with other nodes, you must log into each node directly and separately apply the `fabric-comm-ports-modify` command.

Configuring a Fabric Over a Layer 3 Network

As described in previous sections, you can create a NetVisor OS fabric over the management network or over in-band, i.e., between switches with direct (Layer 2) connectivity.

Another option is to establish NetVisor OS fabric over a Layer 3 network. NetVisor OS supports this capability over a Layer 3 network running either the *BGP* or the *OSPF routing protocol*.

Before you can configure this feature on your network, make sure you configure BGP or OSPF routing on your switches. For more information on Layer 3 configuration, refer to the *Configuring Layer 3 Features* section.

In a simple topology with a network of two BGP neighbors as in ***Figure 6-8.*** NetVisor OS uses in-band communication to establish TCP sessions between the two switches to exchange fabric messages.

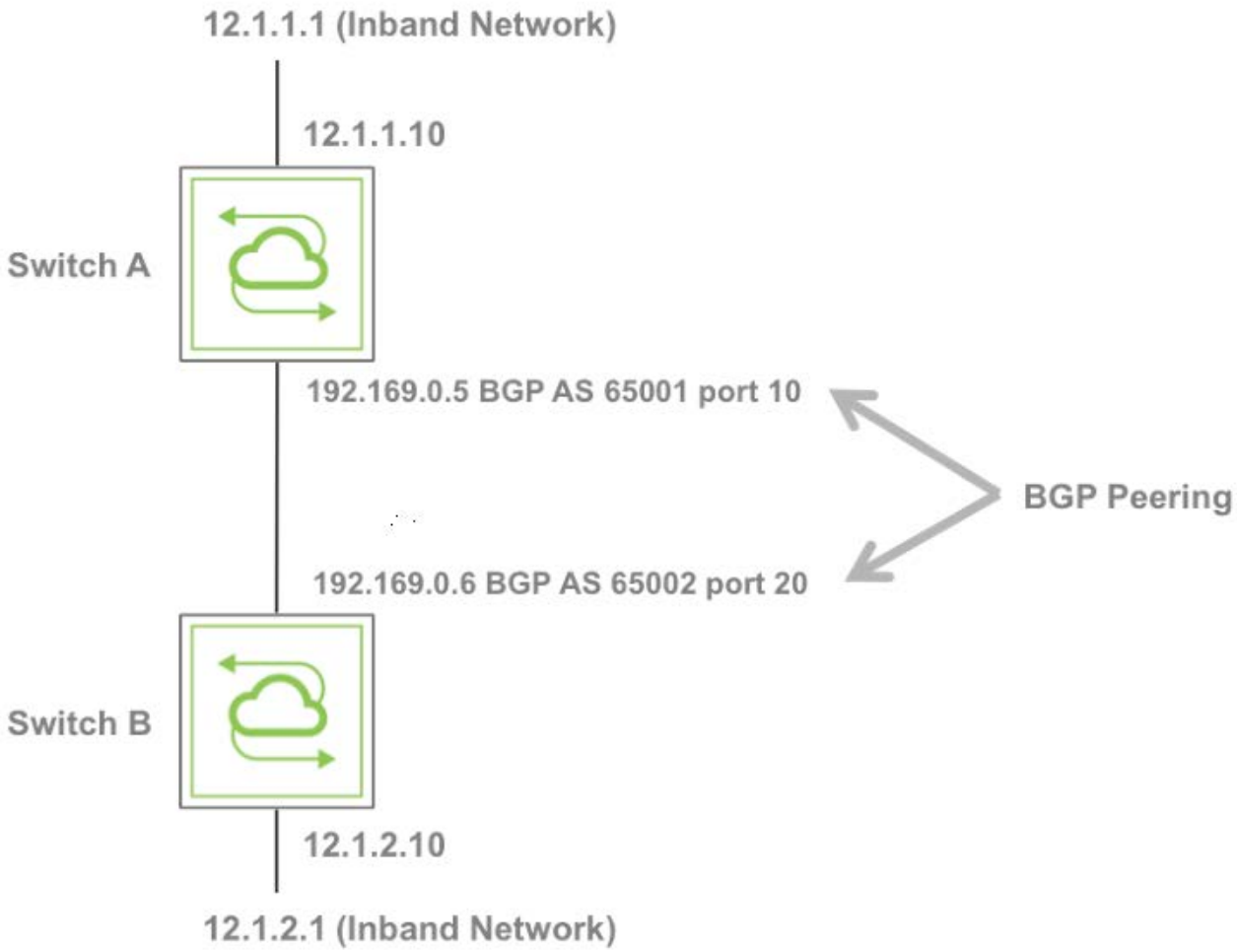


Figure 6-8: Example Network of Two BGP Neighbors Topology

To achieve the BGP-based configuration and establish fabric communication between the two switches, you must perform the following steps:

1) Modify the in-band IP in the same subnet of in-band-nic-ip address.
2) Create a vRouter on each switch.

3) Setup BGP peering and redistribute routes.
4) Setup up global static routes using the respective vRouter as the gateway. You must perform this step because the routes reside inside the vRouter table and each switch must have an appropriate route to reach the in-band network of another switch.
5) Add global static route on switch B to reach switch A in-band network with switch B in-band-nic-ip as the gateway IP address.
6) Then you can configure switch B to join the fabric previously created by switch A.

Use the following commands to accomplish the necessary configuration:

Creating a vRouter for Fabric Communication

Use the `fabric-comm-vrouter-bgp-create` command to create a vRouter for fabric communication.

This command has numerous options including the following:

`name`	Name of the fabric communication vRouter.
`bgp-as`	BGP Autonomous System number of the vrouter from 1 to 4294967295.
`bgp-nic-vlan`	Interface VLAN number between 0 and 4095. You can enable BGP over VLAN interface also.
`bgp-redistribute`	BGP route redistribution. The default value is connected.
`bgp-max-paths`	Maximum BGP paths.
`bgp-ibgp-multipath`	Number of IBGP multipath connections.
`bgp-nic-ip`	IP address of the peering interface.
`bgp-nic-netmask netmask`	BGP NIC netmask.
`bgp-nic-linklocal ip-address`	IPv6 link local address.
`bgp-nic-vnet`	Interface VLAN vNET.
`bgp-nic-bd`	Interface bridge domain.
`bgp-nic-vlan-type`	Interface VLAN type.
`bgp-nic-l3-port`	Layer 3 port
`bgp-nic-mtu`	Interface MTU
`bgp-nic-if-nat-realm`	NAT interface realm
`in-band-nic-ip`	IP address range of the in-band management interface.
`remote-as`	BGP remote AS number from 1 to 4294967295
`neighbor`	IP address of the BGP neighbor.

The BGP protocol reserves Autonomous System (AS) numbers 64512–65534 for private use and the example below uses some of the private AS numbers.

This dedicated CLI performs the majority of the actions in the logical steps 1-4 described in the *Configuring a Fabric Over a Layer 3 Network* section above. However, it does not change the inband IP address or create the fabric.

After creating the fabric, the root switch, Switch A, does not have to specify the fabric network in the switch configuration, but the switch must create a route to the other inband network on the switch with the `fabric-in-band-network-create` command.

Switch B, a non-root switch, does need to specify the fabric network by using the `fabric-network` parameter:

See an example on how to establish fabric communication between two switches (switch A and switch B) detailed in steps 1 through 6 above.

Switch A

```
CLI (network-admin@switch) > switch-setup-modify in-band-ip 12.1.1.1/24

CLI (network-admin@switch) > fabric-comm-vrouter-bgp-create name vrouter-a
bgp-nic-l3-port 10 bgp-as 65001 bgp-nic-ip 192.169.0.5/30 in-band-nic-ip
12.1.1.10/24 remote-as 65002 neighbor 192.169.0.6 bgp-redistribute
connected fabric-network 12.1.1.0/24

CLI (network-admin@switch) > fabric-in-band-network-create network
12.1.2.1/24
```

Switch B

```
CLI (network-admin@switch) > switch-setup-modify in-band-ip 12.1.2.1/24

CLI (network-admin@switch) > fabric-comm-vrouter-bgp-create name vrouter-b
bgp-nic-l3-port 20 bgp-as 65002 bgp-nic-ip 192.169.0.6/30 in-band-nic-ip
12.1.2.10/24 remote-as 65001 neighbor 192.169.0.5 bgp-redistribute
connected fabric network 12.1.2.0/24

CLI (network-admin@switch) > switch-route-create network 12.1.1.0/24
gateway-ip 12.1.2.10

CLI (network-admin@switch) > fabric-join switch-ip 12.1.1.1

CLI (network-admin@switch) > Joined fabric myfabric. Restarting nvOS...
```

Connecting Multiple Fabric Nodes over a Layer 3 Fabric

In a topology with more than two switches configured for BGP, the first peer for each switch can be created using the `fabric-comm-vrouter-bgp-create` command. You must create the other switches separately.

See ***Figure 6-9*** where a third switch, Switch C, is added to the previous two-switch topology.

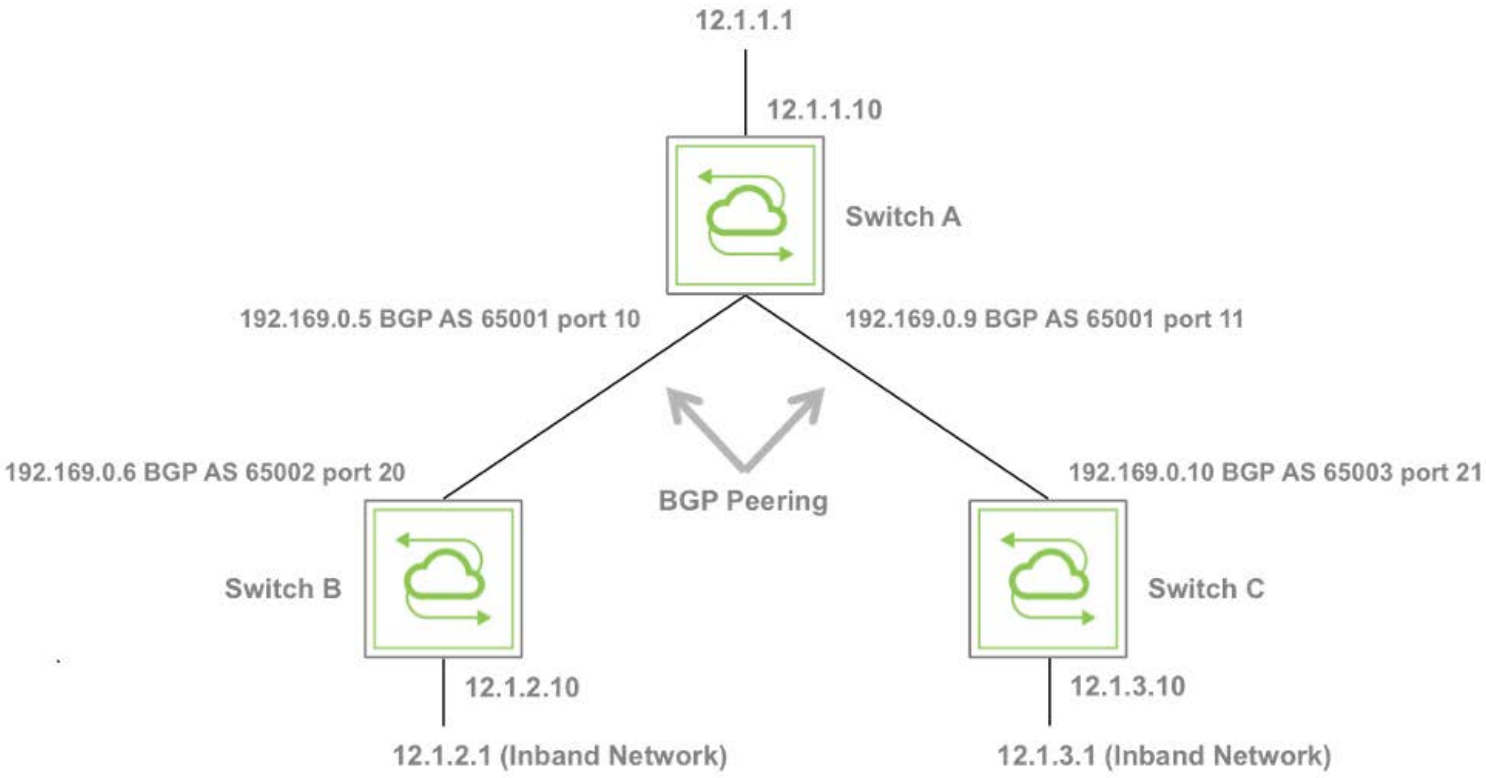


Figure 6-9: An Example of Fabric over Layer 3 Topology

The configuration for this example follows the same scheme as the two switch topology. However, NetVisor OS requires the following additional steps:

1) On Switch A, configure BGP peering with Switch C and create a static route to reach the inband network on Switch C. To extend this configuration with more than three-switch example, use this step on every switch with more than one BGP peer.

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrouter-a
ip 192.169.0.9/30 l3-port 11

CLI (network-admin@switch) > vrouter-bgp-add vrouter-name vrouter-a
neighbor 192.169.0.10 remote-as 65003

CLI (network-admin@switch) > fabric-in-band-network-create network
12.1.3.0/30
```

2) On Switch C, use the `fabric-comm-vrouter-bgp-create` command and connect:

```
CLI (network-admin@switch) > switch-setup-modify in-band-ip 12.1.3.1/24

CLI (network-admin@switch) > fabric-comm-vrouter-bgp-create name vrouter-c
bgp-nic-l3-port 21 bgp-as 65003 bgpnic-ip 192.169.0.10/30 in-band-nic-ip
12.1.3.10/24 remote-as 65001neighbor 192.169.0.9 fabric-network 12.1.1.0/24
bgp-redistribute connected fabric-network 12.1.3.0/24

CLI (network-admin@switch) > switch-route-create network 12.1.1.0/24
gateway-ip 12.1.3.10

CLI (network-admin@switch) > switch-route-create network 12.1.2.0/24
gateway-ip 12.1.3.10

CLI (network-admin@switch) > switch-route-create network 12.1.3.0/24
gateway-ip 12.1.2.10

CLI (network-admin@switch) > fabric-join switch-ip 12.1.1.1

Joined fabric myfabric. Restarting nvOS...
```

Limiting the Transaction Log File Size

In releases prior to Netvsior ONE version 6.1.0, Arista had not restricted the size of the transaction log files, due to which the transaction files increased in size occasionally on some customer premises, causing memory issues.

This issue occurs when some commands such as `stp-port-modify` and `transaction-settings-modify` commands are executed in a loop. These commands do not cause any changes to the configuration after the command is executed for the first time, but the transaction log file is appended every time the command is executed adding duplicate entries and causing the file to grow.

Starting with NetVisor OS version 6.1.0, this limitation is corrected by enabling you to limit the file size of the transaction logs. You can truncate the transaction log file to a desired number of entries by using the `transactions-settings-modify` command depending upon your configuration. By default, this functionality is disabled.

Depending upon the limit you set (for example *N* entries), the transaction log file limits the log entries to the latest *N* entries. Further, NetVisor OS adds an additional *X* entries to the threshold limit of N entries you have configured. When the threshold of *N+X* limit is crossed, the first *X* entries gets overwritten by the newest or latest entries.

This functionality is supported only on *local xact log* file (`local` scoped configurations).

Warning: If you set the transaction limit (*N*), you may lose the older entries in the transaction log file and hinder the rollback possibility. However, this feature is available only for local transaction log files.

Note: If you do not set the threshold limit, then this functionality remains disabled and there is no limit enforced on the log file size.

To restrict the transaction log file size, you can set a limit by using the `transactions-settings-modify` command:

```
CLI (network-admin@switch) > transactions-settings-modify max-local-xact-log-entries max-local-xact-log-entries-number
```

Where, the `max-local-xact-log-entries` is the maximum number of entries (*N*) to be added to the transaction log file (`xact_log` file).

Note: The default value of `max-local-xact-log-entries` is 0, which means, the feature is disabled.

As explained earlier, an additional *X* entries are saved by NetVisor OS before truncating the log file size. NetVisor OS has set the value of *X* to 100 entries. That is, for example, if you set the `max-local-xact-log-entries` to `500` entries, then an additional `100` entries are also saved (see *N+X* explanation above) to the log file before NetVisor enforces the truncation process of the `local-xact-log` file. When the threshold is crossed, the file gets truncated by deleting the previously stored additional *X* (100) entries.

A warning message is written to the *nvOSd.log/perror.log* file:

- When the entries in the transaction log file exceeds the configured value (*N*).
- When the entries exceed the *N+X* value and NetVisor truncates the *X* entries.

To truncate the `local xact log file` to a given `tid` (*N*), use the command:

```
CLI (network-admin@switch) > transaction-log-file-truncate-to trunc_tid
trunc_tid-number
```

Where the `trunc_tid` is the transaction ID (the latest `tid` entries are retained when this feature is enabled)

> **Note**: All truncated commands are written to a new file `/nvOS/log/xact-truncate.log.`

To view the total entries in the file, use the command:

```
CLI (network-admin@switch) > transaction-show transaction-scope local
count-only
```

Where, `count-only` is the formatting option to display only the number of transaction entries.

General Guidelines while Configuring this Feature:

- Only the transactions in `transaction-show transaction-scope local` output and in `/var/nvOS/etc/Local/xact_config.log` file are removed. All the configurations remain intact.

- Transaction rollback is possible even after the log file is truncated to an available TID. Note that the TIDs get renumbered after truncation.

Sample Configuration

Below is a sample configuration for further understanding:

- First view the transaction details using the command:

```
CLI (network-admin@switch) > transaction-show transaction-scope local count-only
Count: 3
```

- Use the below command to view the transaction details:

```
CLI (network-admin@switch) > transaction-show transaction-scope local
start-time:     2021-03-04,16:07:58.030
tid:            1
trunc_tid:      0
state:          commit
command:         switch-setup-modify eula-accepted true eula_timestamp 2021-
03-04,16:07:57
undo-command:   switch-setup-modify eula-accepted false eula_timestamp 1969-
12-31,16:00:00
----------------------------------------
```

```
start-time:     2021-03-09,06:05:23.932
tid:            2
trunc_tid:      0
state:          commit
command:         stp-port-modify port 31 edge
undo-command:   stp-port-modify port 31 no-edge
----------------------------------------
start-time:     2021-03-12,01:56:30.020
tid:            3
trunc_tid:      0
state:          commit
command:         vlan-create id 3040 type public vxlan-mode standard scope
local stats
undo-command:   vlan-delete id 3040
```

- Use the command to set the max-local-xact-log-entries to 50:

```
CLI (network-admin@switch) > transactions-settings-modify max-local-xact-
log-entries 50
```

View the details using the command:

```
CLI (network-admin@switch) > transaction-show transaction-scope local count-only
Count: 150
```

The count 150 indicates the additional 100 (*N+X*).

You can verify the transactions using REST API commands as well.

Troubleshooting the Fabric

Displaying the Transaction History

When configuring and administering the fabric, you can display the list of executed transactions by using the `transaction-show` command. For example in this case committed transaction 1 corresponds to a command in which a vRouter was created:

```
CLI (network-admin@switch) > transaction-show

start-time:     2020-05-14,15:38:34.622
tid:            1
state:          commit
command:        vrouter-create id b001b66:0 name vr1 scope fabric vnet test-fab-2-global
dedicated-vnet-service  locati
on 1 storage-pool rpool zone-id b001b66:1 router-type hardware router-ipstack frr hw-router-
mac 66:0e:94:66:b3:0f clu
ster-active-active-routing hw-vrid 0 hw-vrrp-id -1 bgp-redist-static-metric none bgp-redist-
connected-metric none bgp
-redist-rip-metric none bgp-redist-ospf-metric none bgp-max-paths 1 bgp-delayed-startup 1 bgp-
update-delay-strict 0 b
gp-keepalive-interval 60 bgp-holdtime 180 bgp-global-nh-preference true ospf-redist-static-
metric none ospf-redist-st
atic-metric-type 2 ospf-redist-connected-metric none ospf-redist-connected-metric-type 2 ospf-
redist-rip-metric none
ospf-redist-rip-metric-type 2 ospf-redist-bgp-metric none ospf-redist-bgp-metric-type 2 ospf-
default-information none
 ospf-default-info-originate-metric none ospf-default-info-originate-metric-type 2
undo-command:   vrouter-delete name vr1
----------------------------------------
start-time:     2020-05-14,15:39:02.820
tid:            2
state:          commit
command:        vrouter-interface-add vrouter-name vr1 nic eth0.4092 ip 10.0.120.2 netmask 24
mac 66:0e:94:66:b3:0f vl
an 4092 vlan-type public public-vlan 4092 no-exclusive nic-enable no-pim pim-dr-priority 1 no-
pim-cluster  no-fabric-n
ic vrrp-priority 100 vrrp-adv-int 1000 l3-port 45 mtu 1500 no-sriov-vf mirror-traffic false
no-priority-tag
undo-command:   vrouter-interface-remove vrouter-name vr1 nic eth0.4092
```

The NetVisor OS transaction log file contains the list of commands corresponding to the executed transactions as well as the undo command(s) required to undo each transaction. Commands are shown in the `transaction-show` output for *informational purposes* only.

Each transaction can have a *local*, *cluster* or *fabric* scope. For example acceptance of the EULA is a transaction that is local to each device:

```
CLI (network-admin@switch) > transaction-show transaction-scope local

start-time:     2019-06-02,21:50:35.497
tid:            1
state:          commit
command:        switch-setup-modify eula-accepted true eula_timestamp 2019-06-02,21:50:35
undo-command:   switch-setup-modify eula-accepted false eula_timestamp 1969-12-31,16:00:00
```

On the other hand, for instance transactions of VLAN creation and deletion are usually (but not necessarily

always) executed with a fabric scope. Other objects such as vNETs are created and deleted with a fabric scope too, as shown in the example below:

```
CLI (network-admin@switch) > transaction-show transaction-scope fabric

start-time:     2019-06-02,22:10:54.895
tid:            1
state:          remote-commit
command:        vlan-create id 100 type public vxlan-mode standard scope fabric stats
undo-command:   vlan-delete id 100
----------------------------------------

start-time:     2019-06-02,22:11:25.834
tid:            2
state:          remote-commit
command:        vnet-create id c00025b:1 name vn1 scope fabric vrg c00025b:0 vlan-type public
vlans 5 config-admin create-vnet-mgr vnet-mgr-storage-pool rpool vnet_mgr_id c00025b:0
vnet_mgr_zone_id c00025b:1 vnet_mgr_location 1 vrg_created_by_vnet true admin_role c00025b:400
undo-command:   vnet-delete name vn1
----------------------------------------

start-time:     2019-06-02,22:11:36.907
tid:            3
state:          remote-commit
command:        vlan-delete id 100
undo-command:   vlan-create id 100 type public hw-vpn 0 hw-mcast-group 0 replicators "" repl-
vtep :: in_hw false public-vlan 100 user_public_id 0 public_id_set_by_user false scope fabric
description vlan-100 label "vlan 100" active yes stats uses_refcnt no refcnt 0 vrg 0:0 ports
1-69 untagged-ports none send-ports 21,23,31-34,53,57,61,65,69,272-275 active-edge-ports 272
non-auto-ports none ports-specified false initialized true flags ""
----------------------------------------

start-time:     2019-06-02,22:11:51.550
tid:            4
state:          remote-commit
command:        vnet-delete name vn1
undo-command:   vnet-create id c00025b:1 name vn1 scope fabric vrg c00025b:0 vlan-type public
vlans 5 config-admin admin 40000 create-vnet-mgr vnet-mgr-name vn1-mgr vnet-mgr-storage-pool
rpool vnet_mgr_id c00025b:0 vnet_mgr_zone_id c00025b:1 vnet_mgr_location 1 vrg_created_by_vnet
true global false allow_admin_access false admin_role c00025b:400
```

In case of clusters (that is, pairs of redundant devices described in more detail later in this guide) certain transactions are applied to both nodes as if they were a single logical device.

You can choose the transaction information format for a node by typing the `transaction-show` command followed by the `format` parameter to choose a tabular or a vertical output. For example:

```
CLI (network-admin@switch) > transaction-show format all layout vertical

start-time:    03-19,13:46:42
end-time:      03-19,13:46:43
scope:             fabric
tid:               33
state:             remote-commit
command:           --unrecoverable-- vlan-delete id 22
undo-command: --unrecoverable-- vlan-create id 22 nvid a000030:16 scope fabric name vlan-22
active yes stats vrg 0:0 ports 1-72,128-129,255 untagged-ports none send-ports 31,41,47-
48,51,65-66 active-edge-ports none ports-specified false flags
```

```
----------------------------------------

start-time:    09:36:09
end-time:      09:36:09
scope:             fabric
tid:         34
state:             remote-commit
command:           vlan-create id 35 scope fabric stats ports-specified true
```

Please note that, in this command's output, the `scope` parameter indicates which set of transactions are displayed, as each scope has an independent set of transactions associated with it. As shown in this example, NetVisor OS uses `fabric` as the *default scope* unless another scope is specified.

Also note that you should not copy and paste commands and undo-commands from this output because they may include information that does not apply to a different context. As a matter of fact, as mentioned earlier, NetVisor OS displays the fields for *informational purposes* only so as to allow you to see exactly what happens to the configuration when you roll forward or roll back the transaction ID.

Once you decide which node to modify and the transaction to roll forward or roll back to, you can use the `transaction-rollforward-to` or `transaction-rollback-to` commands to re-run the command (roll forward) or undo the command (roll back) on the node being *troubleshooted*, or even in a broader scope, as described in the following sections.

Keeping Transactions in Sync with Auto-Recovery

When transactions are executed with a fabric scope, they get applied to all nodes that are part of the same fabric instance as the local node. This process requires coordination and synchronization across the nodes.

You can display the ID of the last executed transaction by using the `fabric-node-show` command:

```
CLI (network-admin@pnswitch1) > fabric-node-show format name,fab-name,fab-tid,state,device-state,

 name      fab-name   fab-tid  state   device-state
--------- ---------- -------  ------  ------------
pnswitch2  pnfabric    2       online    ok
pnswitch1  pnfabric    2       online    ok
```

The ID is the same because the fabric has made sure that the same transaction be executed consistently on all online nodes.

What happens, though, if one of the nodes is temporarily offline and gets out of sync?

NetVisor OS's fabric synchronization logic possesses an auto-recovery function (called auto-recover) that makes sure that the transactions get automatically resynchronized when they accidentally go out of sync. This capability is enabled by default:

```
CLI (network-admin@pnswitch1) > transaction-settings-show

switch:                             pnswitch1
allow-offline-cluster-nodes:        on
auto-recover:                       on
auto-recover-retry-time:            5m
reserve-retry-maximum:              10
reserve-retry-interval-maximum(s):  8
```

In special scenarios, this capability can be turned off *on purpose* to force the configuration of a specific node to be rolled back to a previous state:

```
CLI (network-admin@pnswitch1) > transaction-settings-modify no-auto-recover

CLI (network-admin@pnswitch1) > transaction-settings-show
switch:                             pnswitch1
allow-offline-cluster-nodes:        on
auto-recover:                       off
auto-recover-retry-time:            5m
reserve-retry-maximum:              10
reserve-retry-interval-maximum(s):  8

switch:                             pnswitch2
allow-offline-cluster-nodes:        on
auto-recover:                       on
auto-recover-retry-time:            5m
reserve-retry-maximum:              10
reserve-retry-interval-maximum(s):  8
```

However, this change is local to the device where the command was issued. Therefore, in those cases in which it is necessary to apply this change to all nodes in the fabric, the `switch *` directive as shown in the

following command can be used:

```
CLI (network-admin@pnswitch1) > switch * transaction-settings-modify no-auto-recover

CLI (network-admin@pnswitch1) > transaction-settings-show
switch:                             pnswitch1
allow-offline-cluster-nodes:        on
auto-recover:                       off
auto-recover-retry-time:            5m
reserve-retry-maximum:              10
reserve-retry-interval-maximum(s):  8

switch:                             pnswitch2
allow-offline-cluster-nodes:        on
auto-recover:                       off
auto-recover-retry-time:            5m
reserve-retry-maximum:              10
reserve-retry-interval-maximum(s):  8
```

Rolling Back and Rolling Forward Transactions

NetVisor OS maintains a log file with the list of transactions with their respective *undo commands* to be able to revert back, when necessary, to a previous state, that is, to *roll back* one or more transactions starting from the latest one. On the other hand, the list of executed commands can be used to redo certain transactions, in other words to *roll forward* one or more transactions.

However, this is only desirable under special circumstances, because the *auto-recover* feature by default automatically makes sure that all nodes are synchronized to the latest transaction.

For example in case of rare conditions in which transactions diverge on different nodes (despite auto-recover), a roll back or roll forward action may be performed manually through the corresponding command.
However, the auto-recover function may need to be *temporarily disabled* on the affected node(s) to permit the desired action.

The `transaction-rollback-to` command is used to roll back to an earlier fabric transaction number. The `transaction-rollforward-to` command is instead used to roll forward to a subsequent fabric transaction number.

For instance, the fabric state gets accidentally out of sync according to the `fabric-node-show` command output with, say, a missing interface addition transaction:

```
CLI (network-admin@pnswitch1) > fabric-node-show format name,fab-name,fab-tid,state,device-state,

name      fab-name   fab-tid  state  device-state
--------- ---------- -------  ------ ------------
pnswitch2  pnfabric  1        online  ok
pnswitch1  pnfabric  2        online  ok
```

Hence the state can be rolled back to a previously synced ID to restore fabric-wide (`scope fabric`) consistency:

```
CLI (network-admin@pnswitch1) > transaction-rollback-to scope fabric tid 1
Warning: rolled back transactions are unrecoverable unless another fabric node has them. Proceed? [y/n] y
```

After successfully rolling back the transaction (i.e., no error message is printed on the console), the change completes and the transaction is removed from the transaction log.

Alternatively the state can be rolled forward to reattempt to successfully redo the previously failed fabric-wide interface addition:

```
CLI (network-admin@pnswitch1) > transaction-rollforward-to scope fabric tid 2
Added interface eth2.13
```

After successfully rolling forward a transaction (i.e., no error is printed on the console), the change completes and the transaction log is updated.

If multiple nodes go out of sync, you must *recover each node separately*.

An alternative approach (*usually reserved to customer support for special cases*) is to try to force a roll back or roll forward action when the configuration is in sync but the transaction ID fails to get updated:

```
CLI (network-admin@pnswitch1) > transaction-rollforward-to scope fabric tid
2 ignore-error
Added interface eth2.13
```

When a node is out of sync with the other nodes in the fabric, it can catch up by rolling forward all the missing transactions, which it can obtain from another fabric node. If auto-recovery is enabled, this is done without user intervention. If auto-recovery is disabled, the `transaction-rollforward-to` command can be invoked.

Starting from NetVisor OS release 7.0.0, a round-robin enhancement to fabric synchronization has been implemented that helps in dealing with connection failures. The software attempts to contact the best possible node, which is the one that has the highest TID amongst all the peer nodes in the fabric (unless an explicit `remote-node` is specified in the CLI to be used). If that attempt fails, instead of retrying with the same node, NetVisor OS skips any node that it had previously attempted and failed to sync up with.

In other words, when auto-recovery is enabled, the failed nodes are skipped in every resynchronization attempt until all nodes have been tried (Offline nodes are skipped as well). Similarly, when auto-recovery is disabled and a (manual) attempt to roll-forward fails with a node, in the next iteration any failed node(s) is/are skipped. Once a synchronization attempt has failed with all the nodes in the fabric, the software restarts from the first node in a round-robin fashion.

In release 7.0.0 a new column is added to the `fabric-node-show` command output to display the roll-forward status of the fabric peer node(s):

```
CLI (network-admin@switch) > fabric-node-show format rollforward-failed,

rollforward-failed
------------------
false
```

In the rare cases when you apply a configuration to the fabric and a node does not respond to such configuration, as a last resort, you may want to evict the node from the fabric to troubleshoot the problem on the specific device.

To evict a node, that node must be offline, otherwise the eviction command will fail. Then you can use the `fabric-node-evict` command to perform the eviction process like so:

```
CLI (network-admin@switch) > fabric-node-evict name pnswitch2
```

or

```
CLI (network-admin@switch) > fabric-node-evict id b000021:52a1b620
```

Rolling Back the Configuration of the Fabric

In addition to troubleshooting rare conditions, the roll back action can be used in certain circumstances in which it is beneficial to *turn back time* on all the nodes in the same fabric instance.
As discussed in earlier sections, the fabric natively (i.e., by default) stays *synchronized automatically to the last transaction executed*. Therefore, in order to revert all switches to a previous state, it is necessary to temporarily disable this automatic function (auto-recover) like so:

```
CLI (network-admin@pnswitch1) > switch * transaction-settings-modify no-
auto-recover
```

Then it is possible to perform a roll back action to a certain ID on all those switches. You can check the latest transaction ID with the command:

```
CLI (network-admin@pnswitch1) > fabric-node-show format name,fab-name,fab-
tid,state,device-state,

name         fab-name    fab-tid state  device-state
-----------  ----------- ------- ------ ------------
pnswitch1    myfabric    1533    online ok
pnswitch2    myfabric    1533    online ok
```

Let us say you want to roll back all nodes to transaction ID 1530. You can do that with the following command:

```
CLI (network-admin@pnswitch1) > switch * transaction-rollback-to
transaction-scope fabric tid 1530
Warning: rolled back transactions are unrecoverable unless another fabric
node has them.
Please confirm y/n (Default: n):y
CLI (network-admin@pnswitch1) > fabric-node-show format name,fab-name,fab-
tid,state,device-state,

name         fab-name    fab-tid state  device-state
-----------  ----------- ------- ------ ------------
pnswitch1    myfabric    1530    online ok
pnswitch2    myfabric    1530    online ok
```

In this case, though, rolled back transactions are unrecoverable because the fabric logic is forced to *forget* about them on a per node basis where auto-recover was disabled. (So no roll forward is then possible).

Once all the nodes have been rolled back to the chosen transaction, you can re-enable auto-recover to make sure that the fabric stays in sync automatically from that moment onward:

```
CLI (network-admin@pnswitch1) > switch * transaction-settings-modify auto-
recover
```

How to Fix the Cluster Transaction Divergence Issue

When a cluster transaction divergence occur, you cannot execute any further cluster-scoped configurations on either of the cluster nodes. With the release of NetVisor OS version 5.1.2, we provide a solution to fix the cluster transaction divergence issue.

To ensure high availability, NetVisor OS allows cluster scope transactions to proceed even when one of the peer nodes in a cluster goes offline (enabled, by default, by using the command: `transaction-settings-modify` with `allow-offline-cluster-nodes` parameter; which can be disabled, you can disable this option, if required) during configuration changes. However, this may cause a transaction divergence as explained in ***Figure 6-10***. For example, consider a cluster with two nodes- node 1 and node2, both are online and configurations commands are executed on both nodes, and the transactions are synchronized until a certain time where the TID=T. After certain time, node2 goes offline, but transactions continue on node1 and the TID on node1 changes to T1. Now, both the nodes go offline and then node2 comes back online, and transactions continue on node2, where the TID changes to T2. Now, when node1 also comes back online, the transactions get diverged as illustrated in ***Figure 6-10***.

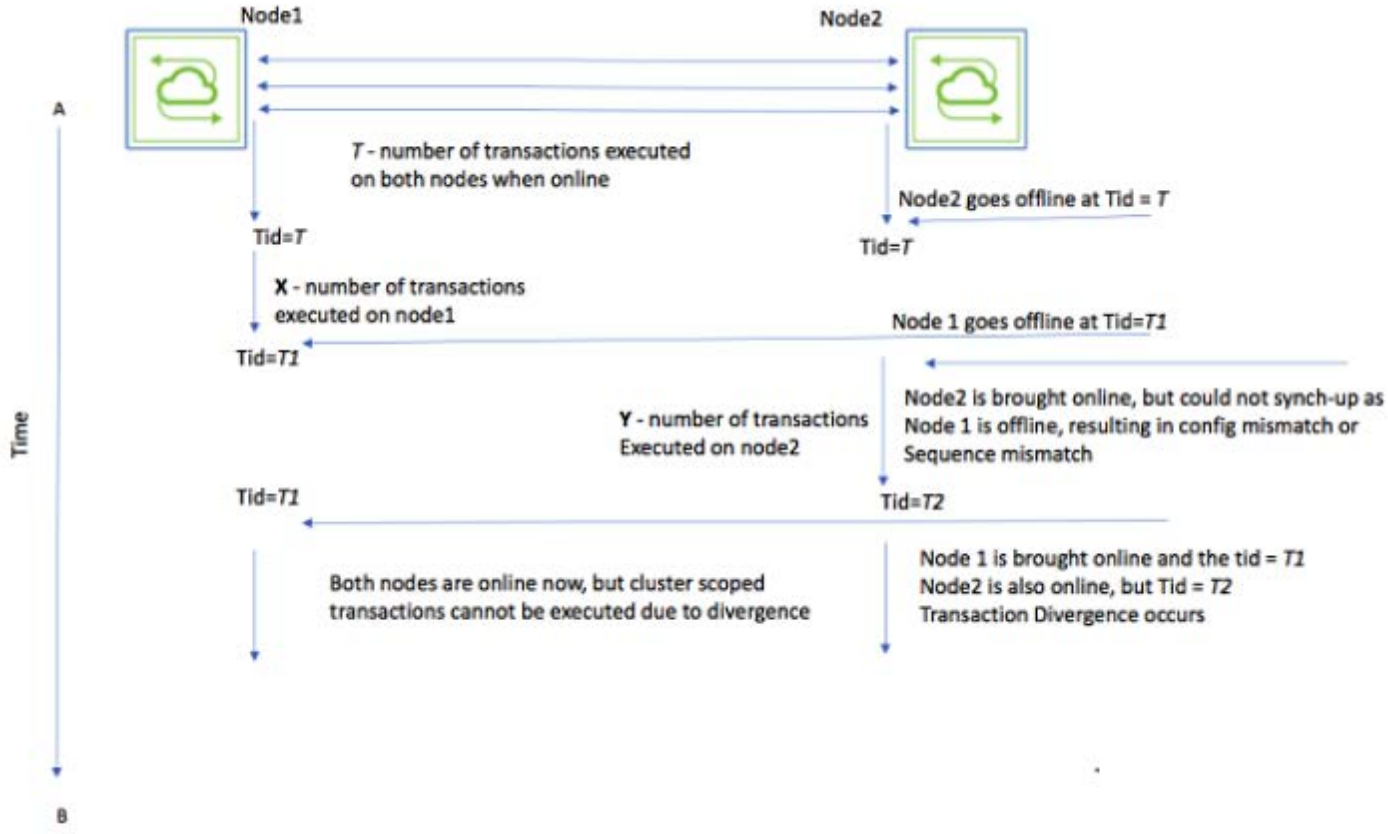


Figure 6-10: Cluster Transaction Divergence

The two possible transaction divergence cases as illustrated in ***Figure 6-10*** include:

- Case 1: Even if X number of transactions and Y number of transactions are the same and is also executed in the same sequence (T1 = T2), still there will be transaction divergence due to the change in cluster-change IDs.
- Case 2: If the X and Y transactions on the nodes are a separate (different) set of commands or is executed in different sequence, then also transaction divergence occurs.

The auto-recovery feature, which is enabled by default on NetVisor OS, automatically rolls forward the transactions with the peers when the offline cluster node is brought up online when there is no transaction divergence. But, in the case of transaction divergence, you have to manually roll-back to a common transaction point for the auto-recovery feature to synchronize (for details, see the previous sections of *Troubleshooting the Fabric* section) . As this is a manual process, it is error prone (because if you mistakenly roll-back to a different transaction point, then you have to repeat the process to get to the common transaction point) .

To mitigate this issue, NetVisor OS introduces a new command: `transaction-cluster-divergence-fix`. You can run this `local-scoped` command on any of the cluster nodes. However, Arista recommends to run this command on a slave node where the divergence occurred.

In case 1 above, when you run the `transaction-cluster-divergence-fix`. command on a node, NetVisor OS updates the change ID s of all diverged transaction IDs to match with the peer node.

In case 2 above, when you run the command on a node, NetVisor fetches the transaction details from the cluster peer and rolls-back the configuration to a common transaction ID on the node on which the command is executed. Then, the node re-synchronizes the configuration with the cluster peer configuration if the auto-recovery function (called auto-recover) that makes sure that the transactions get automatically re-synchronized is set to *ON (*This capability is enabled by default). For details on the auto-recovery function, see the Keeping Transactions in Sync with Auto-Recovery section.

> **Note**: It is recommended to take a backup of the current configuration before issuing the `transaction-cluster-divergence-fix` command. Ensure that the fabric communication between the nodes in the cluster is enabled for this functionality to work.

To fix the cluster transaction divergence issue, use the command on the failed cluster node (recommended to run on a slave node):

```
CLI (network-admin@node_slave) > transaction-cluster-divergence-fix
Warning: this will download config from cluster peer and rollback to tid
before it is diverged. It is recommended to take config backup before
running this command.
Please confirm y/n (Default: n):y
```

Related Commands

```
CLI (network-admin@pn-test-2) > transaction-settings-modify
```

Specify one or more of the following options:	
`allow-offline-cluster-nodes|no-allow-offline-cluster-nodes`	Specify to allow transactions to proceed even if a cluster node is offline. By default, the `allow-offline-cluster-nodes` parameter is enabled on nvOS.
`auto-recover|no-auto-recover`	Specify to automatically recover missed transactions.
`auto-recover-retry-time duration: #d#h#m#s`	Specify the retry time for transaction to auto-recover.

`reserve-retry-maximum reserve-retry-maximum-number`	Specify the maximum number of retires for transaction reservation; 0 for infinite.
`reserve-retry-interval-maximum reserve-retry-interval-maximum-number(s)`	Specify the maximum number of seconds to wait between transaction reservation retries.

About the Fabric Default Parameters

The following is a list of NetVisor default port numbers, multicast addresses, and communication information about the fabric:

Internal keepalive message
- Multicast IP: 239.4.9.7
- UDP Destination Port: 23399
- NetVisor sends the packet from the CPU to the internal port to ensure that the CPU path to the switch is working and the internal port is up.

Fabric keepalive message
- UDP Destination Port: 23394
- Point to point UDP fabric keepalives: if NetVisor cannot receive these messages, the fabric node may go into an offline state.

Global discovery message
- Multicast IP: 239.4.9.3
- UDP destination port: 23399
- Each node periodically multi-casts a message about the fabric. This enables the `fabric-show` command on directly connected nodes, Layer 2-adjacent, to display available fabric instances and also enables the `fabric-join name name-string` command.

Proxy commands
- TCP Destination Port: 23397 SSL
- Used for NetVisor OS-to-NetVisor OS communication. Used for internal purposes and also to implement commands executed on other switches from a local switch.

Status propagation
- TCP Destination Port: 23398 SSL
- Port changes and vport changes propagated to other nodes in the fabric.

TCP API clients
- TCP Destination Port: 23396 SSL
- API clients connect to this port. Disable using the `admin-service-modify if <mgmt/data> no-net-api` command.

File System replication
- TCP Destination Port: 23392 SSL
- For ZFS send and ZFS receive messages when replicating file systems across the fabric.

L2 ARP/DMAC miss/Broadcast encapsulation
- UDP Destination Port: 23389
- These are special VXLAN-encapsulated packets sent from one management CPU to another management CPU between two Layer 2 adjacent switches.

L3 ARP/DMAC miss/Broadcast encapsulation
- UDP Destination Port: 23388
- These are special VXLAN-encapsulated packets sent from one management CPU to another management CPU between two switches interconnected at Layer 3.

vPORT status

- Multicast IP: 239.4.9.4
- UDP Destination Port: 23390
- vPort update messages associated to status changes to hypervisors or hosts in the fabric.

vFlow CPU packets

- UDP Destination Port: 23398
- These packets are sent point-to-point for vflow-snoop of a fabric-scoped vFlow

> **Note:** All of these messages must not drop or become lost in order to keep a fabric healthy. However, the multicast messages do not propagate through routers and therefore NetVisor OS does not rely on them for fabrics created over Layer 3 networks.

The `fabric-node-show` command displays information about internal data structures for each node in the fabric, including the node state.

For example:

```
CLI (network-admin@switch) > fabric-node-show

fab-name   mgmt-ip       in-band-ip  in-band-vlan-type fab-tid  version          state
-------- -------------- ----------- ----------------- ------- -------------     ------
WDTest   10.36.10.45/24 10.0.1.1/24   public            7       5.0.0-5000014540  online
```

If NetVisor OS does not receive any keepalives or other messages from a fabric node for about 30 seconds, then NetVisor marks the node as `offline`.

Anything that prevents keepalives or other kinds of messages from flowing freely between fabric nodes can cause problems for fabric connectivity.

Supported Releases

Command/Parameter	NetVisor OS Version
`fabric-create` `fabric-show`	Commands added in *version 1.2*
`fabric-network` `control-network`	Parameters added in *version 2.4*
`fabric-advertisement-network`	Parameter added in *version 2.4.1*
`fabric-node-show` `fabric-join`	Commands introduced in *version 1.2*
`fabric-info` `fabric-unjoin`	Commands introduced in *version 1.2.1*
`transaction-show`	Command introduced in *version 2.2.3*
`fabric-comm-vrouter-bgp-create`	Command introduced in *version 2.4.1*
`fabric-comm-ports-show` `fabric-comm-ports-modify`	Commands introduced in *version 3.0.0*
`rollforward-failed`	Column added in the `fabric-node-show format all` output in version 7.0.0

Please also refer to the *Arista Command Reference* document.

Related Documentation

Refer to these sections of the Configuration Guides:

- *Implementing a Fabric Upgrade*
- *Configuring Layer 2 Features*
- *Configuring Layer 3 Features*
- *Configuring High Availability*
- *Configuring Virtual Networks (vNETs)*

for further information on concepts mentioned in this section (such as Layer 2, Layer 3, etc.)

Configuring High Availability

This chapter provides information about the High Availability Feature in NetVisor OS.

Understanding the High Availability Feature in NetVisor OS

High availability (HA) refers to the characteristic of a system (an individual device or a group of devices) to ensure that service availability (i.e., uptime) is higher than normal and hence downtime is minimized. In case of networking, this is achieved by increasing the resilience (i.e., fault tolerance) of the network so as to be able to provide and maintain an acceptable level of service during faults and challenges to normal operation.

High availability is generally achieved through multiple strategies that include:

- Redundancy of device components (such as power supplies and fans) to maximize the reliability of individual network nodes.
- Device and link redundancy to maximize end-to-end service continuity by eliminating single points of failure and guaranteeing fast re-convergence.
- Simplified operation as well as network automation and orchestration to minimize potential service-disrupting human errors.

Arista's Unified Cloud Fabric control plane provides intuitive and less error-prone mechanisms to automatically configure and manage a number of physical or logical components as a virtual functional super-entity referred to as a fabric). For details, see the *Configuring and Administering the Unified Cloud Fabric* chapter.

Moreover, as part of the redundant fabric architecture, the NetVisor OS software supports additional advanced capabilities to increase network automation, resiliency and scalability, such as automatic link aggregation and switch clustering with vLAGs.

Understanding Link Aggregation

Link aggregation is a standard technology used to combine multiple network connections in order to provide redundancy in case of single or multiple link failure. The augmented ***data pipe*** generated by aggregating N individual links behaves as a single logical high-bandwidth data path whose capacity can be up to N times the bandwidth of each individual link it comprises. Such logical connection is called a Link Aggregation Group (LAG).

Sometimes it is also called a (port) trunk, link bundle, or port channel as it **bundles** a number of connected physical ports together to implement a single augmented logical communication channel capable of traffic load sharing with fast re-convergence and redundancy.

Therefore, when the number of active bundled ports in a LAG changes for some reason (for example user configuration, hardware fault, etc.), traffic patterns dynamically adapt to reflect the re-balanced state of the LAG with minimal service disruption. Load splitting is achieved by hashing each packet's L2 through L4 headers (when present) to select a physical link to follow.

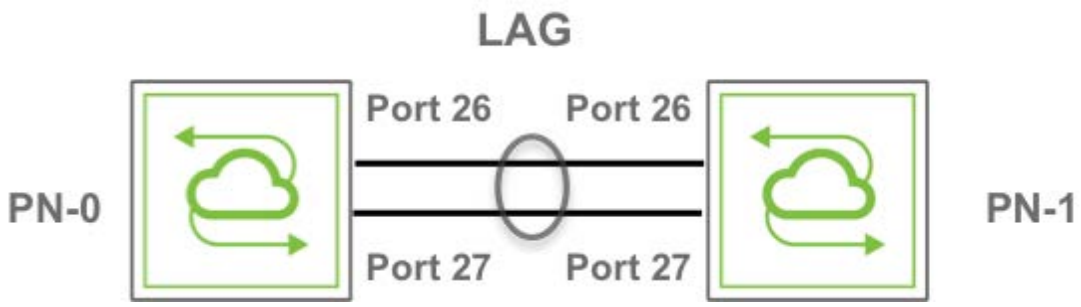


Figure 7-1 - Link Aggregation Group (LAG)

Other terms commonly used to describe this technology included with servers: network interface controller (NIC) bonding or NIC teaming.

Trunks can be either statically set up or dynamically created. In the former case the LAG creation is very fast but it lacks the automation and connectivity checks of the latter case.

For link aggregation the IEEE organization has defined a standard, vendor-independent implementation with the IEEE 802.3ad specification (later transferred to the IEEE 802.1AX-2008 document and superseded by 802.1AX-2014) and has also defined a standard dynamic protocol, called Link Aggregation Control Protocol (LACP), to automate this process.

Understanding the Link Aggregation Control Protocol (LACP)

The IEEE 802.3ad Link Aggregation Control Protocol (LACP) supports the automatic creation of Ethernet LAGs by exchanging special LACP frames (called LACPDUs) down all the links that have the protocol enabled. If it finds an LACP-capable neighbor device (peer) on the other end of the links, it will negotiate the dynamic creation of a LAG comprising all the compatible links (only full-duplex point-to-point Ethernet links running at the same speed can be matched and grouped together).

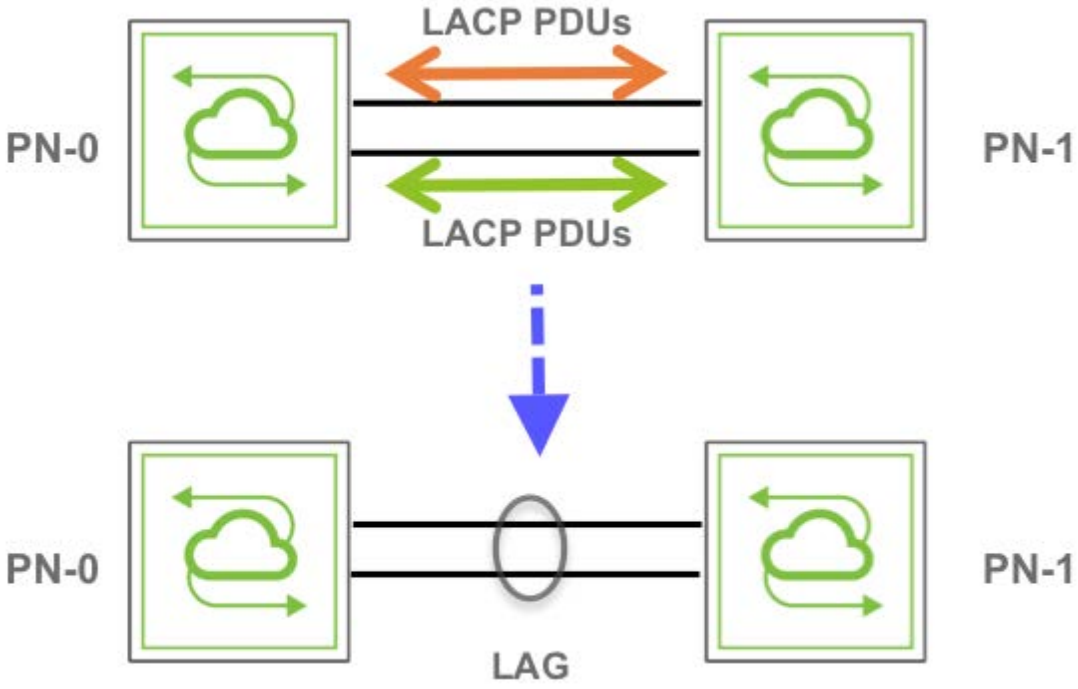


Figure 7-2 - Link Aggregation Control Protocol (LACP)

The LACP dynamic negotiation can be configured in one of two modes: active or passive.

In active mode LACP always (unconditionally) sends frames along the configured links that are intended to be bundled together. In passive mode, instead, LACP does not initiate a conversation/negotiation until it hears from the peer (this is typically the default configuration). Hence, for the negotiation to start at least one of the peers needs to be switched to active mode by the network administrator.

Once the negotiation is complete and the configured number of ports is aggregated, LACP periodically transmits keepalive frames over a LAG to verify connectivity of each link member and to react (i.e., trigger a link failover) in case of detection of a malfunction. LACP can be configured with a slow or fast lacp-timeout value (90s vs 3s) to react more gracefully or more promptly to connectivity changes.

LACP therefore is very helpful with the function of creating and/or augmenting a LAG as it always performs the necessary validations of the added links (such as bidirectional connectivity).

The maximum number (N) of bundled ports in a LAG is typically (4 or) 8. On top of that a significant number of LAGs (L) may need to be provisioned in a large network design to achieve optimal scalability. Therefore, especially in complex deployments with many LAGs and several ports per LAG, the LACP dynamic function plays a really crucial role both in simplifying the port bundling process and in making sure that such process runs smoothly and error free over a large number of network entities (N x L).

In addition, in case of link failure, both LACP and the hardware algorithms will then make sure that the re-convergence/failover process is prompt and minimally impactful for the traffic. (On the other hand, static LAG configuration would not guarantee an equally easy and effective behavior in all possible scenarios.)

In summary, the LACP logic performs the following functions on the switch:

- Maintains configuration and port state information to control link aggregation.
- Exchanges configuration and connectivity information with other peer devices.
- Attaches or detaches ports from LAGs based on the exchanged configuration and state information.

LACP frames are always exchanged between neighboring ports when in active mode, whereas in passive mode frames are exchanged as long as the peer is active. This allows the network admin to leverage the automated LACP negotiation logic to control the LAG bring-up or bring-down process by simply configuring one side of a trunk.

Off mode is instead used to disable the LACP function on a trunk when for example the network admin wants to manage the trunk bring up process manually (however that prevents any dynamic negotiation and link verification).

Furthermore, LACP employs two priority values to determine which ports to aggregate into an active LAG. If for example a hardware limitation prevents all configured compatible ports from aggregating, then the priority values will be used by LACP to select which ports to add and which ports to keep in standby mode.

LAG members in standby mode don't actively participate in traffic forwarding but can later be promoted to forwarding state to be used to replace a failed LAG member.

The LACP logic uses a system priority value, which is generated by the combination of the device MAC address with a user-configurable two-octet priority parameter. It also uses a port aggregation priority value, which is generated by the combination of the port number with a user-configurable two-octet priority parameter. Both priority values are used by LACP to dynamically select which port to aggregate in a bundle and which device (with higher priority) is entitled to that decision. The values for both user-configurable priority parameters can vary from 1 to 65535. Their default configuration is 32768.

Just as their member links, LAGs are point-to-point logical connections, so they can only be as resilient as the networking nodes that they interconnect. In other words, if either peer device at the end of the LAG fails, the entire LAG goes down.
Therefore, in order to further improve the network's fault tolerance, it is of paramount importance to introduce multiple redundant traffic paths in the design (also known as multi-pathing).

This can be achieved in numerous ways at Layer 2 and/or at Layer 3. A very effective Layer 2 strategy to implement dual fully redundant paths with very fast traffic failover is to extend the capability of LAG across two distinct switches to create a virtual LAG device pair.

In Arista terminology this cross-chassis LAG capability is implemented with switch clusters.

Understanding Switch Clusters

In general, in computer science the term ***cluster*** (also known as high-availability/HA cluster or fail-over cluster) is used to identify a group of devices that are functionally equivalent and structurally redundant so that they are able to provide continuity of service (without user intervention) when any component or an entire device fails. Therefore they are used for critical applications when all single points of failure must be eliminated.

Arista Networks builds upon this concept to bring end-to-end multi-pathing with advanced redundancy to high-performance networks where HA is a critical requirement, especially in case of mission-critical high-uptime data center and enterprise deployments.

For this reason Arista's clustering technology represents a key pillar of Arista Unified Cloud Fabric architecture as well as a powerful tool for network designers. From a practical standpoint it allows customers to deploy in many critical points of the network redundant pairs of switches (simply referred to as switch clusters or clusters) with both upstream and downstream traffic load balancing and fast failover capabilities.

A common scenario is depicted in ***Figure 7-3*** below.

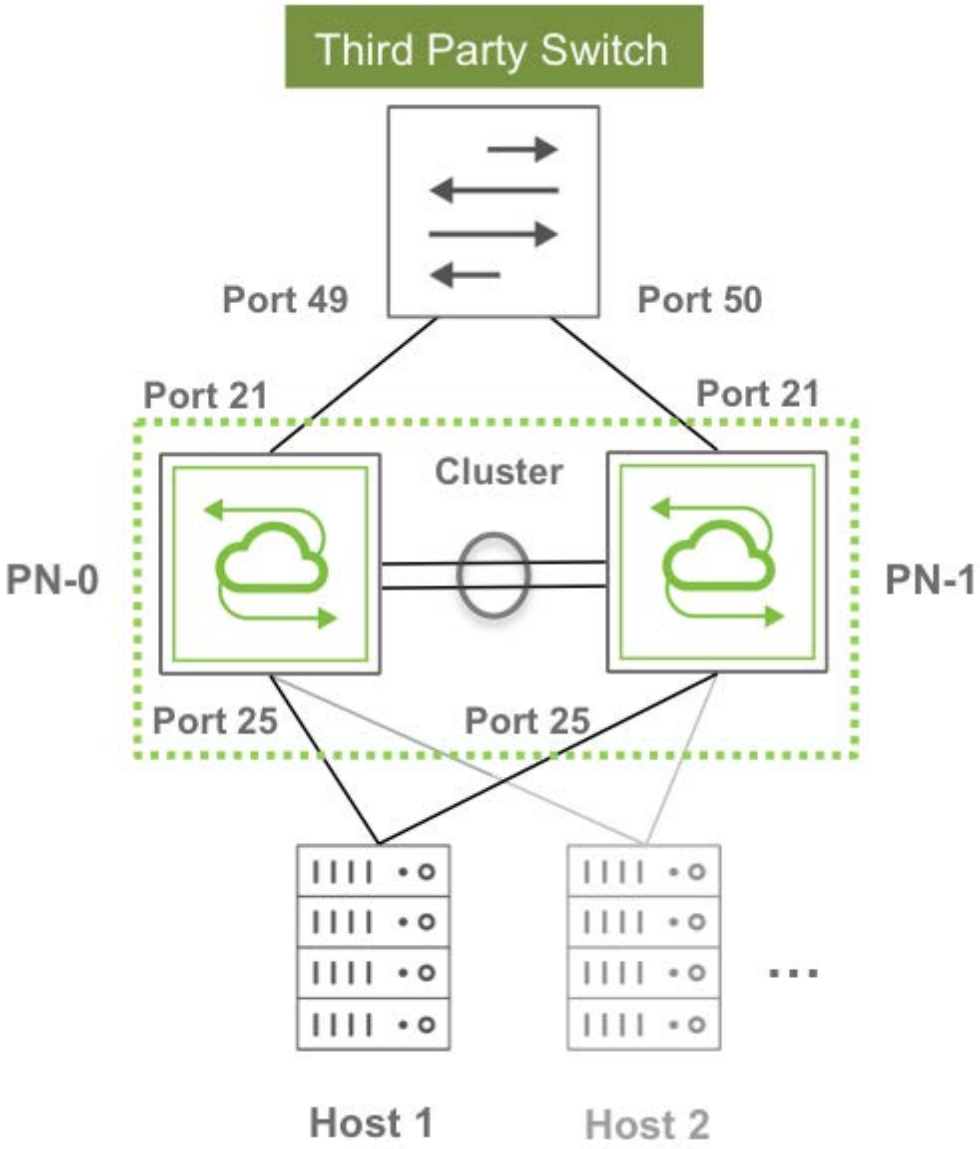


Figure 7-3 - Example of Requirement for Upstream and Downstream Path Redundancy

As seen in the figure above, clusters can interconnect third-party switches as well as servers in a redundant fashion simply by using standard technologies (link bundling or NIC bonding) on those devices.

The key is that each redundant pair of Arista switches functions as a single logical entity capable of inter-switch state synchronization to guarantee proper Layer 2 operation as well as traffic load balancing and failover. To external devices this switch pair appears as a single virtual neighbor both for traffic forwarding and for protocol communication purposes.

As a matter of fact, in computer networking an alternate name that is sometimes used for a cluster is virtual chassis. And this latter designation is perhaps more useful to describe the concept: a pair of separate physical switches is joined in a virtual entity that behaves as if ports belonging to each device were unified within a common virtual chassis.

In Arista's advanced networking architecture clustering is a natural fit and represents a logical extension of the fabric. The requirement for clustering is in fact that the two switches be members of the same fabric instance.

In addition, due to the asymmetric nature of certain interactions, when you create a cluster you must designate one cluster node as cluster-node-1 (that is, primary node) and the other as cluster-node-2 (that is, secondary node).

In order to set up a cluster, an active link between the primary and the secondary node must exist (as shown above in Figure 3). This is called a “cluster link” and can comprise one or more ports directly connecting the two cluster nodes together. If there are more than one port, they will be automatically bundled and the cluster link will effectively be a “cluster trunk” (that is, the LAG of those ports).

Using an intrinsically redundant cluster trunk for cluster node interconnection is the recommended best practice.

The aggregate bandwidth of such trunk should be such that no major traffic bottleneck is experienced in case of failover scenario and/or in the presence of single-homed upstream or downstream devices (while in the former case the cluster trunk is used as backup path only in case of failure, in the latter case a portion of the traffic is always forced to cross the cluster links). This bandwidth calculation varies depending on the network design, the traffic type, the number and requirements of the single-homed devices, etc.

VLAN 4094 is reserved and is used to carry cluster synchronization traffic. It is automatically added to the in-band interface port and to the cluster link when you create the cluster configuration. This is possible because NetVisor OS detects cluster links using an extra data set sent in LLDP messages. Therefore, when a cluster link is detected through this mechanism, VLAN 4094 is automatically added to it.

NetVisor OS performs cluster synchronization over the control network of the fabric. For the in-band interface, synchronization uses VLAN 4094 over the cluster links. For management, synchronization is performed over the management interface.

Each cluster node sends a synchronization message (with cluster state, versioning info, device uptime, etc.) to the peer node every 2 seconds. These messages function also as keepalives: so if three messages in a row are missed, the cluster pair goes offline (that is, symmetrically).

On the other hand, the process of going online for the cluster happens asymmetrically, for two main reasons: to avoid race conditions and to determine an additional important state of each node, the ***master*** or ***slave*** role, for protocol support purposes. Synchronization messages are thus exchanged and used to select the master and slave roles via an internal negotiation algorithm based on parameters like uptime.

When a cluster comes online, it triggers:

- A resend of all status updates to the peer node
- A re-synchronization of all vLAGs
- The transition of STP to run in cluster mode

Synchronization of vLAGs (see also the next sections) is used to make sure that they behave as a single logical entity across both nodes. This is achieved by synchronizing port state, as well as Layer 2 entries when needed.

vLAGs in a cluster can work in active-standby mode (where only the vLAG members on one node are active) or in active-active mode (where all the vLAG members are active and forwarding). The latter mode requires Layer 2 entry/vPort synchronization across nodes.

vLAG state synchronization typically happens from the primary node to the secondary node. However, the secondary node can synchronize a (local) port state when that port comes up. In case of necessity it can also request the primary node to (re)start the synchronization process.

Layer 2 entry synchronization is instead performed symmetrically by both nodes upon detection of a vPort change for performance reasons.

About the Cluster Re-peer Process

When a cluster node dies or needs replacement, it is necessary to use another switch to substitute the failed/missing node to rebuild the cluster pair as soon as possible so as to restore network redundancy.

Initially, the `cluster-repeer` command was used to rebuild a cluster with a new switch node. Subsequently, as an enhancement, the rebuild process was integrated into the fabric join procedure. A cluster pair is now restored via the `fabric-join` command followed by the option:

`repeer-to-cluster-node`	Replace a dead cluster node by restoring against the existing cluster node.

Before adding a new cluster node, though, the node to be replaced must first be evicted from the fabric with the `fabric-node-evict` command.

Then, in order to be able to rebuild a cluster via the `fabric-join` command with the `repeer-to-cluster-node` option, the new node needs to be directly connected to the remaining cluster node that is still active.

The joining node that performs a `fabric-join` to repeer with the cluster's master node will initiate direct communication with it to get up-to-date fabric and cluster configuration files and transaction IDs.

After the joining node installs the new configuration, it transmits a fabric-join transaction to notify all fabric members that it has joined the fabric. In addition, it runs a cluster-update command which rebuilds the cluster pair with itself and the existing cluster peer as members. The existing cluster peer will also modify any cluster objects that may require an update as a consequence of the repeer procedure (for example, trunks).
Finally, the joining node restarts and the process is complete.

The aforementioned procedure implicitly applies to Layer 2-based fabric topologies where direct Layer 2 connectivity with the other fabric nodes is possible for message exchange.

In case of Layer 3 fabric designs, instead, a further enhancement has been introduced in NetVisor OS release 5.1.0 to perform the repeer operation over a Layer 3 fabric interconnect.

A new `fabric-join` option has been added to allow the users to specify that a cluster pair needs to be rebuilt over a Layer 3 fabric configuration:

`over-l3`	Specify to establish fabric is over layer 3.

In addition, a new `over-l3` flag has been added to the output of the `fabric-info` command to indicate whether it's a Layer 2 or Layer 3 case.

Understanding Cluster over Layer 3

In releases prior to NetVisor OS version 7.0.0, direct links between switch pairs were required for clusters to form and function. This means that you must reserve one or more links between the cluster nodes.

Starting from release 7.0.0, as an option, NetVisor OS supports the formation of switch cluster pairs without requiring any direct Layer 2 connections.
This feature leverages the VXLAN technology and is referred to as “cluster over Layer 3” in short. (Refer to the Configuring VXLAN chapter for more details on the VXLAN technology.)

The implementation of cluster over Layer 3 uses the uplinks connected to the spine nodes (which can be third-party devices) in order to form a cluster pair communicating over a Layer 3 network.

See the *Configuring Cluster over Layer 3* section below for platform support, a deployment model, and the required configuration steps.

About the Spanning Tree Protocol (STP) in Cluster Mode

In cluster mode NetVisor OS ensures that the two physical cluster nodes behave as one single logical spanning tree node.

It achieves this by automatically sharing across both devices the same STP bridge ID and STP priorities, and by using dedicated and disjoint STP port ID ranges: STP port IDs on the primary node are 0 to PORTS_MAX – 1 whereas on the secondary node are PORTS_MAX to (2*PORTS_MAX - 1). This happens regardless of the online/offline state of the cluster and thus guarantees protocol ID consistency/persistency and a seamless failover in case of node failure.

In addition, in case of active-active vLAGs (that is, dual-homed links), NetVisor OS makes sure that they behave as single logical ports in the STP state machine. For this purpose, rather than creating a new port ID range for these logical ports, the master uses its local STP port ID as the logical port ID (which is preserved across failovers). Therefore, in case of single vLAG member link failure no STP re-convergence occurs. (Please note that with active-standby vLAGs a single link failure can cause an STP topology change.)

If an entire cluster node fails and therefore the cluster goes offline, STP will run the STP state machine locally and independently from the (down) peer. This is therefore called *independent mode*.

Then, when the cluster comes back online and needs to run in STP cluster mode, as mentioned in the previous section, the cluster synchronization process is employed to elect a master node (with the non-elected peer becoming the slave).

Only the master node runs the STP state machine, which encompasses all VLANs and all ports, both local (on the master switch) and remote on the slave switch (cluster links are excluded and are managed separately, as described in a subsequent paragraph).

The master's STP state machine, user configuration changes (of the STP mode, MSTP instances, bridge ID, etc.) and vLAG port states are mirrored to the slave node so that, if the master fails, the slave can seamlessly take over from where the master left off.

In conjunction with the ownership of the STP state machine, it's required for the master switch to manage the very important BPDU transmission function, even for any (single-homed) ports belonging to the slave switch. Not all the aggregate burden of this function falls on the master node, though: the slave switch provides some basic PDU relay support to the master to ensure proper load splitting of this function and hence optimal scalability when running in STP cluster mode. In addition, the slave relays to the master all BPDUs it may directly receive.

As for cluster links, they follow specific STP state transitions: when a cluster goes online, the cluster links start in STP blocked state (except for VLAN 4094 used for synchronization) to prevent loops. However, once the state from the newly elected master gets fully synchronized to its slave, the cluster links are put into STP forwarding state to provide a back-up forwarding path for traffic.

In special cases in which the transmission of the synchronization messages from the master to the slave fails (or times out), the cluster links are put back into the STP blocking state until synchronization can

complete to avoid potential loops.

If the master node goes offline, the slave will continue running the STP state machine from the last mirrored state. If the former master then comes back online, it is then selected to become the slave and the state of the node that was already up is mirrored to it.

In this way, there is never any interruption in STP state preservation even after multiple failures of either node, as long as one node is always up.

About Split Brain and Detection Script

In Arista parlance, a cluster has two switches (nodes) that operate as a single logical switch. These switches periodically as well as on an event driven basis, exchange control messages over the control network to keep the status and tables (such as L2 tables, vLAG, STP, cluster states, etc) between the two switches synchronized. If one of the nodes in the cluster fails to communicate with the peer node for three consecutive cluster-sync messages, then the node sets the cluster to offline mode and attempts to function in an independent mode. If one of the cluster nodes is down, then operating as an independent node helps to maintain continuity.

However, if both nodes are up and the nodes are unable to sync-up (for example, due to cluster network going down), then both nodes operating in independent mode is not desirable. This situation can lead to duplicate packets such as *broadcast-unknown unicast* or *multicast* (BUM) traffic or traffic loss. This condition is known as split brain.

To mitigate the split brain condition, NetVisor OS provides two scenarios:

1. Handling of Split Brain by disabling STP (starting with NetVisor OS version 7.0.1)
2. Handling of Split Brain using Detection Script (in NetVisor OS version 6.1.1 HF4)

Handling of Split Brain by Disabling STP

Starting from NetVisor OS version 7.0.1 onward, the split brain handling capability of NetVisor has been enhanced by allowing seamless ingress traffic on the orphan ports. That is, during a split brain condition, when STP is disabled on the nodes of a layer 3 fabric setup:

- All ports on both nodes of the cluster remain "*up*" even when a cluster node goes offline while STP is disabled. However, the flood traffic (towards vLAG ports) gets dropped after crossing over cluster because of the egress filtering rules.
- For ingress traffic on orphan ports, traffic entering the peer switch gets forwarded through the cluster port (assuming L2 table has the entry and is not aged out) and traffic on orphan ports remain active.
- When the cluster node comes back online, the cluster node re-synchronizes without having to reboot the switch.

System Behavior with STP enabled

When STP is enabled, if the control network goes down (due to disabling or any network issue), then the cluster port goes to discarding state (use the `stp-state-show` command to confirm) because the cluster STP sync messages cannot be exchanged. In other words, *in-band* connectivity is lost and data traffic cannot flow over the cluster link. This behavior applies even if the control network is configured as `mgmt`. Since the node reachability over both *in-band* and *management* is down, the state in `fabric-node-show` command displays as *offline*. See example below:

```
CLI (network-admin@switch) > fabric-node-show

fab-name  mgmt-ip          in-band-ip  in-band-vlan-type  fab-tid  cluster-tid  out-port  state
--------  ------------  ----------  -----------------  --------  ----------  --------  ------
fab1    10.13.48.59/24 1.1.1.1/24  public             49       5           272     offline
```

```
CLI (network-admin@switch*) > stp-state-show ports 272

vlan: 1,4093,4095
ports: none
instance-id: 0
name: stg-default
bridge-id: 66:0e:94:5e:fb:b2
bridge-priority: 32768
root-id: 66:0e:94:5e:fb:b2
root-priority: 32768
root-port: 0
hello-time: 2
forwarding-delay: 15
max-age: 20
disabled: none
learning: none
forwarding: none
discarding: 272 >>>>>>>>>> discarding
edge: none
designated: 272
```

Note: If management network IP is changed to make it unreachable for unicast fabric messages when the connectivity is available for fabric multicast messages, then the state in `fabric-node-show` command oscillates between `mgmt-only-online` and `offline` states.

Handling of Split Brain Using Detection Script

Caution: We recommend disabling the *split brain* script before performing any software upgrades on the switches in the cluster.

If you are using NetVisor OS version 6.1.1 HF1, you must use the script provided as part of NetVisor package for the detection and recovery of split brain condition.

As a pre-requisite, you must install split brain detection script as a service on both nodes of the cluster pair. NetVisor OS supports control network over *Management* or over *in-band* IP. The control network can be set to `mgmt` or `in-band` by using the `fabric-local-modify control-network [in-band|mgmt]` command.

Note: Starting with NetVisor OS version 6.1.1 HF4, the split brain *install/uninstall* scripts are available in */opt/nvOS/bin/pn-scripts* directory. Having the scripts in */opt/nvOS/bin/pn-scripts* directory enables you to invoke the scripts from both the NetVisor OS CLI and REST API shell prompt.

When the script is installed on leaf switches:

- The script detects the split brain condition based on the following factors:
  - When the control network is over *management*, losing the management network connection results in split brain.
  - When the control network is over *in-band*, cluster links going down results in split brain
- Once the cause is detected, the script proceeds to quarantine the cluster slave (backup) switch. That is, all ports on the slave switch are brought down except the cluster ports. The method employed by split

brain script to disable the ports is not persistent with switch-reboot, hence no manual intervention to enable ports is required.

To install and run the script as a system service, run the *pn_split_brain_install.sh* command by using NetVisor OS CLI. The service will persist a switch reboot. It is not required to re-run the script on every reboot or power-cycle. To stop the script from running, use the *pn_split_brain_uninstall.sh* command.

The script can also detect when the cluster is back online.

Below is an example on how to use the script. To install and run the script, use the command:

```
CLI (network-admin@switch*) > pn-script-run name pn_split_brain_install.sh
Executing  /opt/nvOS/bin/pn-scripts/pn_split_brain_install.sh:
Created symlink from /etc/systemd/system/multi-user.target.wants/svc-nvOS-split-brain.service
to  /etc/systemd/system/svc-nvOS-split-brain.service.
```

To know or detect the status of the script, use the command:

```
CLI (network-admin@switch*) > exit
root@switch*:~#
root@switch*:~# systemctl status svc-nvOS-split-brain.service
● svc-nvOS-split-brain.service - Service to check for split brain functionality in cluster-
slave and disable ports
    Loaded: loaded (/etc/systemd/system/svc-nvOS-split-brain.service; enabled; vendor preset:
enabled)
   Active: active (running) since Wed 2021-11-17 02:27:24 PST; 7s ago
 Main PID: 25780 (perl)
    Tasks: 2
   Memory: 12.8M
      CPU: 1.697s
    CGroup: /system.slice/svc-nvOS-split-brain.service
            ├─25780 /usr/bin/perl /usr/bin/pn_split_brain.pl
Dec 06 19:39:18 switch* systemd[1]: Started Service to check for split brain functionality in
cluster-slave and disable ports.
```

To stop the script from running, use the *pn_split_brain_uninstall.sh* command as below:

```
CLI (network-admin@switch*) > pn-script-run name pn_split_brain_uninstall.sh
Executing  /opt/nvOS/bin/pn-scripts/pn_split_brain_uninstall.sh:
Removed  symlink  /etc/systemd/system/multi-user.target.wants/svc-nvOS-split-brain.service.
```

To know or detect the status of the script, use the command:

```
CLI (network-admin@switch*) > exit
root@switch*:~#
root@switch*:~# systemctl status svc-nvOS-split-brain.service
●  svc-nvOS-split-brain.service
    Loaded: not-found (Reason: No such file or directory)
   Active: inactive (dead)
```

To install the script on all switches of the fabric, use the command:

```
CLI (network-admin@switch*) > switch * pn-script-run name pn_split_brain_install.sh
```

To uninstall the script on all switches of the fabric, use the command:

```
CLI (network-admin@switch*) > switch * pn-script-run name pn_split_brain_uninstall.sh
```

Guidelines and Limitations

The following guidelines/limitations apply during split brain condition:

- The `allow-offline-cluster-nodes` command option in the `transaction-settings-modify` command is turned *OFF* by default and further fabric and cluster scoped transactions are not allowed until the cluster node is back online.
- Quarantine of cluster slave (backup) node mitigates the traffic loss due to split brain issue.
- In Fabric over Layer 3 to detect split brain, you should have '*allow-as*' enabled (so that in-band-ip of cluster master is reachable via spine switch) or the *management* link should be present.

Split Brain Script Status Verify Command

A new command, `pn-service-status-show`, is introduced in Netvsior ONE 6.1.1 HF5 to verify the status of split brain script. An example of how to check the status on both the CLI and vREST API prompt is provided below:

When the split-brain script is not installed:

```
CLI (network-admin@plu007switch*) > pn-service-status-show
name        status
----------- -------
split-brain offline

CLI (network-admin@plu007switch*) >
pluribus@plu-srv001:~$
pluribus@plu-srv001:~$ curl  -s -u network-admin:test123 -X GET http://plu007swl01/vRest/pn-
services -k  | python -m json.tool
{
    "data": [
        {
            "name": "split-brain",
            "status": "offline"
        }
    ],
    "result": {
        "result": [
            {
                "api.switch-name": "local",
                "code": 0,
                "message": "",
                "scope": "local",
                "status": "Success"
            }
        ],
        "status": "Success"
    }
}
pluribus@plu-srv001:~$
```

When the split-brain script is installed:

```
CLI (network-admin@plu007switch*) > pn-script-run name pn_split_brain_install.sh
```

```
Executing  /opt/nvOS/bin/pn-scripts/pn_split_brain_install.sh:
Created  symlink  from  /etc/systemd/system/multi-user.target.wants/svc-nvOS-split-brain.service
to  /etc/systemd/system/svc-nvOS-split-brain.service.

CLI  (network-admin@plu007switch*)  >  pn-service-status-show
name        status
----------      ---------------------------------------------------------------
split-brain  Active: active (running) since Wed 2022-02-16 00:45:17 PST; 5s ago

CLI  (network-admin@plu007switch*)  >

pluribus@plu-srv001:~$ curl  -s -u network-admin:test123 -X GET http://plu007swl01/vRest/pn-
services -k  | python -m json.tool
{
    "data": [
        {
            "name": "split-brain",
            "status": "Active: active (running) since Wed 2022-02-16 00:45:17 PST; 19s ago\n"
        }
    ],
    "result": {
        "result": [
            {
                "api.switch-name": "local",
                "code": 0,
                "message": "",
                "scope": "local",
                "status": "Success"
            }
        ],
        "status": "Success"
    }
}
pluribus@plu-srv001:~$
```

Note: When split brain script is installed or uninstalled on a switch, a message along with time-stamp is printed in nvOSd.log and perror.log file. See example below:

```
2022-02-13,20:04:41.358.: split-brain service is installed
2022-02-14,20:05:21.680.: split-brain service is uninstalled
```

Recovery handling

To recover from split brain, the quarantined switch should be brought back to active service after the cluster comes back online. The script detects when the cluster is back online and proceeds to reboot the prior cluster slave switch (which had detected the split brain condition and was quarantined) and bring it back to active service with the ports enabled.

To check the status of ports, use the command:

```
CLI  (network-admin@switch)  >  cluster-bringup-show
```

Where the status displays `ports-enabled` indicating that both devices of cluster are now active and up for service.

Linux Disk (BTRFS) Mirroring on NRU02, NRU03 and NRU-S0301 Platforms

Disk mirroring, also known as RAID1 (Redundant Array of Independent Disks), is the replication of logical disk volumes onto separate disk(s) in real time that ensures continuous availability of data. A mirrored volume is a complete logical representation of separate volume copies.

In NetVisor OS, the disk mirroring functionality is available on the following platforms:

- NRU03 and NRU-S0301 platforms (NetVisor OS version 6.1.0 onward)
- NRU02 (NetVisor OS version 7.0.1 onward)

On the above platforms, the *BTRFS* partition is mirrored using *BTRFS* RAID1 support and the swap partition is mirrored using *MDADM* RAID support, thereby enabling high availability (HA).

When you upgrade the software from an earlier release version to NetVisor OS version 6.1.0 on NRU03 and NRU-S0301 platforms or to version 7.0.1 on NRU02 platforms, the disk mirroring capability is enabled automatically. The software upgrade process ensures that you do not have to schedule separate maintenance window for disk mirroring.

Note:

- Disk mirroring is enabled by default as part of the upgrade process on NRU03 and NRU-S0301 platforms (in version 6.1.0) and on NRU02 platforms (in version 7.0.1).
- The time taken for upgrade process is longer on these two platforms due to the mirror create process, however, subsequent upgrades do not require the additional time.

It is recommended to disable the *FS Monitor service (FS_MON)* when Disk Mirror capability is enabled on NRU02, NRU03, and NRU-S0301 nodes that were running the FS_MON service. This service takes a recovery action to isolate the node during an SSD failure. However, the Disk Mirror capability eliminates the need to isolate the node provided the switch was replaced after an initial disk failure.

To get the current status of the disk, use the command:

```
root@nru03-sff-tm-1:~#    /opt/nvOS/bin/pn-scripts/fs_mon_status.sh
Read/Write
```

To disable the FS Monitor service from SHELL, use the command:

```
root@nru03-sff-tm-1:~#    /opt/nvOS/bin/pn-scripts/fs_mon_disable.sh
Removed   symlink   /etc/systemd/system/multi-user.target.wants/svc-
fs_mon.service."
```

As mentioned earlier, disk mirroring is enabled by default during the upgrade process. However, you can toggle (enable/disable) disk mirroring using the `switch-setup-modify` command on the above platforms. In this case, you must schedule a separate maintenance window for disk mirroring.

Note: When you toggle the disk mirroring functionality, the switch automatically reboots if the mirroring action is successful.

To toggle disk mirroring, use the `switch-setup-modify` command as shown in the below example:

```
CLI (network-admin@nru-sff) > switch-setup-modify enable-disk-mirror
OR
CLI (network-admin@nru-sff) > switch-setup-modify disable-disk-mirror
```

Below is a sample output displayed during the software upgrade to NetVisor OS version 6.1.0:

```
CLI (network-admin@nru-sff) > software-upgrade package nvOS-6.1.0-
6010018137-onvl.pkg
Scheduled background update. Use software-upgrade-status-show to check.
Switch will reboot itself. DO NOT reboot manually.
-----------------------------------------------------------------------
 [Apr11.07:48:12] Starting software upgrade ...
 [Apr11.07:48:13] Checking available disk space...
 [Apr11.07:48:13] Avbl free space: 200.29G, Required: 1.28G
 [Apr11.07:48:13] Unpacking local package bundle...
 [Apr11.07:48:13] Extracting initial bundle.
 [Apr11.07:48:29] Decrypting signed bundle.
 [Apr11.07:48:30] Extracting signed bundle.
 [Apr11.07:48:46] Extracting packages.
 [Apr11.07:49:03] Fetching repository metadata.
[Apr11.07:49:04] Skipping dpkg update in current boot image
 [Apr11.07:49:05] Computing package update requirements.
 [Apr11.07:49:05] Upgrade agent version: 6.0.1-6000116966
 [Apr11.07:49:05] Upgrading software upgrade framework
 [Apr11.07:49:09] Fetching repository metadata.
[Apr11.07:49:09] Skipping dpkg update in current boot image
 [Apr11.07:49:09] Computing package update requirements.
 [Apr11.07:49:10] Upgrade agent version: 6.1.0-6010018137
 [Apr11.07:49:10] Upgrading nvOS 6.0.1-6000116966 -> 6.1.0-6010018137
 [Apr11.07:49:56] Mirroring ONVL disk.
 [Apr11.07:52:20] Upgrading nvOS 6.0.1-6000116966 -> 6.1.0-6010018137
 [Apr11.07:52:20] Software upgrade completed. Rebooting.
 Shared connection to nru-sff closed.
```

Below is a sample configuration on enabling and disabling disk mirroring on a NRU-S0301 platform:

```
CLI (network-admin@nru-sff) > switch-setup-modify enable-disk-mirror
Warning: This will change disk mirroring state and reboot automatically.
Please confirm y/n (Default: n):y
Successfully enabled mirroring. Rebooting.
```

To verify the switch setup details, use the command:

```
CLI (network-admin@nru-sff) > switch-setup-show
switch-name:              nru-sff
mgmt-ip:                  10.13.6.48/23
```

```
mgmt-ip-assignment:        static
mgmt-ip6:                  fe80::9ac5:dbff:fe43:e3fa/64
mgmt-ip6-assignment:       autoconf
mgmt-link-state:           up
mgmt-link-speed:           1g
in-band-ip:                192.168.6.48/24
in-band-ip6:               fe80::640e:94ff:feff:9e66/64
in-band-ip6-assign:        autoconf
gateway-ip:                10.13.6.1
dns-ip:                    10.135.2.13
dns-secondary-ip:          10.20.4.1
domain-name:               pluribusnetworks.com
ntp-secondary-server:       0.ubuntu.pool.ntp.org
ntp-tertiary-server:        1.ubuntu.pool.ntp.org
timezone:                  America/Los_Angeles
date:                      2021-11-04,18:01:44
hostid:                    150998527
location-id:               2
enable-host-ports:         yes
banner:                    * Welcome to Arista Networks Inc. Netvisor(R). This is a monitored
system.   *
mgmt-lag:                  disable
mgmt-lacp-mode:            off
device-id:                 R1279-F0001-01XXXXXXXXX
ntp:                       on
disk-mirror:               enabled
banner:                    *ACCESS RESTRICTED TO AUTHORIZED USERS ONLY
    *
banner:                    * By using the Netvisor(R) CLI,you agree to the terms of the Arista
Networks *
banner:                    * End User License Agreement (EULA). The EULA can be accessed via*
banner:                    * http://www.arista.com/eula or by using the command "eula-show" *
```

To disable disk mirroring, use the command:

```
CLI (network-admin@nru-sff) > switch-setup-modify disable-disk-mirror
Warning: This will change disk mirroring state and reboot automatically.
Please confirm y/n (Default: n):y
Successfully disabled mirroring. Rebooting.
```

> **Note**: Once the disk-mirror functionality is disabled using the above CLI command, the subsequent software-upgrades do not enable disk mirroring on that switch.

To verify the change:

```
CLI (network-admin@nru-sff) > switch-setup-show
switch-name:               nru-sff
mgmt-ip:                   10.13.6.48/23
mgmt-ip-assignment:        static
mgmt-ip6:                  fe80::9ac5:dbff:fe43:e3fa/64
mgmt-ip6-assignment:       autoconf
mgmt-link-state:           up
mgmt-link-speed:           1g
in-band-ip:                192.168.6.48/24
in-band-ip6:               fe80::640e:94ff:feff:9e66/64
in-band-ip6-assign:        autoconf
gateway-ip:                10.13.6.1
```

```
dns-ip:                    10.135.2.13
dns-secondary-ip:          10.20.4.1
domain-name:               pluribusnetworks.com
ntp-secondary-server:       0.ubuntu.pool.ntp.org
ntp-tertiary-server:        1.ubuntu.pool.ntp.org
timezone:                  America/Los_Angeles
date:                      2021-11-04,18:40:03
hostid:                    150998527
location-id:               2
enable-host-ports:         yes
banner:                    * Welcome to Arista Networks Inc. Netvisor(R). This is a monitored
system.   *
mgmt-lag:                  disable
mgmt-lacp-mode:            off
device-id:                 R1279-F0001-01XXXXXXXXX
ntp:                       on
disk-mirror:               disabled
banner:                    *ACCESS RESTRICTED TO AUTHORIZED USERS ONLY
    *
banner:                    * By using the Netvisor(R) CLI,you agree to the terms of the Arista
Networks *
banner:                    * End User License Agreement (EULA). The EULA can be accessed via*
banner:                    * http://www.arista.com/eula or by using the command "eula-show" *
```

The disk mirroring configuration changes are logged into the disk mirror log file and is available at: */var/nvOS/log/disk_mirror.log* .

Additionally, an alert (that can help network admins to take required action) is generated and are saved to the console, event, and syslog messages when:

- One of the disk goes missing in BTRFS RAID1 configuration
- BTRFS device statistics has non-zero IO errors

For example, below is the sample output in different locations when a disk goes missing:

- From the output of `log-event-show` command:

```
CLI (network-admin@nru-sff) > log-event-show
```

- From the */var/log/syslog* file:

```
Feb 22 20:52:19 nru-sff root: BTRFS error: missing device in BTRFS mirror
```

- From the console:

```
nru-sff login: BTRFS error: missing device in BTRFS mirror
root@nru-sff:~#BTRFS error: missing device in BTRFS mirror
```

General Guidelines

- On NRU02 platforms (available from NetVisor OS version 7.0.1 onward), the Solaris disk gets overwritten and is used for mirroring the BTRFS partition. That is, you cannot boot to Solaris BE after the mirroring operation. Note that disabling the mirror functionality using CLI command does not bring back the Solaris partitions.

- On NRU02, NRU03, and NRU-S0301 platforms, to avoid the disk mirroring as part of upgrade process to NetVisor OS version 7.0.1, */etc/do_not_mirror_disk* should be present. This can be achieved by entering "*touch /etc/do_not_mirror_disk*" at the Linux prompt.
- Both disks on the switch should be identical in size for disk mirroring to be successful. If the disks are not identical, then the mirroring process aborts.
- If one of the disks goes offline due to hardware issues, then it is expected that the disk stays offline because BTRFS may not work if the disk goes offline intermittently and considers the offline disk as corrupt disk.
- If either disk fails while disk mirroring is enabled, then a switch replacement via RMA process is required as the disk cannot be replaced in the field.
- If the second disk fails for any reason, then the switch changes to read-only mode and loses normal functioning capability. This scenario can be avoided by promptly replacing the switch if and when the first disk fails.
- *MDADM* package is installed on all BEs (boot-environments), during software upgrade, to enable swapping of disks.
- While upgrading to NetVisor OS version 6.1.0, all existing BEs (including BEs from older versions) on both NRU03 and NRU-S0301 platforms get the mirroring functionality enabled. Therefore, after upgrading to version 6.1.0, if you later rollback to a previous version on the same switch, then the mirroring functionality is preserved and the older BEs continue to get mirrored onto the second disk.
- Only the latest version of OS/ONL field diagnostics released in November 2020 is retained during disk mirroring. All older versions or other partitions on disk 2 are removed during the disk mirroring process.
- If it is required to install OS/ONL field diagnostics, then first disable disk mirroring, install the field diagnostics, and enable disk mirroring again. The expected mirror creation time is 10-15 minutes.
- As part of mirroring process, the SWAP size is reduced from 32 GB to 24 GB to accommodate the ONL field diagnostics partition. This is a one-time operation and cannot be rolled back, which means that disabling disk mirroring does not increase the swap size back to 32 GB.

About Layer 3 Hardware Forwarding in a Cluster

In a cluster, Layer 3 hardware forwarding takes place on both nodes running as a VRRP pair (see also the next sections) even if the local VRRP interface is in standby.

In symmetry of routing each node is expected to forward traffic in hardware on behalf of the peer for any packets destined to the VRRP virtual IP address. This can be achieved by synchronizing L3 entries, L3 adjacencies (vPorts) and vRouter state between the two peers.

For single-homed L3 interfaces, instead, nodes may act independently and may even have an L3 adjacency over the cluster link for failover purposes.

Understanding vLAGs

The virtual chassis logical model can be effectively used to enable an advanced capability called multi-chassis link bundling/port channeling. This feature is also referred to as MC-LAG (Multi-Chassis Link Aggregation Group).

An MC-LAG is a cross-device link bundle (i.e., trunk). It typically has at least one port member on one switch and another port member on a backup switch. It can be facing downstream toward an end point (configured with a regular LAG on its end) or upstream toward another switch or router (also using a regular LAG or even an MC-LAG).

The IEEE 802.1AX-2008 standard for link aggregation (LAG) does not mention MC-LAG but does not preclude it, because the implementation of the required synchronization technology can be vendor-specific without affecting interoperability. Such additional under the hood (i.e., mostly transparent) synchronization technology performs advanced functions such as automatic MAC address learning synchronization across members, sharing of the same switch MAC and STP system ID, sharing of port state and information even during dynamic protocol negotiations, etc.

Aista's implementation of MCLAG is called vLAG (Virtual Link Aggregation Group) on a switch cluster. That in practice means that in order to be able to configure a vLAG a cluster pair needs to be set up first.

As mentioned in an earlier section, a necessary requirement for a cluster to be created is to provision a redundant ***horizontal*** connection between the two cluster members: in Arista parlance this process is called automatic LAG (link aggregation) of the intra-cluster trunk. As long as at least two ports on both cluster members are physically interconnected, the automatic LAG logic will take care of bundling them together and provisioning them as intra-cluster trunk.

The ***cluster trunk*** is just a regular LAG, but is the pivot around which various important functions revolve. First off, it provides a critical communication channel utilized for cluster synchronization activities as well as protocol exchanges. It is also instrumental for supporting the MC-LAG (a.k.a. vLAG) function on the cluster, as it can function as a backup forwarding path during failover scenarios. (Therefore it must be a fat enough data pipe to prevent potential bottlenecks in case of failures.)

From a high level point of view, a vLAG is a single logical but physically redundant upstream or downstream connection across a pair of cluster switches (see also Figure 4 below). In other words, physically it's a triangular topology that includes the cluster trunk, while logically it appears just as if it were a point-to-point multi-link connection.

This property makes active-active vLAGs ideal for a number of Layer 2 or Layer 3 network designs, where standard port bundling/port channeling technologies (IEEE 802.3ad) can be used for high availability and very fast traffic steering (due to multi-pathing) in case of link failure. (See also the examples in the figures below.)

Using clusters with vLAGs for instance is a technique often chosen to avoid using the Spanning Tree Protocol (STP) for Layer 2 redundancy.

Figure 7-4 below shows how a vLAG is used to present a router/switch with a single logical connection instead of two separate ones. Therefore, in case of a Layer 2 configuration, loop prevention is not needed on that network segment (as no loop is present).

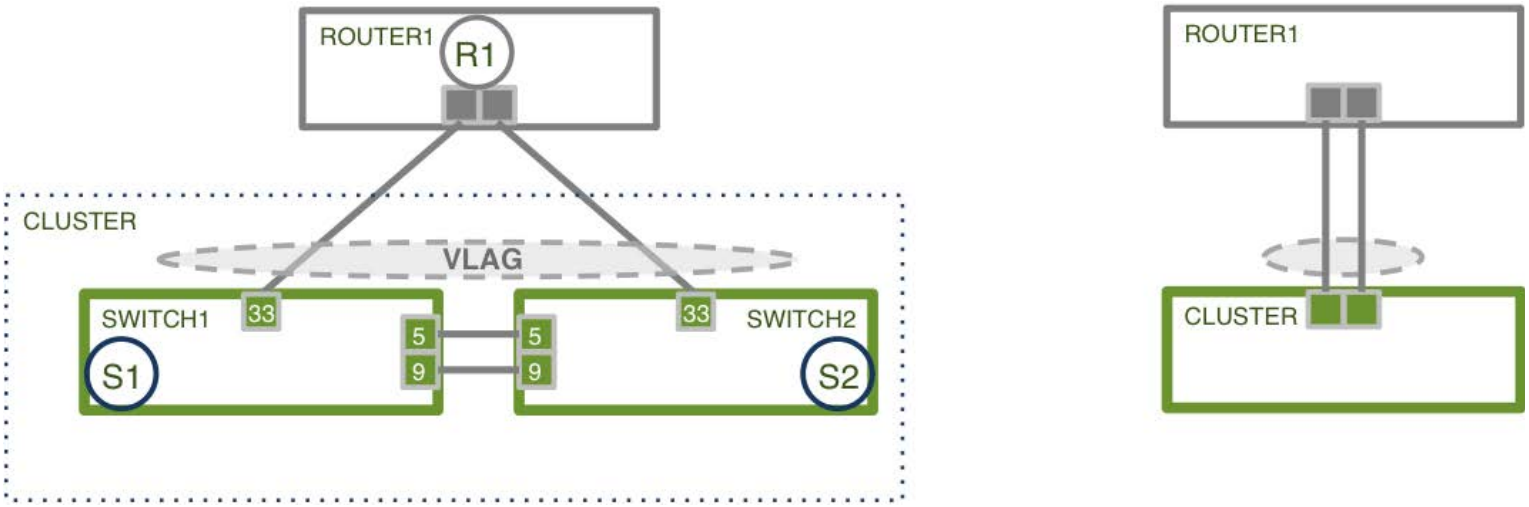


Figure 7-4 - Logical View of a vLAG from an External Device

Furthermore, the same strategy can be used downstream too to interconnect dual-homed servers to leaf switches.

In fact, as long as the neighboring device (whether a router, switch or server) supports the standard LACP protocol, port bundling negotiation is taken care of by the protocol itself automatically even when a vLAG is used on the Arista switches' side. (Note that even static trunking can be effectively used in these scenarios, especially when dynamic negotiation is not required.) This is by virtue of the interoperability characteristics built in to the Unified Cloud Fabric's control plane.

In general, switch clusters with vLAGs are extremely capable and flexible configurations so they can be employed in a number of network designs in conjunction with other standard technologies.

For example, as shown in Figure 7-5 below, vLAGs can connect to other vLAGs or to other third-party virtual chassis/multi-chassis LAG solutions. This creates a "four-way" bundle with the interconnection of two multi-chassis LAGs on opposite sides.

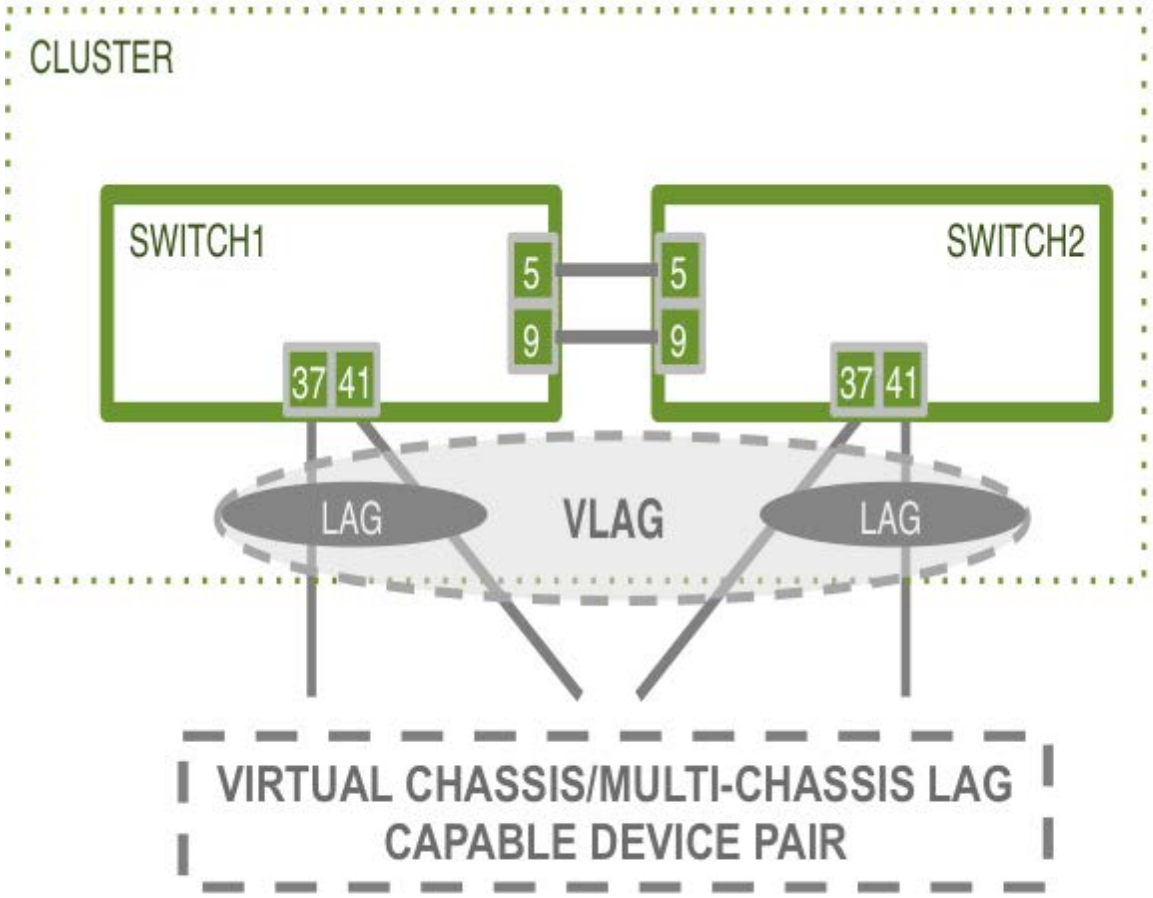


Figure 7-5 - Four-way vLAG between a Arista Cluster Pair and Another Multi-chassis LAG

Figure 7-6 below shows the common use case of interconnection of leaf and spine clusters to provide high availability using the virtual chassis/four way vLAG technique.

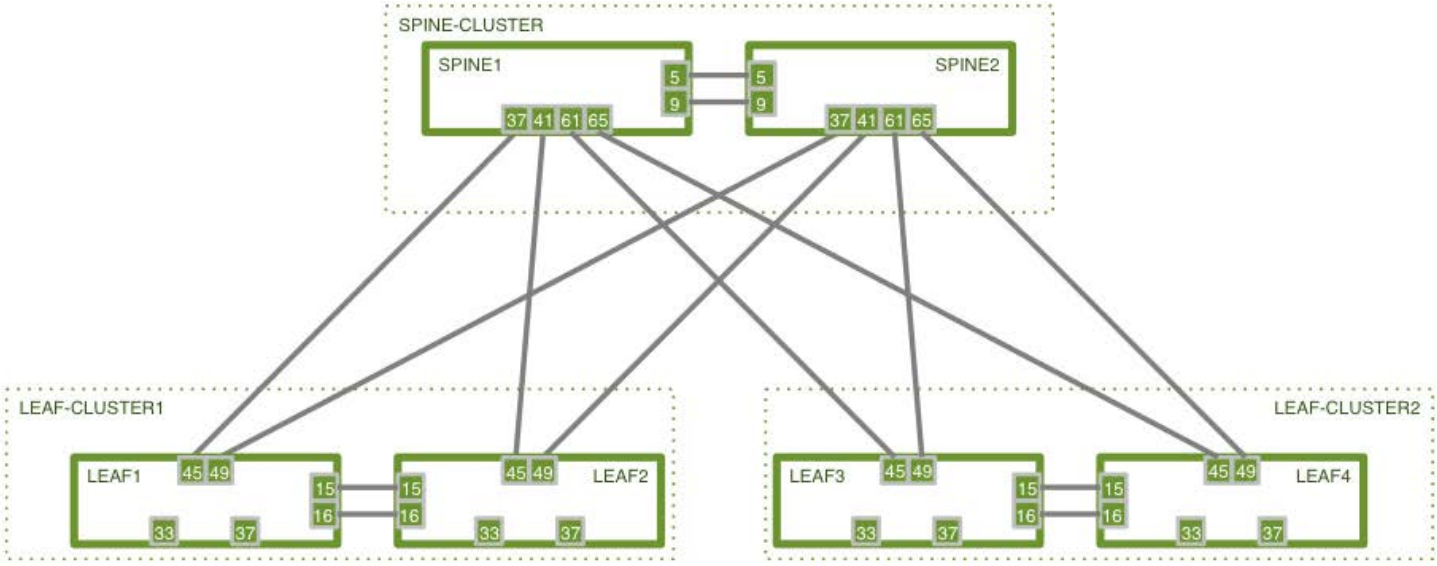


Figure 7-6 - Arista High Availability Cluster-based Spine/Leaf Design

In these network designs using active-active vLAGs (and cluster links), to prevent potential Layer 2 forwarding loops NetVisor OS automatically installs special internal forwarding rules that drop any packets that would egress a vLAG port if the ingress port is found to be the cluster link.

On top of all of the above, clusters and vLAGs can also be used in conjunction with the standard VRRP technology to provide redundant active-active first-hop Layer 3 gateways to multi-homed end-points.

This capability is particularly important in the context of DCI designs, when used in conjunction with another standard technology, VXLAN, as we will see in a subsequent chapter.

Understanding Virtual Router Redundancy Protocol (VRRP)

The Virtual Router Redundancy Protocol (VRRP) is a standard high-availability protocol initially defined by the Internet Engineering Task Force (IETF) in RFC 2338, later superseded by RFC 5798 for version 3 of the protocol with support for both IPv4 and IPv6.

The goal is to eliminate the single point of failure inherent in the first hop router (default gateway) for the connected hosts. Therefore, when at least two potential first hop routers are deployed, VRRP can be used to dynamically assign responsibility for the function of “virtual next hop” to either of the active routers on the (V)LAN. This is done by implementing an election protocol to select a so-called master router out of the two (or more) VRRP-capable routers available.

A master router performs the function of virtual router, i.e., controls the IPv4 or IPv6 address(es) (called virtual addresses or VIPs) used as default gateway(s) by the hosts and forwards packets sent to these address(es).

Non-elected router(s) are called backup router(s). The VRRP logic supports dynamic failover in the forwarding responsibility to a backup router, if the master router becomes unavailable.

- For IPv4, the critical benefit provided by VRRP is a higher-availability default gateway function with a resilient VIP address.
- For IPv6 configurations in which standard IPv6 Neighbor Discovery mechanisms could potentially help with the selection of the default gateway, the key benefit provided by VRRP is to provide a fast failover and a resilient VIP address.

VRRP supports rapid transition from master to backup router in case of node failure. The master router sends VRRP advertisements every second to the backup(s). If the master router’s advertisements are not received within a time window of three seconds, then a backup router is elected as the master. If the failed master router becomes active again, it can reclaim the role of master or allow the former backup to continue as the master router. The role depends on the value assigned to a parameter called VRRP priority.

VRRP routers are configured with a priority of between 1 and 254 and the router with the highest priority is elected to be the master. The default priority is 100.

At Layer 2 a VRRP virtual router must use an address in the 00-00-5E-00-01-XX Media Access Control (MAC) address range. In particular, the last byte of the address (XX) corresponds to the Virtual Router Identifier (VRID), which is distinct for each virtual router in the network. A VRRP virtual router will reply with this special MAC address when an ARP request is sent for the virtual router's IPv4 address.

As we will see in subsequent sections, having a resilient VIP that survives device failures is critical for simpler and more effective high availability of advanced services.
In Arista’s network designs VRRP can be used in conjunction with clusters and vLAG to combine fast switchover capabilities at Layer 2 with a redundant Layer 3 VIP for first hop routing toward the upstream part of the network (i.e., toward spine switches or other network devices).

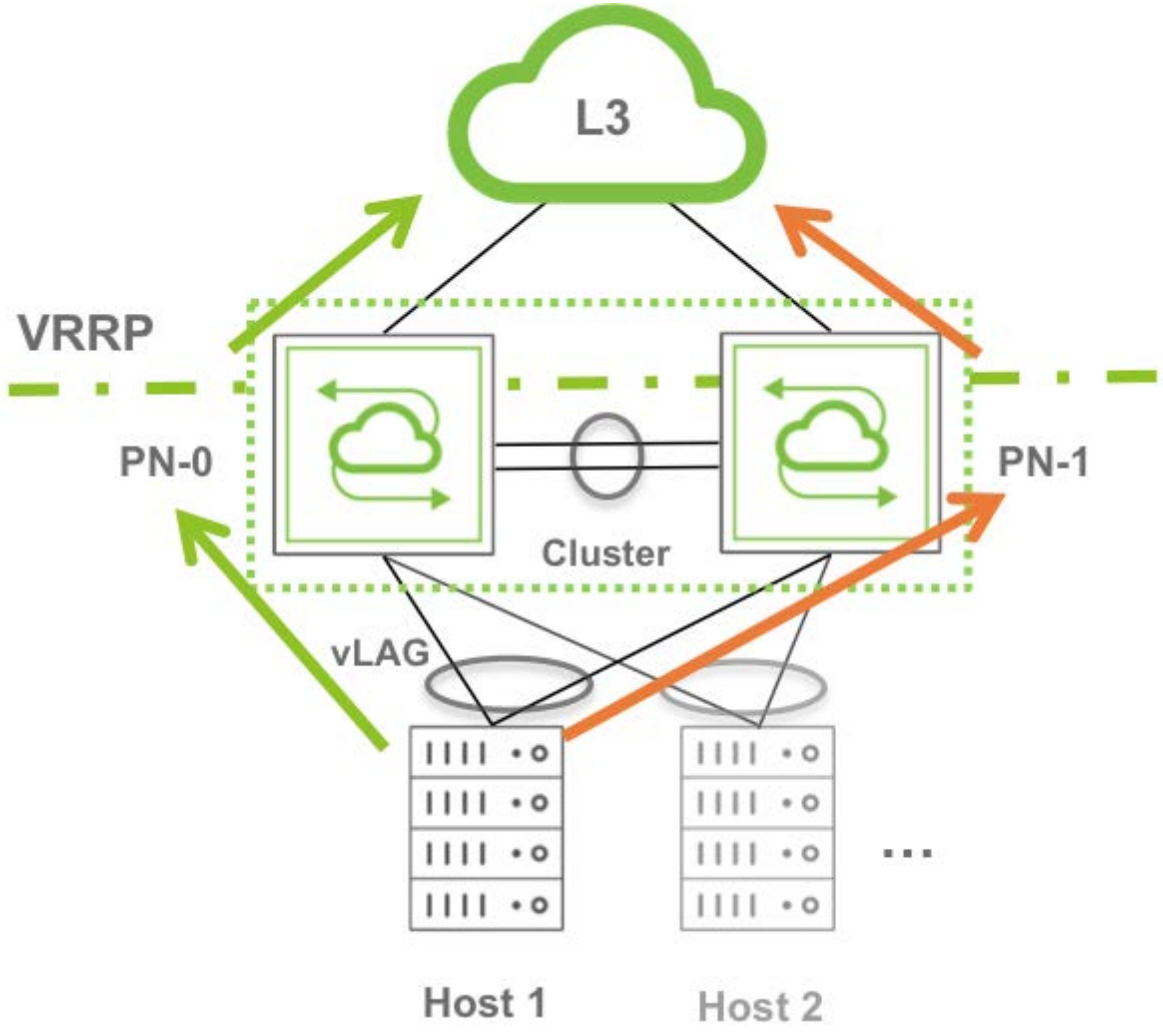


Figure 7-7 - Example of Use of VRRP in Conjunction with a Cluster

In addition, Arista's implementation optimizes the performance of this technology combination by supporting active-active Layer 3 forwarding on both VRRP routers in a cluster pair.

In other words, when a VIP is configured, each router is expected to route in hardware on behalf of the other peer for any packets destined to the VIP.

About Cluster Active-Active Routing for IPv6 Addresses

With cluster active-active routing two cluster peers act as forwarding proxies for each other's destination MAC addresses. This works for IPv4 as well as IPv6 traffic.
But if the two routers try to communicate with each other, packets may not route correctly on the network.

In particular, in case of the Neighbor Discovery function IPv6 uses ICMPv6 packets (instead of L2 ARP packets, as in the case of IPv4). Therefore, when one cluster node PN-0 sends a Neighbor Solicitation message to its peer, the other node PN-1 responds with a Neighbor Advertisement message with the destination IP address of the requester (PN-0). When transmitted by PN-1 the Neighbor Advertisement message will have both the destination IP and the destination MAC address of the peer PN-0. Since both nodes act as forwarding proxies for each other, the destination MAC address of the peer will be matched by the hardware of PN-1 and the packet will be routed (back to the CPU of PN-1) without reaching its correct destination (i.e., PN-0). This behavior would therefore break the Neighbor Discovery function of IPv6.

To obviate this problem, NetVisor OS adds a host route in hardware that matches the link-local IPv6 address of the cluster peer.

This host route entry properly routes packets such as Neighbor Advertisement messages to their destination.

Guidelines and Limitations

For recommended cluster and vLAG design topologies, refer to the section "About Symmetric Routing over vLAGs".

For cluster over Layer 3 the following guidelines apply:

- You cannot configure the cluster over Layer 3 feature on switches that are configured as EVPN border gateways. (See the *Configuring EVPN* section for more details on border gateways.)
- You must disable any direct links between cluster nodes before configuring cluster over Layer 3.
- To create a cluster over Layer 3 configuration, you must first configure VTEPs on each cluster node. For details, see *Configuring the Overlay: VTEP Interconnections and VNIs* section in the *Configuring VXLAN* chapter of the *NetVisor OS Configuration Guide*.
- To forward traffic over the cluster's local auto-tunnel, you must associate all VLANs with VXLAN IDs and the VXLAN IDs with VTEPs.
- Arista Networks recommends out-of-band connectivity (using a dedicated management network) for the cluster over Layer 3 feature.
- The `fabric-join repeer-to-cluster-node` operation requires direct inter-switch links. Therefore, this functionality is not supported with a cluster over Layer 3.
- The cluster over Layer 3 functionality supports using a physical interface's IP address only. It does not support a loopback interface's IP address.

Configuring Static Trunking for Link Aggregation

To statically configure a Link Aggregation Group (LAG), also known as trunk, you can use the `trunk-create` command.

```
CLI (network-admin@switch) > trunk-create
```

`trunk-create`	Create a trunk configuration.
`name` *`name-string`*	Specify the name for the trunk configuration.
`ports` *`port-list`*	Specify the port number(s) for the link(s) to aggregate into the trunk.
Specify any of the following options:	
`speed disable|100m|1g|10g|40g|50g|100g`	Specify the port speed or disable the port.
`egress-rate-limit unlimited`	Specify an egress rate limit for the configuration.
`autoneg|no-autoneg`	Specify if you want the physical port to autonegotiate port speed.
`jumbo|no-jumbo`	Specify if the port can receive jumbo frames.
`lacp-mode off|passive|active`	Specify the Link Aggregation Control Protocol (LACP) mode for the configuration.
`lacp-priority` *`number`*	Specify the LACP priority. This is a number between 1 and 65535 with a default value of 32768.
`lacp-timeout slow|fast`	Specify the LACP time out as slow (30 seconds) or fast (4 seconds). The default value is slow.
`lacp-fallback bundle|individual`	Specify the LACP fallback mode as individual or bundled.
`lacp-fallback-timeout seconds`	Specify the LACP fallback timeout in seconds. The range is between 30 and 60 seconds with a default value of 50 seconds.
`reflect|noreflect`	Specify physical port reflection
`edge-switch|no-edge-switch`	Specify if the switch is an edge switch.
`pause|no-pause`	Specify if pause frames are sent.
`description` *`description-string`*	Specify a description for the trunk configuration.
`loopback|no-loopback`	Specify loopback if you want to use loopback.
`vxlan-termination|no-vxlan-termination`	Specify if a VXLAN can terminate on the trunk.
`unknown-ucast-level` *`unknown-ucast-level-string`*	Specify an unknown unicast level in percent. The default value is 100%.
`unknown-mcast-level` *`unknown-mcast-`*	Specify an unknown multicast level in percent.

level-string	The default value is 100%.
broadcast-level *broadcast-level-string*	Specify a broadcast level in percent. The default value is 100%.
port-mac-address *mac-address*	Specify the MAC address of the port.
loop-vlans *vlan-list*	Specify a list of VLANS for looping.
routing|no-routing	Specify if the port participates in routing on the network.
host-enable|host-disable	Enable or disable host facing ports.
defer-bringup|no-defer-bringup	Specify to delay bringing up the host.
dscp-map dscp-map name|none	Specify the DSCP map name to enable on a port
local-switching|no-local-switching	Specify local-switching or no-local-switching. A no-local-switching port cannot bridge traffic to another no-local-switching port
allowed-tpid q-in-q|q-in-q-old	Specify an allowed TPID in addition to 0x8100 for VLAN headers.
fabric-guard|no-fabric-guard	Specify if fabric guard is enabled.
fec|no-fec	Specify Port Forwarding Error Correction (FEC) mode.

For example, create a trunk configuration named trunk-1 for ports 1, 2, and 3 by entering the following command:

```
CLI (network-admin@switch) > trunk-create name trunk-1 ports 1,2,3
```

To verify the result of the configuration, use the `trunk-show` command:

```
CLI (network-admin@switch) > trunk-show

name     port  speed   autoneg  jumbo
-------  ----  ------  -------  -----
trunk-1  1-3   10g     off      off
```

You can modify the trunk configuration by using the `trunk-modify` command:

```
CLI (network-admin@switch) > trunk-modify
```

trunk-modify	Modify a trunk configuration.
name *name-string*	Specify the name for the trunk configuration.
trunk-id *trunk-id-number*	Specify the trunk ID.
description *description-string*	Specify the trunk description. **Note:** If you add ports to the trunk, the trunk description applies to the new ports. If you remove ports from the trunk, the trunk description is removed from the ports.

Specify any of the following options:	
`ports` *`port-list`*	Specify the physical ports.
`speed disable|100m|1g|10g|40g|50g|100g`	Specify the port speed or disable the port.
`egress-rate-limit unlimited`	Specify an egress rate limit for the configuration.
`autoneg|no-autoneg`	Specify if you want the physical port to autonegotiate port speed.
`jumbo|no-jumbo`	Specify if the port can receive jumbo frames.
`lacp-mode off|passive|active`	Specify the Link Aggregation Control Protocol (LACP) mode for the configuration.
`lacp-priority` *`number`*	Specify the LACP priority. This is a number between 1 and 65535 with a default value of 32768.
`lacp-timeout slow|fast`	Specify the LACP time out as slow (30 seconds) or fast (4 seconds). The default value is slow.
`lacp-fallback bundle|individual`	Specify the LACP fallback mode as individual or bundled.
`lacp-fallback-timeout seconds`	Specify the LACP fallback timeout in seconds. The range is between 30 and 60 seconds with a default value of 50 seconds.
`reflect|noreflect`	Specify physical port reflection
`edge-switch|no-edge-switch`	Specify if the switch is an edge switch.
`pause|no-pause`	Specify if pause frames are sent.
`description` *`description-string`*	Specify a description for the trunk configuration.
`loopback|no-loopback`	Specify loopback if you want to use loopback.
`vxlan-termination|no-vxlan-termination`	Specify if a VXLAN can terminate on the trunk.
`unknown-ucast-level` *`unknown-ucast-level-string`*	Specify an unknown unicast level in percent. The default value is 100%.
`unknown-mcast-level` *`unknown-mcast-level-string`*	Specify an unknown multicast level in percent. The default value is 100%.
`broadcast-level` *`broadcast-level-string`*	Specify a broadcast level in percent. The default value is 100%.
`port-mac-address` *`mac-address`*	Specify the MAC address of the port.
`loop-vlans` *`vlan-list`*	Specify a list of VLANs for looping.
`routing|no-routing`	Specify if the port participates in routing on the network.
`host-enable|host-disable`	Enable or disable host facing ports.

`defer-bringup|no-defer-bringup`	Specify to delay bringing up the host.
`dscp-map dscp-map name|none`	Specify the DSCP map name to enable on a port
`local-switching|no-local-switching`	Specify local-switching or no-local-switching. A no-local-switching port cannot bridge traffic to another no-local-switching port
`allowed-tpid q-in-q|q-in-q-old`	Specify an allowed TPID in addition to 0x8100 for VLAN headers.
`fabric-guard|no-fabric-guard`	Specify if fabric guard is enabled.
`fec|no-fec`	Specify Port Forwarding Error Correction (FEC) mode.

For example, to remove port 2 from the trunk, issue the command:

```
CLI (network-admin@switch) > trunk-modify name trunk-1 port 1,3
```

View the updated trunk configuration:

```
CLI (network-admin@switch) > trunk-show

name      port  speed  autoneg  jumbo
------- ---- ----- ------- -----
trunk-1  1,3   10g    off       off
```

To remove the trunk configuration from the switch, use the `trunk-delete` command:

```
CLI (network-admin@switch) > trunk-delete
```

`trunk-delete`	Delete a trunk configuration.
`name` *`name-string`*	Specify the name of the trunk configuration.
`trunk-id` *`trunk-id-number`*	Specify the trunk ID.
`description` *`description-string`*	Specify the trunk description to delete the trunk configuration. When you delete a trunk configuration, the trunk description is removed from the member ports.

```
CLI (network-admin@switch) > trunk-delete name trunk-1
```

Verify that the trunk configuration is removed again by using the `trunk-show` command.

Configuring Active-Standby Link Failover on Management Interfaces

With switches that have dual management ports, you can aggregate them in a LAG with both of them being active and forwarding. This is useful when increased bandwidth is required for management purposes or when the upstream management network supports solutions like MG-LAG (aka virtual chassis) to remove single points of failure.

Alternatively, when it's desirable that only one of the two management interfaces be forwarding, the so-called "active-standby" or "active-backup" mode of operation is supported for LAG on management ports.

In this case, the management interfaces form a LAG with only one of the two members active at any time. The other link is in standby mode, which means that it becomes forwarding only when the primary link goes down. When the primary link comes back up, it rejoins the LAG and the backup link returns to standby state.

Use the `switch-setup-modify` command to set up a management LAG.

Use the active-standby option to enable the active-backup mode.

```
CLI (network-admin@switch) > switch-setup-modify mgmt-lag <disable| enable
|  active-standby>
```

Configuring Link Aggregation Control Protocol (LACP)

The Link Aggregation Control Protocol (LACP) is a dynamic LAG protocol that is controlled by user configuration. The negotiation process and the addition/removal of ports is controlled by parameters like LACP mode (active, passive and off) and LACP priorities.

To dynamically bring up a trunk the ports on at least one of the two peer LACP switches must be set to active mode.

For example use the following command to dynamically bundle ports 20-23:

```
CLI (network-admin@switch) > trunk-create name trunk23 ports 20-23 lacp-mode active
```

To modify the mode of a trunk running LACP, use the command:

```
CLI (network-admin@switch) > trunk-modify name trunk23 lacp-mode passive
```

To modify a port configuration with a non default LACP priority, use the command:

```
CLI (network-admin@switch) > port-config-modify port 33 lacp-priority 34
```

To enable or disable LACP on a switch, and/or to change the default system priority value, you can use the command:

```
CLI (network-admin@switch) > lacp-modify enable|disable system-priority priority
```

The default system-priority value is 32768 with a range of 1 to 65535.

To display LACP information, use the command:

```
CLI (network-admin@switch) > lacp-show

switch:                 switch
enable:                 yes
system-priority:        32768
systemid:               800640e942c007a
```

Configuring a Cluster

To set up a cluster of two switches, say, `pleiades1` and `pleiades2`, you must first verify that they both belong to the same fabric instance, in this case `corp-fab`:

```
CLI (network-admin@switch) > fabric-node-show layout vertical

id: 184549641
name: pleiades1
fab-name: corp-fab
fab-id: b000109:5695af4f
cluster-id: 0:0
local-mac: 3a:7f:b1:43:8a:0f
fabric-network: in-band
control-network: in-band
mgmt-ip: 10.9.19.203/16
mgmt-mac: ec:f4:bb:fe:06:20
mgmt-secondary-macs:
in-band-ip: 192.168.168.203/24
in-band-mac: 3a:7f:b1:43:8a:0f
in-band-vlan: 0
in-band-secondary-macs:
fab-tid: 1
cluster-tid: 0
out-port: 0
version: 2.4.204009451,#47~14.04.1-Ubuntu
state: online
firmware-upgrade: not-required
device-state: ok
ports: 104
```

To create a cluster configuration, use the following command:

```
CLI (network-admin@switch) > cluster-create name cluster1 cluster-node-1
pleiades1 cluster-node-2 pleiades2
```

To verify the status of the cluster, use the `cluster-show` command:

```
CLI (network-admin@switch) > cluster-show

name       state    cluster-node-1 cluster-node-2
-------- ------- -------------- --------------
cluster1 online   pleiades1         pleiades2
```

To replace a failed cluster node, use the fabric-join `repeer-to-cluster-node` command.

To display information about the cluster, use the `cluster-info` command:

```
CLI (network-admin@switch) > cluster-info format all layout vertical

 name:                          cluster1
 state:                         online
 cluster-node-1:                pleiades1
```

```
cluster-node-2:                 pleiades2
tid:                            2
mode:                           master
ports:                          69-71,129
remote-ports:                   69-71,128
```

Performing the Cluster Re-peer Process

In order to rebuild a cluster over a Layer 2 or Layer 3 fabric it's necessary to use the `fabric-join` command with the `repeer-to-cluster-node` option.

For example, if one wants to join `switch1` from `switch3` to rebuild a cluster pair (that previously lost `switch2`), the following commands can be used:

- In case of a Layer 2 fabric:

```
CLI (network-admin@switch3) > fabric-join repeer-to-cluster-node switch1
```

- In case of a Layer 3 fabric (using for example OSPF or BGP for interconnection), the command changes to:

```
CLI (network-admin@switch3) > fabric-join repeer-to-cluster-node switch1 ?

password            plain text password
abort-on-conflict  fabric join is aborted
delete-conflicts    conflicts are deleted
over-l3             fabric is over layer 3
```

Note: The *`over-l3`* option is applicable only on Dell and Edgecore switches, where the fabric is over Layer 3 network.

```
CLI (network-admin@switch3) > fabric-join repeer-to-cluster-node switch1
over-l3

Joined fabric myFabric. Restarting nvOS...
Please enter username and password:
     Username (network-admin):
     Password:
Connected to Switch switch3; nvOS Identifier:0x900093c; Ver: 6.0.1-
6000116966
```

You can display the information about the fabric instance of the local switch using the `fabric-info` command:

```
CLI (network-admin@switch3) > fabric-info format all layout vertical

name:                           myFabric
id:                             a000030:5537b46c
vlan:                           3
fabric-network:                 in-band
control-network:                in-band
tid:                            365
fabric-advertisement-network:   inband-only
over-l3:                        true
```

Configuring Cluster over Layer 3

Clustering switch pairs over Layer 3 requires the creation of VXLAN tunnels between the cluster nodes through the intermediate (spine) switch(es). The VXLAN tunnels function as *virtual* cluster links and carry both control traffic (if control is over in-band) and data traffic.

Please refer to the *Configuring VXLAN* chapter for more information on the VXLAN technology and configuration.

Note: A cluster over Layer 3 configuration takes longer to achieve traffic convergence in comparison to a traditional cluster using direct cluster links. This is because in the former case cluster communication is made possible by VXLAN tunnels, which come up only after Layer 3 convergence.

Note: Currently cluster over Layer 3 uses automatic tunnels called auto-tunnel-cluster (as shown in the figure below). It is not compatible with manual VXLAN tunnels.

Note: Starting from release 7.0.0/7.0.1, NetVisor OS supports the cluster over Layer 3 feature on the following platforms:

- On NetVisor 7.0.0: Freedom (Edgecore)- F9480-V (AS7326-56X), F9460-X (AS5835-54X), F9460-T (AS5835-54T)
- On NetVisor 7.0.0: Dell S5200 Series, NRU03, and NRU-S0301.
- On NetVisor 7.0.1: Dell S4100 Series

Consider for example, the topology below where the cluster over Layer 3 feature is configured. In this figure, two cluster pairs are connected via Layer 3 over a spine node.

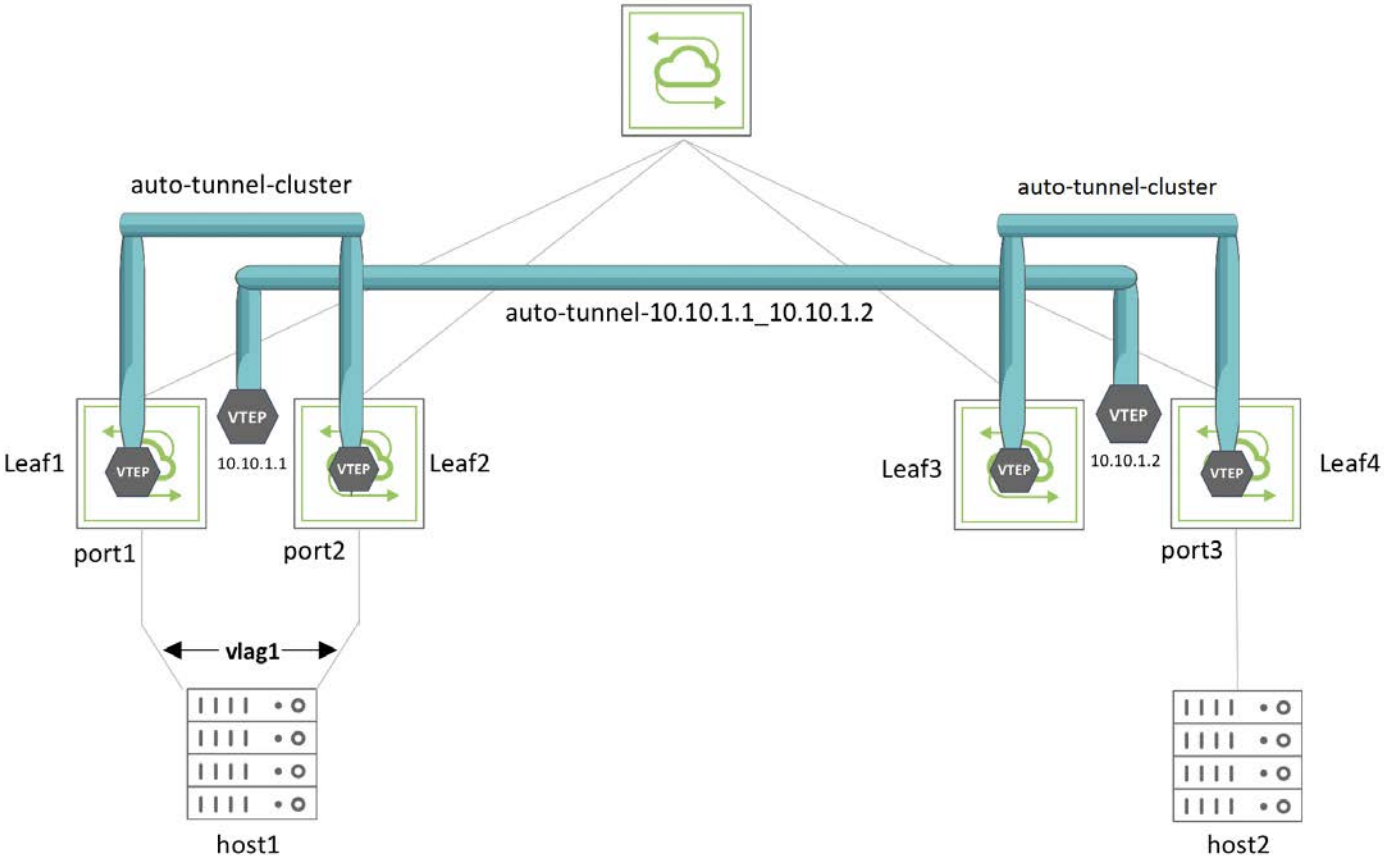


Figure 7-8 – Cluster over L3 Topology with vLAG

The VXLAN tunnel called `auto-tunnel-cluster` in ***Figure 7-8*** is a special local tunnel between the leaf switch pairs created over the interconnections with the spine switch. The tunnel called `auto-tunnel-10.10.1.1_10.10.1.2` is the regular VXLAN connection between VTEP VIPs 10.10.1.1 and 10.10.1.2. The vLAG named `vlag1` is formed using port1 of Leaf1 and port2 of Leaf2.

To configure cluster over Layer 3, use the `cluster-create` command:

```
CLI (network-admin@Leaf1) > cluster-create
```

`cluster-name`	Specify the name of the cluster.
`cluster-node-1 fabric-node name`	Specify the name of the first switch in the cluster
`cluster-node-2 fabric-node name`	Specify the name of the second switch in the cluster
`vxlan 0..16777215`	Specify the VXLAN ID for cluster over L3.
`node1-ip ip-address`	Specify the IP address of cluster node 1.
`node2-ip ip-address`	Specify the IP address of cluster node 2.
`validate|no-validate`	Validate the inter-switch links and state of the switches in the cluster.
`cluster-sync-timeout milliseconds`	Specify the amount of time before a cluster times out during synchronization. The allowed range is between 500ms and 2000ms.

`cluster-sync-offline-count`	`number`	Specify the number of missed synchronizations before the cluster goes offline.

Note: The VXLAN ID should be a non-null value to enable cluster over L3.

For example, to configure the cluster that includes Leaf1 and Leaf2, use the command:

```
CLI (network-admin@Leaf1) > cluster-create name cluster1 cluster-node-1
Leaf1 cluster-node-2 Leaf2 vxlan 40940 node1-ip 192.168.1.1 node2-ip
192.168.1.2
```

This command creates the bidirectional tunnel `auto-tunnel-cluster` between Leaf1 and Leaf2 with a local scope and adds the cluster VXLAN ID to the tunnel.

To view the cluster configuration, use the command:

```
CLI (network-admin@Leaf1) > cluster-show format state,cluster-node-
1,cluster-node-2,vxlan,tunnel-name,node1-ip,node2-ip,mode,ports,remote-
ports
```

```
state  cluster-node-1  cluster-node-2  vxlan tunnel-name         local-ip    remote-ip   mode   ports   remote-ports
------ --------------- --------------- ----- ------------------- ----------- ----------- ------ ------- ------------
online Leaf1           Leaf2           40940 auto-tunnel-cluster 192.168.1.1 192.168.1.2 slave  none    none
online Leaf1           Leaf2           40940 auto-tunnel-cluster 192.168.1.1 192.168.1.2 master none    none
online Leaf3           Leaf4           0                         ::          ::          master 1-3,272 1-3,272
online Leaf3           Leaf4           0                         ::          ::          slave  1-3,272 1-3,272
```

To view the tunnel configuration, use the command:

```
CLI (network-admin@Leaf1) > tunnel-show format name,scope,local-ip,remote-
ip,state
```

```
name                             scope   local-ip    remote-ip   state
-------------------------------  -------  ----------  ----------  -----
auto-tunnel-cluster               local    192.168.1.1 192.168.1.2 ok
auto-tunnel-10.10.1.1_10.10.1.2   cluster  10.10.1.1    10.10.1.2    ok
```

The auto-created tunnel `auto-tunnel-10.10.1.1_10.10.1.2` between endpoints 10.10.1.1 and 10.10.1.2 links the cluster of Leaf1 and Leaf2 to the cluster of Leaf3 and Leaf4.

To display the tunnels and the associated VXLAN IDs, use the command:

```
CLI (network-admin@Leaf1) > tunnel-vxlan-show
```

```
name                                vxlan
----------------------------------  -----
auto-tunnel-cluster                 40940
auto-tunnel-cluster                 1000
auto-tunnel-cluster                 1001
auto-tunnel-cluster                 1002
auto-tunnel-10.10.1.1_10.10.1.2     1000
```

Use the `vlag-show` command to display the vLAG configuration:

```
CLI (network-admin@Leaf1*) > vlag-show format name,switch,port,peer-
```

```
switch,peer-port,status,local-state,peer-state,

name   switch  port  peer-switch  peer-port  status  local-state  peer-state
-----  ------  ----  -----------  ---------  ------  -----------  ----------
vlag1 Leaf1   1     Leaf2         2          normal enabled,up   enabled,up
```

You can modify a traditional cluster with direct links to a cluster over Layer 3 with VXLAN tunnels by using the `cluster-modify` command. You must **disable** the direct links before making this change in configuration:

```
CLI (network-admin@Leaf1) > cluster-modify name cluster10 vxlan 8188 node1-ip 10.10.11.1 node2-ip 10.10.11.2
```

Note: Arista Networks requires that the direct links be disabled and recommends rebooting both the cluster switches after you modify a traditional cluster with direct links to a cluster over Layer 3.

To perform the reverse transition from cluster over Layer 3 to traditional cluster, you must enable the direct cluster link(s) and specify `vxlan` as 0 in the `cluster-modify` command, like so:

```
CLI (network-admin@Leaf1) > cluster-modify name cluster10 vxlan 0
```

Note: Arista Networks recommends rebooting both the cluster switches after you modify a cluster over Layer 3 to a traditional cluster with direct links.

Note: When configuring BGP on the network to support the cluster over L3 functionality, you can use either of the two following options: (a) use eBGP with `allow-as-in-origin` or (b) use iBGP with the recommended practices as below:

a. You allocate a dedicated port (not a cluster port or a loopback port, which are not suitable) and assign a VLAN with a vRouter interface to it so that the iBGP interface stays up. Also, ensure that the VNI associated to the VLAN is added to the cluster's VTEP.

b. Use a route map, as shown below, on both cluster nodes, configured for in-band communication to stay up:

```
CLI (network-admin@Leaf1) > vrouter-prefix-list-add vrouter-name <vrouter-name> name <list-name> seq <number> action permit prefix <peer ip/mask>

CLI (network-admin@Leaf1) > vrouter-route-map-add vrouter-name <vrouter-name> name <map-name> seq <number> action permit match-prefix <list-name>

CLI (network-admin@Leaf1) > vrouter-bgp-add vrouter-name <vrouter-name> neighbor <bgp-neighbor-ip> remote-as <as-number> next-hop-self bfd multiprotocol ipv4-unicast,ipv6-unicast weight 0 neighbor-keepalive-interval 10 neighbor-holdtime 30 route-map-out <map-name>
```

Starting from release 7.0.0 NetVisor OS also supports the configuration of VXLAN-based bridge domains (BDs) on clusters over Layer 3. (Refer to the *Configuring Advanced Layer 2 Transport Services* chapter for more details on bridge domains.)

With this enhancement it is not required to allocate reserved VLANs for clusters over Layer 3. However, a reserved VLAN is still needed if the BD needs to traverse a traditional cluster. (Within a fabric it's possible

to have a mix of traditional clusters and clusters over Layer 3.)

It is also possible to modify a traditional cluster to become a cluster over Layer 3, and vice versa, when BDs are configured on it. Cluster links must be *disabled* before configuring a cluster over Layer 3.

On the other hand, the transition from cluster over Layer 3 to a traditional cluster requires a reserved VLAN to have been provisioned and the cluster links to be operational.

To configure a cluster over Layer 3 you can use the `cluster-create` command as shown above:

```
CLI (network-admin@Leaf1) > cluster-create name cluster1 cluster-node-1
Leaf1 cluster-node-2 Leaf2 vxlan 40940 node1-ip 192.168.1.1 node2-ip
192.168.1.2
```

Then a BD can be configured like so:

```
CLI (network-admin@Leaf1) > bridge-domain-create name bd-cluster-o-l3 scope
fabric vxlan 800800 [rsvd-vlan 4080]
```

where the reserved VLAN is no longer needed if the BD doesn't need to traverse a traditional cluster.

Then the BD can be configured on the desired set of ports, for example like so:

```
CLI (network-admin@Leaf1*) > bridge-domain-port-add name bd-cluster-o-l3
port 1 outer-vlan 1000 inner-vlan 800

CLI (network-admin@Leaf1*) > l2-table-show bd bd-cluster-o-l3 format
mac,bd,vlan,inner-vlan,vxlan,ip,ports,state,status,

mac               bd              vlan inner-vlan vxlan  ip           ports state  status
----------------- --------------- ---- ---------- ------ ------------ ----- ------ ------
00:20:fc:81:4d:08 bd-cluster-o-l3       800         800800 205.95.95.62        tunnel host
00:12:c0:80:33:1e bd-cluster-o-l3 1000 800         800800 205.95.95.51 1      active host

CLI (network-admin@Leaf1*) > bridge-domain-show□

name □□□□□□   scope  □ports vxlan-inner-packet mac-learning l2-tunneling□
--------------- ------ ----- ----------------- ------------ ------------□
bd-cluster-o-l3 fabric 1 □□   auto □□□□□□□     on □□□□□   none□
```

Configuring a vLAG

If you want to connect the two cluster nodes to an upstream switch or to a downstream host in a redundant fashion, you can configure a vLAG between the uplinks or the downlinks on the cluster nodes.

For example, if `switch1` has port 53 connected to an upstream switch and switch2 has port 19 connected to the same upstream switch, create a vLAG by executing the `vlag-create` command on either of the cluster switches (switch1 in this example):

```
CLI (network-admin@switch1) > vlag-create name uplink port 53 [peer-switch
switch2] peer-port 19
```

To verify the configuration, use the following command:

```
CLI (network-admin@switch1) > vlag-show

name   cluster   mode          switch  port peer-switch peer-port status local-state lacp-mode
------ --------- ------------- ------- ---- ----------- --------- ------ ----------- ---------
uplink cluster1  active-active switch1 53   switch2     19        normal enabled     active
```

> **Note:** Before you can create a vLAG, you must configure the two switches in a cluster.

Starting from release 7.0.0, NetVisor OS supports an enhancement for active-standby mode, for example when a single main vLAG link needs to be active while the backup link is required to become (and remain) active only when the main one fails.

You can enable this mode with the command:

```
CLI (network-admin@switch1) > vlag-create name <name> port <id> peer-switch
<peer-name> peer-port <id> mode active-standby
```

> **Note**: `lacp-mode` must be set to `active` for mode `active-standby` to work. `lacp-mode off` is not supported with mode `active-standby`.

In this release, in `active-standby` mode the active port election would happen through LACP message exchanges and would be based on the first port to become operationally active (i.e., in `up` state). Consequently, the other port would become the standby port and would be in `phy-up,vlag-blocked` state, as shown in the example below:

```
CLI (network-admin@switch1) > port-show port 121

port bezel-port status                                    config
---- ---------- ------------------------------------------ ------
121  31         up,host,LLDP,LACP-PDUs,vlag-active,vlan-up fd,10g

CLI (network-admin@switch2) > port-show port 122

port bezel-port status                              config
---- ---------- --------------------------------- ------
122  31.2       phy-up,LLDP,vlag-blocked,LACP-wait 10g
```

```
CLI (network-admin@switch1) > vlag-show

name   cluster mode              switch   port peer-switch peer-port status local-state peer-state lacp-mode
-----  ------- --------------  -------  ---- -----------  ---------  ------ -----------  ----------  ---------
vlag1 cluster active-standby  switch1 121   switch2      122        normal enabled,up             active
```

Note: The active-standby mode and release 7.0.0's enhancements work with both traditional clusters and clusters over Layer 3.

Note: With active-standby mode, for Layer 3 connectivity and redundancy, a single VTEP VIP (VRRP VIP) would still be used along with active-active L3 forwarding. Due to the asymmetric nature of active-standby vLAG mode, upstream traffic arriving on the cluster node with the standby vLAG port will have to take an extra physical (or virtual) hop (i.e., it will have to traverse the cluster link, or a VXLAN tunnel in case of a cluster over Layer 3) to reach its destination over an asymmetric forwarding path.

In release 7.0.0 VXLAN bridge domains are supported on vLAG members configured in mode `active-standby`. It is recommended that the bridge domains be configured before configuring the vLAG(s) to carry them.

Modifying LACP Mode and Parameters on an Existing vLAG Configuration

You can modify the LACP mode as part of an existing vLAG configuration using the following command to change parameters like LACP mode or timeout:

```
CLI (network-admin@switch) > vlag-modify
```

`name name-string`	Specify the vLAG name.
`failover-move-L2|failover-ignore-L2`	If you specify the parameter failover-move-L2, NetVisor sends gratuitous ARPs.
`lacp-mode off|passive|active`	Specify the LACP mode as off, passive or active.
`lacp-timeout slow|fast`	Specify the LACP timeout as slow (90 seconds) or fast (3 seconds).
`lacp-fallback bundle|individual`	Specify the LACP fallback mode as individual or bundled.
`lacp-fallback-timeout 30..60`	Specify the LACP fallback timeout in seconds. The default is 50 seconds.

Configuring a (v)LAG in LACP fallback mode *individual (*vs. *bundle)* allows it to operate as individual ports in the absence of proper LACP negotiation with a network peer. This configuration is useful with hosts with multiple network interfaces that for example need to boot from a server on the network prior to booting the operating system on the local hard drive (this is known as PXE boot, based on the Pre-Boot Execution Environment). In this scenario the hosts would have to use individual ports until they can boot the operating system and start using LACP.

Once LACP is running and port members receive LACP PDUs from their peers, then the port members of the configured vLAG can get bundled to operate as a trunk.

With this configuration, NetVisor OS logically prepares the trunk on the switch but does not add any of the port members to it. The ports continue to operate individually until LACP PDUs are heard on them. At that point, the dynamic negotiation process is started, proper compatibility and communication is verified between the peers and the bundling process is carried out if all conditions are met. Only then all member ports cease to operate individually and are aggregated together as an operational trunk.

To give hosts enough time to PXE boot, a fallback timeout interval parameter is used (with a default value of 50 seconds). If no LACP PDUs are received for the number of seconds configured as the fallback timeout, the LACP logic checks if the LACP negotiation interval (`lacp-timeout`) has expired. If it has, then the port members fall back to individual mode. If it has not, another fallback timer is scheduled with a value equal to the fallback timeout.

For example:

- The LACP fallback timeout is set to 50 seconds and the LACP negotiation interval is set to 90 seconds (default).
- After 50 seconds, the fallback timer is rescheduled because the LACP negotiation interval has not expired yet.
- After an additional 40 seconds (90 seconds total), the LACP negotiation interval expires.

- Another 10 seconds pass (100 seconds total) when the fallback timer expires, no LACP PDUs have been received (for example due to the host still booting), then the member ports fall back to individual ports.

Both the negotiation/bundling and the fallback cases described above represent the on-demand characteristic of LACP-based port aggregation process (applicable to both vLAGs and normal trunks).

Configuring the Cluster Slave Switch Bring-up Process

Note: This feature is applied only during the initial start-up of the cluster slave switch.

When a (previously down) slave node comes back up and a cluster is thus restored to online state, it can take measurable time to change the hardware routing and Layer 2 tables. This set-up delay may cause some non-negligible traffic loss on the network if lots of hardware changes are performed in bulk.

Therefore, to minimize service disruption to a negligible interval, the bring-up process is performed in an optimized staggered fashion by incrementally restoring/updating the ports. (This mode can be changed as shown below.) Some delays are also added during the staggered bring-up to make sure that state transitions and synchronization complete before moving to the next set of changes. (Such delays are configurable as shown below.)

In order to minimize any traffic impact, all non-Layer 3 and non-vLAG ports are restored first. This allows the cluster links to activate and the cluster configuration and state to synchronize.

After synchronization is complete, Layer 3 ports are brought up first. After that, NetVisor OS waits for a configurable `l3-to-vlan-interface-delay` before bringing up vRouter vlan interfaces. This is so that routing protocols can delay advertising directly-attached subnets upstream.

After Layer 3 ports are brought up, NetVisor OS also waits for another configurable l3-to-vlag-delay to allow time for the routing protocols to converge and program the routes. This time defaults to 15 seconds. After that delay, the vLAG ports are brought up staggered. (Note that `l3-to-vlan-interface-delay` should be less than or equal to `l3-to-vlag-delay`.)

The maximum-sync-delay parameter controls the maximum time to wait for synchronization in the case where the cluster cannot synchronize information.

Note that, if the node coming up is a cluster's master, then no staggering and no Layer 3-to-vLAG delay is applied (because it's not beneficial).

With version 5.1.1, NetVisor OS also supports the staggered bring up of vRouter VNICs. When configured, the vRouter VNICs, which are not configured on Layer3 ports. are brought up in a staggered manner during the nvOS boot-up on a cluster peer. You can specify the wait time (0-60000 ms) between NIC bring up.

By default, the vRouter interfaces for which the VLAN is up is brought up simultaneously. The bring up can be staggered by specifying a non-zero vrouter-if-staggered-interval. The staggered bring up process is helpful to reduce the traffic loss caused by the simultaneous bring up of all VNICs.

Note: The vRouter interfaces on Layer 3 ports are brought up first and is not staggered. Also, after bootup, the subsequent VLAN interfaces being down or up is not affected by the staggered configuration.

In most cases it is not necessary to change the default parameter values.

However, in certain scenarios to be able to further optimize the slave bring up process to match one's specific requirements, it is possible to modify the parameters by using the following command:

```
CLI (network-admin@switch) > cluster-bringup-modify
```

You can specify one or more of the following options:

`l3-port-bringup-mode staggered| simultaneous`	Specify the Layer 3 port bring up mode during start up.
`l3-port-staggered-interval duration: #d#h#m#s`	Specify the interval between Layer 3 ports in Layer 3 staggered mode. This can be in days, hours, minutes, or seconds.
`vlag-port-bringup-mode staggered| simultaneous`	Specify the vLAG port bring up mode during start up.
`vlag-port-staggered-interval duration: #d#h#m#s`	Specify the interval between vLAG ports in vLAG duration: This can be in days, hours, minutes, or seconds.
`maximum-sync-delay duration: #d#h#m#s`	Specify the maximum delay to wait for cluster to synchronize before starting Layer 3 or vLAG port bring up. This can be in days, hours, minutes, or seconds.
`l3-to-vlag-delay duration: #d#h#m#s`	Specify the delay between the last Layer 3 port and the first vLAG port bring up. This can be in days, hours, minutes, or seconds. The default value is 15 seconds.
`l3-to-vlan-interface-delay duration: #d#h#m#s`	Specify the delay between last L3 port and vRouter vlan interface bring up. This can be in days, hours, minutes, or seconds.
`port-defer-bringup-delay duration: #d#h#m#s`	Specify the global timer value to be used for port delay-bring up.
`port-defer-bringup-mode staggered| simultaneous`	Specify the port defer bring up mode during start up.
`port-defer-bringup-staggered-interval duration: #d#h#m#s`	Specify the interval between ports in defer bring up mode.
`vrouter-if-bringup-mode staggered| simultaneous`	Specify the vRouter VLAN interface bring up mode.
`vrouter-if-staggered-interval 0..60000`	Specify the interval in ms between vRouter VLAN interface bring up in staggered mode. The value ranges between 0-60000 ms.

To display the status of the cluster bring up process, use the `cluster-bringup-show` command:

```
CLI (network-admin@switch) > cluster-bringup-show

state: ports-enabled
l3-port-bringup-mode: staggered
l3-port-staggered-interval: 3s
vlag-port-bringup-mode: staggered
vlag-port-staggered-interval: 3s
maximum-sync-delay: 1m
```

```
l3-to-vlag-delay:  15s
l3-to-vlan-interface-delay:  0s
```

Restoring Ports for Cluster Configurations

Note: This feature is applied only during the initial start-up of the network.

Cluster configuration supports sub-second traffic loss in case of fail-over events.

There are two types of ports providing redundant data paths:
- Layer 3 ports over ECMP redundant routed paths
- Virtual LAGs (vLAGs) providing redundant Layer 2 paths

During failover and recovery port events, it may take measurable time to change the hardware routing and MAC tables on larger networks. This delay incurs traffic loss on the network. To reduce delay, this feature allows you to incrementally restore the ports at start-up. By incrementally restoring the ports, the changes to the hardware are prevented from contending with each other. This reduces the delay between a port up and the hardware updates with the appropriate Layer 3 and Layer 2 information for the port. This process ensures sub-second fail over.

All non-Layer 3 and non-vLAG ports are restored first. This allows the cluster links to activate and the cluster configuration to synchronize information. Layer 3 and vLAG port restoration starts after the cluster synchronizes. This is predicated on the cluster becoming active, all Layer 2 and Layer 3 entries, such as status updates, exchanged, cluster STP status synchronized, and all router interfaces initialized.

NetVisor OS enforces the following sequence for port bring up (not guaranteed for first upgrade to NetVisor OS and is allowed only for subsequent reboots):

1. Cluster ports, VXLAN-loopback-trunk ports, and Loopback ports
2. Layer 3 ports
3. vLAG ports
4. All other ports

The sequence for port bring-down is:

1. Orphan ports (non-vLAG, non-Layer3, and non-Cluster)
2. vLAG ports
3. Layer 3 ports
4. All other ports

If a port is configured with `defer-bringup` parameter, then that port is brought up along with other ports. All ports except cluster ports can be configured for `defer-bringup` using the `port-config-modify` command. You can specify the global timer value for delaying the port bringup by using the `cluster-bringup-modify` command with `port-defer-bringup-delay duration` parameter.

The parameter, `maximum-sync-delay`, controls the maximum time to wait for synchronization in the case where the cluster cannot synchronize information. After synchronization is complete, Layer 3 ports are restored first, since Layer 3 traffic can traverse the cluster link to the peer VLAG port if needed. Currently the reverse is typically not true.

If vLAG ports are restored first, a Layer 3 adjacency between the two cluster nodes may be needed but may not exist in some network configurations. After Layer 3 ports are restored, NetVisor OS waits a configurable Layer 3 port to vLAG delay to allow time for the routing protocols to converge and insert the

routes. The delay time defaults to 15 seconds.

After the delay, the vLAG ports are restored incrementally and restoring ports incrementally allows enough time to move Layer 2 entries from the cluster link to the port. This incremental restoration of ports also allows the traffic loss to occur in small, 200-300ms per port, rather than one large time span. This is particularly important for server clusters where temporary small losses are not problematic, but fail or timeout for a large continuous traffic loss. If the node coming up is the cluster master, then no staggering and no Layer 3 to VLAG wait is applied. And if the node is the cluster master node, that means the peer is down or coming up, and not handling traffic. Therefore NetVisor OS safely restores the ports as soon as possible to start traffic flowing between the nodes.

In addition, with version 5.1.1, NetVisor OS supports the staggered bring up of vRouter vNICs. When configured, the vRouter vNICs, which are not configured on Layer3 ports, are brought up in a staggered manner during the nvOS boot-up on a cluster peer. You can specify the wait time (0-60000 ms) between NIC bring up.

By default, the vRouter interfaces for which the VLAN is up is brought up simultaneously. The bring up can be staggered by specifying a non-zero `vrouter-if-staggered-interval`. The staggered bring up process is helpful to reduce the traffic loss caused by the simultaneous bring up of all VNICs.

> **Note:** The vRouter interfaces on Layer 3 ports are brought up first and is not staggered. Also, after bootup, the subsequent VLAN interfaces being down or up is not affected by the staggered configuration.

To configure or modify the port bring up process, use the command:

```
CLI (network-admin@Leaf1) > cluster-bringup-modify
```

`cluster-bringup-modify`	Modifies the cluster bring up configuration.
Specify one or more of the following options	
`l3-port-bringup-mode staggered| simultaneous`	Specify the Layer 3 port bring up mode during start up.
`l3-port-staggered-iterval duration: #d#h#m#s`	Specify the interval between Layer 3 ports in Layer 3 staggered mode. This can be in days, hours, minutes, or seconds.
`vlag-port-bringup-mode staggered| simultaneous`	Specify the vLAG port bring up mode during start up.
`vlag-port-staggered-interval duration: #d#h#m#s`	Specify the interval between vLAG ports in vLAG staggered mode. This can be in days, hours, minutes, or seconds.
`cluster-trunk-port-staggered-interval duration: #d#h#m#s`	Specify the interval between cluster trunk ports in staggered mode. This can be in days, hours, minutes, or seconds.
`maximum-sync-delay duration: #d#h#m#s`	Specify the maximum delay to wait for cluster to synchronize before starting Layer 3 or vLAG port bring up. This can be in days, hours, minutes, or seconds.

`l3-to-vlag-delay duration: #d#h#m#s`	Specify the delay between the last Layer 3 port and the first vLAG port bring up. This can be in days, hours, minutes, or seconds. The default value is 15 seconds.
`l3-to-vlan-interface-delay duration: #d#h#m#s`	Specify the delay between the last Layer 3 port and the vRouter VLAN interface bring up.
`port-defer-bringup-delay duration: #d#h#m#s`	Specify the global timer value to be used for port delay-bring up.
`port-defer-bringup-mode staggered| simultaneous`	Specify the port defer bring up mode during start up.
`port-defer-bringup-staggered-interval duration: #d#h#m#s`	Specify the interval between ports in defer bring up mode.
`vrouter-if-bringup-mode staggered| simultaneous`	Specify the vRouter VLAN interface bring up mode.
`vrouter-if-staggered-interval 0..60000`	Specify the interval in ms between vRouter VLAN interface bring up in staggered mode. The value ranges between 0-60000 milli-seconds.
`start-port-enable-delay 0..240`	Specify the time delay (sec) to start enabling ports (including cluster ports) on the standby switch.

To display the status of the cluster bring up process, use the `cluster-bringup-show` command:

```
CLI (network-admin@Leaf1) > cluster-bringup-show
switch:                                 Leaf1
state:                                  ports-enabled
l3-port-bringup-mode:                   staggered
l3-port-staggered-interval:             3s
vlag-port-bringup-mode:                 staggered
vlag-port-staggered-interval:           3s
maximum-sync-delay:                     1m
l3-to-vlag-delay:                       15s
l3-to-vlan-interface-delay:             0s
port-defer-bringup-delay:               30s
port-defer-bringup-mode:                 staggered
port-defer-bringup-staggered-interval:   0s
vrouter-if-bringup-mode:                 staggered
vrouter-if-staggered-interval(ms):       0
```

To display the status of ports with `defer-bringup` details on select ports, use the command:

```
CLI (network-admin@Leaf1) > port-config-show format port,cluster-
port,defer-bringup,vlag, port 11,15,42,49

port  cluster-port  defer-bringup  vlag
----  ------------  -------------  ---------
11    no            no             host-vlag
15    no            yes
42    yes           no
```

```
49    no              no
```

To view the details of a specified port, use the command:

```
CLI (network-admin@Leaf1) > port-show port 49 format
port,ip,mac,vlan,status,config

port ip         mac               vlan status                                          config
---- ---------- ----------------- ---- ------------------------------------------------ ------
49   50.50.50.2 66:01:01:01:01:01 4092 up,PN-fabric,LLDP,l3-port,remote-l3-port,vlan-up fd,10g
```

Configuring Active-Active vLAGs: a Step-by-Step Example

Note: As displayed in ***Figure 7-9***, there must be a physical connection between PN-0 and PN-1 before you can configure a vLAG.

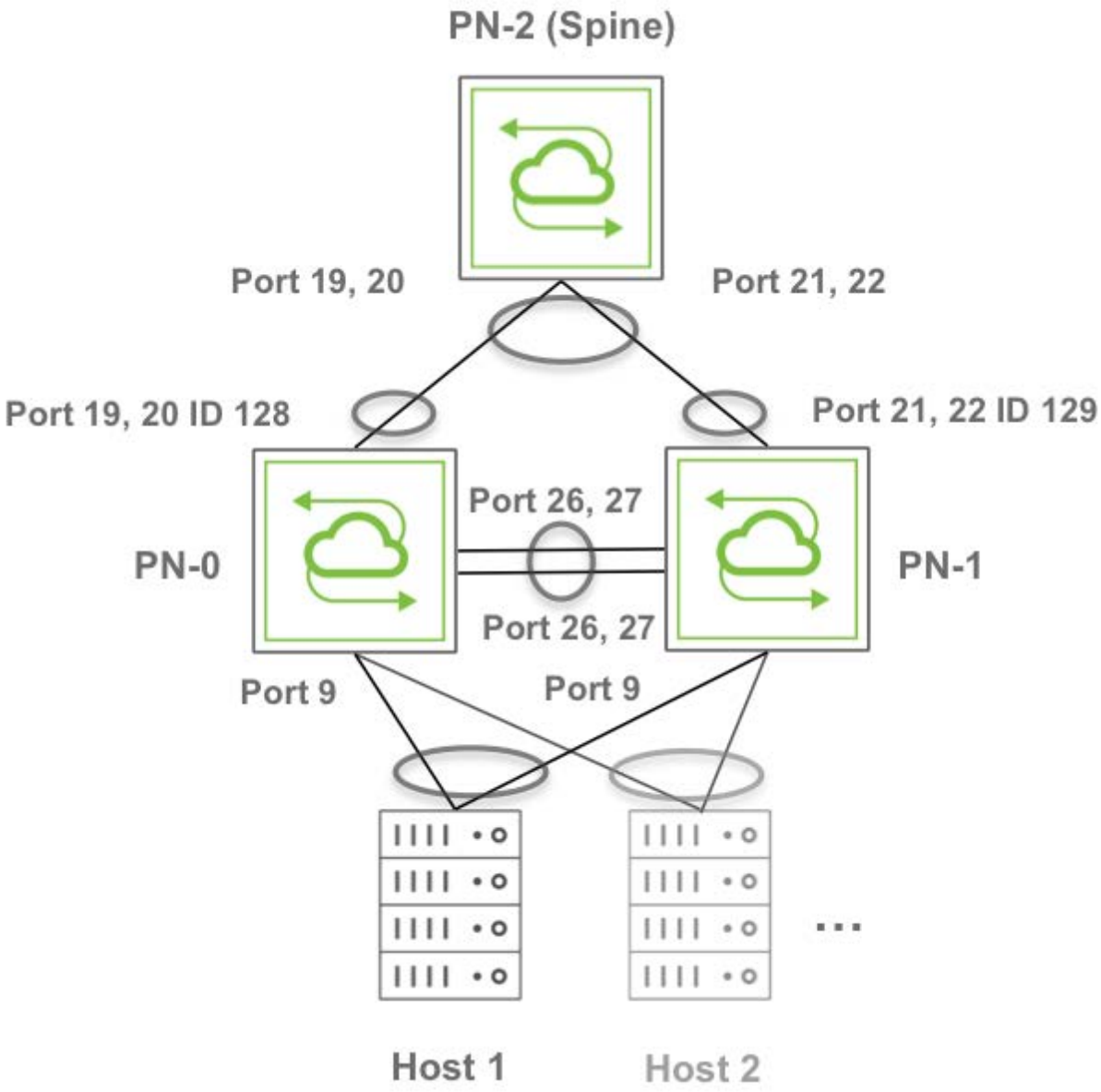


Figure 7-9 - Active-Active vLAGs Toward a Dual-Homed Host and a Spine Switch

The sample topology in ***Figure 7-9*** above comprises three Arista switches that are part of the same fabric instance, fab-vLAG. The spine switch is set as RSTP root.

It's important to note that ports 19-20 on PN-0 and ports 21-22 on PN-1 are connected to PN-2 (Spine). Ports 26, 27 interconnect PN-0 and PN-1 for the cluster link configuration required to set up a vLAG.

You can use the following steps to configure the above-pictured configuration with active-active vLAGs.

1) On the spine switch PN-2 use the following command:

```
CLI (network-admin@switch) > stp-modify bridge-priority 4096
```

2) Create the fabric and add the switches to it. On PN-2 use the `fabric-create` command:

```
CLI (network-admin@switch) > fabric-create name fab-VLAG
```

On PN-0 and PN-1 join the fabric:

```
CLI (network-admin@switch) > fabric-join name fab-VLAG

CLI (network-admin@switch) > fabric-join name fab-VLAG
```

3) Create VLAN connectivity from the spine switch all the way to the host. On PN-2 create for example VLAN 25 with scope fabric:

```
CLI (network-admin@switch) > vlan-create id 25 scope fabric
```

On PN-0 and PN-1 add VLAN 25 and untag the port connected to the host:

```
CLI (network-admin@switch) > vlan-port-add vlan-id 25 untagged ports 9

CLI (network-admin@switch) > vlan-port-add vlan-id 25 untagged ports 9
```

On PN-0 and PN-1 make the port connected to the host be an edge port:

```
CLI (network-admin@switch) > stp-port-modify port 9 edge

CLI (network-admin@switch) > stp-port-modify port 9 edge
```

4) Create a cluster configuration between PN-1 and PN-0. This creates the cluster link between ports 26, 27. On PN-0 enter the `cluster-create` command:

```
CLI (network-admin@switch) > cluster-create name VLAG cluster-node-1 PN-0
cluster-node-2  PN-1
```

5) In this example, for simplicity's sake we will use static trunks toward the spine switch to create a vLAG. (LACP's active mode can be used too, instead of off mode.) In this case you would first disable ports between PN-2 and PN-0 and then create a static trunk between them. Therefore, on PN-0 modify the ports facing PN-2 like so:

```
CLI (network-admin@switch) > port-config-modify port 19,20 disable
```

6) Create a two-port trunk between PN-0 and PN-2 with those ports:

```
CLI (network-admin@switch) > trunk-create name pn0-to-pn2 ports 19,20 lacp-
mode off
```

```
CLI (network-admin@switch) > trunk-show format all layout vertical

switch: PN-0
trunk-id: 128
name: pn0-to-pn2
ports: none
speed: disable
egress-rate-limit: unlimited
autoneg: off
jumbo: on
enable: off
lacp-mode: off
lacp-priority: 0
lacp-timeout: slow
lacp-fallback: bundle
lacp-fallback-timeout: 50
lacp-individual: none
stp-port-cost: 2000
stp-port-priority: 128
reflect: off
edge-switch: no
pause: no
description:
loopback: off
receive-only: on
unknown-ucast-level: %
unknown-mcast-level: %
broadcast-level: %
lport: 0
rem-rswitch-port-mac: 00:00:00:00:00:00
rswitch-default-vlan: 0
status:
config:
trunk-hw-id: 0
send-port: 4294967295
routing: yes
host-enable: no
```

From the above output, you can verify the name and ID of the trunk configuration *pn0-to-pn2*. You need this information to create the vLAG.

7) On PN-1 repeat the same commands to create a static two-port trunk (LACP mode off) between PN-1 and PN-2. You would, therefore, disable ports between PN-2 and PN-1 and then create a static trunk between them. On PN-1 modify the ports facing PN-2 like so:

```
CLI (network-admin@switch) > port-config-modify port 21,22 disable

CLI (network-admin@switch) > trunk-create name pn1-to-pn2 ports 21,22 lacp-
mode off

CLI (network-admin@switch) > trunk-show format all layout vertical
```

```
switch: PN-1
intf: 129
name: pn1-to-pn2
port: 21-22
speed: 10g
autoneg: off
jumbo: off
enable: off
lacp-mode:  off
lacp-priority:  32768
lacp-timeout:  slow
reflect: off
edge-switch: no
pause: no
description:
loopback: off
mirror-only: off
lport: 0
rswitch-default-vlan:  0
port-mac-address: 06:60:00:02:10:80
status:
config:
send-port: 0
```

8) Create the vLAG from the bottom switches going upstream. Keep one side of the vLAG disabled while you configure this step. On PN-0 use the `vlag-create` command:

```
CLI (network-admin@switch) > vlag-create name to-spine port 128 peer-port
129 peer-switch PN-1 lacp-mode off mode active-active
```

On PN-2 create a normal 4-way trunk with the name `trunk-pn`:

```
CLI (network-admin@switch) > trunk-create name trunk-pn ports 19,20,21,22
lacp-mode off
```

9) Enable ports on all switches. On PN-2, PN-0 and PN-1 enter the `port-config-modify` command:

```
CLI (network-admin@switch) > port-config-modify port 19,20,21,22 enable

CLI (network-admin@switch) > port-config-modify port 19,20 enable

CLI (network-admin@switch) > port-config-modify port 21,22 enable
```

10) As a final step, create the server-facing active-active vLAG. In this case we will make it dynamic (LACP mode active). On PN-0 enter the `vlag-create` command:

```
CLI (network-admin@switch) > vlag-create name to-host port 9 peer-port 9
peer-switch PN-1 lacp-mode active mode active-active
```

Display and verify the vLAG configuration information:

```
CLI (network-admin@switch) > vlag-show format all layout vertical

id: a000024:0
name: to-host
cluster: VLAG
mode: active-active
switch: PN-0
port: 9
peer-switch: PN-1
peer-port: 9
failover-move-L2: no
status: normal
local-state: enabled, up
lacp-mode: active
lacp-timeout: slow
lacp-key: 26460
lacp-system-id: 110013777969246
```

Understanding LAG Path Selection and Load Balancing

For Layer 2 back-to-back connectivity, Arista NetVisor OS supports the standard link aggregation technology in order to combine multiple network connections into a logical pipe called a Link Aggregation Group (LAG), or '(port) trunk', which can provide redundancy in case of single or multiple link failure.

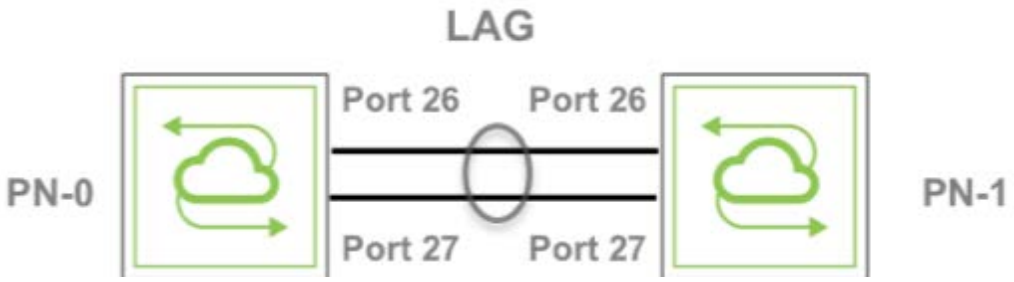


Figure 7-10: Two-port LAG Example

NetVisor OS also supports link aggregation across two redundant chassis to implement multi-pathing without requiring the creation of Layer 2 loops and the use of the Spanning Tree protocol. This feature is called *vLAG* (Virtual Link Aggregation Group) on a *switch cluster*.

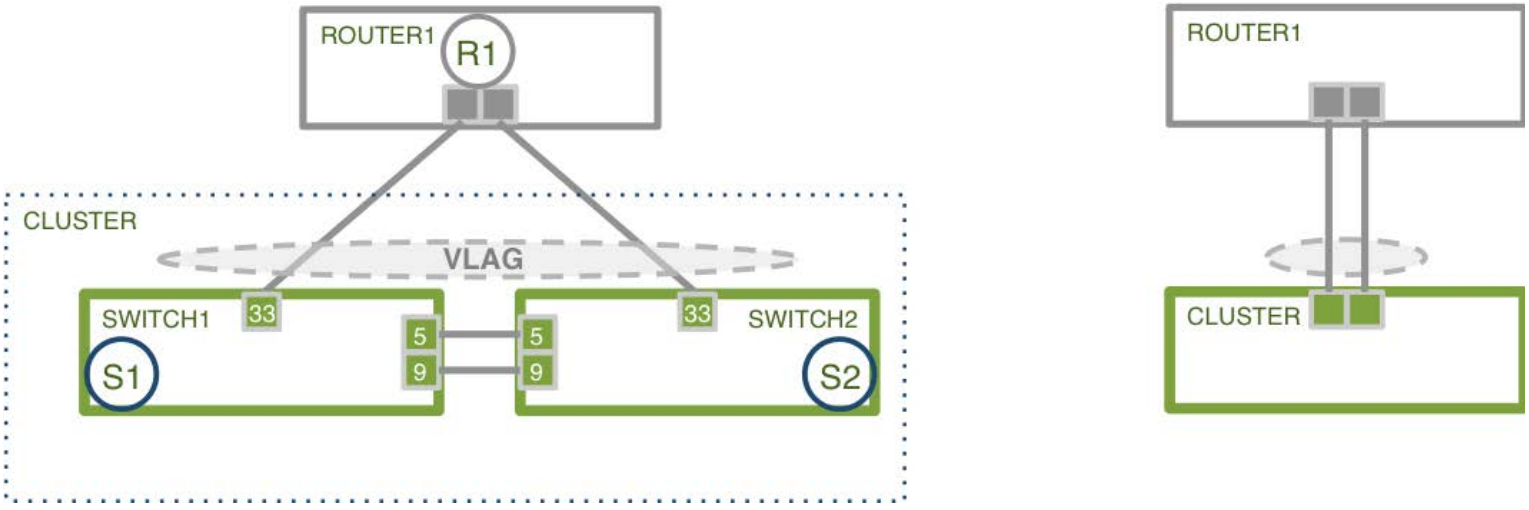


Figure 7-11: Two-port vLAG Example

Both the aforementioned technologies perform path selection in hardware using a high-performance technology called *packet field hashing*.

What that means is that the hardware extracts a number of packet fields and with them performs a special calculation to generate a *hardware index*. This hardware index is then used to select an egress physical port for a (v)LAG. Refer to the *About Layer 2 Hardware Hashing* section for more details on hashing field selection.

Understanding the vLAG Forwarding Rule

In order to prevent potential duplication of multicast or broadcast traffic, a forwarding rule is installed on all ports that are part of vLAGs to drop traffic whose ingress point is the cluster links. This rule is shown in the figure below:

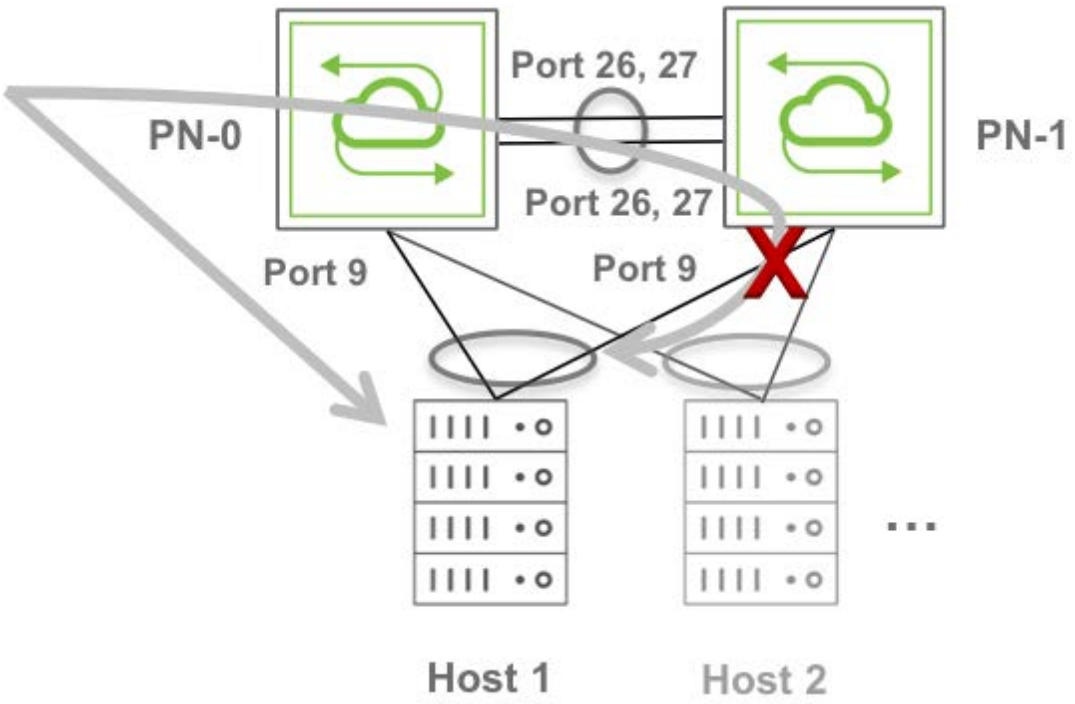


Figure 7-12 - vLAG Forwarding Rule

The figure above shows a packet entering cluster switch PN-0 that needs to be sent out to Host 1 over the vLAG. To prevent duplication of traffic on the vLAG, any traffic traversing the cluster links and egressing on vLAG ports on cluster switch PN-1 is dropped.

Configuring Trunk Hashing

By default, NetVisor OS supports the trunk hashing algorithm based on the fields listed in the *About Layer 2 Hardware Hashing* section (***Table 4-1***).

Starting from NetVisor OS release 6.1.0, this default mode is called `enhanced` hashing. Starting from the same release, NetVisor OS also supports other less granular hashing modes that can be useful in special cases. The supported alternate modes are:

```
l2-src    src mac based hashing
l2-dst    dst mac based hashing
l2         src+dst mac based hashing
l3-src    src ip based bashing
l3-dst    dst ip based hashing
l3         src+dst ip based hashing
```

They can be selected with the new `hash-mode` parameter in the `trunk-create` and `trunk-modify` commands. For example like so:

```
CLI (network-admin@switch) > trunk-modify name trunk1 hash-mode l3
```

The mode can be verified with the `trunk-show` command:

```
CLI (network-admin@switch) > trunk-show format name,trunk-id,ports,hash-
mode,status

name                 trunk-id ports hash-mode status
-------------------- -------- ----- --------- --------------------------------------
trunk1               272      65-68 l3        up,PN-switch,PN-cluster,multicast-router
vxlan-loopback-trunk 397      13-16 enhanced   up,stp-edge-port
```

Configuring Resilient Trunk Hashing

Starting from NetVisor OS release 6.1.0, the `resilient` hash mode can be configured on trunks to help prevent unnecessary traffic disruption when the number of trunk member ports changes.

Note: Resilient hashing is supported on all platforms except:

- Dell Z9100-ON, S5048F, Z9264-ON
- Freedom F9532L-C/Edgecore AS7712-32X
- Freedom F9532-C/Edgecore AS7716-32X
- Freedom F9572L-V/Edgecore AS7312-54XS
- Freedom F9572-V/Edgecore AS7316-54XS
- Freedom F9532C-XL-R/Edgecore AS7716-32X
- Freedom F9664-C/Edgecore AS7816-64X.

Due to a hardware restriction, this feature cannot be changed on the fly and needs to be configured during trunk creation like so:

```
CLI (network-admin@switch) > trunk-create name trunk2 hash-mode resilient
ports 5,6,7

CLI (network-admin@switch) > trunk-show format name,trunk-id,ports,hash-
mode,status

name                trunk-id ports hash-mode status
------------------- -------- ----- --------- --------------------------------------
trunk1              272      65-68 l3        up,PN-switch,PN-cluster,multicast-router
trunk2              273      5,6,7 resilient up,PN-switch
vxlan-loopback-trunk 397      13-16 enhanced  up,stp-edge-port
```

If you then try to dynamically modify resilient mode (for example to revert back to the default enhanced mode), you get an error message:

```
CLI (network-admin@switch) > trunk-modify trunk-id 273 hash-mode enhanced
trunk-modify: Trunk is configured with resilient hash. Delete and create
trunk to change hash mode
```

So you need to delete the trunk first and then recreate it if modification of resilient mode is required.

Configuring Symmetric Trunk Hashing

When a specific network design or monitoring device requires that the same trunk port be selected for all packets in both directions in a connection, starting from release 6.1.0 NetVisor OS supports the *Symmetric Trunk Hashing* mode.

Note that:

- Symmetric hashing is agnostic to intrinsically unidirectional parameters, such as the ingress physical interface.
- Symmetric hashing mode is an alternative configuration to the default (asymmetric) mode.
- Configuring symmetric hashing does not require a switch restart.
- You can configure symmetric hashing on a single switch or on all switches of a fabric by prepending the command with switch*.

To configure symmetric hashing, use the following per-device system command:

```
CLI (network-admin@switch) > system-settings-modify <symmetric-hash | no-symmetric-hash>
```

Where, the `symmetric-hash` parameter enables symmetric trunk hashing on the switch. By default, this feature is disabled. If enabled, you can use the `no-symmetric-hash` parameter to disable it.

To configure symmetric hashing on all switches in a fabric, use the command:

```
CLI (network-admin@switch) > switch * system-settings-modify <symmetric-hash | no-symmetric-hash>
```

In case of GRE or IPSec traffic, ignoring the unidirectional GRE key or IPSec SPI fields in the symmetric hash computation can be configured using the CLI:

```
CLI (network-admin@switch) > system-settings-modify <hash-suppress-unidir-fields | no-hash-suppress-unidir-fields>
```

This feature is disabled by default, using the `no-hash-suppress-unidir-fields` parameter.

To verify the configuration details, use the `system-settings-show` command. Below is a sample output, where both `symmetric-hash` and `hash-suppress-unidir-fields` parameters are disabled.

```
CLI (network-admin@switch) > system-settings-show layout vertical
switch:                                 switch
optimize-arps:                          off
lldp:                                   on
policy-based-routing:                   off
optimize-nd:                            off
reactivate-mac:                         on
```

```
reactivate-vxlan-tunnel-mac:            on
manage-unknown-unicast:                 off
manage-broadcast:                       off
block-loops:                            off
auto-trunk:                             on
auto-host-bundle:                       off
cluster-active-active-routing:          on
routing-over-vlags:                     off
source-mac-miss:                        copy-to-cpu
igmp-snoop:                             use-l3
vle-tracking-timeout:                   3
pfc-buffer-limit:                       40%
cosq-weight-auto:                       off
port-cos-drop-stats-interval(s):        disable
lossless-mode:                          off
snoop-query-stagger:                    no-stagger-queries
host-refresh:                           off
proxy-conn-retry:                       on
proxy-conn-max-retry:                   3
proxy-conn-retry-interval:              500
manage-l2-uuc-drop:                     on
xcvr-link-debug:                        disable
fastpath-bfd:                           off
linkscan-interval:                      150000
linkscan-mode:                          software
single-pass-l2-known-multicast:         off
single-pass-flood:                      off
batch-move-mac-hw-group-for-vlan-only:  off
memory-tracker:                         on
symmetric-hash:                         off
hash-suppress-unidir-fields:            off
```

Configuring Asymmetric Routing over vLAGs

vLAGs are symmetric Layer 2 redundant paths that work best with symmetric traffic flows. In these cases the aforementioned vLAG forwarding rule works fine, as discussed in the next section.

In case of Layer 3 forwarding over vLAGs, there can exist certain corner-case network designs in which a routing protocol (e.g., BGP or OSPF) is used to peer over a vLAG in an asymmetric fashion.

Let's consider the example in Figure: Asymmetric Routing over a vLAG in which the network admin has chosen to configure BGP peering only with cluster member PN-1. The downstream hosts are dual-homed and are properly connected to the rest of the network via a default gateway (provided by VRRP).

PN-2 is connected via a vLAG as well to the cluster, however is configured with a /30 subnet that is only present on one cluster member (PN-1 in this example, with PN-2 peering with it via BGP and using the address 10.75.1.1/30).

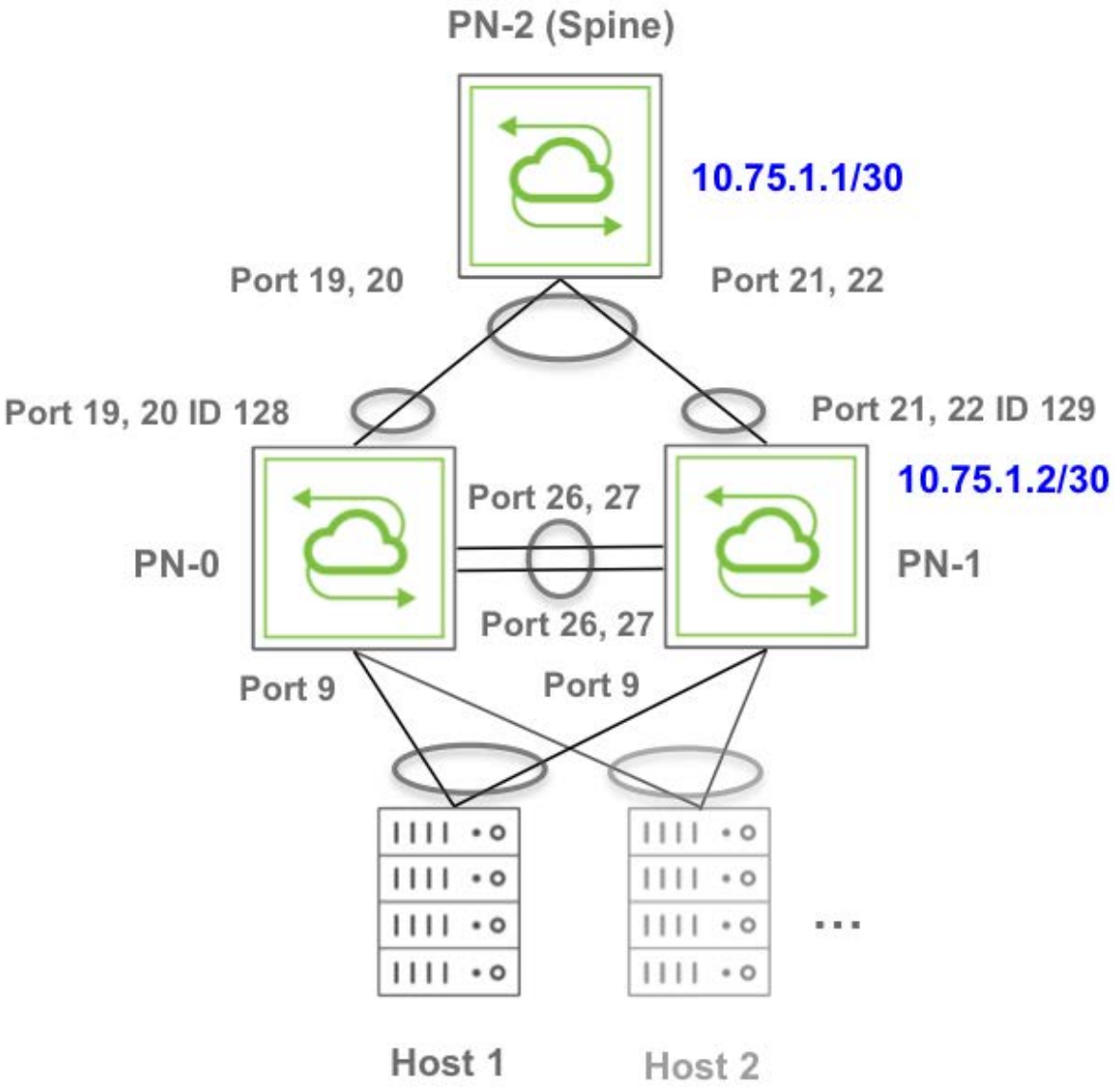


Figure 7-13 - Asymmetric Routing over a vLAG

If Host 1 wants to talk to PN-2's 10.75.1.1 (or to another asymmetric route advertised by it), its traffic can go left or right depending on the vLAG load-balancing algorithm. If it goes right, PN-1 can route it to PN-2. However if the traffic goes left, PN-0 has no route it has learned locally to reach PN-2. Therefore, in the best case it will direct traffic over the cluster link (assuming there is some IGP running over it). However, PN-1 will not be able to send the packet to H1 due to the egress filtering rules enforced on vLAGs (see previous section).

For this reason asymmetric routing configurations should be avoided.

However a knob is provided as a last resort to bypass the cluster egress filtering rules in case the customer still wants to utilize this sub-optimal design, which is not recommended. In general, it is sub-optimal for two main reasons:

- Even when there is no failure (vLAG links are up), traffic is forced to cross the cluster links, thereby potentially oversubscribing them.
- There is no redundancy at L3 level. When BGP peering to cluster member PN-1 fails, there is no backup path to direct the traffic to.

To enable support of this design a new configuration option is provided to allow packets crossing the cluster link to be routed without being dropped when egressing vLAGs.

To modify the default configuration use the `system-settings-modify routing-over-vlags|no-routing-over-vlags` command like so:

```
CLI (network-admin@switch) > system-settings-modify routing-over-vlags
```

You can use this command to verify the configuration change:

```
CLI (network-admin@switch) > system-settings-show

switch:                          PN-1
routing-over-vlags:  on
switch:                             vanquish4
routing-over-vlags:  off
switch:                             vanquish3
routing-over-vlags:  off
```

About Symmetric Routing over vLAGs

As stated above, vLAGs work best with symmetric traffic flows. In order to achieve that with routing, two main designs are recommended: VRRP + ECMP and symmetric VRRP.

The goal of either design is to distribute traffic equally across the different paths both upstream and downstream, making sure that the redundant cluster nodes can steer the traffic to its destination without having to rely upon the cluster links.

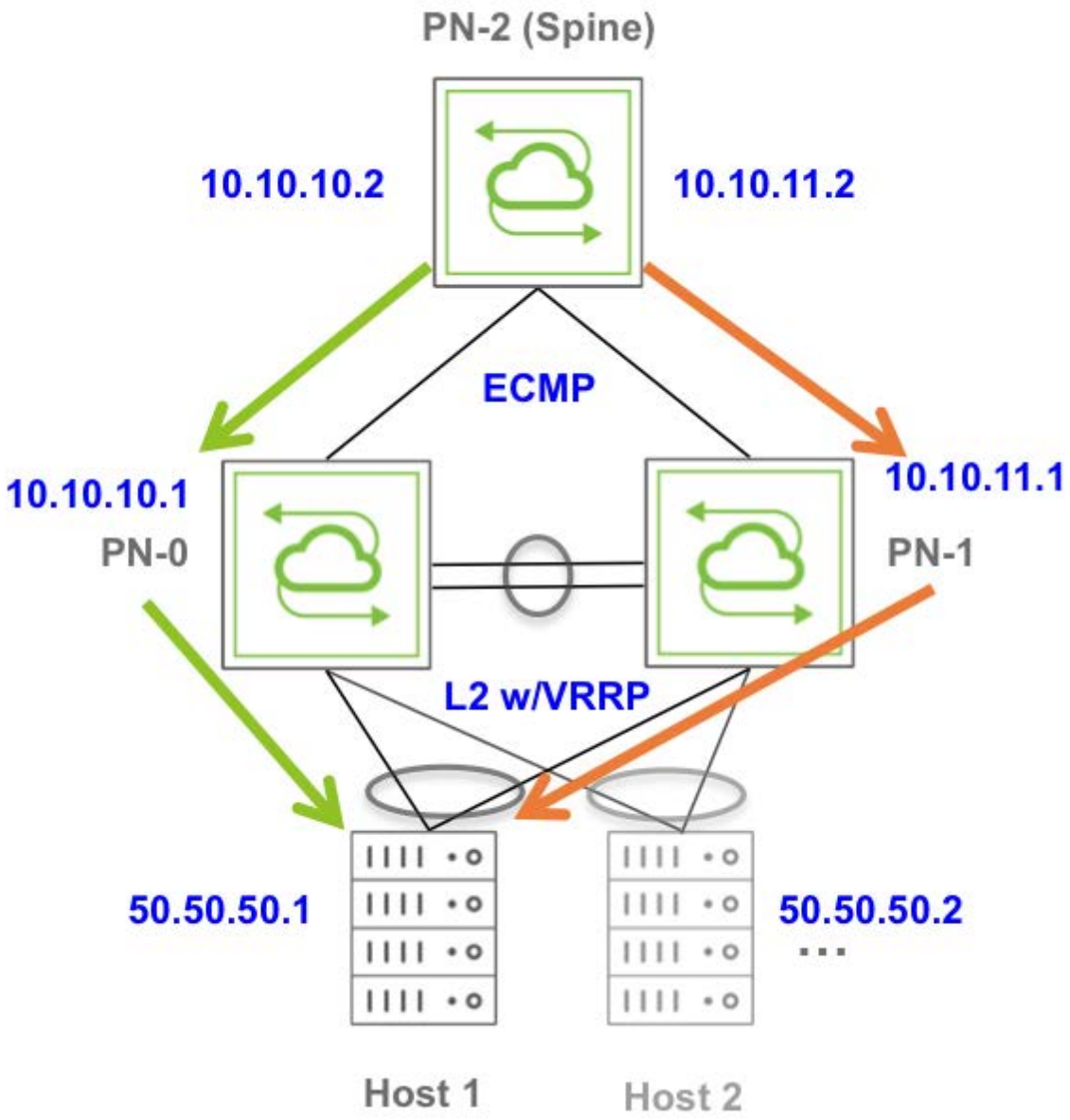


Figure 7-14 - Symmetric Routing over a vLAG with ECMP and VRRP

In this figure traffic is, for example, directed to and from host 50.50.50.1: downstream the spine router/switch has ECMP routes toward both cluster nodes, which have Layer 3 adjacencies to all the hosts, so either of them can properly steer traffic to 50.50.50.1. Upstream they implement the active-active default gateway function via VRRP and support traffic load-balancing with vLAGs from the host(s) toward the spine.

Hence this design implements optimal Layer 3 forwarding both ways (without relying on the cluster links as active paths, only as backups). It also supports running Layer 3 routing protocols on the cluster switches.

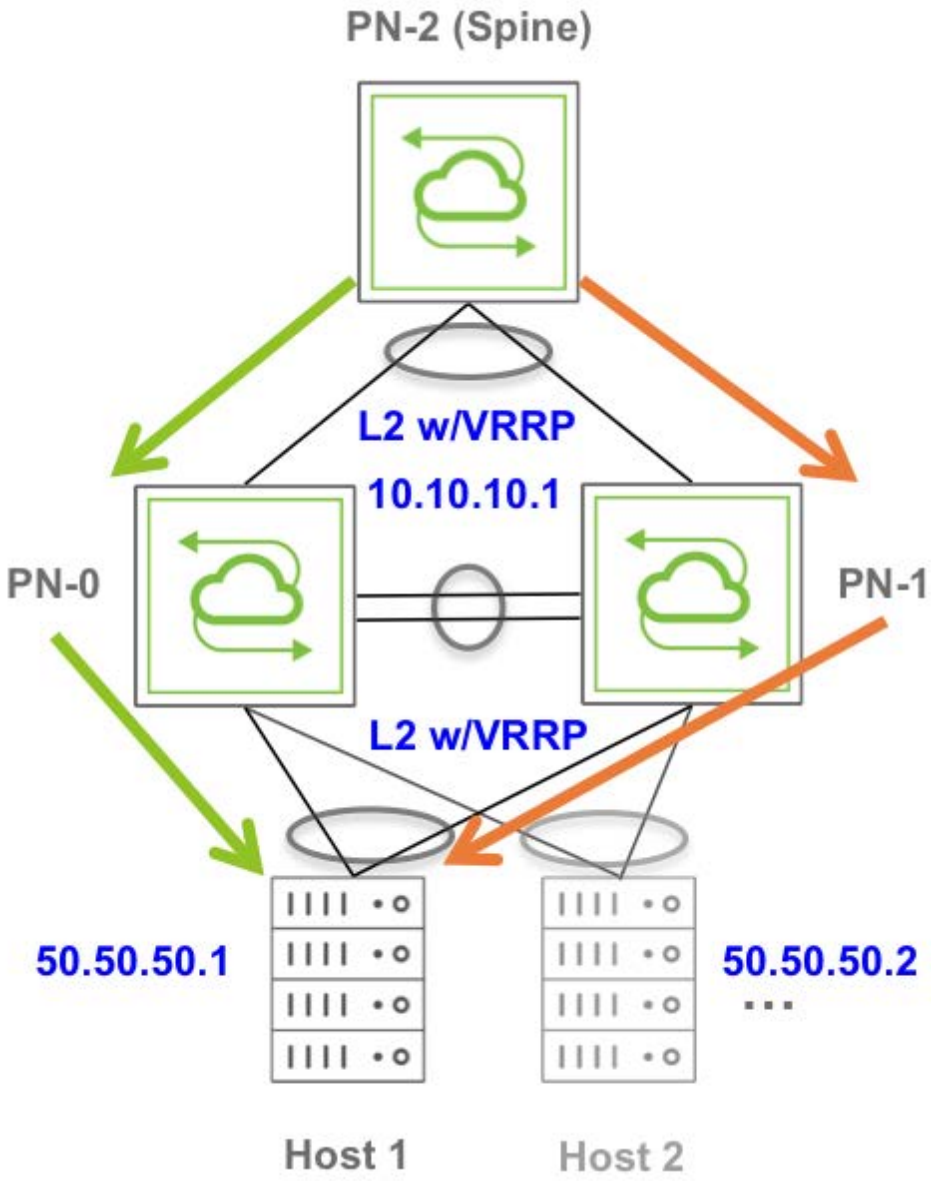


Figure 7-15 - Symmetric Routing over a vLAG with VRRP

In this other scenario two cluster switches run vRouters with active-active VRRP in order to provide redundant Layer 3 next hops (using virtual IPs) to both upstream and downstream devices.

This design achieves symmetric Layer 3 forwarding purely via vLAG load-balancing and VRRP active-active forwarding. However, note that it does not lend itself to the use of dynamic routing protocols on vRouters because with VRRP routing adjacencies would only form on the vRouter acting as VRRP master, preventing the slave vRouter to process and install routes.

NetVisor OS supports the active-active dual-forwarding logic by default with VRRP. However, if needed, you can disable it or re-enable it on a per vRouter basis with this command:

```
CLI (network-admin@switch) > vrouter-modify name vRouter-PN-0 cluster-
active-active-routing|no-cluster-active-active-routing
```

To display the configuration, use the `vrouter-show` command:

```
CLI (network-admin@switch) > vrouter-show format all layout vertical

switch: PN-0
id: b000f1e:1
name: vRouter-PN-0
type: vrouter
scope: local
vnet: test
vnet-service: dedicated
state: enabled
location: sw45
zone-id: b000f1e:2
template: no
failover-action: stop-old
router-type: hardware
fabric-comm: false
router-ipstack: frr
hw-router-mac: 66:0e:94:1e:7a:6a
cluster-active-active-routing: disable
hw-vrid: 0
hw-vrrp-id: -1
proto-multi: none
proto-routing: static,routesnoop
bgp-redist-static-metric: none
bgp-redist-connected-metric: none
bgp-redist-rip-metric:   none
bgp-redist-ospf-metric: none
bgp-dampening: false
bgp-scantime(s):  60
bgp-delayed-startup(s): 1
bgp-keepalive-interval(s): 60
bgp-holdtime(s): 180
ospf-redist-static-metric: none
ospf-redist-static-metric-type: 2
ospf-redist-connected-metric:   none
ospf-redist-connected-metric-type: 2
ospf-redist-rip-metric:   none
ospf-redist-rip-metric-type: 2
ospf-redist-bgp-metric: none
ospf-redist-bgp-metric-type: 2
ospf-stub-router-on-startup: false
ospf-bfd-all-if: no
ospf-default-information: none
ospf-default-info-originate-metric: none
ospf-default-info-originate-metric-type: 2
bgp-snmp:  false
bgp-snmp-notification: false
ospf-snmp:  false
ospf-snmp-notification: false
ospf6-snmp:  false
ospf6-snmp-notification: false
ip-snmp: false
```

Configuring Active-Active vLAG Forwarding with Loopback Recirculation

In network designs in which vLAG load-balancing with fast failover is required to work in conjunction with dynamic routing protocol peering, the aforementioned vLAG forwarding rule may cause traffic drops.

Therefore, for certain switch models with spare bandwidth an additional configuration option has been implemented.

Let's consider the example in Figure: Symmetric Routing over a vLAG with Loopback Recirculation below where a cluster pair uses standard OSPF peering over a Layer 2 domain represented by a redundant (v)LAG.

To the routing protocol running on PN-2 the (v)LAG looks like a share medium over which both cluster nodes, 10.10.10.1 and 10.10.10.2, are reachable. However, the (v)LAG performs traffic load-balancing: packets sent toward 10.10.10.1 can be steered to either PN-0 or PN-1; likewise, packets sent toward 10.10.10.2 can be steered to either PN-0 or PN-1. When packets are sent to the "wrong" next-hop, the latter will have to use the cluster links to steer the packets back to their "correct" destination.

However, packets traversing the cluster links that need to egress a vLAG (going downstream to reach one of the hosts) are dropped.

For this case, NetVisor OS supports the configuration of internal loopback recirculation for routed packets entering the switch from cluster links in order to bypass the vLAG forwarding rule.

The approach consists in provisioning a set of physical ports in loopback mode without requiring external cabling. In particular, the E28-Q model platform offers up to 12x10GE ports (that is, which are not exposed to users in the front panel) that can be configured in loopback mode for an additional forwarding capacity of 120 Gbps.

An internal forwarding rule can be manually installed on both cluster nodes to redirect the traffic to the internal loopback when it ingresses from the cluster links with a destination MAC address matching that of the next-hop. When a packet is recirculated by this rule over the loopback, it gets normally routed and hence is not subject to the vLAG forwarding rule (that would otherwise drop it).

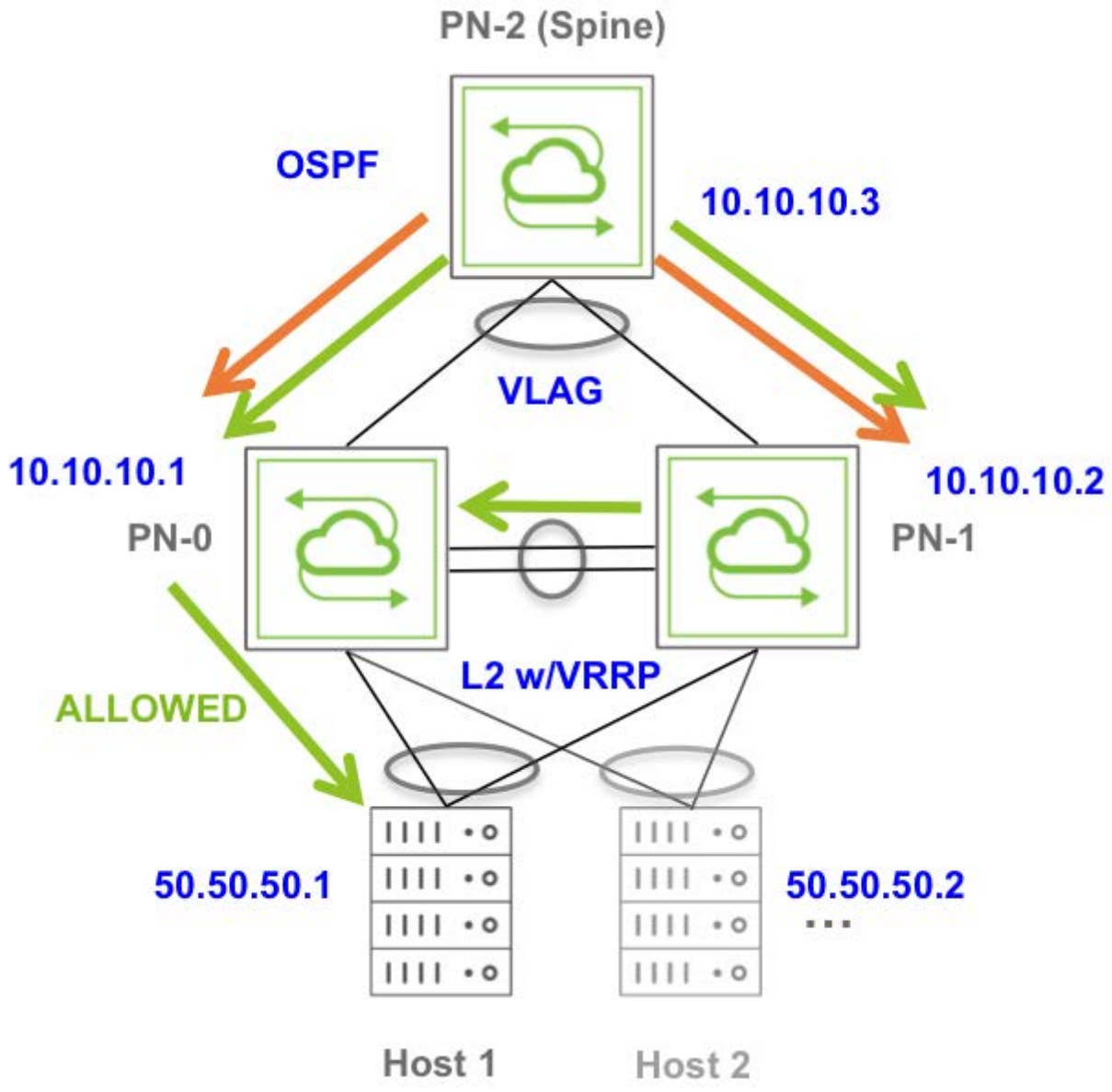


Figure 7-16 -Symmetric Routing over a vLAG with Loopback Recirculation

Note that, when this option is configured, approximately 50% of the traffic traverses the cluster links, which is not ideal, hence it is not recommended as a general solution but can be useful in certain designs.

For each vRouter that needs redirection of the traffic to the internal loopback, first find the MAC addresses of the (two, in case of a cluster) corresponding interfaces using the `vrouter-interface-show` command:

```
CLI (network-admin@switch) > vrouter-interface-show vrouter-name UP1

vrouter-name: UP1
nic: eth11.200
ip: 200.200.200.1/24
assignment: static
mac: 66:0e:94:10:29:e1
vlan: 200
vxlan: 0
```

```
if: data
vm-nic-type:  data
exclusive:  no
nic-config:  enable
nic-state:  up
```

In this case the interface MAC address is `66:0e:94:10:29:e1`.

Then configure a loopback (with a single port or a multi-port trunk):

```
CLI (network-admin@switch) > port-config-modify port 89 loopback
```

Or for the trunk case:

```
CLI (network-admin@switch) > port-config-modify port 89-100 loopback

CLI (network-admin@switch) > trunk-create name loopbackUP ports 89-100
loopback
```

Then create the special forwarding rule using the `vFlow` command:

```
CLI (network-admin@switch) > vflow-create name LB_vflow scope local in-port
129 dst-mac 66:0e:94:10:29:e1 action set-dmac-to-port action-value 89
action-set-mac-value  66:0e:94:10:29:e1
```

Or for the trunk case:

```
CLI (network-admin@switch) > vflow-create name LB_vflow scope local in-port
129 dst-mac 66:0e:94:10:29:e1 action set-dmac-to-port action-value 130
action-set-mac-value  66:0e:94:10:29:e1
```

For `action-value` use the loopback port number or the loopback trunk ID (130, in this example, assigned during creation).

For a cluster pair, you must configure each node using the above configuration steps.

Configuring Virtual Router Redundancy Protocol

The configuration of the VRRP protocol is tied to the creation and configuration of vRouters. It requires entering various parameters such as a VRRP priority, a VRRP ID to uniquely identify the virtual router, etc.

Configuring VRRP Priority

The priority is a value used by the VRRP master election process.

The valid priority range is 1-254, where 1 is the lowest priority and 254 is the highest priority.

The default value is 100. Higher values indicate higher priority for the master router election, therefore a backup router (also called a slave) can be configured for example with a priority value lower than the default.

Configuring the VRRP ID

A virtual router is identified by its virtual router identifier (VRID) and by a set of virtual IPv4 and/or IPv6 address(es).

Each virtual IPvX address is paired to a MAC address in the 00-00-5E-00-01-XX address range where the last byte of the address (XX) corresponds to the VRID.

The VRID is also used to tag and differentiate protocol messages exchanged by VRRP routers.

The virtual router identifier is a user-configurable parameter with a value between 1 and 255. There is no default value.

In the configuration this parameter has to be associated to a vRouter entity and to the VRRP interface, as shown in the example below.

Example Configuration

In this example two switches, named switch1 and switch2, are going to share a subnet and VLAN over which to set up VRRP's virtual router function (with an ID of 10):

- VLAN 100 with IP address range 192.168.11.0/24

The corresponding vRouters are going to share a common vNET:

- The vrrp-router vNET with scope fabric

To configure VRRP, start with switch1 and create a vRouter that is associated with the aforementioned vNET and a VRRP ID of 10. Before configuring the vrouter-create command, you must create the corresponding vnet:

```
CLI (network-admin@switch) > vrouter-create name vrrp-rtr1 vnet vrrp-router
router-type hardware hw-vrrp-id 10 enable
```

Add a vRouter interface that corresponds to the router's own real IP address:

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrrp-rtr1
ip 192.168.11.3 netmask 24 vlan 100 [if data]
```

The above command will output a message such as:

```
Added interface eth0.100 with ifIndex 24
```

You can also use the `vrouter-interface-show` command to check the name of the newly created interface (eth0.100):

```
CLI (network-admin@switch) > vrouter-interface-show

format all layout vertical
vrouter-name: vrrp-rtr1
nic: eth0.100
ip: 192.168.11.3/24
assignment: static
mac: 66:0e:94:dd:18:c4
vlan: 100
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: up
```

Create the VRRP interface on the master switch with virtual IP 192.168.11.2, VRRP ID 10 and default priority (100):

```
CLI (network-admin@switch1) > vrouter-interface-add vrouter-name vrrp-rtr1
ip 192.168.11.2 netmask 24 vlan 100 [if data] vrrp-id 10 vrrp-primary
eth0.100 vrrp-priority 100
```

The above command will output a message such as:

```
Added interface eth1.100 with ifIndex 25
```

Then create a vRouter and an interface (with real IP `192.168.11.4`) also on switch2:

```
CLI (network-admin@switch) > vrouter-create name vrrp-rtr2 vnet vrrp-router
router-type hardware hw-vrrp-id 10 enable

CLI (network-admin@switch) > vrouter-interface-add vrouter-name vrrp-rtr2
ip 192.168.11.4 netmask 24 vlan 100 [if data]
```

Use the `vrouter-interface-show` command to check the name of the newly created interface (`eth3.100`):

```
CLI (network-admin@switch2) > vrouter-interface-show format all layout
vertical

vrouter-name: vrrp-rtr2
nic: eth3.100
ip: 192.168.11.4/24
```

```
assignment: static
mac: 66:0e:94:21:a9:6c
vlan: 100
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: up
```

Create the VRRP interface for the backup switch with the same VIP `192.168.11.2`, same VRRP ID (10) and a lower-than-default priority (say, 50):

```
CLI (network-admin@switch2) > vrouter-interface-add vrouter-name vrrp-rtr2
ip 192.168.11.2 netmask 24 vlan 100 [if data] vrrp-id 10 vrrp-primary
eth3.100 vrrp-priority 50
```

Display the information about the VRRP setup:

```
CLI (network-admin@switch2) > vrouter-interface-show format all layout
vertical

vrouter-name: vrrp-rtr1
nic: eth0.100
ip: 192.168.11.3/24
assignment: static
mac: 66:0e:94:dd:18:c4
vlan: 100
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: up
vrouter-name: vrrp-rtr1
nic: eth1.100
ip: 192.168.11.2/24
assignment: static
mac: 00:00:5e:00:01:0a
vlan: 100
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: up
vrrp-id: 10
vrrp-primary: eth0.100
vrrp-priority: 100
vrrp-state: master
vrouter-name: vrrp-rtr2
nic: eth3.100
ip: 192.168.11.4/24
```

```
assignment:  static
mac:  66:0e:94:21:54:07
vlan:  100
vxlan:  0
if:  data
alias-on:
exclusive:  no
nic-config:  enable
nic-state:  up
vrouter-name:  vrrp-rtr2
nic:  eth4.100
ip:  192.168.11.2/24
assignment:  static
mac:  00:00:5e:00:01:0a
vlan:  100
vxlan:  0
if:  data
alias-on:
exclusive:  no
nic-config:  enable
nic-state:  down
vrrp-id:  10
vrrp-primary:  eth3.100
vrrp-priority:  50
vrrp-state:  slave
```

When you intentionally disable the master's VRRP interface, the backup interface becomes the new master:

```
vrouter-name:  vrrp-router2
nic:  eth4.100
ip:  192.168.11.2/24
assignment:  static
mac:  00:00:5e:00:01:0a
vlan:  100
vxlan:  0
if:  data
alias-on:
exclusive:  no
nic-config:  enable
nic-state:  up
vrrp-id:  10
vrrp-primary:  eth3.100
vrrp-priority:  50
vrrp-state:  master
```

When you re-enable the disabled interface on the former master, that interface becomes the master again, and the second interface returns to be a backup (slave):

```
vrouter-name:  vrrp-router2
nic:  eth4.100
ip:  192.168.11.2/24
assignment:  static
mac:  00:00:5e:00:01:0a
vlan:  100
```

```
vxlan: 0
if: data
alias-on:
exclusive: no
nic-config: enable
nic-state: down
vrrp-id: 10
vrrp-primary: eth3.100
vrrp-priority: 50
vrrp-state: slave
```

Configuring VRRP Pre-emption Mode

VRRP supports rapid transition from master to backup router in case of node failures. The master router sends VRRP advertisements based on the configured VRRP advertisement interval or a default interval of 1000 ms to the backup router(s). If the master router's advertisements are not received within the configured time window, then a backup router is elected as the master router. If the failed master router becomes active again, it can reclaim the role of master or allow the former backup to continue as the master router. The role depends on the value assigned to a parameter called VRRP priority.

To minimize network traffic, only the master router sends periodic VRRP advertisement messages. A backup router do not attempt to preempt the master unless it has higher priority, thereby eliminating service disruption.

It is also possible to administratively prohibit all preemption attempts. When the master router becomes unavailable due to a node failure or other network issues, then the highest priority backup router transitions to the master role after a short delay, providing a controlled transition of the virtual router responsibility with minimal service interruption.

With NetVisor OS version 6.0.0, you can enable or disable the *VRRP preempt mode* while configuring a VRRP interface. The preempt (or pre-empt) mode controls when a starting or restarting higher-priority backup router preempts a lower-priority master router.

By default, the preemption mode is enabled on all VRRP instances in NetVisor OS. Enabling the preemption feature is useful when a high priority router represents a much preferred path and hence takes over the master role whenever it comes back online, albeit with a potential (small) traffic disruption.

On the other hand, by disabling the preemption feature, you can prevent a high priority backup VRRP router from taking over the master router role after a restart (caused for example by a failure or other network issue). In other words, when the VRRP preemption mode is disabled and when a high priority master router becomes unavailable, the next high priority VRRP instance on the backup router assume the role of master router. When the previously unavailable VRRP device is back online and starts receiving advertisements, it does not take over the master role and remains as the backup (slave) router.

However, it is recommended not to disable the preemption mode on all the VRRP instances on a router. For example, if you have seven VRRP instances, you may disable the preemption mode on a few instances (say 3-4 VRRP instances) to maintain load balancing by avoiding a lower priority router to continue as the master router causing the control plane traffic to be handled by a single node.

Disabling the preemption mode is useful in scenarios when a higher priority backup router tries to become the master router, while a master router already exists. For example, when a higher priority master router becomes unavailable and the next high priority router assumes the master role, and later when the original master router comes back online and does not receive any advertisements for a short period (due to any

network issue or power failure), then the original master router reclaims the role of master router while the second high priority router continues in the master role. In this case, the higher priority router's claim is accepted and that becomes the master router.

Note: In NetVisor OS version 6.0.0, you must use the `vrrp-preempt-mode disable` option only when the VRRP is enabled between two switches. This functionality is not applicable for cases where VRRP is enabled on more than two switches.

You can enable or disable preemption feature on IPv4 and IPv6 addresses of the vrouters.

This feature includes the addition of a new OID to the Arista proprietary PN-VRRP-MIB, which is mapped to the standard VRRP-MIB v3.

Use either the CLI command or the REST API syntax to enable or disable the preemption feature. To enable or disable the VRRP preemption mode by using the CLI, use the `vrouter-interface-add` or `vrouter-interface-modify` commands:

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name name-string
vrrp-priority <0..254> vrrp-preempt-mode disable|enable
```

`vrouter-interface-add`	Adds an interface to a vRouter.
`vrouter-name name-string`	Enter a name for the service configuration.
`vrrp-priority 0..254`	Enter a VRRP priority for the vrouter interface. The default priority is 100.
`vrrp-preempt-mode disable|enable`	Specify one of the options to allow or prevent a high priority Backup router from becoming Master router. The default value is `enabled`.

For example, to add an interface, VR1 with priority 150, and enable the preempt mode, use the command:

```
CLI (network-admin@switch) > vrouter-interface-add vrouter-name vr1 vrrp-
priority 150 vrrp-preempt-mode enable
```

To view the details, use the `vrouter-interface-show` or `vrouter-interface-ip-show` commands. The `format all` option displays a separate column '`vrrp-preempt-mode`' with the configured value.

```
CLI (network-admin@switch) > vrouter-interface-show format all
```

To modify an existing vrouter interface, use the command:

```
CLI (network-admin@switch) > vrouter-interface-modify vrouter-name name-
string nic nic-string vrrp-preempt-mode disable|enable
```

Assuming that you had already configured other parameters such as `vlan`, `nic`, `ip-address`, and `vrrp-priority` for a vrouter interface (refer to the Configuring VRRP section for details), below is a sample configuration and show output when you enable and disable the `vrrp-preempt-mode` parameter on a vrouter interface.

View the details when the `vrrp- preempt-mode` is enabled on a VLAN:

```
CLI (network-admin@switch) > vrouter-interface-show vlan 2 format nic,nic-
state,vlan,ip,vrrp-priority,vrrp-state,vrrp-preempt-mode,

vrouter-name  nic   nic-state vlan    ip         vrrp-priority vrrp-state vrrp-preempt-mode
------------ ------ --------- ---- ------------ ------------ ---------- -----------------
ghq-spine1   eth0.2  up        2    10.10.2.2/24
ghq-spine1   eth1.2  down      2    10.10.2.1/24  100           slave      enable
ghq-spine2   eth2.2  up        2    10.10.2.3/24
ghq-spine2   eth3.2  down      2    10.10.2.1/24  110           slave      enable
uss-spine1   eth4.2  up        2    10.10.2.4/24
uss-spine1   eth5.2  down      2    10.10.2.1/24  120           slave      enable
uss-spine2   eth6.2  up        2    10.10.2.5/24
uss-spine2   eth7.2  up        2    10.10.2.1/24  130           master     enable
```

Disable the VRRP preempt by using the `vrouter-interface-modify` command on vrouter instances:

```
CLI (network-admin@ghspine01-new) > vrouter-interface-modify vrouter-name ghq-spine1 nic
eth1.2 vrrp-preempt-mode disable

CLI (network-admin@ghspine01-new) > vrouter-interface-modify vrouter-name ghq-spine2 nic
eth3.2 vrrp-preempt-mode disable

CLI (network-admin@ghspine01-new) > vrouter-interface-modify vrouter-name uss-spine1 nic
eth5.2 vrrp-preempt-mode disable

CLI (network-admin@ghspine01-new) > vrouter-interface-modify vrouter-name uss-spine2 nic
eth7.2 vrrp-preempt-mode disable
```

View the details after disabling the VRRP preemption mode:

```
CLI (network-admin@ghspine01-new) > vrouter-interface-show vlan 2 format
nic,nic-state,vlan,ip,vrrp-priority,vrrp-state,vrrp-preempt-mode,

vrouter-name  nic   nic-state vlan    ip         vrrp-priority vrrp-state vrrp-preempt-mode
------------  ------ --------- ---- ------------ ------------ ---------- -----------------
ghq-spine1   eth0.2  up        2   172.16.2.2/24
ghq-spine1   eth1.2  down      2   172.16.2.1/24  100           slave      disable
ghq-spine2   eth2.2  up        2   172.16.2.3/24
ghq-spine2   eth3.2  down      2   172.16.2.1/24  110           slave      disable
uss-spine1   eth4.2  up        2   172.16.2.4/24
uss-spine1   eth5.2  down      2   172.16.2.1/24  120           slave      disable
uss-spine2   eth6.2  up        2   172.16.2.5/24
uss-spine2   eth7.2  up        2   172.16.2.1/24  130           master     disable
```

Troubleshooting High Availability

A trunk (also called LAG or port channel) can be configured automatically or can be defined manually with or without the LACP protocol.

Due to their resiliency and traffic load balancing, trunks can be used for inter-switch communication within a cluster (auto-LAG) or for general network connectivity (user-configured LAG).

You can verify a trunk (LAG) status with the command `trunk-show`:

```
CLI (network-admin@switch) > trunk-show format
switch,name,ports,speed,lacp-mode,status
```

```
switch       name         ports      speed    lacp-mode     status
----------   -------      -------    ------   -----------   --------
pnswitch1    ports1-4     1-4        10g      off           up
pnswitch1    ports5-8     5-8        10g      off           up
pnswitch1    ports9-12    9-12       10g      off           up
pnswitch1    ports13-16   13-16      10g      off           up
```

Trunks can be configured with or without LACP. The following example shows the LACP options available when creating a trunk:

```
CLI (network-admin@switch) > trunk-create name port1-4 ports 1,4 lacp-mode
```

`Off`	LACP is off
`Passive`	LACP passive mode
`Active`	LACP active mode

When LACP is set to active or passive mode it helps detect link and configuration changes, whereas in off mode it is up to the network admin to deal with the trunk bring up process and to avoid configuration mistakes/oversights (such as asymmetries in the configuration).

Clusters and vLAGs provide the underlying redundancy structure for network communications. You can check that a cluster and a vLAG are functioning properly with the following commands. First verify the cluster status (online or offline) with the command `cluster-show`:

```
CLI (network-admin@switch) > cluster-show
```

```
name          state  cluster-node-1   cluster-node-2   tid    ports      remote-ports
-----------   ----   -------------    -------------    ---    -----      ------------
pnclusterodd  online pnswitch1        pnswitch3         15    4,36,128   4,36,129
pnclustereven online pnswitch2        pnswitch4          0   4,8,128    4,8,129
```

Then verify the vLAG status(es) with the command `vlag-show`:

```
CLI (network-admin@pnswitch1) > vlag-show layout vertical
```

```
name:         pnvlag1
cluster:      pnclusterodd
```

```
mode:         active-active
switch:       pnswitch1
port:         trunk-to-plus
peer-switch:  pnswitch3
peer-port:    trunk-to-plus
status:       normal
local-state:  enabled
lacp-mode:    up  off

name:         pnvlag2
cluster:      pnclustereven
mode:         active-active
switch:       pnswitch2
port:         49
peer-switch:  pnswitch4
peer-port:    18
status:       normal
local-state:  enabled
lacp-mode:    up  active
```

A vLAG is a logical entity that relies upon its port members (physical ports with an operational Layer 1 status) and upon the underlying cluster.

Therefore, first check that the vLAG status is normal and the state is “enabled,up”.

If there are problems with the vLAG, work back through the objects it depends on the cluster, and ultimately the physical ports and the cables.

Supported Releases

Command/Parameter	NetVisor OS Version
trunk-create, *trunk-modify* (with LACP parameters and loopback)	Commands with parameters first supported in version 2.1
cluster-create	Command added in version 1.2
cluster-bringup-modify	Command added in version 2.5.4
l3-to-vlan-interface-delay	Parameter introduced in version 2.6.2
vlag-create, vlag-modify (with parameters for mode lacp-mode, active-standby and active-active)	Commands with parameters first supported in version 2.2
lacp-mode, lacp-fallback, and lacp-fallback-timeout	Parameters introduced in version 2.4
lacp-port-priority	Parameter introduced in version 2.5
hw-vrrp-id	Parameter introduced in version 2.2 for vrouter-create
vrrp-id, vrrp-priority, and vrrp-primary	Parameters introduced in version 2.2 for vrouter-interface-modify
vxlan, node1-ip, node2-ip	Parameters introduced in version 7.0.0 for cluster-create

Refer to the Arista Networks NetVisor OS Command Reference documents for more details on the commands.

Related Documentation

For more information on concepts mentioned in this section (such as Layer 2, Layer 3, and the Fabric), refer to below topics from the Configuration Guide:

- Configuring Layer 2 Features
- Configuring Layer 3 Features
- Configuring and Administering the Unified Cloud Fabric
- Configuring VXLAN

Configuring VXLAN

This chapter provides information about understanding and configuring the VXLAN features on a NetVisor OS switch using the NetVisor OS command line interface (CLI).

Understanding VXLAN

In modern networks, Virtual eXtensible Local Area Network (VXLAN) has become a key technology enabler for advanced data center network designs such as those employed by cloud service providers and large enterprises. RFC 7348 is the IETF *informational* document that describes in detail the VXLAN encapsulation scheme.

Historically, the primary need for the introduction of a new encapsulation scheme originated in the data center where high server density on top of server virtualization placed increased demands on the available physical and logical resources of the network.

In particular, each virtual machine (VM) requires its own Media Access Control (MAC) address. Therefore, highly scalable data centers with N VMs running on M servers would require MAC address tables potentially larger (N x M) than what is available in a switched Ethernet network.

In addition, bare metal servers as well as VMs in a data center need to be grouped and isolated according to various management and security policies: this is usually achieved by assigning them to Virtual LANs (VLANs). However, the maximum number of VLANs is limited to 4094 and hence it's inadequate for the largest network designs, especially when *multi-tenancy* and *VLAN reuse* are required.

In fact, data centers are often required to host multiple tenants, which must have their own isolated management and data forwarding domains. This means that each tenant should be able to independently assign MAC addresses and VLAN IDs without causing resource conflicts in the physical network. In order to achieve this, an extra *layer of network virtualization* is required.

While various solutions for each aforementioned resource limitation/challenge have been proposed in the past, thanks to its flexibility and ample software and hardware support VXLAN has risen above all alternatives to become the *solution of choice* to address all of the above limitations. Moreover, its powerful UDP-based encapsulation has been leveraged to enable novel capabilities as well as to replace legacy solutions for highly scalable network designs (as further discussed in the paragraphs below).

One important use case is when virtualized environments require having Layer 2 forwarding capabilities scale across the entire data center or even between different data centers for efficient allocation of compute, network and storage resources. A solution that enables a network to scale across data centers is oftentimes referred to as *Data Center Interconnect* (*DCI*) and is an important high availability design.

In the DCI scenario traditional approaches that use the Spanning Tree Protocol (STP) or an MPLS-based technology (such as VPLS) for a loop-free topology are not always optimal or flexible enough. In particular, network designers typically prefer to use an *IP transport for the interconnection, redundancy and load balancing of resources and traffic* .

Nonetheless, despite the preference for a Layer 3-based interconnect, in VM-based environments there is often a need to preserve Layer 2-based connectivity for inter-VM communication. How can both models coexist then?

VXLAN is the answer to this question: it employs a UDP-based header format which can be used to route traffic within a physical Layer 3 network (also called an *underlay*) while its encapsulation capabilities can seamlessly *preserve and augment* Layer 2's inherent characteristics (such as MAC addresses and VLANs) and communication rules.

From this point of view VXLAN marries the best of both worlds. Hence it can be used to implement a virtual

Layer 2 network (a so-called *Layer 2 overlay*) on top of a traditional Layer 3 network design.

This capability is also an ideal fit for example for DCI deployments where a Layer 2 overlay is required to carry the Ethernet traffic to and from geographically dispersed VMs.

The approach of transporting Layer 2 traffic in a UDP-encapsulated format is akin to a logical 'tunneling function'.

However, VXLAN's informational RFC does not enforce any particular control-plane and data-plane scheme (like a tunnel-like model), nor does it limit the number of possible use cases. (It simply illustrates how to overcome certain limitations while suggesting one communication scheme for unicast traffic and one for multicast/broadcast/unknown unicast traffic).

In practical terms, then, VXLAN is primarily an encapsulation format. The way the control plane uses it varies from vendor to vendor, sometimes with fully proprietary implementations and sometimes with open software components based on common interoperability standards.

Despite this heterogeneity and flexibility of implementation, the VXLAN format has become widely popular: it is supported in hardware by most data center switches and it is also supported in software on virtualized servers running for example an open virtual switch.

Most hardware vendors however don't implement verbatim the entire RFC in particular with regard to the 'flood and learn' data plane mechanism described in it, because deemed not scalable enough.

The following sections will describe Arista Networks' innovative VXLAN integration with the *Unified Cloud Fabric* (UCF, or simply the *fabric*), the related optimizations and use cases, and their configuration.

About VXLAN's Packet Format and Its Implications

From a pure encapsulation perspective, the VXLAN format has a few important general implications related to the specific fields that it adds as well as to the UDP encapsulation.

The VXLAN header is shown in ***Figure 8-1*** below.

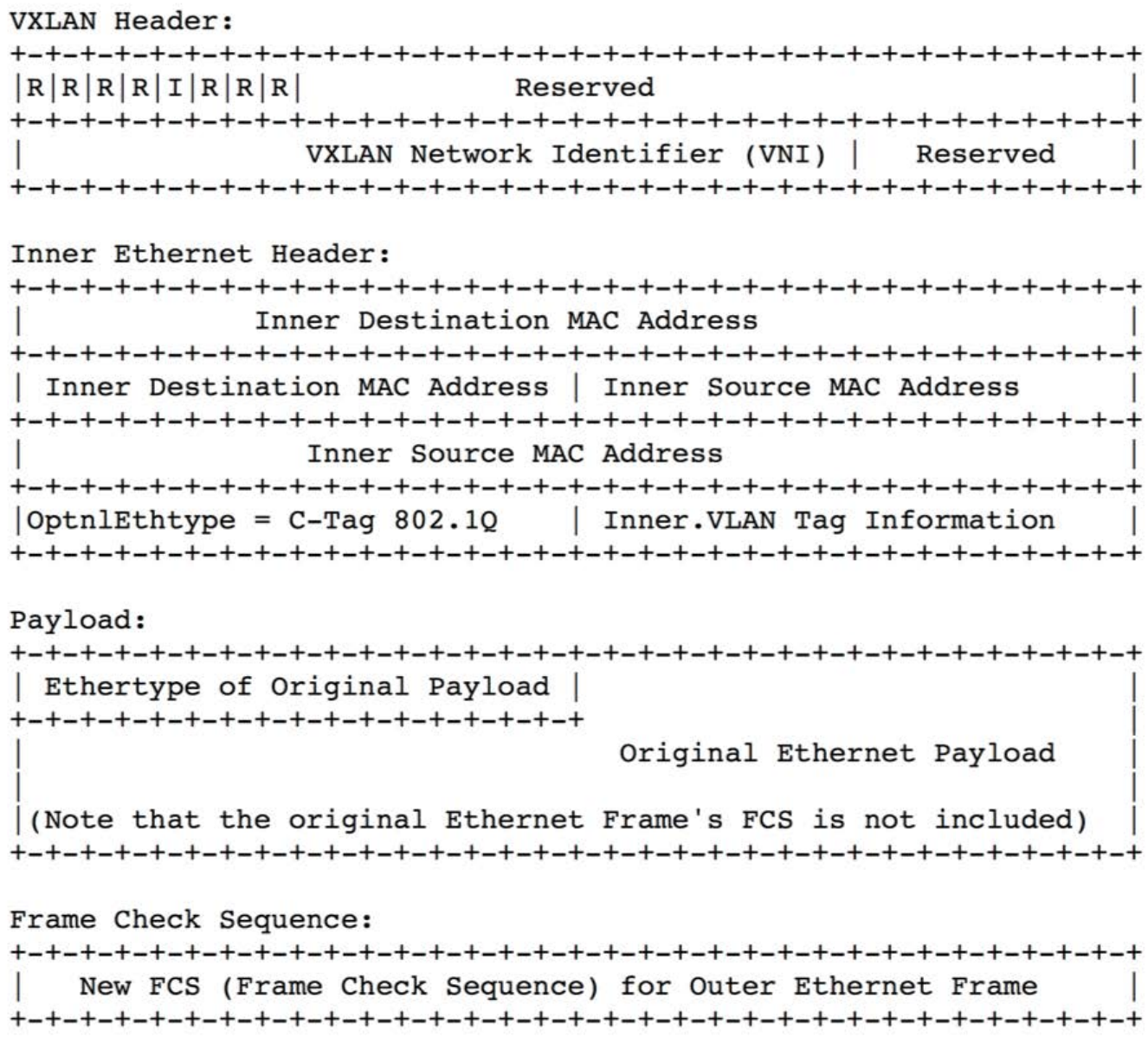


Figure 8-1: VXLAN Header Added on Top of a Plain UDP Packet (Ethernet + IP + UDP)

VXLAN requires the addition of a variable number of extra octets (i.e., overhead) to all the frames it encapsulates: as shown above, Inner Ethernet Header + Payload + FCS represent the Ethernet frame that originates in the underlay network from a host (virtualized or not). Such original frame is then encapsulated into an 8-octet VXLAN header, which is then encapsulated into a regular UDP (+ IP + 802.3) packet. An outer IEEE 802.1Q VLAN tag is optional and, when present, adds *four additional octets* to the total.

For example, in case of UDP over IPv4, this equates to a total of 50/54 additional octets of VXLAN encapsulation overhead. In case of UDP over IPv6 an additional 20 octets (compared to IPv4's case) have to be added due to the longer IPv6 header (for 70/74 bytes of total extra overhead).

About the MTU Configuration

The first obvious implication is that, in order to make sure that variable-length VXLAN encapsulated packets can be successfully carried across a generic Ethernet underlay network, the *Maximum Transmission Unit* (*MTU*) size must be properly increased end-to-end on all the physical and logical interfaces connecting the network nodes.

In the IPv4 case the MTU needs to be raised to at least *1554 octets* (from the standard 1500 value). In the IPv6 case it needs to be set to at least *1574 octets*.

In an Ethernet network, usually this requirement is supported for both cases by enabling *jumbo frame forwarding* on the device interfaces (at Layer 2 and 3) that have to carry any VXLAN traffic.

About the VXLAN IDs

The second important implication of the VXLAN packet format is that it includes a very useful 24-bit field called *VXLAN Network Identifier* (*VNI*). Per the definition in RFC 7348:

> *VXLAN Segment ID/VXLAN Network Identifier (VNI): this is a 24-bit value used to designate the individual VXLAN overlay network on which the communicating VMs are situated. VMs in different VXLAN overlay networks cannot communicate with each other.*

Traditional network segmentation at Layer 2 is implemented with IEEE 802.1Q VLANs to provide logical segmentation of the traffic in different broadcast domains. The use of VLANs, however, imposes a number of limiting factors, especially in large network designs, for example in large multi-tenant data centers.

By leveraging the VNI field as well as a number of VXLAN-based forwarding optimizations it is possible to provide the same Ethernet Layer 2 network services as VLANs but with greater extensibility and flexibility.

In particular, the 24-bit VNI field supports 16 million possible IDs, which is usually considered more than sufficient for the largest network designs.

When compared to VLANs, VNIs offer the following benefits:

- *Increased scalability* to address more Layer 2 segments as it is possible to map such segments to up to 16 million IDs that can coexist in the same administrative domain.

- *Flexible allocation* of segment IDs in multi-tenant environments, as it is possible to map each VLAN ID to different unique VNIs (that is, VLAN ID reuse is easily supported). Furthermore, VXLAN's UDP transport enables the *flexible interconnect* of data center pods across a generic IP underlay, even when the pods are geographically distributed for redundancy and load sharing purposes.

- *Standard routing technologies* (Layer 3 forwarding and equal-cost multipath (ECMP) load-balancing) are applied to VNI-based segments in the underlying IP network thanks to VXLAN's UDP-based encapsulation. This can translate into better network utilization, simpler interoperability in

heterogeneous networks and greater scalability enabled by *hardware-based forwarding and replication*.

About VXLAN's Layer 2 Extension

The third important implication of the VXLAN packet format is that it can be used to transport Layer 2 traffic from a source to a destination where both source and destination use a single MAC address each (or a limited number of addresses). That is why VXLAN is often likened to a 'tunnel' from a source to a destination that carries traffic from any number of end devices.

In more general terms, as we will see in the following, it's a *versatile Layer 3 forwarding path* from a source to a destination.

Both source and destination are called *VXLAN Tunnel End Points* (VTEPs) per the RFC.

One important challenge in today's virtualized environments is that there is increased demand on MAC address tables of switches that connect to servers. Instead of learning one MAC address per physical server, the switch now has to learn the MAC addresses of the individual VMs, and if the MAC address table overflows, the switch will stop learning new MAC addresses until idle entries age out.

With VXLAN's *end points using a single MAC address*, only that address gets exposed to and learned by adjacent transit switches. Therefore, there is no requirement for global learning of all MAC addresses anymore (like in a flat Layer 2 network), and *MAC address scalability* is only limited by the table size of the switch where the VTEP is configured.

Furthermore, the VNI identifies the scope of the inner MAC frame originated from each host. Therefore, it is possible to utilize overlapping MAC addresses across segments without conflict since, just like with VLANs, the traffic is logically isolated by using different identifiers (the VNIs in case of VXLAN segments).

The aforementioned advantageous implications bring even more benefits to network designers when combined with Arista Network's innovative *Unified Cloud Fabric* and its advanced feature set.

In the following sections we will describe Arista's distinctive implementation of the VXLAN technology, its forwarding optimizations and benefits, as well as its specific configuration steps.

About NetVisor VXLAN Implementation and Benefits

If a network designer were tasked with selecting the best and most desirable technology among a number of implementation options, he or she would be hard pressed not to go with an open and standard approach. That is simply because a standard technology can maximize interoperability and hence offer network designers a wealth of software and hardware choices.

In contrast to other closed proprietary approaches, which require significant investments in complex and rigid hardware, Arista Networks has chosen the user-friendly path of a *software-defined networking technology* based on the VXLAN industry standard, where the fabric and its features seamlessly integrate with the advanced capabilities enabled by the VXLAN transport. (For more details on Arista's fabric configuration, refer to the *Setting up and Administering the Unified Cloud Fabric* chapter.)

In particular, Arista has integrated the VXLAN implementation with the fabric's capability of distributed MAC learning and VLAN transport, its redundant path support through vLAGs, virtual addresses and vRouters, as well as with the fabric VRFs' distributed forwarding capability for maximum redundancy and scalability.

The power of the *Unified Cloud Fabric* distributed control plane has been leveraged to greatly augment and simplify VXLAN's basic capabilities without requiring complex proprietary (and potentially not inter-operable) enhancements. *Fabric VRFs and distributed multicast forwarding* (described later in this section) are two examples of the powerful synergies that have been achieved.

As part of the Unified Cloud Fabric, two inter-related logical layers have to be designed and configured: the underlay and the overlay network.

Let's look at each one individually.

About the Underlay Network's Architecture

For flexibility and openness reasons, NetVisor OS supports a spectrum of different options based on *standard protocols* for fabric underlay architectures: which can be built using Layer 2 only, or with a combination of Layer 2 and Layer 3, or only with Layer 3, as shown in ***Figure 8-2*** below.

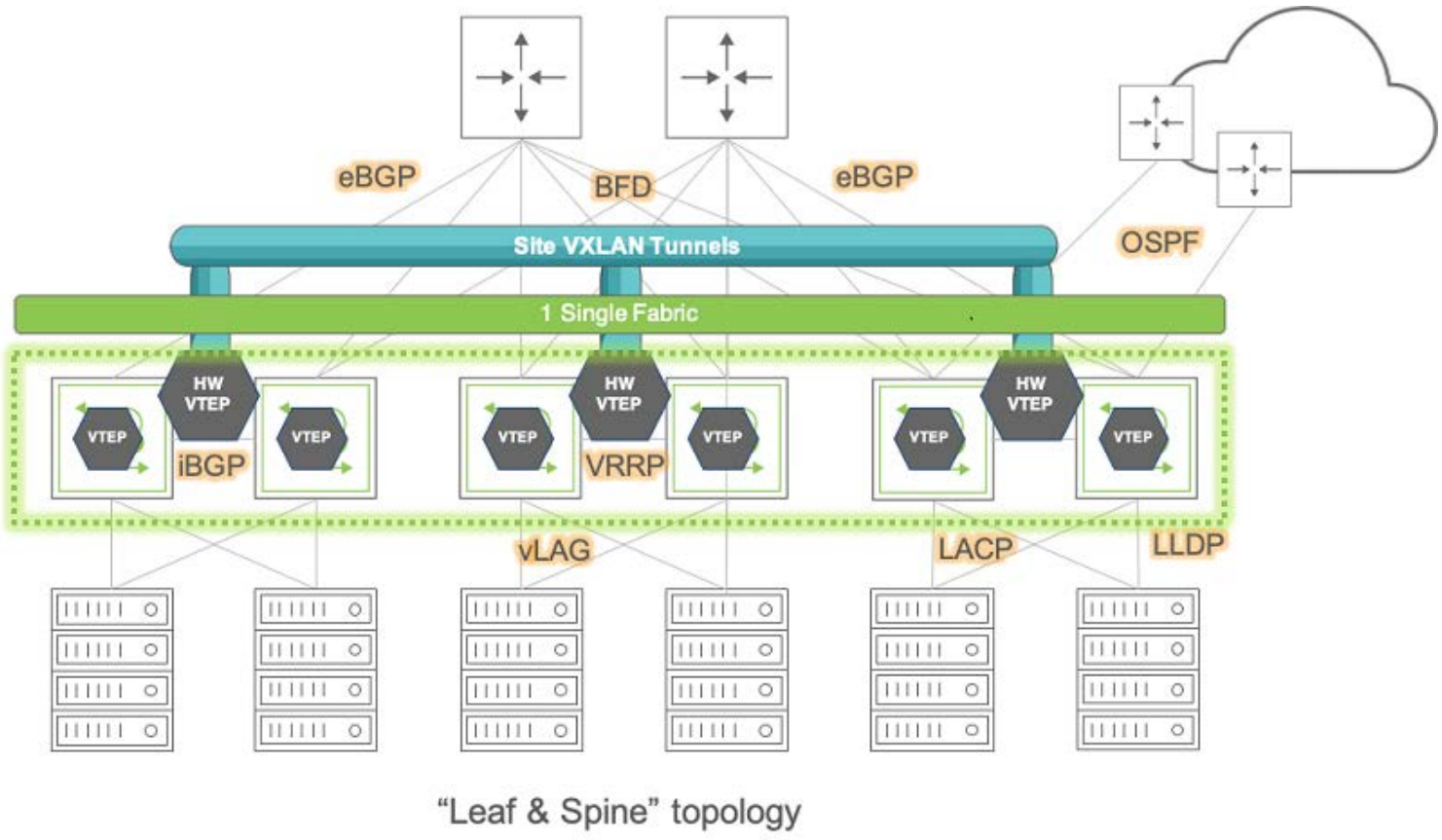


Figure 8-2: Arista Fabric Over Any Underlay Network

As depicted above, network underlays typically include both path and device redundancy.

As explained in the *Configuring High Availability* chapter, either with Layer 2 or with Layer 3 designs Arista supports switch clustering and vLAGs in order to minimize reliance on the Spanning Tree Protocol (in case of Layer 2 networks) as well as to optimize redundancy and failover speed.

For Layer 3 underlay networks Arista also supports standard routing protocols for device interconnection (such as OSPF and BGP) as well as the VRRP protocol when default gateway redundancy is required.

In particular, having the possibility of leveraging an IP-based routed transport for intra-DC and inter-DC interconnection makes the entire architecture more open and leaner, with the option to use any switch vendor in the core (since any vendor nowadays can provide a broad, inter-operable and robust support of Layer 3 technologies).

In summary, for the network underlay the designer will have to decide which topology and forwarding scheme to use, which devices to deploy in each network location, and, as explained earlier, will have to enable jumbo frames wherever the VXLAN transport is required (typically it is required end-to-end for pervasive deployment models).

Arista Networks' open networking switches can be used as both leaf and spine switches, and can be connected to 3rd party switches, routers and appliances using just the aforementioned standard underlay technologies (whose configuration is described in detail in other chapters of this guide).

About the VXLAN Fabric Overlay and End Points

Regarding the VXLAN transport's architecture and configuration, network designers have to decide where to place the VXLAN end points, i.e., the sources and the destinations of the VXLAN forwarding paths (referred to as *tunnels* in the RFC).

This decision depends on many different factors (such as level of software and hardware support, scalability, virtualization, traffic patterns, etc.) but the choice usually falls on the edge nodes of the fabric (whether hardware or software-based).

Redundancy can also be a key decision factor, as discussed in the next section.

A common scalable choice for end point (VTEP) placement is represented by the leaf switch(es), that is, the first-hop switches physically connected to the hosts (bare metal servers or VMs).

Furthermore, when configured in redundant pairs, leaf switches are the ideal locations for VTEP placement for improved service continuity, as shown in the figure below. (On the other hand, it is also possible that in certain other designs the VTEPs can be placed on spine switches, if so required by design prerequisites and/or network constraints.)

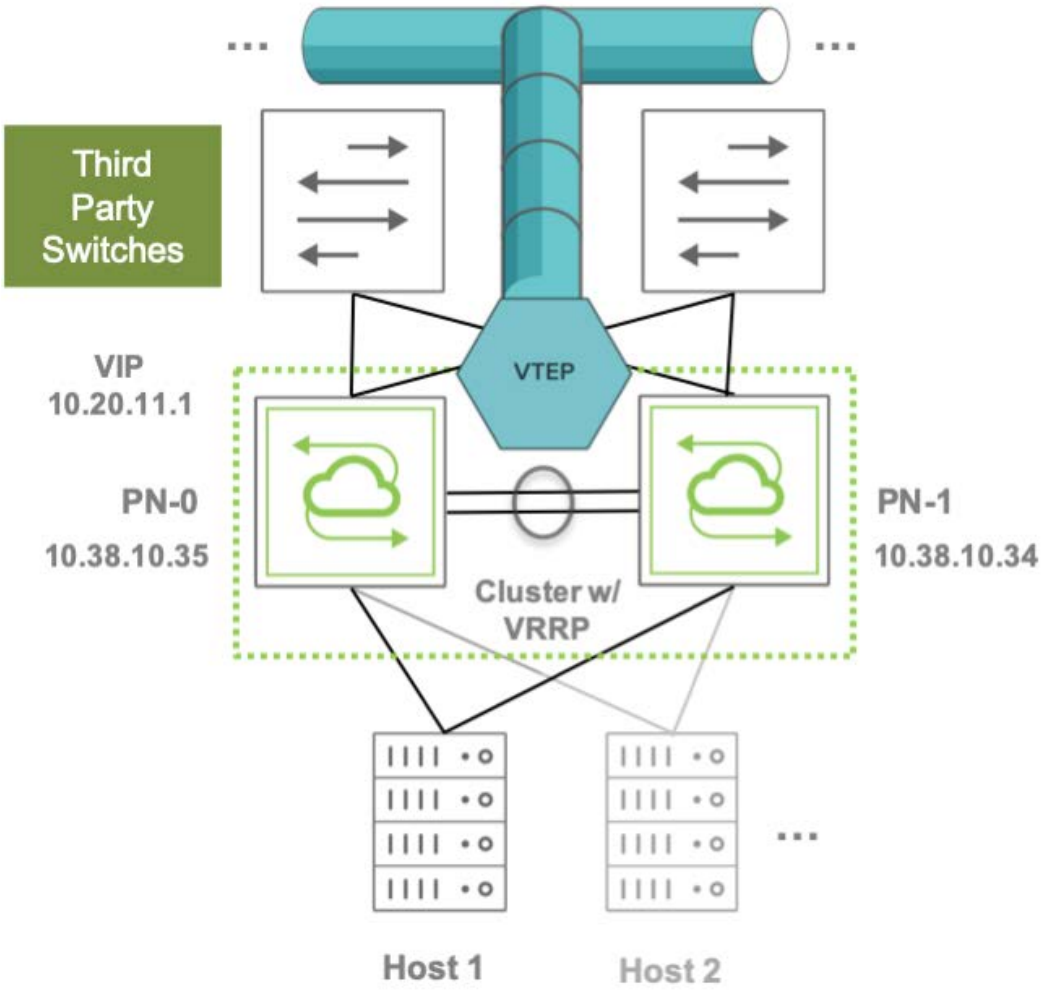


Figure 8-3: VTEP Placement on Leaf Switch Clusters

Arista switches support flexible VTEP placement. For example, VTEPs can be placed on pairs of Arista leaf

switches that are connected to a third-party Layer 3 core network (as shown in the figure above), or they can be placed on pairs of Layer 3 Arista spine switches that interconnect two DC sites (or pods) over a generic IP-based network, or the design choice can even be a combination of the two above.

In any of these cases, the VXLAN traffic flows are terminated on the switches, which are responsible for encapsulating packets that arrive from servers or hosts on a Layer 2 network and transmitting them over to their destination. Similarly, the VXLAN packets arriving from the opposite direction are decapsulated and the inner packets are forwarded over to their destination end devices. The switches also collect traffic statistics, optimize ARP requests and MAC learning.

In addition to the above, Arista also supports scenarios in which VXLAN is not terminated on the switch itself, that is, in which the switch does not participate in the encapsulation and/or decapsulation of the VXLAN traffic.

In such case a host would typically implement a VTEP in software and therefore the switch would act mainly as an underlay node for the pass-through VXLAN traffic directed to that host.

On that VXLAN traffic, though, NetVisor OS can still perform an important function:

Analytics collection: All TCP control packets as well as ARP packets traversing a tunnel are captured. By default, the ARP and TCP packets are captured only at the ingress and egress tunnel switches. With inflight analytics enabled, the analytics are captured on the traversing packets as well. These packets are used to build connection statistics and provide visibility as to which VXLAN nodes are on specific ports. With inflight analytics enabled, analytics packets are captured on the traversing packets.

About VTEP High Availability

A critical design requirement for various DC networks is to deploy the VXLAN transport in conjunction with a *high availability (HA) configuration* at the edge of the fabric. This guarantees path redundancy and service continuity in case of link or device failure at the edge.

In VXLAN parlance edge redundancy support is known as *VXLAN Tunnel Endpoint High Availability* (*VTEP HA*).

Arista switches support VTEP HA through the combination of multiple features: in particular, Arista implements a VTEP HA switch pair by harnessing the power of switch clustering in conjunction with the vLAG technology and with the standard VRRP protocol (these features are described in detail in the *Configuring High Availability* chapter).

This feature combination enables the configuration of redundant active-active Layer 3 switches that function as VRRP peers (with mutual liveliness checks) and that can provide a common *virtual IP* (*VIP*) address as active-active default gateway.

Thanks to the cluster synchronization function, a VTEP pair can act as a single logical VXLAN end point using a single shared VIP as source address. Similarly, a destination VXLAN end point can be reached by using its HA pair's common VIP as destination address.

This interconnection of virtual IPs is exemplified in the figure below.

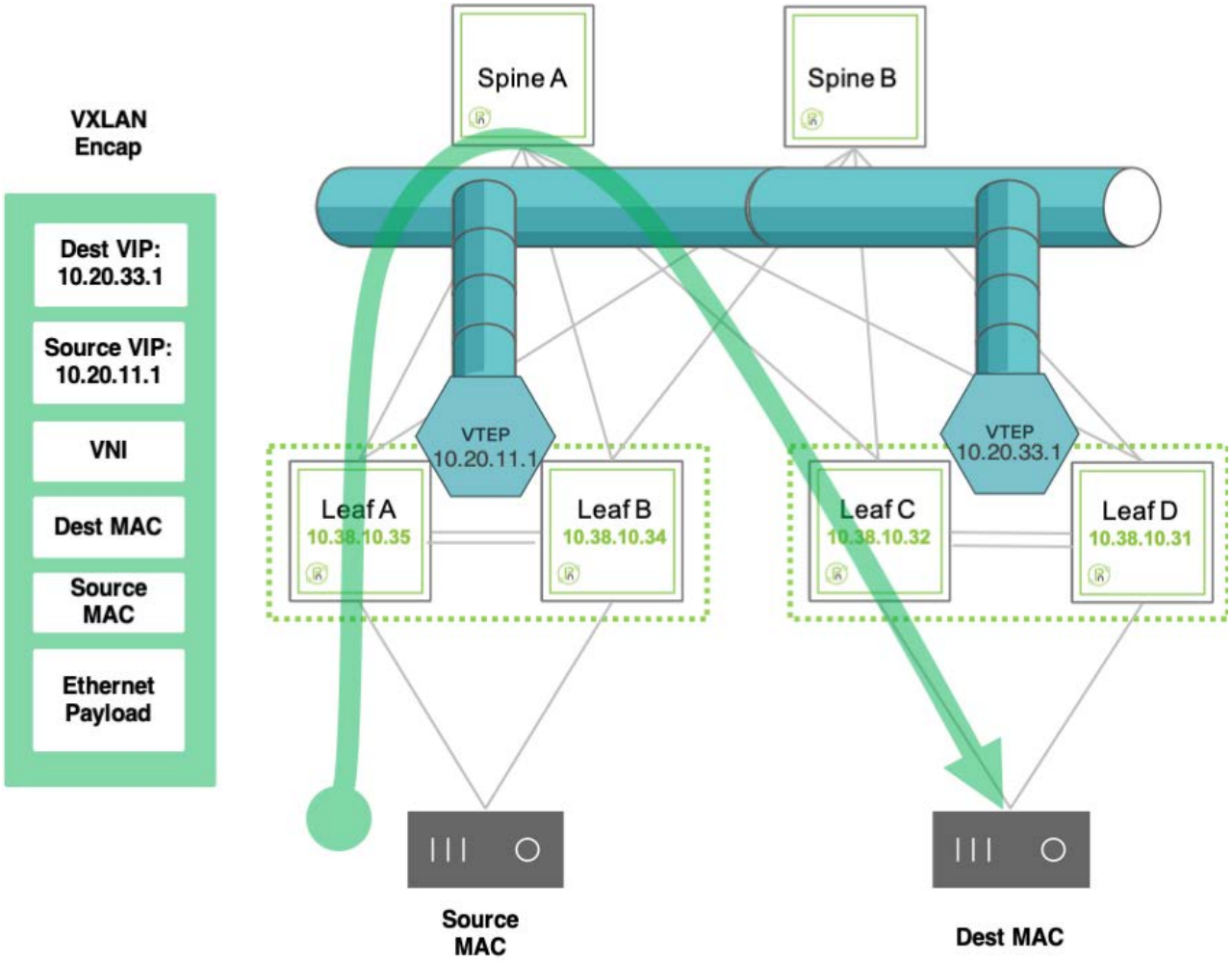


Figure 8-4: VIP Use for Inter-VTEP Communication

Setting up multiple of the above logical pairs enables the creation of an overlay network of *interconnected VXLAN end points* based on virtual (not physical) addresses which offers *embedded physical as well as logical redundancy*.

Configuration of these VXLAN pairs can be performed manually for each individual source and destination VTEP. However, Arista can further leverage the power of the Unified Cloud Fabric's distributed control plane to automate this tedious and potentially error prone process (for more details refer to the VTEP configuration section below).

Similarly, configuration of VLAN extensions over a VXLAN fabric requires a standard VLAN ID to VXLAN Network ID (VNI) mapping scheme. This ID mapping process too can be repetitive and error-prone, but with Arista's distributed control plane such global configuration can be propagated to all the nodes belonging to the same fabric instance, thus saving significant time and effort.

About the VXLAN Fabric Overlay and its Forwarding Optimizations

A fabric overlay comprises a number of VXLAN end point pairs, as mentioned above, whose interconnection is used to transport end stations' traffic end-to-end based on their VLAN-to-VNI mappings.

A typical example of the logical overlay interconnect (of leaf switches in this instance) that VXLAN can be

used to create over a physical underlay network is depicted in the figure below:

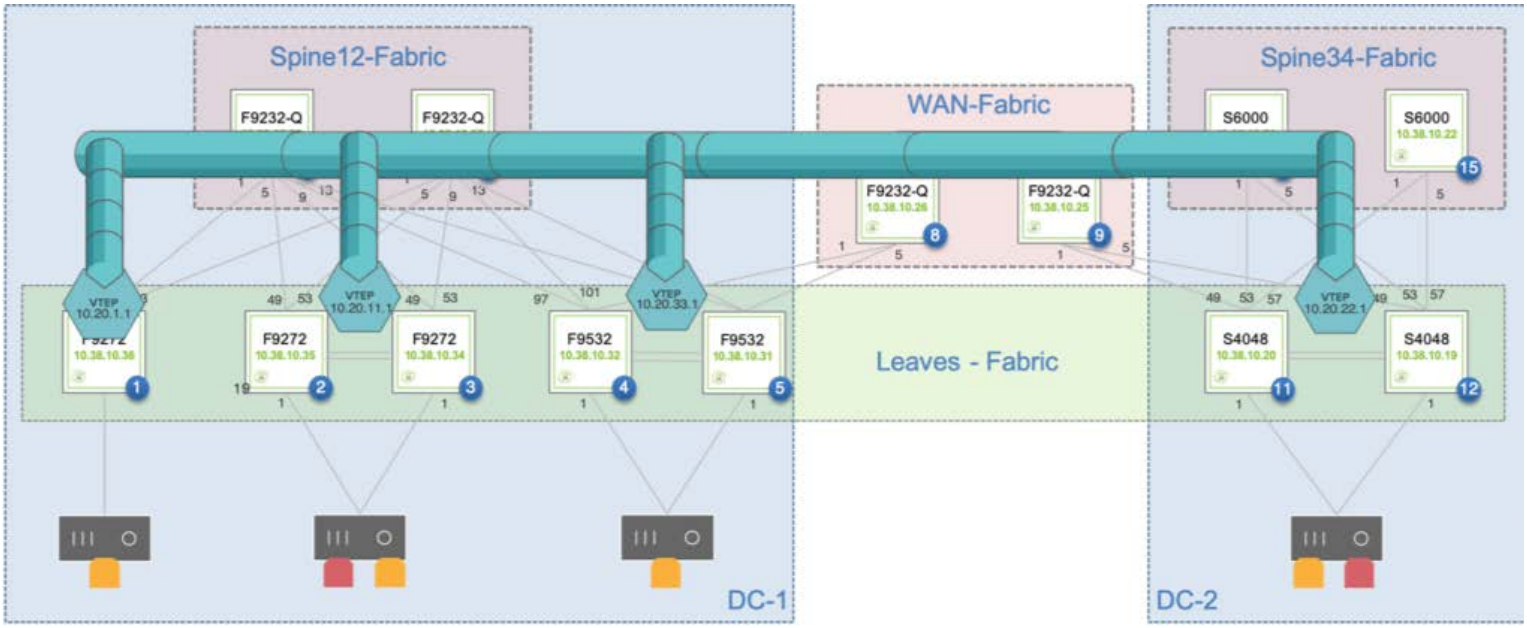


Figure 8-5: Overlay Network Interconnecting Single and Redundant VTEPs

NetVisor's forwarding logic and distributed control plane employ the standard *split horizon algorithm* to create a loop-free logical forwarding topology in the fabric overlay network so as to steer any VXLAN traffic over it without creating potential switching loops.

Furthermore, the implementation of the forwarding logic does not require complex address tracking and propagation mechanisms (such as IETF's MP-BGP-based EVPN solution). Arista instead uses its own *native address* registration and tracking technology, called *vPort database*, which is particularly useful to track host moves.

While a plain-vanilla hardware Layer 2 table is limited in its capacity by a switch's dedicated ASIC memory size, the NetVisor OS software runs in the control plane processor's much larger DRAM memory space and hence is capable of tracking a large list of Layer 2 entries, much larger than what can fit into the limited space of a hardware table.

This logical software extension of the Layer 2 table is the vPort database, which can be considered as the *authoritative distributed endpoint directory and switching activity history book* of the entire Unfied Cloud Fabric.

In addition, as an optimization option (disabled by default), the fabric can use vPort information in conjunction with the *ARP optimization technology* to help manage the dynamic process of binding MAC/IP address pairs.

Arista's ARP optimization technology harnesses the versatility of the vPort database to perform the dynamic endpoint discovery and ARP proxy functions that are required to support the mobility of end-points across the fabric, as described in the figure below.

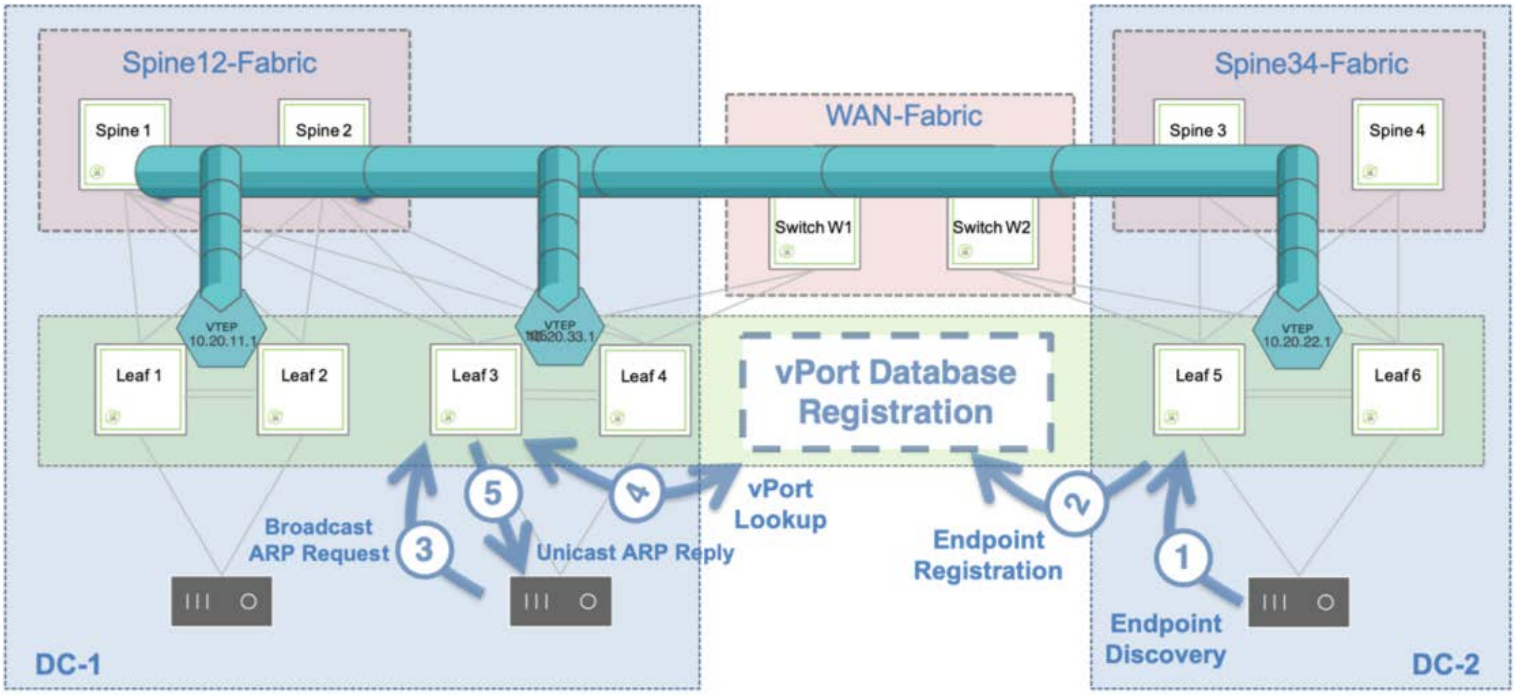


Figure 8-6: ARP Optimization Process

In the figure above, on the right-hand side a host is discovered through the traffic exchange with its neighboring switch. As a consequence, its information is recorded in the vPort database.

On a remote switch (on the left-hand side of the figure) whenever someone sends a broadcast ARP request to learn about the MAC address associated with the IP address of the newly discovered host, an *ARP suppression function* is executed.

In other words, the ARP response comes *proxied* from the fabric based on the contents of the distributed vPort database, which is available on all fabric nodes. Therefore, there is no need for any direct ARP communication between requester and registered host, nor for any flooding of broadcast ARP packets.

> **Note:** The ARP optimization can be useful in various scenarios but may also impact the CPU utilization of the switches. Therefore, it is disabled by default and its use needs to be evaluated on a case-by-case basis.

Moreover, starting from release 2.6.0, Arista has added a further under-the-hood enhancement to the fabric's control plane logic that significantly reduces the need for the less-than-optimal 'flood and learn' scheme for unknown destination traffic.

This enhancement, called *vPort forwarding*, implements a more optimized forwarding logic thanks to the versatility of the vPort database: this logic is analogous to ARP optimization's use of the vPort database information to avoid unnecessary ARP flooding, but it's applied to the general case when the destination Layer 2 address of a generic packet can be successfully looked up in the distributed vPort database so as to avoid having to flood the packet to all its possible destinations.

About ECMP Load Distribution for VXLAN Traffic

Equal-cost multi-path routing (ECMP) is a routing strategy where next-hop packet forwarding can occur over multiple paths on a per flow basis.

Hardware platforms usually perform a hash function on a number of packet fields to perform path selection and determine the load balancing traffic distribution.

In case of VXLAN-encapsulated traffic, for backward-compatibility purposes, the same *legacy* ECMP algorithm can be applied to VXLAN packets too without requiring deeper inspection.

This is because the VXLAN RFC recommends that the encapsulation function add some amount of *entropy* to a VXLAN packet by generating a variable source UDP port:

> *Source Port: It is recommended that the UDP source port number be calculated using a hash of fields from the inner packet -- one example being a hash of the inner Ethernet frame's headers. This is to enable a level of entropy for the ECMP/load-balancing of the VM-to-VM traffic across the VXLAN overlay.*

In other words, that recommendation makes sure that some amount of variability is present when creating VXLAN connections so that they can be easily load-balanced using most ECMP functions implemented by networking platforms.

In particular, any ECMP algorithm that takes into account (also) the *Layer 4 source port* for path determination will be able to take advantage of the level of entropy added to the VXLAN packets as per the RFC.

In case of NetVisor OS this is achieved *by default* by implementing a granular *7-tuple h*ashing scheme that uses the following fields: source IP address, destination IP address, VLAN number, destination Layer 4 port, *source Layer 4 port*, IP protocol number, source physical port.

Other networking devices may use a simpler ECMP configuration referred to as *5-tuple hashing*, which includes just the source and destination IP addresses, the protocol type, and the source and destination Layer 4 ports for UDP or TCP flows.

Adding VXLAN traffic entropy and applying ECMP hashing are basic hardware functions supported by default on Arista switches. These functions are supported by most 3rd party switches and routers as well.

See also the *Displaying ECMP Load Balancing Info for VXLAN* section.

About VXLAN Routing

In most designs the VXLAN overlay network requires that overlay traffic be routed as well as bridged.

This is typically required for the host traffic to be routed across VNI-mapped VLANs and/or to reach the data center gateways, firewalls (or other L3/L4 appliances), so as to be forwarded to the Internet. The reverse path from the Internet also requires that the traffic be routed as well to reach the hosts.

> **Note**: Arista also refers to VXLAN routing with the acronym RIOT, which stands for Routing In and Out of Tunnels.

For hosts on different VLANs to communicate with each other and with the Internet, a cluster of *border leaf*

switches is deployed for redundancy and load splitting purposes to become the overlay network's default gateway. VLAN interfaces are configured as required by the network administrator on the cluster's vRouters (up to 32 vRouters are supported).

This model is sometimes referred to as *centralized routing* for the VXLAN fabric (as opposed to the *distributed routing* model implemented with VRFs, as explained in a subsequent section).

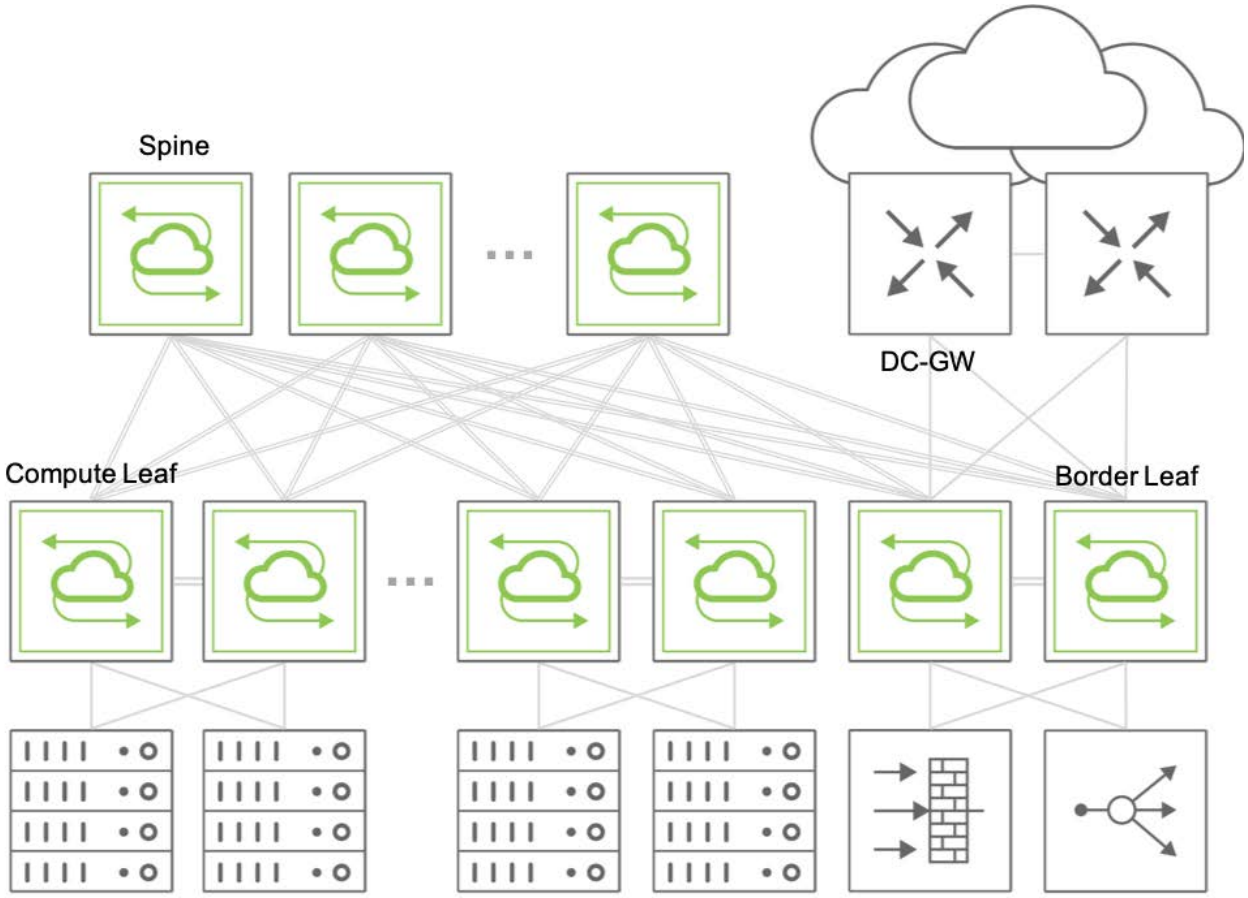


Figure 8-7: Example of Fabric Design Using Centralized Routing

In the centralized routing model, any fabric packets that need to be routed between two hosts in different VNI-mapped VLANs are sent to a centralized overlay vRouter and then VXLAN-encapsulated or -decapsulated depending on source and/or destination's host location.

When multiple sequential hardware operations are required, such as in the forwarding –> decapsulation –> routing sequence, or in the reverse routing –> encapsulation –> forwarding sequence (or also when any packet replication is needed), Arista leverages the hardware's capabilities to implement *multiple packet lookup and rewrite passes* using a technology called *recirculation* (or hardware loopback).

This technique is used whenever a certain switch model is incapable of applying all the necessary operations to certain traffic in a single instance (that is, in a single pass). Hence, such traffic will have to be recirculated at least once through the hardware forwarding engine.

Arista supports the recirculation technique with the dedicated `vxlan-loopback-trunk` configuration.

About the VXLAN Loopback Trunk

NetVisor OS uses the `vxlan-loopback-trunk` configuration command to set aside a number of (internal or front-panel) ports that are configured as a logical group (i.e., a dedicated port channel) for recirculation.

This special trunk link is auto-created and is reserved to be used to carve out the necessary amount of bandwidth required for the recirculation of packets.

Typically, the network administrator allocates a number of ports (e.g., 2 or more for redundancy purposes) to a VXLAN loopback trunk that is sufficient to provide an amount of aggregate bandwidth *in excess of* what is required for routing and/or replication of all Broadcast/Unknown Unicast/Multicast (BUM) VXLAN traffic.

From an operational standpoint, if a packet needs to be recirculated after decapsulation as part of the routing operation, Layer 2 entries for the vRouter MAC address or the VRRP MAC address on VNI-mapped VLANs are programmed to point to the `vxlan-loopback-trunk` link in hardware.

Therefore, to verify that the traffic can be recirculated, the network administrator can simply look at output of the `l2-table-show` command, which should display a special `vxlan-loopback` flag to indicate the recirculation-related hardware state (see also the Checking VXLAN Recirculation L2 and L3 Entries section below).

In summary, it is important to size the VXLAN loopback trunk appropriately and, in certain cases, it may be helpful to verify the operational state of the Layer 2 entries associated to it in the hardware tables.

When monitoring the fabric, it is also useful to periodically check the hardware recirculation counters (as discussed in the *Showing VXLAN Trunk Replication Counters* below) so that the network admin can stay abreast on the peak recirculation bandwidth needs of the various fabric switches.

For sizing purposes, note that the VXLAN loopback infrastructure is used for routing traffic before VXLAN encapsulation and/or after VXLAN decapsulation. It is also used for bridging broadcast , unknown unicast or multicast (BUM, in short) traffic in the overlay network.

On the other hand, non-routed known unicast traffic is forwarded and encapsulated, or decapsulated and forwarded, in one single pass *without requiring the VXLAN loopback trunk* .

Starting from NetVisor OS releases 6.0.0 and 6.0.1 new single-pass configurations are supported on certain switch models using newer hardware forwarding engines. In particular, it is possible to enable single-pass unicast VXLAN routing, single-pass broadcast/unknown unicast and multicast flooding, as well as single-pass distributed multicast bridging. Therefore, by leveraging these hardware capabilities when supported, it is possible to eliminate the need for loopback trunks in certain configurations. For more details, refer to the Configuring the VXLAN Loopback Trunk section.

About the VXLAN Configuration Models

NetVisor OS offers three powerful VXLAN configuration models.

The first model is aimed at full configuration automation and therefore is likely to be used more often. It leverages a special configuration abstraction called a *VTEP object*. When VTEP objects are created on the nodes, they trigger the automatic creation of VXLAN tunnels from such nodes, thus creating a full mesh of bidirectional connections between the nodes. In addition, starting from NetVisor OS release 6.0.0 it is possible to automatically add a certain VLAN/VNI pair to all the VXLAN connections with the `auto-vxlan` keyword (note that, if the VNI value is not specified by the user, it is picked automatically by the system

provided that the configuration scope is *fabric*.)

This any-to-any model is the most convenient one to quickly set up a full VXLAN fabric between multiple leaf nodes and associate to it all of the configured VLAN/VNI mappings. For more details, refer to the Configuring VTEP Objects with Automatic Fabric Connections section.

Figure 8-8 shows an example of mesh with three leaf nodes configured with three VTEP objects.

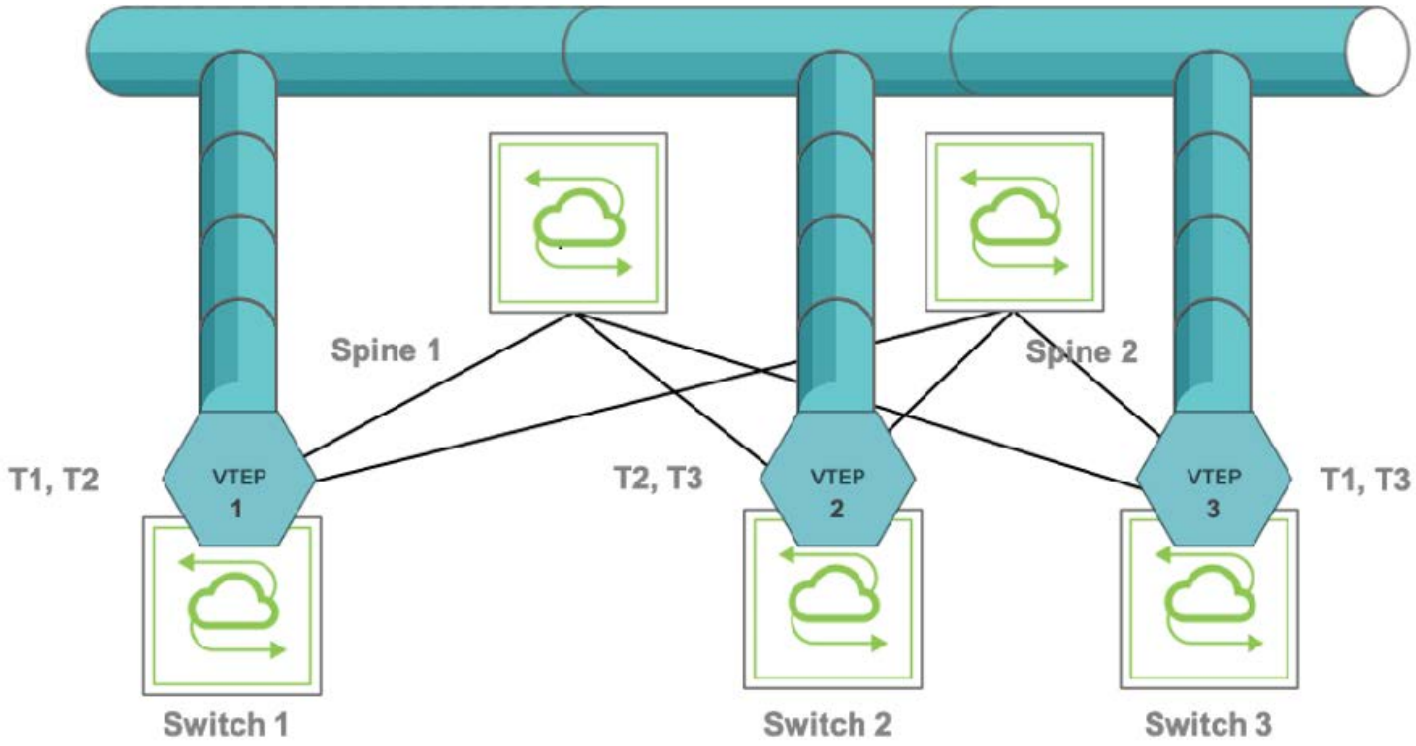


Figure 8-8: Example of VTEP Object Configuration

Bidirectional connections T1 through T3 are called *auto-tunnels* because they are automatically configured by the system when the VTEP objects are created with the `vtep-create` command. See the *Configuring VTEP Objects with Automatic Fabric Connections* section below for the naming of the tunnels automatically generated for the topology in ***Figure 8-8***.

The second model is more granular because it requires more explicit configuration, as it doesn't use the `auto-vxlan` feature (which also means it does not require release 6.0.0 or later).

In this scenario, the user may not want to extend all the VNIs to all the VTEPs and their mesh of VXLAN connections. The user may instead want to extend a VNI, say, to VTEP1 and VTEP2 only (in the figure above) by using the `vtep-vxlan-add` command *selectively*. In such case, only two tunnels are automatically created between VTEP1 and VTEP2 for the chosen VNI (the two auto-tunnels are represented by the bidirectional T2 connection in the figure above), whereas T1 and T3 would not be created for the specific VNI per explicit user choice.

In this model, semi-manual establishment of VXLAN connections is required since the user is responsible to explicitly select which VNIs should be added to each VTEP.

Starting from NetVisor OS release 6.0.1 a third model has been introduced: it's the *hub-and-spoke configuration* model. When used in conjunction with bridge domains, it mimics the behavior of the Metro Ethernet Forum (MEF) *E-Tree service*.

This model leverages a new keyword, `isolated`, and is not compatible with the `auto-vxlan` assignment. The VTEP on the hub node is configured normally, just as in case 2 above, whereas the VTEPs on the spokes are configured with the `isolated` keyword because they are not supposed to communicate with each other.

Figure 8-9 below shows an example of hub-and-spoke VXLAN topology where the hub node has bidirectional connections to the spokes but the spokes don't have connections between each other.

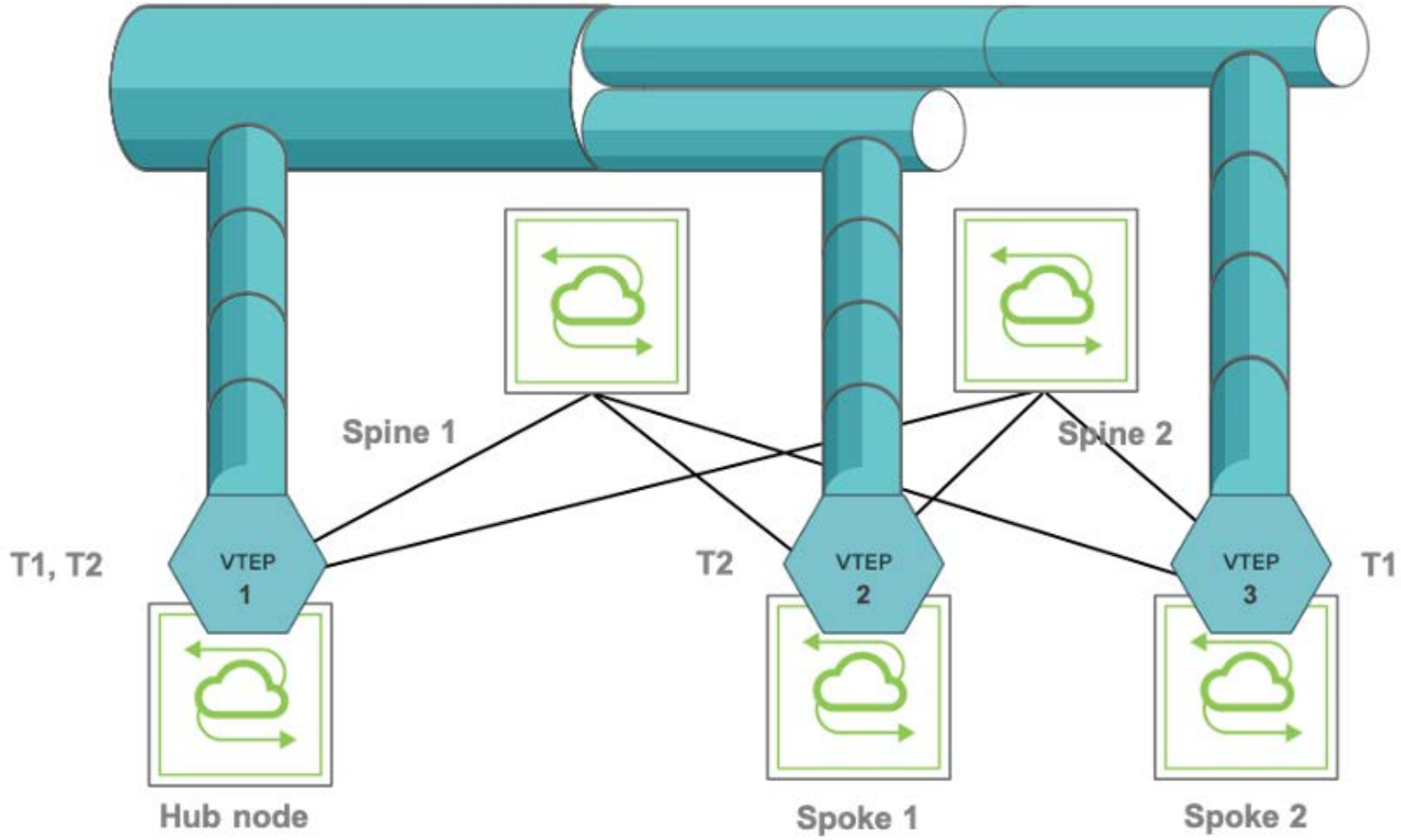


Figure 8-9: Example of Hub-and-spoke VXLAN Configuration

> **Note:** The above two figures actually show a *virtual connection view on a per-VNI basis*: that means that tunnels are automatically created but then it's the per-VNI connectivity policy that determines whether nodes associated with each VNI are actually connected in a virtual mesh or in a virtual hub-and-spoke configuration.

For configuration details and examples, refer to the Configuring VTEP Objects with Automatic Fabric Connections section below.

About Unicast Fabric Virtual Routing and Forwarding (VRF) with Anycast Gateway

NetVisor OS Unified Cloud Fabric adds Layer 3 segmentation to VXLAN interconnections with the support of VRF (Virtual Routing and Forwarding) instances, complementing the vRouter construct and offering a highly scalable distributed routing solution to network architects.

NetVisor supports VRF as a hardware technology allowing multiple routing spaces to coexist on the same distributed fabric architecture. Furthermore, with the addition of the Anycast Gateway functionality, the Unified Cloud Fabric enables distributed forwarding at the first hop router as well as intrinsic VM mobility capabilities across complex multi-site data center designs. This guarantees the maximum VRF scalability possible, limited only by the specific forwarding ASIC capabilities.

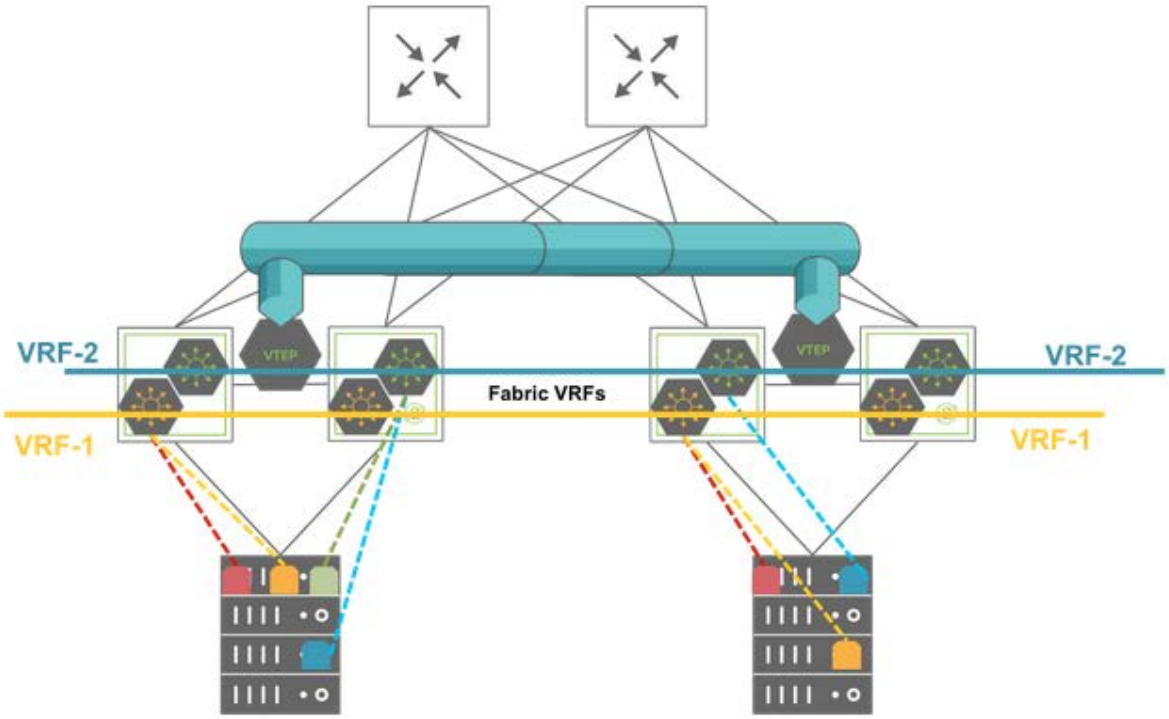


Figure 8-10: East-West Traffic Segmentation with Multiple VRF Instances

NetVisor Fabric VRFs have the following advantages:

- High scalability with support for a large number of VRF instances on a single fabric node (in the order of thousands depending on hardware capacity especially with newer ASICs and as an aggregate number fabric-wide).
- High performance distributed routing of East-West traffic at the Top-of-Rack (ToR) switch level. The distributed routing capability hosted on each leaf node avoids the need for hair pinning traffic to a centralized vRouter.
- Small forwarding state to manage on each node.
- Native redundancy without needing dedicated redundancy protocols (and potentially extra overhead).

- Dual stack support for IPv4 and IPv6 subnets.

- Simple fabric-wide configuration and management (typical provisioning overhead is proportional to: (number_of_VRFs + number_of_VLANs + number_of_switches) instead of the industry average of up to (number_of_VLANs * number_of_switches).

- IPv4 and IPv6 subnets can be automatically stretched to multiple locations without extra configuration.

- Starting from NetVisor OS release 6.0.1, subnet prefixes can be imported/exported between VRFs by using an innovative feature called virtual service group (vSG).

Fabric VRFs are lightweight distributed atomic constructs created without the need for a local vRouter and they do not currently support any routing protocols on VRF instances. This choice enables very high scalability and very low overhead in the management of the distributed segmentation and routing function.

You can connect fabric VRFs to third party VRF routers or gateways either directly using static routing or through a redundant group of border leaf switch(es) running the vRouter function mapping 1:1 to the Fabric VRF instances. In the latter case, border leaf switches can run any supported IGP protocol to interconnect with third party VRF routers or gateways.

For redundancy purposes, you can configure two VRF routers or gateways, sometimes referred to as *DC gateways*, can be configured per VRF (`vrf-gw` and `vrf-gw2`).

To be more precise, these two important configuration parameters represent two static default routes for northbound traffic. They can be quite flexible: after VRF global creation, they can be locally modified by using the `vrf-modify` command and allowing the implementation of different exit points for a VRF depending on the switch location.

In addition, static routing (with the `vrf-route-add` command) can be leveraged to augment them, for example, to install more than two routes or to change the VRF exit point for specific destination prefixes.

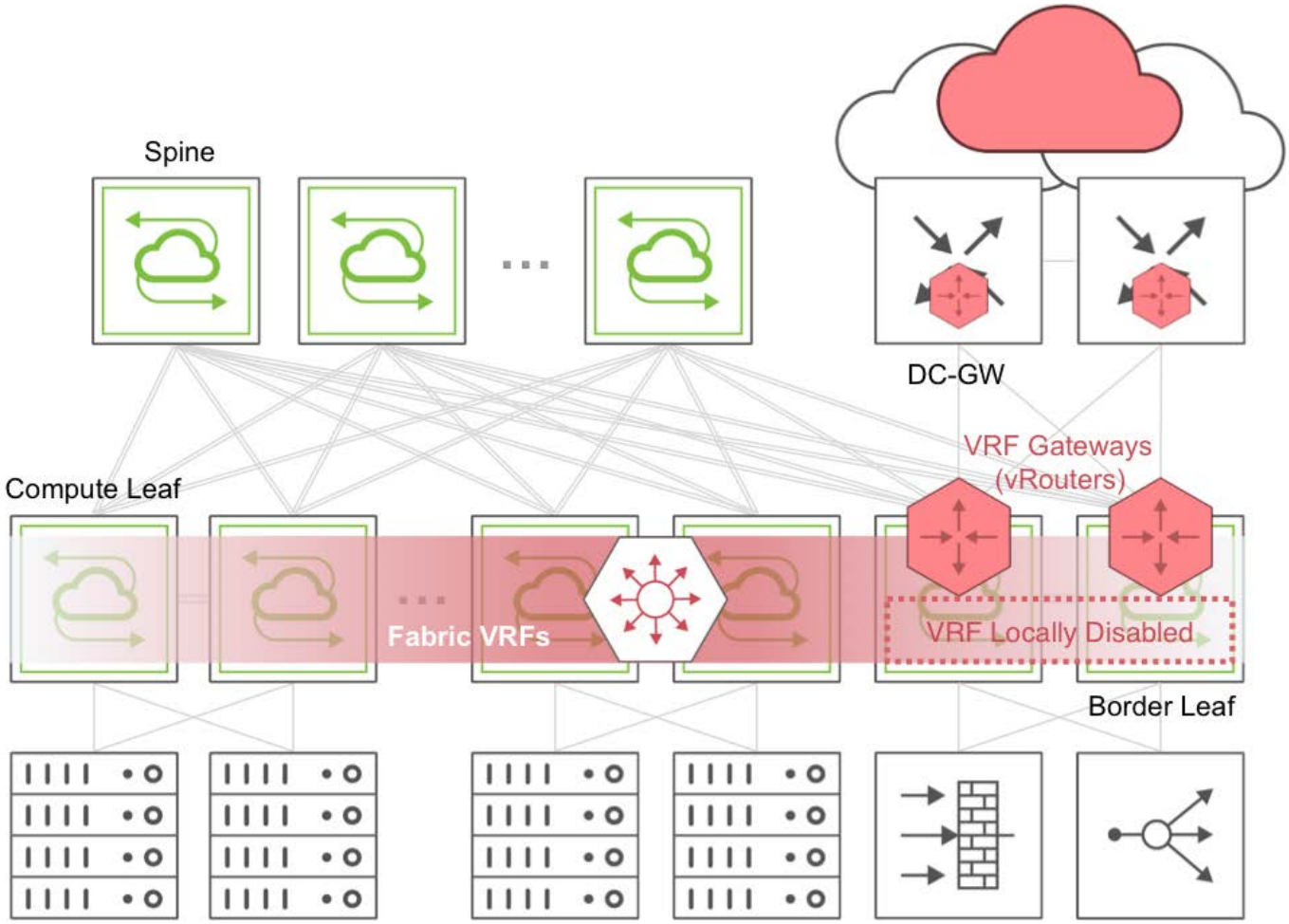


Figure 8-11: East-West Traffic Segmentation with North-South VRF (DC) Gateways

As part of the Fabric VRFs configuration, you can create IPv4 and IPv6 subnets, which are atomic objects in the Fabric data plane to associate to the VRF instances in order to implement distributed traffic segmentation.

In particular, Fabric leaf switches use subnet objects for management purposes to represent groups of directly connected hosts with a fabric wide scope across the VXLAN interconnect. NetVisor OS also uses them to program subnet routes into the hardware to send Layer 3 packets corresponding to unresolved adjacencies to the software so that next-hop resolution through ARP requests can be performed. When a host responds to the ARP request(s), more specific Layer 2 and Layer 3 host entries are configured in the hardware so that end-to-end forwarding ensues.

In addition, NetVisor OS supports the anycast gateway routing function for the Fabric VRFs to enable distributed first-hop routing, redundancy and mobility. This capability uses a dedicated virtual MAC address, called the anycast gateway MAC address, which gets associated with configurable anycast gateway IP addresses as part of the subnet object configuration.

The default MAC address for the anycast gateway function is 64:0e:94:40:00:02. It can be displayed with the `fabric-anycast-mac-show` command. If necessary, you can also modify it using the `fabric-anycast-mac-modify` command.
Furthermore, as a key VRF-aware service, NetVisor OS supports end host address assignment through the DHCP packet relay function for up to two DHCP servers.

About IGMP Snooping Support with VXLAN

Starting with version 3.1.1, NetVisor OS adds support for selective replication of (instead of always flooding) multicast traffic based on IGMP join messages received over ports and tunnels.

With this enhancement multicast traffic belonging to a group is forwarded only to member ports and relevant remote VTEPs.

This feature uses the head-end replication (HER) model for replication of packets to be sent over to remote VTEPs. NetVisor OS also forwards IGMP join messages over VXLAN tunnels, for other fabric switches to see those messages and as a consequence build the group membership list accordingly.

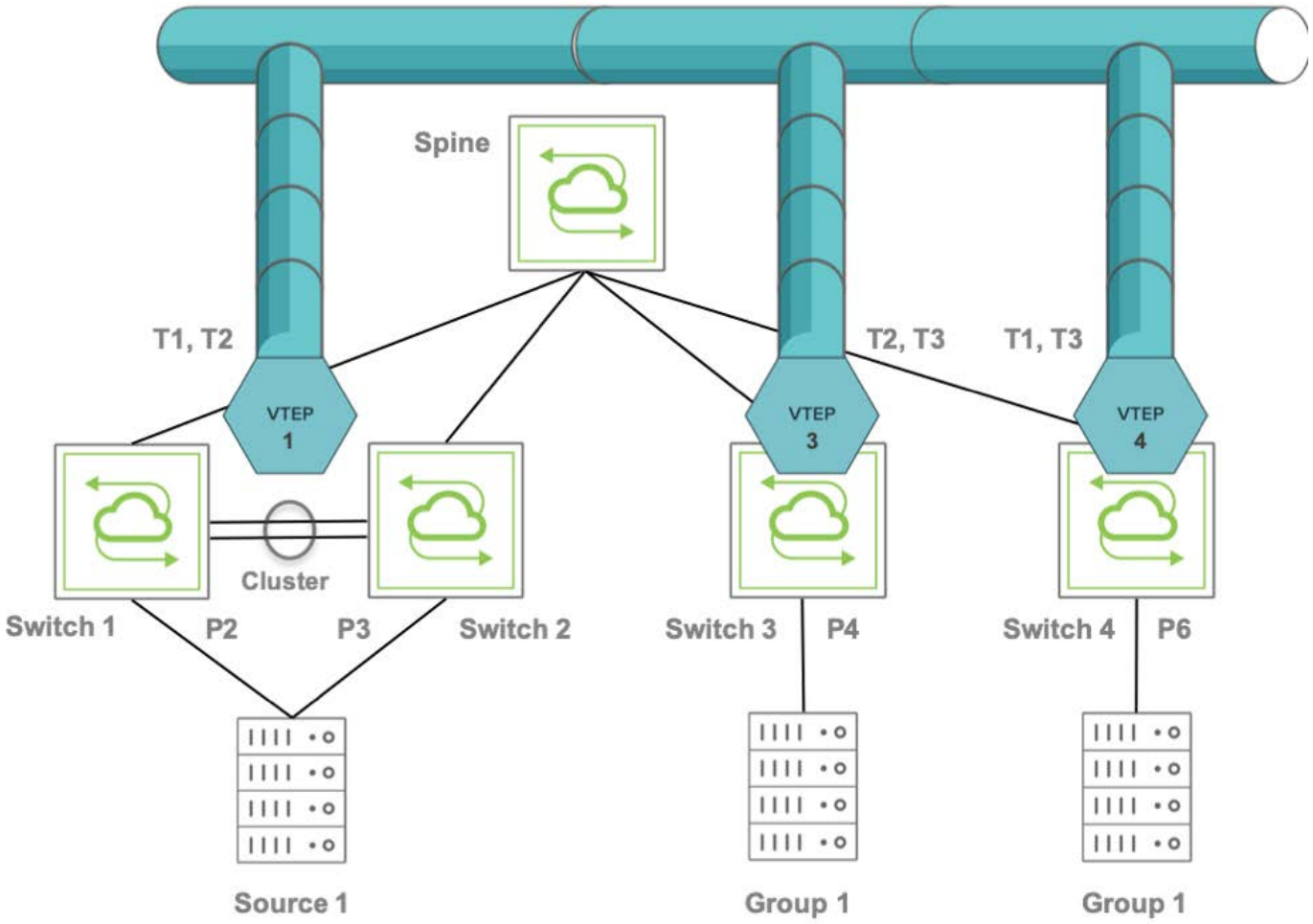


Figure 8-12: Example of Fabric Topology with Multicast Distribution

The topology in the ***Figure 8-12*** includes Switch 1 and Switch 2 configured as a cluster pair that uses a common virtual IP address (VIP) as the source for two tunnels T1 and T2.

The cluster pair, Switch 1 and Switch 2, appears as one logical switch with a common VXLAN endpoint VTEP1. Two tunnels, T1 and T2, are created with the same local VIP toward the other two end points, VTEP3 and VTEP 4.

The spine switch hashes the traffic from Switch 3 or Switch 4 to the cluster pair, load balancing

between Switch 1 and Switch 2.

The above topology includes Switches 1, 2, 3, and 4 with ports 2,3,4, and 6, as part of the same broadcast domain (say, VXLAN ID 10).

Initially, port 4 sends an IGMP join messages for the G1 multicast group. Hence Switch 3 adds local port 4 (P4) as an IGMP member for Layer 2 multicast group G1.

In addition, Switch 3 floods IGMP packets to the remote VTEPs 1 and 4. Hence the remote switches associated with those VTEPs receive IGMP packets and add G1 as Layer 2 multicast group.

On VTEP1's cluster the flooded IGMP join message is also forwarded (synced) to the cluster peer through the out-of-band channel so that both cluster peers can see and program the same Layer 2 multicast group entry.

Next, P6 joins group G1 too, and the IGMP join packet is flooded to the remote VTEPs 1 and 3.

Now the group membership is:

- On both Switch 1 and Switch 2, VTEP 3 and 4
- On Switch 3, local port P4 and VTEP4
- On Switch 4, local port P6 and VTEP 3

Then, if source S1 sends multicast traffic on P3, that traffic matches the MAC address corresponding to G1's DMAC on VXLAN ID 10 in the Layer 2 table therefore the hardware bridges that traffic to the remote VTEPs 3 and 4. After receiving it, Switch 3 and Switch 4 check the Layer 2 table and forward the traffic to local receivers on P4 on Switch 3 and on P6 on Switch 4.

Refer to the *Configuring IGMP Snooping with VXLAN* section for the configuration details.

About Distributed Multicast Forwarding with Fabric VRFs

Note: This feature is supported only on certain switch models as described in the section Configuring Multicast Fabric VRFs below.

By leveraging the distributed control plane of the Unified Cloud Fabric, Arista has extended unicast Fabric VRFs (as described above) to also include multicast routing support.

This enhancement is referred to as *Multicast Fabric VRFs with distributed routing support*.

It leverages Arista fabric's native capabilities to route multicast traffic without employing a multicast routing protocol (such as PIM) in the underlay or overlay networks.

In particular, Multicast Fabric VRFs are a *logical extension* of the IGMP Snooping feature described in the previous section.

That is because Multicast Fabric VRFs use techniques analogous to IGMP Snooping to handle and selectively forward multicast traffic.

In particular, with Multicast Fabric VRFs, the multicast forwarding logic is enhanced to also handle inter-VLAN forwarding and replication.

These are the lookup steps that are implemented.

First, multicast traffic may need to be locally bridged and/or remotely forwarded to multicast receivers in the same VXLAN ID-mapped VLAN on the other overlay nodes.

In addition, multicast receivers may be located locally in one or more distinct VLANs and hence multicast traffic may need to be routed and replicated to those VLANs as well.

Arista leverages Layer 2 hardware table entries to point to local ports and tunnels (represented by logical ports, as seen in the previous section). Those entries point to a loopback trunk to implement a *second pass lookup*. This pass corresponds to a hardware (*, G, VLAN) L3 multicast lookup for any of the local receiver ports located in different output VLANs.

Furthermore, once a multicast packet reaches a remote VTEP in the overlay network, it is decapsulated and then L3 multicast entries route the traffic locally on that node. The decision to which VXLAN ID(s) to replicate the decapsulated packets is based on the IGMP snooping logic (which knows where all receivers attached to the node are).

Refer to the Configuring Multicast Fabric VRFs section for more details on how to enable the feature and to configure the appropriate loopback trunk.

About Arista Open Networking Multi-site Fabric

Data Center Interconnect (DCI) is the process of connecting two or more geographically distributed data center locations (also known as *multi-site fabric network designs*).

It is employed to ensure that data is exchanged consistently and in a timely manner among different locations for redundancy, high availability and load sharing purposes, to support a number of important design requirements that include: service resiliency (i.e., fault tolerance), performance and scale, disaster recovery, operational efficiency, local country regulations.

A multi-site Unified Cloud Fabric design is Arista's best-in-class standards-based solution to address customer requirements for multi-site data center connectivity. It leverages the VXLAN-based feature set to be able to extend across a generic IP transport core network, as depicted in the ***Figure 8-13*** below.

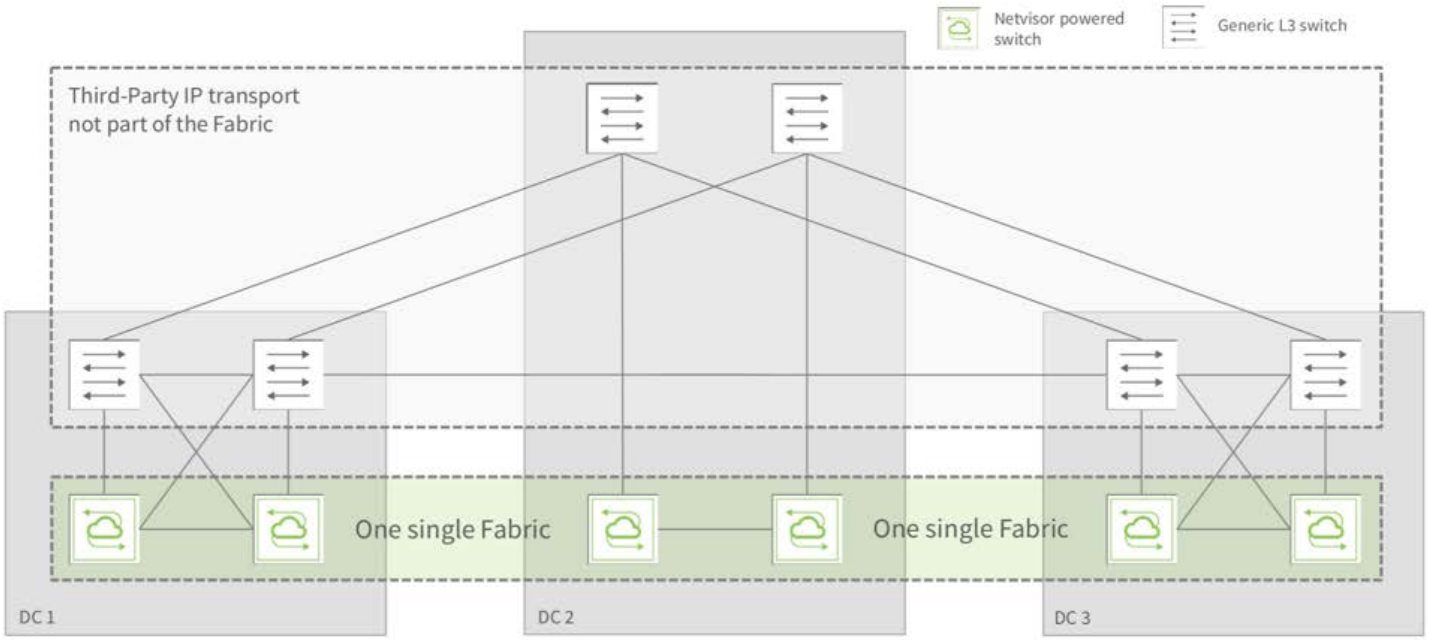


Figure 8-13: Multisite Fabric Topology Over an Agnostic IP-routed Core

Thanks to VXLAN, it boasts a number of very desirable characteristics: it's simple, very high performance, highly redundant and can scale up to support dozens of sites, while providing high availability (HA) with ~200 ms failover times.

It's flexible, multi-tenant, interoperable and topology agnostic: it works with any L3 core/underlay and uses sophisticated VXLAN-based Layer 2 extension and pseudo-wire technologies to achieve transparent inter-site communication with end-point tracking.

This enables the support for important use cases such as *VM mobility* for business continuity and geographical load sharing. In addition, it natively supports end-to-end visibility of client-server application traffic as well as of server-to-server communication flows.

In order to implement an Open DCI architecture that can scale to a multi-site design it is important to bring together all the foundational technologies discussed in the previous sections:

- Unified Cloud Fabric's distributed control plane
- Switch clusters with vLAGs
- Fabric traffic optimizations
- Virtual Networks (vNETs)-based network provisioning
- Analytics collection

Arista provides support for different underlay architectures as well as for different routing options (centralized or distributed) for the fabric overlay.

Distributed forwarding with *Fabric VRFs* represents the most scalable multi-site DC design option (with both unicast and multicast support), whose configuration steps are detailed in the following sections.

Guidelines and Limitations

For most fabric designs the general recommendation is to implement a Layer 3 underlay for scalability and ease of interoperability: BGP and OSPF are solid and mature protocols that in a great number of scenarios offer the most effective way to design and run a fabric.

Whenever possible, for redundancy, scalability and optimal trunk load splitting it is recommended to allocate at least 50% extra bandwidth to VXLAN loopback trunks in excess of the peak aggregate bandwidth expected to be used by all the fabric traffic when recirculated.

These are some common limitations to consider when planning and configuring VXLAN deployments:

1. For VXLAN centralized routingdesigns up to 32 vRouters are supported (the actual maximum number is platform-dependent).
2. For Fabric VRF distributed designs the number of supported VRFs is in in the order of thousands, depending on the specific switch model's hardware capacity.

Configuring the VXLAN Underlay Network

Configuring the underlay network involves the configuration of (at least) the following base features:

- Layer 2
- Layer 3
- The fabric
- Clustering
- vLAGs
- Optionally (even though it's highly recommended), traffic analytics

Readers are referred to the respective configuration sections of each feature for more details.

In addition, on Arista switches jumbo frame support can be enabled on the appropriate interfaces to accept frames with an MTU larger than 1500 bytes (namely, with a 'jumbo' size). Refer to the *Enabling Jumbo Frame Support* section earlier in this guide for details.

This guarantees that all standard maximum size packets to be transmitted in the overlay network are not dropped by the underlay.

Moreover, starting from release 2.5.0, NetVisor OS enforces a routing MTU size in hardware, which can be set with the `mtu` option when adding vRouter interfaces (see examples below).

Therefore, in order to avoid large frames to be dropped during Layer 3 forwarding or during encapsulation, in addition to enabling jumbo frames on physical interfaces, it is also important to *configure routed interfaces with the proper MTU settings* (typically between 1580 and 9398 bytes).

Changes to interface attributes (such as the MTU) are recommended to be applied in the initial phases of the underlay configuration. Moreover, changing the MTU on certain higher-level entities such as trunk ports may require more specific commands (such as `trunk-modify` instead of the basic `port-config-modify` command).

Configuring the Overlay: VTEP Interconnections and VNIs

VTEPs can be configured as individual vRouter interfaces. However, as discussed in the *About VTEP High Availability* section, VTEPs are more commonly configured on switch pairs running VRRP to support redundant logical VIPs for VXLAN termination.

In this latter case, the first step is to create a VIP instead of a regular interface.

Both cases are exemplified below (a. and b.) in the list of steps required to set up the overlay:

1. First configure the underlay's vRouter interfaces, with the proper MTU:

   a) Create a vRouter and add a vRouter interface for each VTEP:

   ```
   CLI (network-admin@switch) > vrouter-create name <vr-name> vnet <vnet-name> router-type hardware hw-vrrp-id <id>

   CLI (network-admin@switch) > vrouter-interface-add vrouter-name <vr-name> ip <network/netmask> vlan <y> mtu <mtu>
   ```

   b) For VTEP HA instead add a vRouter interface using VRRP:

   ```
   CLI (network-admin@switch) > vrouter-interface-add vrouter-name <vr-name> ip <network/netmask> vlan <y> vrrp-id <id> vrrp-primary <ethz.y> mtu <mtu>
   ```

   `<mtu>` can be set for example to 1580 bytes (or more).

2. Once the VTEPs are created, configure the VTEP connections (also referred to as 'tunnels') from sources to destinations. On non-redundant switches, the tunnel is created with scope `local` whereas on redundant switches the tunnel is created with scope `cluster`:

   ```
   CLI (network-admin@switch) > tunnel-create name <tunnel-name> local-ip <ip1> remote-ip <ip2> scope local vrouter-name <vr-name>

   CLI (network-admin@switch) > tunnel-create name <tunnel-name> local-ip <vip1> remote-ip <vip2> scope cluster vrouter-name <vr-name> peer-vrouter-name  <peer-vr-name>
   ```

3. Then create the mappings between VNIs and VLANs on the respective switches:

   ```
   CLI (network-admin@switch) > vlan-create scope <scope> id <vlan-id> vxlan <vnid>
   ```

   > **Note:** A VLAN can be associated to a VNI when created on a VTEP HA pair with the `vlan-create scope cluster id <vlan-id> vxlan <vnid>` command. Also, the mappings can be set up also with the `vlan-modify id <vlan-id> vxlan <vnid>` command *after VLAN creation*.

   Then, to add ports to a VLAN created with `vlan-create` command, use the `vlan-port-add` command, for example:

```
CLI (network-admin@switch) > vlan-port-add vlan-id <vlan-id> ports <port
numbers>
```

To delete a VLAN with its mapping, use the `vlan-delete` command.

Lastly, to display the information about a VLAN, for example to verify a VNI mapping and the list of ports added to it, use the `vlan-show` command:

```
CLI (network-admin@switch) > vlan-show id 70 format id, type, vxlan, scope,
description, ports, untagged-ports layout vertical

id:  70
type:     public
vxlan:    70000
scope:    cluster
description:  vlan-70
ports:    0-2, 5-48, 50-52, 54-56, 63-70,272-273, 275-276, 278-280, 397
untagged-ports:  none
```

4. Add the required VNI mappings to the VXLAN connections:

   ```
   CLI (network-admin@switch) > tunnel-vxlan-add name <tunnel-name> vxlan
   <vnid>
   ```

5. For monitoring VXLAN specific states and statistics, use the following commands:

`vlan-show`	Displays the VXLAN ID associated with the VLAN ID.
`tunnel-show`	Displays the configured tunnel and the state.
`trunk-show`	Displays the port used for BUM traffic re-circulation.
`port-stats-show`	Displays statistics for each port.
`tunnel-stats-show`	Displays statistics for each tunnel.
`vxlan-stats-show`	Displays statistics for each VXLAN ID.

Note: The above configuration model is called ‘manual VXLAN tunnel creation’ and is preferred for example when the user needs to have maximum control and granularity of configuration. On the other hand, when more automation is needed, an alternate configuration model is available and is described below in the *Configuring VTEP Objects with Automatic Fabric Connections* section.

Configuring the VXLAN Loopback Trunk

In order to perform routing on the overlay and to replicate Layer 2 broadcast, unknown unicast and multicast (BUM) traffic over multiple VXLAN tunnels on Arista switches, you must add ports to a `vxlan-loopback-trunk` using the following command:

```
CLI (network-admin@switch) > trunk-modify name vxlan-loopback-trunk ports
<list of ports> jumbo
```

Where `<list of ports>` may vary based on the hardware architecture of the switch (on most models it is likely to be a list of front panel ports set aside for this use.)

Check the data sheet of your switch model to verify the number of internal ports as well as front-panel ports available and their respective speeds. Depending on the expected peak amount of routed and BUM traffic, you can use either N x 10 GbE ports or two (or more) 40 GbE ports (or even 100 GbE).

The above command also prints out the following informational message:

```
!!!! Ports updated on vxlan-loopback-trunk. Ports added only support vxlan
recirculation. Any features configured on these ports may not work. Please
update the necessary config. Any ports removed need necessary adjustment on
autoneg,jumbo,etc  !!!
```

If you create a VLAN with associated VXLAN ID when `vxlan-loopback-trunk` is without any active ports, you will get this warning message:

```
CLI (network-admin@switch) > vlan-create id 100 scope local vxlan 1001000

!!!! Vlan 100 created, but there are no ports configured in vxlan-loopback-
trunk. vxlan forwarding may not function correctly. !!!!
```

Single-pass VXLAN Forwarding

Starting from NetVisor OS release 6.0.0, on NRU03 and NRU-S0301 platforms and on the Dell S5200 Series, more advanced hardware capabilities enable the switches to perform routing in and out of tunnels (RIOT) in a single pass (that is, without requiring the loopback trunk).

> **Note**: On the Dell S4100 series single pass RIOT is supported for distributed VXLAN routing with Unicast Fabric VRFs, however single pass RIOT is not supported for centralized VXLAN routing on a DC Gateway border leaf.

On the supported platforms, this capability can be enabled with the command:

```
CLI (network-admin@switch) > system settings-modify no-single-pass-riot|
single-pass-riot
```

The default setting is `no-single-pass-riot`. A reboot is required for the configuration change to take effect and an appropriate message is displayed to remind the user of this requirement.

Furthermore on all hardware platforms, starting from NetVisor OS release 6.0.0, a new global command is introduced to enable single-pass replication (i.e., flooding) of Layer 2 broadcast, unknown unicast and multicast (BUM) traffic over multiple VXLAN tunnels: `system-settings-modify no-single-pass-flood|single-pass-flood`.

By default, `single-pass-flood` mode is disabled. A reboot is required for the configuration change to take effect and an appropriate message is displayed to remind the user of this requirement.

For example:

```
CLI (network-admin@switch2) > system-settings-modify single-pass-flood
!!!! Please reboot the system for the new single-pass-flood setting to take
effect correctly !!!!

CLI (network-admin@switch2) > system-settings-show format single-pass-flood
single-pass-flood: on

CLI (network-admin@switch2) > switch-reboot
```

After the reboot, the `tunnel-show flood-nexthop` command can be used to display the path that is selected as next hop for flooded traffic out of the available ECMP paths, when `single-pass-flood` is enabled:

```
CLI (network-admin@switch2) > tunnel-show flood-nexthop

switch   name   flood-nh-ip  flood-nh-vlan  flood-nh-port  flood-nh-mac        flood-nh-egress-id
-------  -----  -----------  -------------  -------------  ----------------  ------------------
switch1
switch2 tnl2   12.12.12.7    4088            70           66:0e:94:1a:e3:d3   100008
switch2 tnl1   10.10.10.7    4092            69           66:0e:94:1a:e3:d3   100007
```

In the above output `switch1` has no info as that node is not in `single-pass-flood` mode.

> **Note** that, due to a hardware limitation in single-pass mode, ECMP load balancing of flooded traffic is only performed on a per-tunnel basis. In other words, flooded traffic for a specific tunnel will always take the same active ECMP path, which is selected at random out of the available ECMP next hops.

The `port-stats-show` command can be used to display the traffic statistics on the loopback ports and on tunnel ports:

```
CLI (network-admin@switch2) > port-stats-show format
time,port,ibits,iUpkts,iMpkts,iCongDrops,ierrs,obits,oUpkts,oBpkts,oMpkts,o
CongDrops

time     port ibits iUpkts iBpkts iMpkts iCongDrops ierrs obits oUpkts oBpkts oMpkts oCongDrops
-------- ---- ----- ------ ------ ------ ---------- ----- ----- ------ ------ ------ ----------
18:28:56 1    0     0      0      0      0          0     0     0      0      0      0
18:28:56 49   110K  0      85     58     0          0     174K  0      104    127    0 0     0
18:28:56 69   41.8K 0      0      10     0          0     119K  85     0      11     0 0     0
```

In this example, port 1 is the VXLAN loopback port (with a zero packet count, as the switch is in `single-pass-flood` mode), while port 49 is the input port where the broadcast packets are received and port 69 is the tunnel output port where the encapsulated flooded packets are sent to. The same packet count (along with the corresponding byte count) is displayed in the `HER-pkts` and `HER-bytes` fields of `tunnel-stats-show`:

```
CLI (network-admin@switch2) > tunnel-stats-show format switch,HER-pkts,HER-
bytes

switch          HER-pkts  HER-bytes
--------------  --------  ---------
switch2         85        98K
```

Starting from NetVisor OS release 6.0.1, analogously to the BUM traffic replication case described above, support for single-pass Layer 2 forwarding of *known* multicast traffic is introduced.

IGMP snooping is used to determine the output interfaces corresponding to the multicast receivers as described in the section titled Configuring IGMP Snooping with VXLAN (refer to that section for configuration details). A new parameter is introduced to perform single-pass L2 forwarding for this multicast traffic with *known* receivers:

```
CLI (network-admin@switch) > system-settings-modify no-single-pass-l2-
known-multicast | single-pass-l2-known-multicast
```

The `no-single-pass-l2-known-multicast` is the default setting. A reboot is required for a change in setting to take effect, as reminded by the message shown below:

```
CLI (network-admin@switch) > system-settings-modify single-pass-l2-known-
multicast
!!!! Please reboot the system for the new single-pass-l2-known-multicast
setting to take effect correctly !!!!

CLI (network-admin@switch) > system-settings-show format single-pass-l2-
known-multicast
single-pass-l2-known-multicast:  on
```

Note: that as in the BUM forwarding case, due to a hardware limitation in single-pass mode, ECMP load balancing of multicast traffic is only performed on a per-tunnel basis. In other words, forwarded multicast traffic for a specific tunnel will always take the same active ECMP path, which is selected at random out of the available ECMP next hops.

Note: On capable devices, single-pass features can be used to avoid consuming front panel ports for the recirculation function.

Configuring VLAN 1 with VXLAN

VLAN 1 is also known as the *default VLAN*, as it is assigned by default to switch ports. It is also used by default on Arista switches to transport fabric-specific traffic.

In general, it is a common best practice to not use VLAN 1 network-wide to carry user traffic, especially to avoid potential misconfigurations due to user error.

However, in certain cases, it is required to use and transport VLAN 1 over a VXLAN-based fabric and hence it needs to be associated with a VNI. Before NetVisor OS release 6.0.0 attempting to map VLAN 1 to a VNI would be rejected, like so:

```
CLI (network-admin@switch) > vlan-modify id 1 vxlan 123
vlan-modify: Vlan id 1 is not a valid vlan, or is a reserved vlan
```

Starting from release 6.0.0, if the fabric VLAN is modified to a value different from 1, it is then possible to map VLAN 1 to a VNI like so:

```
CLI (network-admin@switch) > fabric-local-modify vlan 100

CLI (network-admin@switch) >  vlan-modify id 1 vxlan 123
vlan-modify: ports 13,25-26,69-Disabling MAC Address Learning72 are tagged
in vlan 20, but untagged in vlan 1 with vxlan not allowed, Adjust the port
membership and retry the vxlan configuration
```

Note that the `fabric-local-modify` command is—as the name implies—*local to each node* and hence it needs to be executed on all the nodes that are part of the fabric (for example, with the `switch *` command prefix). Moreover, although `fabric-local-modify` was available even before release 6.0.0, the VLAN 1 to VNI mapping was not allowed until version 6.0.0.

Also note that, as shown above, if the *untagged port membership* of VLAN 1 comprises any ports, due to a hardware limitation, the `vlan-modify id 1 vxlan <value>` command will still fail. You need to make sure there are no ports using VLAN 1 as untagged VLAN in order for the command to succeed:

```
CLI (network-admin@switch) >  vlan-modify id 1 vxlan 123
!!!! Vlan has vxlan, but there are no ports configured in
vxlan-loopback-trunk. vxlan forwarding may not function correctly. !!!!
```

The printed message means that VLAN 1 was successfully associated to a VXLAN ID. As explained in the previous section, in order for VXLAN forwarding to be fully functional, you also need to make sure that ports are added to the `vxlan-loopback-trunk`.

Once VLAN 1 is mapped to a VNI, you cannot change the fabric VLAN back to 1, as that would create a conflict:

```
CLI (network-admin@switch) > fabric-local-modify vlan 1
fabric-local-modify: vlan 1 has VXLAN 123, unconfigure and retry
```

However, you may at some point decide to unmap VLAN 1 from the associated VNI (in other words, “unconfigure” it as suggested in the message above), by using the command:

```
CLI (network-admin@switch) > vlan-modify id 1 vxlan 0
```

As explained in the command help, as shown below, the VNI value 0 is used to *unconfigure*, that is, to remove a VLAN/VNI mapping:

```
CLI (network-admin@switch) > vlan-modify id 1 vxlan
```

`vlan-modify`	modify a VLAN
`id 0..4095`	VLAN ID
between 1 and 4 of the following options:	
`description description-string`	VLAN description
`vxlan 0..16777215`	VXLAN identifier for tunnel, **value 0 indicates unconfigure vxlan**
`replicators vtep-group name | none`	Replicator Group
`vnet vnet name`	VNET for this VLAN
`public-vlan 0..4095`	Public VLAN for VNET VLAN

Checking VXLAN Recirculation's L2 and L3 Entries

As discussed earlier, when implementing RIOT at least a recirculation pass is used. That requires that Layer 2 and Layer 3 entries be programmed appropriately to point to the loopback trunk.

With the `l2-table-show` command it's possible to verify that a specific VNI-mapped VLAN is configured to point to the VXLAN loopback trunk to forward and then encapsulate the upstream traffic at the ingress VTEP.

```
CLI (network-admin@switch) > l2-table-show vlan 200

mac:                  00:00:5e:00:01:0a
vlan:                 200
vxlan                 10000
ip:                   2.2.2.2
ports:                69
state:                active,static,vxlan-loopback,router
hostname:             Spine1
peer-intf:            host-1
peer-state:
peer-owner-state:
status:
migrate:
```

When VTEP HA is implemented, the same command can be used to show that the VLAN is configured with VRRP and that it points to the VLAN loopback trunk. For example:

```
CLI (network-admin@Spine1) > l2-table-show vlan 200

mac:                    00:00:5e:b9:01:b0
vlan:                   200
vxlan                   10000
ip:                     2.2.2.2
ports:                  69
state:                  active,static,vxlan-loopback,router,vrrp
hostname:               Spine1
peer-intf:                          host-1
peer-state:             active,vrrp,vxlan-loopback
peer-owner-state:
status:
migrate:
```

Similarly, in order to decapsulate and router the VXLAN traffic originated from a source VTEP, at the destination VTEP at least two passes are required. Therefore, a Layer 3 entry is programmed to point to the `vxlan-loopback-trunk`.

The `l3-table-show` command can be used to verify that the hardware state is properly set with the `vxlan-loopback` flag:

```
CLI (network-admin@Spine4) > l3-table-show ip 3.3.3.2 format all
```

```
mac:              00:00:c0:00:07:75
ip:               3.3.3.2
vlan:             200
public-vlan:      200
vxlan:            10000
rt-if:            eth5.200
state:            active,vxlan-loopback
egress-id:        100030
create-time:      16:46:20
last-seen:        17:25:09
hit:              22
tunnel:           Spine1_Spine4
```

Showing VXLAN Trunk Replication Counters

It is possible to display statistics (incoming and outgoing bytes, packets, etc.) for VXLAN connections (simply called tunnels) that also include the amount of head-end replication (HER) applied to the VXLAN traffic.

Currently, NetVisor OS uses `vxlan-loopback-trunk` port stats counters for HER replication statistics; therefore, they are cumulative not granular, but may provide a useful tool to measure the amount of `vxlan-loopback-trunk` bandwidth required in real world conditions.

The `tunnel-stats-show` command can be used to output all the statistics including the number of head-end replicated packets and the number of head-end replicated bytes. It can also be executed periodically with the `show-interval seconds-interval` option as shown in the example below:

```
CLI (network-admin@Spine1) > tunnel-stats-show show-interval 1 format all
```

```
time     tunnel-name sw-inpkts sw-opkts sw-idrops sw-odrops counter                HER-pkts HER-bytes
-------- ----------- --------- -------- --------- --------- -------               -------- ---------
11:28:03 pip         8.41K     9        3.44K     0         0                       4.37M    0
time     tunnel-name sw-inpkts sw-opkts sw-idrops sw-odrops counter                HER-pkts HER-bytes
-------- ----------- --------- -------- --------- --------- -------               -------- ---------
11:28:04 pip         0         0        1         0         0                         1        0
time     tunnel-name sw-inpkts sw-opkts sw-idrops sw-odrops counter                HER-pkts HER-bytes
-------- ----------- --------- -------- --------- --------- -------               -------- ---------
11:28:05 pip         1         0        1         0         0                         1        0
time     tunnel-name sw-inpkts sw-opkts sw-idrops sw-odrops counter                HER-pkts HER-bytes
-------- ----------- --------- -------- --------- --------- -------               -------- ---------
11:28:06 pip         0         0        1         0         0                         1        0
time     tunnel-name sw-inpkts sw-opkts sw-idrops sw-odrops counter   • • •        HER-pkts HER-bytes
-------- ----------- --------- -------- --------- --------- -------               -------- ---------
11:28:07 pip         0         0        2         0         0                       57.1K    0
time     tunnel-name sw-inpkts sw-opkts sw-idrops sw-odrops counter                HER-pkts HER-bytes
-------- ----------- --------- -------- --------- --------- -------               -------- ---------
11:28:08 pip         0         0        1         0         0                       76.2K    0
time     tunnel-name sw-inpkts sw-opkts sw-idrops sw-odrops counter                HER-pkts HER-bytes
-------- ----------- --------- -------- --------- --------- -------               -------- ---------
11:28:10 pip         0         0        1         0         0                       75.6K    0
time     tunnel-name sw-inpkts sw-opkts sw-idrops sw-odrops counter                HER-pkts HER-bytes
-------- ----------- --------- -------- --------- --------- -------               -------- ---------
11:28:11 pip         0         0        1         0         0                       76.6K    0
time     tunnel-name sw-inpkts sw-opkts sw-idrops sw-odrops counter                HER-pkts HER-bytes
-------- ----------- --------- -------- --------- --------- -------               -------- ---------
11:28:12 pip         0         0        1         0         0                       67.0K    0
```

Figure 8-14: Wide Multi-Column Output of VXLAN Head End Replication Counters

Displaying ECMP Load Balancing Info for VXLAN

As discussed in the earlier sections, equal-cost multi-path (ECMP) load balancing defines a routing strategy in which next-hop packet forwarding to a single destination can occur over multiple paths.

In case of VTEP interconnections, it is possible to verify that multiple next hops are available with the `tunnel-show` command.

It is also possible to verify that ECMP is applied in hardware (provided multiple next hops exist) and to display the selection of the next hops by checking the RIB table with the `vrouter-rib-routes-show` command.

For example, the user has configured a tunnel called `leaf1toleaf2` with a local IP address `22.3.11.1` and a remote IP address `32.4.11.1`. The network design provides *two next hops* to route the VXLAN traffic from the local VTEP to the remote one: `192.178.0.6` and `192.178.0.2`.

The `tunnel-show` and `vrouter-rib-routes-show` commands can be used to display the tunnel's specific next hops as well as the route info for the remote end point (`32.4.11.0/24`) in the RIB table, as shown below:

```
CLI (network-admin@leaf1) > tunnel-show name leaf1toleaf2 ecmp-nexthops
layout vertical

scope:              local
name:               leaf1toleaf2
type:               vxlan
vrouter-name:        scorpius-vxlan-01
local-ip:           22.3.11.1
remote-ip:          32.4.11.1
router-if:          eth0.3
next-hop:           192.178.0.6
next-hop-mac:        66:0e:94:f7:31:f0
nexthop-vlan:       4091
active:             yes
state:              ok
bfd:                disabled
bfd-state:           not-replicator-vtep
route-info:          17.60.11.0/30
ports:              49
auto-tunnel:        static
scope:              local
name:               leaf1toleaf2
type:               vxlan
vrouter-name:        scorpius-vxlan-01
local-ip:           17.60.13.1
remote-ip:          17.60.11.1
router-if:          eth0.3
next-hop:           192.178.0.2
next-hop-mac:        66:0e:94:b7:95:c3
nexthop-vlan:       4092
active:             yes
```

```
state:              ok
bfd:                disabled
bfd-state:           not-replicator-vtep
route-info:          17.60.11.0/30
ports:              272
auto-tunnel:         static

CLI (network-admin@leaf1) > vrouter-rib-routes-show ip 32.4.11.0

vrid ip           prelen number-of-nexthops nexthop      flags       vlan  intf_ip      intf-id
---- ----------   ------ ------------------ -----------  ----------  ----- -----------  -------
0    17.60.11.0  30     2                  192.178.0.6 ECMP,in-hw 4091  192.178.0.5  1
0    17.60.11.0  30     2                  192.178.0.2 ECMP,in-hw 4092  192.178.0.1  0
```

Configuring VTEP Objects with Automatic Fabric Connections

Based on the command sequence in the earlier section, *Configuring the Overlay: VTEP Interconnections and VNIs*, configuring a series of unidirectional end-point interconnections (even in a simple triangular topology) requires the network administrator to manually issue a long list of explicit (and hence tedious) `tunnel-create` commands.

Therefore, starting from release 2.6.0, Arista has introduced an additional user-friendly operational simplification element called a *logical VTEP configuration object* that leverages the intelligence of the fabric's distributed control plane to automate the aforementioned tedious connection setup process.

With this configuration paradigm, instead of having to create individual unidirectional connections between the end points (i.e., the `tunnel-create` model), a user can simply create a single VTEP object per endpoint with the `vtep-create` command and then map the required VXLAN identifiers to it.

This in turn triggers the *automatic creation* of all the required VXLAN connections in both directions between endpoints, yielding a *significant amount of configuration and time savings*.

An example of VTEP object configuration syntax is displayed below:

```
CLI (network-admin@switch) > vtep-create name <vtep-object-name>  vrouter-
name <vrouter name> ip <primary IP> virtual-ip <vip> [location <switch
name>]

CLI (network-admin@switch) > vtep-vxlan-add name <vtep-object-name> vxlan
<vnid1>

CLI (network-admin@switch) > vtep-vxlan-add name <vtep-object-name> vxlan
<vnid2>

CLI (network-admin@switch) > vtep-vxlan-add name <vtep-object-name> vxlan
<vnid3>
```

And so on...

VTEP objects can also be displayed or deleted with the `vtep-show` and `vtep-delete` commands, respectively.

Auto-VXLAN

Starting from NetVisor OS release 6.0.0 the `auto-vxlan` parameter is supported as an option of the VLAN creation process and can be used to automate the `vtep-vxlan-add` operations.

The `auto-vxlan` parameter can be used either in combination with an *explicit VNI value* or implicitly *without specifying it*. In both cases any created VLAN/VNI mapping is automatically added to all the existing VTEP objects. Additionally, in the latter case, the VNI value is automatically picked and assigned out of a *predefined range*.

Therefore, to automatically assign a certain *user-defined* VLAN/VNI mapping to all VTEP objects in the fabric, you can use the following command:

```
CLI (network-admin@switch) > vlan-create id 1234 scope fabric vxlan 5001234
auto-vxlan
```

If you want the software to automatically pick and assign the VNI too, you can simply enter the shorter command below, which requires the `scope` to be set to `fabric` (as the VNI assignment can only be fabric-wide):

```
CLI (network-admin@switch) > vlan-create id 1234 scope fabric auto-vxlan
```

The above `vlan-create` commands with `auto-vxlan` automatically configure the following (assuming there are 3 VTEPs in the fabric, VTEP1 through VTEP3):

```
CLI (network-admin@switch) > vtep-vxlan-add name VTEP1 vxlan 5001234

CLI (network-admin@switch) > vtep-vxlan-add name VTEP2 vxlan 5001234

CLI (network-admin@switch) > vtep-vxlan-add name VTEP3 vxlan 5001234
```

where 5001234 is the VNI value either entered explicitly by the user or automatically selected from the `vtep-auto-vxlan` range.

The range used for automatic VNI assignment can be controlled and modified, if needed, with the `vtep-auto-vxlan-show` and `vtep-auto-vxlan-modify` commands.

> **Note:** You can use the `vlan-create` command with `auto-vxlan` in `local` and `cluster` scope too *provided the VNI is explicitly specified*. If it's not, an error message is printed:

```
CLI (network-admin@switch) > vlan-create id 1000 scope cluster auto-vxlan
vlan-create: auto-vxlan only allowed for fabric scope

CLI (network-admin@switch) > vlan-create id 1000 scope local auto-vxlan
vlan-create: auto-vxlan only allowed for fabric scope
```

Also starting from NetVisor OS release 6.0.0 the `vlan-show` command output displays when `auto-vxlan` is used:

```
CLI (network-admin@switch) > vlan-show id 1234 format id, type, vxlan,
auto-vxlan, scope, description, active, stats, ports

id    type    vxlan    auto-vxlan scope description active stats ports
---- ------ ------- ---------- ----- ----------- ------ ----- -------------
1234 public 5001234  yes        local  vlan-1234    yes     no     0-48,50-52
```

If an associated VNI value needs to be changed after a VLAN has been created using the `auto-vxlan` option, the `vlan-modify` command can be used for that purpose like so:

```
CLI (network-admin@switch) > vlan-modify id 1234 vxlan 4001234
```

For VLANs created with the `auto-vxlan` option, the new VNI will replace the old VNI in the VTEP connections.

Starting from NetVisor OS release 6.0.0, when a VLAN is deleted, if it was previously created with the `auto-vxlan` option, the VLAN/VNI mapping will be removed from all the VTEP objects automatically.

In the example of ***Figure 8-8*** above, the following commands can be used to create three VTEP objects on three nodes:

```
CLI (network-admin@Switch1) > vtep-create name VTEP1 vrouter-name vr1 ip
1.1.1.1 location Switch1

CLI (network-admin@Switch2) > vtep-create name VTEP2 vrouter-name vr2 ip
2.2.2.2 location Switch2

CLI (network-admin@Switch3) > vtep-create name VTEP3 vrouter-name vr3 ip
3.3.3.3 location Switch3

CLI (network-admin@Switch1) > vlan-create id 10 scope fabric vxlan 10010
auto-vxlan
```

The command sequence above automatically creates bidirectional VXLAN connections between the three nodes and also automatically adds VNI 10010 to all of them, thanks to the `auto-vxlan` keyword, as shown in the following outputs:

```
CLI (network-admin@Switch1) > tunnel-vxlan-show

switch    name                        vxlan
--------- --------------------------- -----
Switch1   auto-tunnel-1.1.1.1_2.2.2.2 10010
Switch1   auto-tunnel-1.1.1.1_3.3.3.3 10010
Switch2   auto-tunnel-2.2.2.2_1.1.1.1 10010
Switch2   auto-tunnel-2.2.2.2_3.3.3.3 10010
Switch3   auto-tunnel-3.3.3.3_1.1.1.1 10010
Switch3   auto-tunnel-3.3.3.3_2.2.2.2 10010

CLI (network-admin@Switch1) > vtep-vxlan-show

name  vxlan
----- -----
VTEP1 10010
VTEP2 10010
VTEP3 10010
```

Inter-fabric VTEP Connections

Starting from NetVisor OS release 6.0.0, the automatic VTEP object creation functionality is further extended to include an additional parameter that can be used to connect a VTEP in one fabric instance to another one in a different fabric instance. In other words, this new capability enables *inter-fabric VTEP connections*.

As part of the VTEP object creation, a new reserved location keyword, called `host-external`, is introduced to enable inter-fabric connectivity, as shown in the command output below:

```
CLI (network-admin@switch1) > vtep-create name ext-VTEP location
     host-external
```

```
switch1
switch2
```

Specifying the `host-external` keyword in the command identifies an external VTEP, whose corresponding *unique IP address* should then be specified as part of the command like so:

```
CLI (network-admin@switch1) > vtep-create name FAB2-VTEP1 location host-external ip 20.1.1.1 description "RemoteFabricName:FAB2, SwitchName: FAB2-switch2"
```

> **Note:** The `virtual-ip` option is not supported with external VTEPs (it's meant to be used only with regular VTEPs). For redundancy, the ip option can simply point to the VRRP VIP (virtual IP) of the external VTEP.

Also note that a good practice (which is not enforced in the CLI but is recommended) is to name external VTEPs in a user-friendly and precise way, for example, with the following naming structure: `<fabric-name>-vtep-name`, or with any other suitable structure that allows the naming to *unequivocally and uniquely identify a specific VTEP*.

As shown in the example above, a `description` field is also added to the `vtep-create` command, which can be used to store additional details about the precise remote VTEP location including switch name, fabric name, switch vendor, etc.

External VTEPs can be displayed along with regular VTEPs in the `vtep-show` command like so:

```
CLI (network-admin@switch1) > vtep-show format all name FAB2-VTEP1

scope  name       location      vrouter-name ip         virtual-ip description
------ ---------- ------------- ------------ ---------- ---------- -----------
fabric VTEP1      switch1       vr1          190.11.1.2 190.11.1.1
fabric VTEP2      switch2       vr2          190.11.1.3 190.11.1.1
fabric VTEP3      switch3       vr3          190.12.1.3 190.12.1.1
fabric VTEP4      switch4       vr4          190.12.1.2 190.12.1.1
fabric FAB2-VTEP1 host-external              20.1.1.1
RemoteFabricName: FAB2, SwitchName: FAB2-switch2
```

Hub-and-spoke VXLAN Connectivity

Starting from NetVisor OS release 6.0.1, a new keyword is introduced to support a *hub-and-spoke* VXLAN connectivity model in which a hub node is connected to all the spokes bidirectionally, but the spokes are not connected to each other (see an example in ***Figure 8-9*** above).

> **Note**: This feature is supported only on Dell, Edgecore, and Freedom series switches.

The new `isolated` keyword is used along with manual VNI assignment in the `vtep-vxlan-add` command. Therefore, it is applied on a *per-VNI per-VTEP mapping basis*.

The i`solated` keyword is used to configure the spoke nodes, whereas the hub nodes are configured normally. In the example of ***Figure 8-9*** above, the following sequence can set up one hub and two spoke/isolated nodes for VNI 10030:

```
CLI (network-admin@Switch1) > vlan-create id 30 scope fabric vxlan 10030

CLI (network-admin@Switch1) > vtep-vxlan-add name VTEP1 vxlan 10030
CLI (network-admin@Switch2) > vtep-vxlan-add name VTEP2 vxlan 10030
isolated
CLI (network-admin@Switch3) > vtep-vxlan-add name VTEP3 vxlan 10030
isolated
```

Due to the hub-and-spoke nature of the configuration, bidirectional VXLAN connections are only created (automatically) between the hub node (Switch1) and the spokes (Switch2 and Switch3), but not between the spokes as shown below:

```
CLI (network-admin@Switch1) > tunnel-vxlan-show

Switch   name                          vxlan
--------  ----------------------------  -----
Switch1   auto-tunnel-1.1.1.1_2.2.2.2  10030
Switch1   auto-tunnel-1.1.1.1_3.3.3.3  10030
Switch2   auto-tunnel-2.2.2.2_1.1.1.1  10030
Switch3   auto-tunnel-3.3.3.3_1.1.1.1 10030
```

The default configuration mode is `not-isolated` (in other words, the `isolated` keyword is not assigned by default to any VNI mappings). Also, `not-isolated` is an *optional* keyword so it can be omitted. Note that, if users need to change a VNI from isolated to not-isolated/normal mode, they need to first execute the `vtep-vxlan-remove` command in order to delete the VNI assignment and subsequently they can add it back with the default/normal mode.

The per-VNI isolated configuration can be checked with the `vtep-vxlan-show` command as shown below:

```
CLI (network-admin@switch) > vtep-vxlan-show
name  vxlan  isolated
----  -----  --------
vt1   10021 no
vt1   10020 yes
vt1   10023 yes
vt1   10025 no
vt1   10022 no
vt1   10024 yes
vt2   10021 no
vt2   10020 yes
vt2   10023 no
vt2   10025 no
vt2   10022 yes
vt2   10024 yes
vt3   10021 no
vt3   10020 yes
vt3   10023 yes
vt3   10025 yes
vt3   10022 yes
vt3  10024 no
```

Disabling VXLAN Termination

It is possible, for security purposes, to disable VXLAN termination on certain ports that are not supposed to source VXLAN-encapsulated traffic.

This prevents any malicious host from generating VXLAN encapsulated packets that would normally be subject to (unwanted) VXLAN tunnel termination and subsequent forwarding. For example, the following command disables the termination on port 35:

```
CLI (network-admin@switch) > port-config-modify port 35 no-vxlan-
termination
```

It can be re-enabled, if necessary, like so:

```
CLI (network-admin@switch) > port-config-modify port 35 vxlan-termination
```

The default settings are:

- vNETs with `vlan-type private` rely on the VXLAN functionality to implement their private characteristics. Therefore, when a port is configured to be a managed port with `vlan-type private`, VXLAN termination is disabled by default.

- Underlay ports have VXLAN termination on by default and can use the `port-config-modify` command to disable VXLAN termination as deemed to enforce port level security.

Disabling MAC Address Learning

In certain fabric topologies, the MAC address learning function can be disabled in order to achieve better scalability, or because it is not strictly required (for example on point-to-point connections).

For these cases, starting from NetVisor OS release 6.0.0, it is possible to disable MAC address learning as part of the VXLAN configuration. For this purpose, a new configuration option is added, `mac-learning | no-mac-learning`, to the following commands:

- `tunnel-create`
- `tunnel-modify`
- `vtep-create`
- `vtep-modify`

In the above commands, `mac-learning` is the default setting. The `tunnel-show` and `vtep-show` commands are extended to display the learning state in a new column. For example:

```
CLI (network-admin@switch) > tunnel-create name tun1 scope local local-ip
1.1.1.1 remote-ip 1.1.1.2 vrouter-name vr1 no-mac-learning

CLI (network-admin@switch) > tunnel-show format name,local-ip,remote-
ip,mac-learning,

name  local-ip  remote-ip mac-learning
----- -------- ---------- ------------
tun1  1.1.1.1    1.1.1.2     off
```

Then to re-enable MAC address learning, you can use:

```
CLI (network-admin@switch) > tunnel-modify name tun1 mac-learning

CLI (network-admin@switch) > tunnel-show format name,local-ip,remote-
ip,mac-learning,

name  local-ip  remote-ip  mac-learning
----- -------- ---------- ------------
tun1  1.1.1.1   1.1.1.2     on
```

Similarly, in case of VTEP object creation to disable MAC address learning you can use:

```
CLI (network-admin@switch) > vtep-create name VTEP-01 location switch
vrouter-name vr1 ip 10.16.111.2 virtual-ip 10.16.111.1 no-mac-learning

CLI (network-admin@switch) > vtep-show

scope   name     location vrouter-name     ip          virtual-ip   mac-learning
------  -------  -------- ------------  -----------  -----------  ------------
fabric VTEP-01   switch     vr1         10.16.111.2  10.16.111.1    off
```

To re-enable it:

```
CLI (network-admin@switch) > vtep-modify name VTEP-01 mac-learning

CLI (network-admin@switch) > vtep-show

scope   name     location vrouter-name    ip           virtual-ip   mac-learning
------ -------  -------- ------------  -----------   -----------  ------------
fabric VTEP-01   switch     vr1         10.16.111.2   10.16.111.1   on
```

Note: Disabling MAC address learning at the tunnel or VTEP level means that the learning function is disabled for all of the associated VNIs.

Configuring Unicast Fabric VRFs with Anycast Gateway

The following commands are used for the configuration of VRF instances and of the associated VRF gateway (`vrf-gw` and `vrf-gw2`) IP addresses:

```
CLI (network-admin@switch) > vrf-create
```

`name name-string`	Specify a name for the VRF.
`vnet vnet-name`	Specify the name of the vNET to assign the VRF. If you only have a global vNET configured, omit this parameter.
`scope local|cluster|fabric`	Specify the scope for the VRF.
`vrf-gw ip-address`	Specify the gateway IP address.
`vrf-gw2 ip-address`	Specify the second gateway IP address.
`vrf-gw-ipv6 ip-address`	Specify the IPv6 gateway address.
`vrf-gw2-ipv6 ip-address`	Specify the second IPv6 gateway address.
`enable|disable`	Specify to enable or disable VRF routing.
`description` *`description-string`*	Specify a VRF description. The maximum number of allowed characters is 59.

```
CLI (network-admin@switch) > vrf-delete
```

`name name-string`	Specify VRF name that you want to delete.
`vnet vnet-name`	Specify the name of the vNET assigned to the VRF.

```
CLI (network-admin@switch) > vrf-modify
```

`name name-string`	Specify a name for the VRF.
`vnet vnet-name`	Specify the name of the vNET to assign the VRF.
`scope local|cluster|fabric`	Specify the scope for the VRF.
`vrf-gw ip-address`	Specify the gateway IP address.
`vrf-gw2 ip-address`	Specify the second gateway IP address.
`vrf-gw-ipv6 ip-address`	Specify the IPv6 gateway address.
`vrf-gw2-ipv6 ip-address`	Specify the second IPv6 gateway address.
`enable|disable`	Specify to enable or disable VRF routing.
`description` *`description-string`*	Specify a VRF description. The maximum number of allowed characters is 59.

```
CLI (network-admin@switch) > vrf-show
```

`name name-string`	Displays the name of the VRF.
`vnet vnet-name`	Displays the name of the vNET assigned the VRF.
`scope local|cluster|fabric`	Displays the scope of the VRF.
`vrf-gw ip-address`	Displays the gateway IP address.
`vrf-gw2 ip-address`	Displays the second gateway IP address.
`vrf-gw-ipv6 ip-address`	Displays the IPv6 gateway address.
`vrf-gw2-ipv6 ip-address`	Displays the second IPv6 gateway address.
`enable|disable`	Displays the status of VRF routing as enable or disable.
`description` *`description-string`*	Displays the VRF description.

The following commands are used for the configuration of subnet objects for the associated anycast gateway addresses and the associated VNIs:

```
CLI (network-admin@switch) > subnet-create
```

`name name-string`	Specify the name of the subnet.
`description` *`description-string`*	Specify the subnet description. The maximum number of allowed characters is 59.
`scope local|cluster|fabric`	Specify the scope for the VRF.
`vnet vnet-name`	Specify the name of the vNET to assign the VRF.
`vxlan vxlan-id`	Specify the VXLAN ID to assign to the subnet.
`vrf vrf name`	Specify the VRF to which the subnet belongs to.
`network ip-address`	Specify the IPv4 network IP address.
`netmask netmask`	Specify the netmask for the IPv4 address.
`anycast-gw-ip ip-address`	Specify the anycast gateway IPv4 address for the subnet.
`network6 ip-address`	Specify the IPv6 subnet network address.
`netmask6 netmask`	Specify the IPv6 subnet netmask address.
`anycast-gw-ip6 ip-address`	Specify the anycast gateway IPv6 address for the subnet.
`packet-relay enable|disable|none`	Enable or disable the packet relay.
`forward-proto dhcp`	Specify the protocol type to forward the packets.
`forward-ip ip-address`	Specify the forwarding IPv4 address.
`forward-ip2 ip-address`	Specify the second forwarding IPv4 address.
`forward-ip6 ip-address`	Specify the forwarding IPv6 address.
`forward-ip6-2 ip-address`	Specify the second forwarding IPv6 address.

`flood enable|disable|none`	Specify the flooding state of BUM traffic
`enable|disable`	Specify to enable/disable subnet routing.

`CLI (network-admin@switch) > subnet-delete`

`name name-string`	Specify the name of the subnet.
`vnet vnet-name`	Specify the name of the vNET to assign the VRF.
`vrf name-string`	Specify the VRF to assign the subnet.

`CLI (network-admin@switch) > subnet-modify`

`name name-string`	Specify the name of the subnet.
`description` *`description-string`*	Specify the subnet description. The maximum number of allowed characters is 59.
`vnet vnet-name`	Specify the name of the vNET to assign the VRF.
Specify one or more of the following options:	
`network ip-address`	Specify the IPv4 network IP address.
`netmask netmask`	Specify the netmask for the IPv4 address.
`anycast-gw-ip ip-address`	Specify the anycast gateway IPv4 address for the subnet.
`network6 ip-address`	Specify the IPv6 subnet network address.
`netmask6 netmask`	Specify the IPv6 subnet netmask address.
`anycast-gw-ip6 ip-address`	Specify the anycast gateway IPv6 address for the subnet.
`packet-relay enable|disable|none`	Enable or disable the packet relay.
`forward-proto dhcp`	Specify the protocol type to forward the packets.
`forward-ip ip-address`	Specify the forwarding IPv4 address.
`forward-ip2 ip-address`	Specify the second forwarding IPv4 address.
`forward-ip6 ip-address`	Specify the forwarding IPv6 address.
`forward-ip6-2 ip-address`	Specify the second forwarding IPv6 address.
`enable|disable`	Specify to enable/disable subnet routing.

`CLI (network-admin@switch) > subnet-show`

`name name-string`	Displays the name of the subnet.
`description` *`description-string`*	Displays the subnet description.
`scope local|cluster|fabric`	Displays the scope for the VRF.
`vnet vnet-name`	Displays the name of the vNET to assign the VRF.

`vlan vlan-id`	Displays the VLAN ID to assign to the subnet.
`vxlan vxlan-id`	Displays the VXLAN ID to assign to the subnet.
`vrf name-string`	Displays the VRF to assign the subnet.
`network ip-address`	Displays the network IPv4 address.
`netmask netmask`	Displays the netmask for the IPv4 address.
`anycast-gw-ip ip-address`	Displays the anycast gateway IPv4 address.
`network6 ip-address`	Displays the IPv6 subnet network address.
`netmask6 netmask`	Displays the IPv6 subnet netmask address.
`anycast-gw-ip6 ip-address`	Displays the anycast gateway IPv6 address for the subnet.
`linklocal ip-address`	Displays the IPv6 Link Local address.
`packet-relay enable|disable|none`	Displays the packet relay mode.
`forward-proto dhcp`	Displays the protocol type forwarding the packets.
`forward-ip ip-address`	Displays the forwarding IPv4 address.
`forward-ip2 ip-address`	Displays the second forwarding IPv4 address.
`forward-ip6 ip-address`	Displays the forwarding IPv6 address.
`forward-ip6-2 ip-address`	Displays the second forwarding IPv6 address.
`state init|ok|vxlan not found|vxlan deactivated|not-in-hw|vrouter interface exists`	Displays the subnet state.
`hw-state|no-hw-state`	Displays if there is a hardware state present.
`enable|disable`	Displays the state of the subnet routing.
`format fields-to-display`	Display output using a specific parameter. Use all to display all possible output.
`parsable-delim character`	Display output formatted for machine parsing using a specified delimiter.
`sort-asc`	Display output in ascending order.
`sort-desc`	Display output in descending order.
`show dups`	Display duplicate entries in the output.
`layout vertical|horizontal`	Format the output in a vertical or horizontal layout.
`show-interval seconds-interval`	Repeat the show command at a specified interval.
`show-headers| no-show-headers`	Display column headers or not.
`limit-output number`	Limit the display output to a specific number of entries.
`count-output`	Display the number of entries in the output. This is useful with vRouter show commands.

`count-only`	Displays the number of entries only.
`unscaled`	Display full values in the output instead of scaled approximate values.
`raw-int-values`	Display integer values instead of mapped values

The following commands allow you to modify and display anycast gateway information on the fabric:

```
CLI (network-admin@switch) > fabric-anycast-mac-show
```

`format fields-to-display`	Display output using a specific parameter. Use all to display all possible output.
`parsable-delim character`	Display output formatted for machine parsing using a specified delimiter.
`sort-asc`	Display output in ascending order.
`sort-desc`	Display output in descending order.
`show dups`	Display duplicate entries in the output.
`layout vertical|horizontal`	Format the output in a vertical or horizontal layout.
`show-interval seconds-interval`	Repeat the show command at a specified interval.
`show-headers| no-show-headers`	Display column headers or not.
`limit-output number`	Limit the display output to a specific number of entries.
`count-output`	Display the number of entries in the output. This is useful with vRouter show commands.
`count-only`	Displays the number of entries only.
`unscaled`	Display full values in the output instead of scaled approximate values.
`raw-int-values`	Display integer values instead of mapped values

```
CLI (network-admin@switch) > fabric-anycast-mac-modify
```

`mac mac-address`	`Modify the MAC address for anycast. The default MAC address is 64:0e:94:40:00:02.`

For example, the following `vrf-create` command can be used to create `VRF-1`:

```
CLI (network-admin@switch) > vrf-create name VRF-1 scope fabric
```

The `vrf-create` command can be issued to configure for instance 1000 VRFs on a single node, as shown in this output:

```
CLI (network-admin@switch) > vrf-show count-output
```

```
name    vnet scope  anycast-mac       vrf-gw vrf-gw2 active hw-router-mac     hw-vrid
------- ---- ------ ----------------- ------ ------- ------ ----------------- -------
VRF-1   0:0  fabric 64:0e:94:40:00:02  ::     ::      no    00:00:00:00:00:00 -1
VRF_2   0:0  fabric 64:0e:94:40:00:02  ::     ::      yes   66:0e:94:1b:59:47  1
VRF_3   0:0  fabric 64:0e:94:40:00:02  ::     ::      yes   66:0e:94:1b:6c:91  2
VRF_4   0:0  fabric 64:0e:94:40:00:02  ::     ::      yes   66:0e:94:1b:76:3d  3
VRF_5   0:0  fabric 64:0e:94:40:00:02  ::     ::      yes   66:0e:94:1b:7f:e2  4
VRF_6   0:0  fabric 64:0e:94:40:00:02  ::     ::      yes   66:0e:94:1b:89:87  5
...
VRF_999 0:0  fabric 64:0e:94:40:00:02  ::     ::      yes   66:0e:94:1b:aa:8a   999

Count: 999
```

Note: The newer ASICs can support an even higher count. The maximum number is ASIC *limited*.

The following commands can be used to create two subnet objects associated with `VRF-1` for East-West traffic segmentation:

```
CLI (network-admin@switch) > vlan-create id 12 vxlan 500012 scope fabric
ports none

CLI (network-admin@switch) > vlan-create id 13 vxlan 500013 scope fabric
ports none

CLI (network-admin@switch) > subnet-create name subnet-vxlan-500012 scope
fabric vxlan 500012 network 172.10.2.0/24 anycast-gw-ip 172.10.2.1 vrf VRF-
1

CLI (network-admin@switch) > subnet-create name subnet-vxlan-500013 scope
fabric vxlan 500013 network 172.10.3.0/24 anycast-gw-ip 172.10.3.1 vrf VRF-
1
```

Note: Starting from NetVisor OS release 6.0.0, the VNI assignment in `vlan-create` can be automated with the `auto-vxlan` keyword.

Finally, the following commands can be used to create two smaller subnets (/29) to provide North-South reach-ability in and out of `VRF-1` to/from VRF gateways `172.10.0.2` and `172.10.1.2`:

```
CLI (network-admin@switch) > vlan-create id 10 vxlan 500010 scope fabric
ports none

CLI (network-admin@switch) > vlan-create id 11 vxlan 500011 scope fabric
ports none

CLI (network-admin@switch) > subnet-create name subnet-vxlan-500010 scope
fabric vxlan 500010 network 172.10.0.0/29 anycast-gw-ip 172.10.0.1 vrf VRF-
1

CLI (network-admin@switch) > subnet-create name subnet-vxlan-500011 scope
fabric vxlan 500011 network 172.10.1.0/29 anycast-gw-ip 172.10.1.1 vrf VRF-
1
```

Note: The scope of the VRF and subnet objects typically would be `fabric`; however, to cater to specific needs and designs it is also possible to configure *local VRFs* and *subnets* in certain cases.

The next step is to configure the VRF gateways for `VRF-1`:

```
CLI (network-admin@switch) > switch <switch_list> vrf-modify name VRF-1
vrf-gw 172.10.0.2 vrf-gw2 172.10.1.2
```

Figure 8-15: Fabric VRFs with Border Leaves Connecting External Network

In this example it is assumed that the connectivity is implemented with static routing on the DC gateways (for example, third-party devices). To provide inbound reach-ability for VRF-1, the DC gateways must be provisioned with static routes for the VRF subnets receiving traffic from external networks, using the adjacent anycast gateway addresses as next-hop:

```
DC-Gateway-1# ip route vrf VRF-1 172.10.2.0/23 172.10.0.1
DC-Gateway-1# ip route vrf VRF-1 172.10.2.0/23 172.10.1.1
DC-Gateway-2# ip route vrf VRF-1 172.10.2.0/23 172.10.0.1
DC-Gateway-2# ip route vrf VRF-1 172.10.2.0/23 172.10.1.1
```

In addition, you can also leverage the `vrf-route-add` command to add static routes to specific VRF-enabled networks when required:

```
CLI (network-admin@switch) > vrf-route-add
vrf-route-add                              add vrf route
    one of the following vrf selectors:
```

```
    vrf-name name-string                  vrf name
    vrf-vnet vnet name                    VNET for the VRF
    the following route arguments:
    network ip-address                    IP address
    netmask netmask                       netmask
    gateway-ip ip-address                  gateway IP address
```

`vrf-route-remove` and `vrf-route-show` commands are also available.

Configuring the Anycast Gateway MAC Address as Source Address

Starting from NetVisor OS release 6.1.0, it is possible to select the Anycast Gateway MAC address as source address used for distributed routing of traffic across subnets. (When the Anycast Gateway function is not used or when this capability is not enabled, NetVisor OS only employs the router MAC address as source address.)

This feature is useful when some downstream device (for example, a firewall) gleans the source MAC address from a routed packet to use it for various reasons, such as for example in the response to the original packet.

The default source MAC address used for the Anycast Gateway function is the common router MAC address. During VRF creation or modification, by using the following command it is possible to specify to use the Anycast Gateway MAC address instead:

```
CLI (network-admin@switch) > vrf-create name vrf1 anycast-mac-for-forwarding

CLI (network-admin@switch) > switch * vrf-modify name vrf1 {anycast-mac-for-forwarding | no-anycast-mac-for-forwarding}
```

The default setting is `no-anycast-mac-for-forwarding`.

> **Note**: To modify this capability fabric-wide, use `switch * vrf-modify` as shown above, because `scope fabric` is not supported.

```
CLI (network-admin@switch) > vrf-show format name,anycast-mac,active,hw-router-mac,anycast-mac-for-forwarding

name   anycast-mac        active hw-router-mac      anycast-mac-for-forwarding
-----  ----------------- ------ ----------------- -------------------------
VRF-1 64:0e:94:40:00:02 yes    66:0e:94:b5:9e:c2 yes
VRF-4 64:0e:94:40:00:02 yes    66:0e:94:b5:d5:fb yes
VRF-2 64:0e:94:40:00:02 yes    66:0e:94:b5:be:8c yes
VRF-3 64:0e:94:40:00:02 yes    66:0e:94:b5:6c:97 yes
```

You can verify that the vFlow entry is properly installed with the command:

```
CLI (network-admin@switch) > vflow-show format name,src-mac,action | grep Anycast

name                           src-mac           action
-----------------------------  ----------------  --------------------
Anycast-Src-Miss-Cancel-ToCpu  64:0e:94:40:00:02  cancel-switch-to-cpu
```

Configuring Virtual Service Groups

Note: This feature is supported only on Dell, Edgecore, and Freedom series switches.

Starting from NetVisor OS release 6.0.1 it is possible to import/export subnet prefixes between Unicast Fabric VRFs. This operation is sometimes informally referred to as 'prefix leaking'.

To implement this functionality a new innovative entity called *Virtual Service Group* (vSG) is introduced: it's an abstraction that defines in an intuitive and user-friendly way the policies by which prefixes are shared.

Note: As of NetVisor OS release 6.0.1, this functionality applies only to Unicast Fabric VRFs and not to vRouter-based VRFs.

First a vSG is created with the `vsg-create` command, specifying a name and optionally (but conveniently) a description string:

```
CLI (network-admin@switch) > vsg-create name <vsg-name> [description
<description>]
```

Similarly, an existing vSG can be deleted or its description can be modified with the `vsg-delete` and `vsg-modify` commands, respectively.

A newly created vSG is a grouping that is meant to contain a list of VRFs, which can be added individually with the command:

```
CLI (network-admin@switch) > vsg-vrf-add name <vsg-name> vrf <name> [type
promiscuous | isolated] [vnet]
```

The `vsg-vrf-add` command enables the user to optionally specify the *type of prefix sharing* to apply to each VRF as well as an associated vNET. The `promiscuous` keyword (i.e., the *default* option) identifies VRFs whose prefixes can be freely shared with any other VRF. Instead, the `isolated` keyword identifies VRFs whose prefixes can only be shared with VRFs marked as promiscuous. So when the type is not specified, it defaults to promiscuous.

Analogously to the `vsg-vrf-add` command, VRF removal is performed with the `vsg-vrf-remove` command.

The last vSG configuration step is to identify which subnet(s) and/or prefix(es) to share for each VRF with the command:

```
CLI (network-admin@switch) > vsg-network-add name <vsg-name> vrf <vrf-name>
network <prefix/mask>|subnet <name> [vnet]
```

If all (current and future) subnets within a VRF have to be shared, you can use the handy shortcut: `subnet all`.

Once added, subnets/prefixes can be removed with the corresponding `vsg-network-remove` command.

The sharing of the added prefixes/subnets between VRFs follows the rules dictated by the respective

promiscuous/isolated markings, which is a very simplistic configuration model but can also be rather powerful.

Going through a few examples of use cases is the best way to describe the flexibility of such configuration model, as discussed in the following.

Example 1: Full-mesh Connectivity Model

In this scenario there are for example two tenants who share a common service. The three network entities are associated to three VRFs respectively: TENANT-GREEN, TENANT-GREY and SERVICE-AMBER. All three VRFs need to share at least one subnet with the others for proper connectivity.

This is an any-to-any communication model, in which you can use the promiscuous keyword to (explicitly) state that sharing is allowed. Then you can specify which subnets are shared. For example, TENANT-GREEN and TENANT-GREY share one of their subnets with SERVICE-AMBER like so:

```
CLI (network-admin@switch) > vsg-create name GROUP-A description ANY-TO-
ANY-SHARING
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-A vrf TENANT-GREY
type promiscuous
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-A vrf TENANT-GREEN
type promiscuous
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-A vrf SERVICE-AMBER
type promiscuous
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-A vrf TENANT-
GREY subnet SUBNET-11
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-A vrf TENANT-
GREEN subnet SUBNET-21
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-A vrf SERVICE-
AMBER subnet SUBNET-1
```

SUBNET-11 corresponds to 10.0.11.0/24, SUBNET-21 to 10.0.21.0/24 and SUBNET-1 to 10.1.1.0/24.

```
CLI (network-admin@switch) > vsg-show
switch name    description
------ ------- ------------------
switch GROUP-A ANY-TO-ANY-SHARING

CLI (network-admin@switch) > vsg-vrf-show
switch vsg-name vnet vrf           type
------ -------- ---- ------------ -----------
switch GROUP-A  0:0  TENANT-GREY  promiscuous
switch GROUP-A  0:0  TENANT-GREEN promiscuous
switch GROUP-A  0:0  SERVICE-AMBER promiscuous

CLI (network-admin@switch) > vsg-network-show
switch vsg-name vrf            vnet subnet    network         network_state
------ -------- ------------ ---- --------- --------------- ------------
switch GROUP-A  TENANT-GREY  0:0  SUBNET-11 10.0.11.0/24    ok
switch GROUP-A  TENANT-GREEN 0:0  SUBNET-21 10.0.21.0/24    ok
switch GROUP-A  SERVICE-AMBER 0:0 SUBNET-1  10.1.1.0/24     ok
```

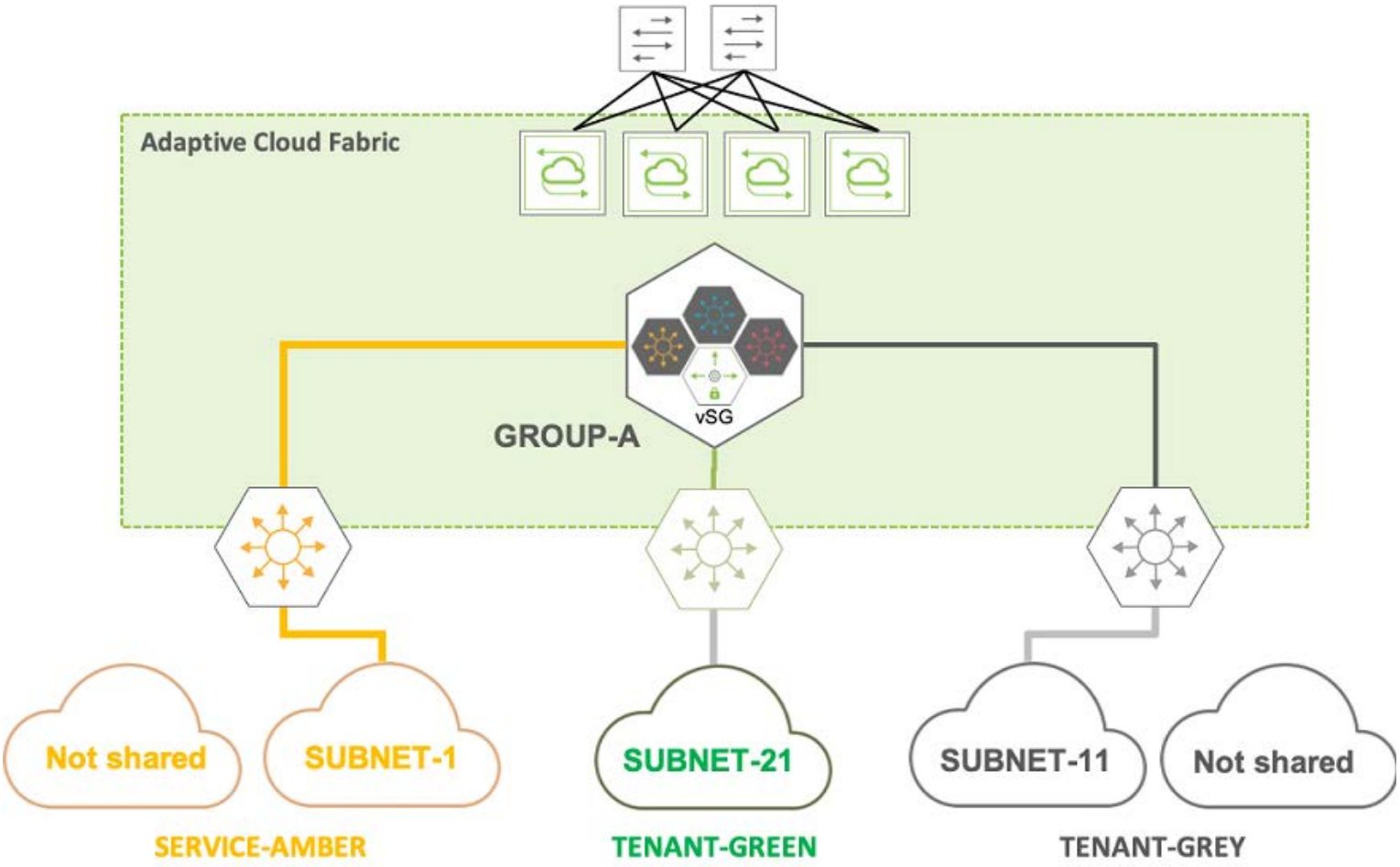


Figure 8-16 – Virtual Service Group Example

Example 2: Hub-and-spoke Connectivity Model

In this scenario, the 'hub' VRF is the one that needs to share its subnet(s) with all other vSG members without distinction. In other words, it's the *promiscuous VRF*. A good example would be a common shared service. Instead, *isolated VRFs (spokes)* don't need to share subnets between each other, they only need to share them with the promiscuous VRF. A common case would be different tenants (say, GREEN and GREY) that require routing segregation without any mutual prefix sharing, like so:

```
CLI (network-admin@switch) > vsg-create name GROUP-B description HUB-AND-
SPOKE-SHARING
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-B vrf SERVICE-BLUE
type promiscuous
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-B vrf TENANT-GREY
type isolated
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-B vrf TENANT-GREEN
type isolated
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-B vrf TENANT-
GREY subnet SUBNET-12
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-B vrf TENANT-
```

```
GREEN subnet SUBNET-22
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-B vrf SERVICE-
BLUE subnet SUBNET-2
```

Example 3: Hierarchical Connectivity Model

This example is analogous to #1 above but applies the vSG grouping function to separate service VRFs (say, AMBER, BLUE and CRIMSON). This allows the user to create a hierarchy where the vSG corresponds to the superset (i.e., the union) of multiple VRFs that remain separate. The advantage of this approach compared to having a single merged 'fat' VRF is that keeping service VRFs separate can enable better granularity in assigning subnets/prefixes, policies or different vNETs:

```
CLI (network-admin@switch) > vsg-create name GROUP-C description VRF-UNION
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-C vrf SERVICE-BLUE
type promiscuous
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-C vrf SERVICE-AMBER
type promiscuous
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-C vrf SERVICE-
CRIMSON type promiscuous
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-C vrf SERVICE-
AMBER subnet SUBNET-1
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-C vrf SERVICE-
BLUE subnet SUBNET-2
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-C vrf SERVICE-
CRIMSON subnet SUBNET-3
```

The `vsg-network-add` enables full granularity of prefix sharing depending on the specific network requirements. If all current and future subnets have to be shared, then you can use these shortcuts instead:

```
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-C vrf SERVICE-
AMBER subnet all
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-C vrf SERVICE-
BLUE subnet all
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-C vrf SERVICE-
CRIMSON subnet all
```

Example 4: Unidirectional Connectivity Model

This example is less common because it is more typical for traffic to flow bidirectionally and hence routing to be applied in both directions. However, it is possible that in some scenarios only unidirectional communication is required, for example, in certain service chaining cases.
To support this use case, you can configure certain VRFs to act only as *traffic sources* (hence not sharing subnets) and other VRFs to act only as *traffic receivers* (hence sharing subnets). In other words, in order to accommodate the unidirectional flow scenario, only partial subnet sharing can be configured like so:

```
CLI (network-admin@switch) > vsg-create name GROUP-E description
UNIDIRECTIONAL-SHARING
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-E vrf RECEIVE type
typepromiscuous
CLI (network-admin@switch) > vsg-vrf-add vsg-name GROUP-E vrf SOURCE
```

```
 promiscuous
CLI (network-admin@switch) > vsg-network-add vsg-name GROUP-E vrf RECEIVE
subnet SUBNET-RCV
```

Furthermore, sharing of subsets is possible too by using the `network` keyword in `vsg-network-add` (say, there are two subnets, 100.1.1.0/24 and 101.1.1.0/24, but only four prefixes have to be shared respectively: 100.1.1.0/30 and 101.1.1.0/30):

```
CLI (network-admin@switch) > vsg-network-add vsg-name vsg1 vrf vrf1 network
100.1.1.0/30
CLI (network-admin@switch) > vsg-network-add vsg-name vsg1 vrf vrf2 network
101.1.1.0/30

CLI (network-admin@switch) > vsg-network-show

Switch vsg-name vrf  vnet subnet network       network_state
------ -------- ---- ---- ------ ------------ -------------
switch vsg1     vrf1 0:0          100.1.1.0/30 ok
switch vsg1     vrf2 0:0          101.1.1.0/30 ok
```

Configuring vSG Route Sharing

In certain configurations it can be convenient to implement the automatic sharing of one or more routes in accordance with the vSG rules. For example, a firewall (or a DC gateway or another similar device) may be a common next hop for all the end devices across the entire vSG. Such device (or devices) is typically deployed to enable shared service(s): in the case of a firewall, it supports external connectivity, security policies, etc.

In these configurations with shared services, it can be convenient to add routes directly to the vSG so that they get automatically shared to *all* the members based on the vSG rules chosen by the user.

Starting from NetVisor OS release 6.1.0 a new capability is introduced to add routes to the vSG, which can save significant configuration overhead especially in large vSG configurations.

For example, let's consider a scenario in which there are two isolated VRFs (`blue` and `red`) that need to communicate through a firewall. (For the sake of simplicity, the example only uses two VRFs but it can be scaled to N VRFs making the new automation even more advantageous.)

The firewall is in a subnet associated with a promiscuous VRF called `transit.` `blue` and `red` are isolated VRFs that share one subnet each (10.1.1.0/24 and 20.1.1.0/24) with `transit`, which uses a third subnet. In particular, the firewall is the actual common service to be shared in such subnet and uses the address 30.1.1.2.

This is the corresponding configuration of the vSG:

```
CLI (network-admin@switch) > vrf-create name blue
CLI (network-admin@switch) > vrf-create name red
CLI (network-admin@switch) > vrf-create name transit vrf-gw 30.1.1.2

CLI (network-admin@switch) > subnet-create name vlan_10 vrf blue vlan 10
network 10.1.1.0/24 anycast-gw-ip 10.1.1.1
CLI (network-admin@switch) > subnet-create name vlan_20 vrf red vlan 20
network 20.1.1.0/24 anycast-gw-ip 20.1.1.1
CLI (network-admin@switch) > subnet-create name vlan_30 vrf transit vlan 30
network 30.1.1.0/29 anycast-gw-ip 30.1.1.1

CLI (network-admin@switch) > vsg-create name vSG-1 description "Firewall
Access"
CLI (network-admin@switch) > vsg-vrf-add name vSG-1 vrf blue type isolated
CLI (network-admin@switch) > vsg-vrf-add name vSG-1 vrf red type isolated
CLI (network-admin@switch) > vsg-vrf-add name vSG-1 vrf transit type
promiscuous

CLI (network-admin@switch) > vsg-network-add name vSG-1 vrf blue subnet
vlan_10
CLI (network-admin@switch) > vsg-network-add name vSG-1 vrf red subnet
vlan_20
CLI (network-admin@switch) > vsg-network-add name vSG-1 vrf transit network
30.1.1.2/32
```

In this example, the need is to automatically share across all the VRFs in the vSG the default gateway route (0.0.0.0/0) pointing to the firewall's IP address (30.1.1.2/32).

That could be achieved by manually configuring in both blue and red the same route by using the `vrf-route-add` command twice.

Alternatively and more conveniently, starting from release 6.1.0 a single command, `vsg-route-add`, can be used to share the default route across *all* of the VRFs that are part of a vSG, like so (by virtue of the promiscuous nature of `transit`):

```
CLI (network-admin@switch) > vsg-route-add vsg-name vSG-1 vrf transit
network 0.0.0.0/0 gateway-ip 30.1.1.2
```

Note: The `gateway-ip` address (i.e., the next hop) in the vSG route must be shared from the promiscuous VRF prior to the `vsg-route-add` command. Hence the sequence of commands is in the order displayed above with the network highlighted in bold. If by mistake this requirement is not met, the `vsg-route-add` command will produce an error message: `network 30.1.1.2 not leaked from vrf transit`

Note: The network in the vSG route should not overlap with local subnets or shared prefixes. For example, the command: `vsg-route-add name vsg1 vrf transit network 10.1.1.2/30 gateway-ip 30.1.1.2` produces the error message: `vsg-route-add: !! Overlapping subnet found for network 10.1.1.2 in vrf blue` because 10.1.1.2 overlaps with the local subnet of VRF `blue`.

To verify that a vSG route was correctly added, you can use the `vsg-route-show` command:

```
CLI (network-admin@switch) > vsg-route-show
switch name vrf      vnet network   gateway-ip
------ ---- ------- ---- --------- ----------
switch vsg1 transit 0:0  0.0.0.0/0 30.1.1.2
```

Moreover, to verify the entire vSG configuration the following commands can be used:

```
CLI (network-admin@switch) > subnet-show
switch name     scope vlan vrf      network      anycast-gw-ip packet-relay
forward-proto state enable
------ ------- ----- ---- ------- ----------- ------------ ------------
------------ ----- ------
switch vlan_10 local 10   blue    10.1.1.0/24 10.1.1.1      disable
dhcp         ok    yes
switch vlan_20 local 20   red     20.1.1.0/24 20.1.1.1      disable
dhcp         ok    yes
switch vlan_30 local 30   transit 30.1.1.0/24 30.1.1.1      disable
dhcp         ok    yes

CLI (network-admin@switch) > vsg-vrf-show
switch vsg-name vnet vrf      type
------ -------- ---- ------- -----------
switch vsg1     0:0  blue    isolated
switch vsg1     0:0  red     isolated
switch vsg1     0:0  transit promiscuous

CLI (network-admin@switch) > vsg-network-show
```

```
switch vsg-name vrf      vnet subnet   network      network-state
------ -------- -------  ---- -------  -----------  -------------
switch vsg1     blue     0:0  vlan_10 10.1.1.0/24 ok
switch vsg1     red      0:0  vlan_20 20.1.1.0/24 ok
switch vsg1     transit 0:0  vlan_30 30.1.1.2/32 ok
```

Once added, a vSG route can also be removed with the corresponding `vsg-route-remove` command:

```
CLI (network-admin@switch) > vsg-route-remove name vsg1 vrf transit network
0.0.0.0/0 gateway-ip 30.1.1.2
```

Configuring IGMP Snooping with VXLAN

By snooping IGMP messages it is possible to determine the (local) port membership for multicast groups. It is also possible to include the logical ports associated with VXLAN tunnels and their remote VTEPs when IGMP messages are snooped on remote overlay network nodes.

The following command supports this feature:

```
CLI (network-admin@switch) > igmp-snooping-modify vxlan|no-vxlan
```

`vxlan|no-vxlan`	Enable IGMP on VXLAN. Disabled by default.

```
CLI (network-admin@switch) > igmp-snooping-modify vxlan

CLI (network-admin@switch) > igmp-snooping-show

enable:                         yes
vxlan:                          yes
enable-vlans:                   1-4092
snoop-link  local-vlans:                 none
```

To disable it:

```
CLI (network-admin@switch) > igmp-snooping-modify no-vxlan

CLI (network-admin@switch) > igmp-snooping-show
enable:                yes
vxlan:                 no
enable-vlans:          1-4092
snoop-linklocal-vlans: none
```

Informational Note: IGMP Snooping is enabled by default while the VXLAN option is disabled by default.

Let us consider an example: Assume that IGMP join messages for group `239.1.1.1` (from source `10.1.1.2`) are received on a tunnel associated with `VLAN 10` (with `VNI 10`), as shown in the command output below:

```
CLI (network-admin@switch) > vlan-show vxlan 10

id type   vxlan vxlan-type replicators scope description active stats     ports          untagged-ports active-
edge-ports
-- ------ ----- ---------- ---------- ----- ----------- ------ ----- --------------- -------------
-------------
10 public 10    user       none       local  vlan-10     yes    no    9,41,69-72,253   9              9
```

Group IP `239.1.1.1` is associated to source IP `10.1.1.2` and its port membership list only contains the logical port ID (*12755068416*) associated with a VXLAN tunnel:

```
CLI (network-admin@switch) > igmp-show group-ip 239.1.1.1
```

```
group-ip  node-ip  vnet vxlan bd vlan port       source node-type expires(s)
--------- --------_ ---- ----- -- ---- ---------- ------ --------- ---------_
239.1.1.1 10.1.1.2           10  12 12755068416 0.0.0.0 host      0
```

You can check the tunnel info (such as its associated VTEP IP addresses) corresponding to logical port *1275068416* with the following command:

```
CLI (network-admin@switch) > tunnel-show tunnelID 1275068416

scope:              local
name:               auto-tunnel-70
type:               vxlan
vrouter-name:       vr1
local-ip:           70.1.1.2
remote-ip:          80.1.1.2
router-if:          eth1.4092
next-hop:           70.1.1.1
next-hop-mac:       66:0e:94:70:61:7f
remote-switch:      0
active:             yes
state:              ok
bfd:                disabled
bfd-state:          unknown
error:
route-info:         80.1.1.0/24
ports:              19
auto-tunnel:        auto
```

You can also verify that the L2 table contains the MAC address corresponding to group IP `239.1.1.1` (i.e., 01:00:5e:01:01:01):

```
CLI (network-admin@switch) > l2-table-hw-show mac 01:00:5e:01:01:01

mac               vlan  vxlan  ports   state            hw-flags  mc-index
----------------- ----- -----  -----  ---------------   ------- ---------
01:00:5e:01:01:01 10    10     none   active,static,hit           201326595
```

Configuring Distributed VRF-aware vFlow

NetVisor OS version 6.0.0 introduces Virtual Routing and Forwarding (VRF) -aware vFlows, which enables vFlows to operate at scale in a distributed VRF environment. This feature extends support for Policy Based Routing (PBR) in a VRF. By using the VRF-aware vFlow enhancement, you can implement L3 or L4 security access lists, micro-segmentation, QoS, and monitoring features in a fabric with distributed VRFs.

Configuring Distributed VRF-aware Policy Based Routing

Distributed PBR enables service insertion (adding services into the forwarding path of traffic) of constructs such as firewalls. When a firewall is inserted into a network, the traffic originating from a VRF instance needs to be redirected to the firewall or DC gateway through a next-hop address or an ECMP group.

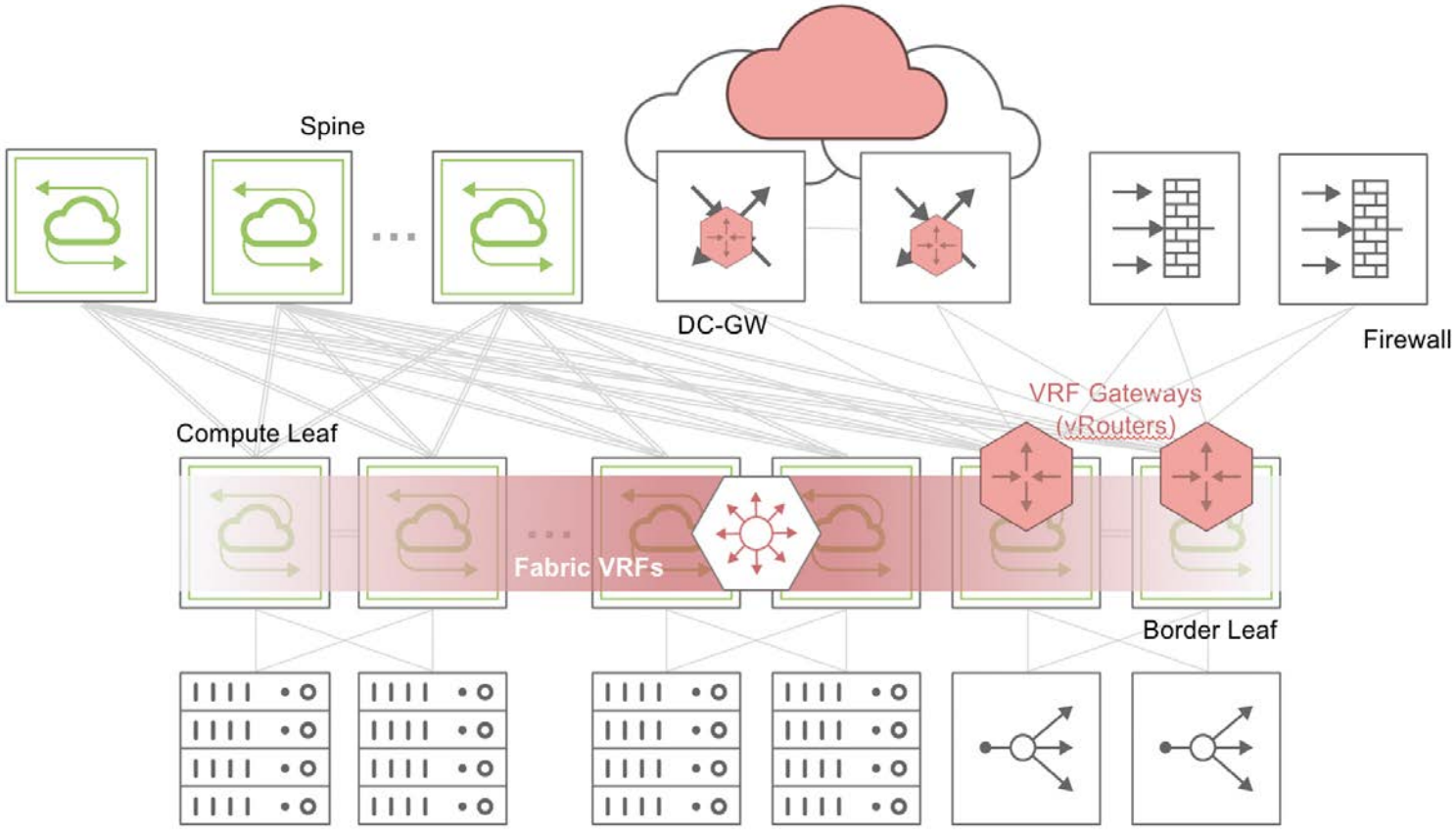


Figure 8-17 - Distributed VRF-aware Policy-based Routing

For instance, in a spine-leaf topology as shown in ***Figure 8-17***, if traffic with a source IP address of 100.1.1.10 in a VRF needs service insertion of a firewall, you can create a vFlow to redirect the traffic to a next hop address or ECMP group. When the traffic matches the source IP address of 100.1.1.10, traffic is sent to the next hop or ECMP group indicated in the vFlow, instead of being forwarded based on the Layer 3 routing entries.

You must enable PBR globally to create vFlows . You can enable PBR by using the command:

```
CLI (network-admin@switch1) > system-settings-modify policy-based-routing
Note: nvOSd must be restarted for this setting to take effect
```

Now, create a vFlow to send the packets from the source IP address to the required next hop IP address by

using the command:

```
CLI (network-admin@switch1) > vflow-create name flow1 scope local vrf vrf-1
src-ip 100.1.1.10 action to-next-hop-ip action-to-next-hop-ip-value
150.1.1.2
```

To view the details of the vFlow, use the command `vflow-show`:

```
CLI (network-admin@switch1) > vflow-show layout vertical
name:                          flow1
scope:                         local
type:                          pbr
src-ip:                        100.1.1.10
vrf:                           vrf-1
burst-size:                    auto
precedence:                    default
action:                        to-next-hop-ip
action-to-ecmp-group-value:    150.1.1.2
enable:                        enable
table-name:                     System-L3-L4-PBR-1-0
```

If there are multiple plausible next hop IP addresses to reach a firewall, you can perform load balancing by creating a static ECMP group and adding the next hop addresses (up to 16) to the ECMP group. Then, configure a vFlow to send the required traffic to the ECMP group and in turn, to the firewall. For details regarding configuration of ECMP groups, refer to the *Sending Network Traffic to an ECMP Group with PBR* section.

For example, to create a vFlow to send packets to a pre-configured ECMP group named `vrf_ecmp` (previously created with the `static-ecmp-group-create` command), use the `vflow-create` command with the action keyword `to-ecmp-group`:

```
CLI (network-admin@switch1) > vflow-create name flow2 scope local in-port
23 src-ip 10.0.11.100 src-ip-mask 255.255.255.0 dst-ip 10.0.12.100 dst-ip-
mask 255.255.255.0 vrf vrf-2 action to-ecmp-group action-to-ecmp-group-
value vrf_ecmp
```

To view the configuration details, use the `vflow-show` command:

```
CLI (network-admin@switch1) > vflow-show
name:                          flow2
scope:                         local
type:                          pbr
in-port:                       23
src-ip:                         10.0.11.100/255.255.255.0
dst-ip:                         10.0.12.100/255.255.255.0
vrf:                           vrf-2
burst-size:                    auto
precedence:                    default
action:                        to-ecmp-group
action-to-ecmp-group-value:    vrf_ecmp
enable:                        enable
table-name:                     System-L3-L4-PBR-1-0
```

Configuring Distributed VRF-aware Security Access Lists

NetVisor OS supports L3 and L4 security access lists for traffic in a VRF. The keywords `drop` or `none` can be supplied to the `action` parameter in the `vflow-create` command, to drop or allow packets.

For example, to drop all packets from the source IP address 100.1.1.10 in a VRF, use the command:

```
CLI (network-admin@switch1) > vflow-create name flow10 scope local src-ip
100.1.1.10 vrf vrf5 action drop
```

To view the vFlow details, use the command `vflow-show:`

```
CLI (network-admin@switch1) > vflow-show layout vertical
name:                       flow10
scope:                      local
type:                       pbr
src-ip:                     100.1.1.10
dst-ip:                     none
vrf:                        vrf5
burst-size:                 auto
precedence:                 default
action:                     drop
action-value:               none
metadata:                   none
enable:                     enable
table-name:                  System-L3-L4-PBR-1-0
```

Configuring Distributed VRF-aware Micro-segmentation

Micro-segmentation is the process of creating secure zones within a fabric to isolate workloads. The distributed VRF-aware vFlow feature allows you to create security zones in a fabric. You can configure micro-segmentation by creating L3 or L4 filters for internal tagging during ingress and making an aggregated egress decision.

For example, define a security zone (A) in a VRF. The zones can be based on various source IP address criteria, such as, across subnets, external subnets, and intra-subnet.

Create vFlows that define zone A as the source:

```
CLI (network-admin@switch1) > vflow-create name ZONE-A-SOURCE-10.1 scope fabric vrf VRF-1 src-
ip 10.10.1.0/24 action set-metadata action-value 10 precedence 5

CLI (network-admin@switch1) > vflow-create name ZONE-A-SOURCE-10.1 scope fabric vrf VRF-1 src-
ip 10.10.128.0/20 action set-metadata action-value 10 precedence 5

CLI (network-admin@switch1) > vflow-create name ZONE-A-SOURCE-10.3 scope fabric vrf VRF-1 src-
ip 10.10.33.0/24 action set-metadata action-value 10 precedence 5

CLI (network-admin@switch1) > vflow-create name ZONE-A-SOURCE-10.4 scope fabric vrf VRF-1 src-
ip 10.10.64.0/20 action set-metadata action-value 10 precedence 5
```

Create vFlows that define zone A as the destination. These vFlows are given a lower precedence in this example.

```
CLI (network-admin@switch1) > vflow-create name ZONE-A-DESTIN-11.1 scope fabric vrf VRF-1 dst-ip 10.10.1.0/24 action set-metadata action-value 11 precedence 3

CLI (network-admin@switch1) > vflow-create name ZONE-A-DESTIN-11.1 scope fabric vrf VRF-1 dst-ip 10.10.128.0/20 action set-metadata action-value 11 precedence 3

CLI (network-admin@switch1) > vflow-create name ZONE-A-DESTIN-11.3 scope fabric vrf VRF-1 dst-ip 10.10.33.0/24 action set-metadata action-value 11 precedence 3

CLI (network-admin@switch1) > vflow-create name ZONE-A-DESTIN-11.4 scope fabric vrf VRF-1 dst-ip 10.10.64.0/20 action set-metadata action-value 11 precedence 3
```

Zone A uses metadata 10 for traffic that leaves specific IP addresses and metadata 11 for traffic that arrives at specific IP addresses. Now, you can create egress micro-segmentation rules based on the security zone. The metadata values are among the parameters that define the egress policies, and these values help in making the egress decision to send the packets out.The egress policies are defined in Egress-Table-1-0 or the Egress Content Aware Processing (ECAP) hardware filter table.

For example, to allow UDP traffic from zone A to the destination IP address 172.10.0.0/8, use the following command:

```
CLI (network-admin@switch1) > vflow-create name ZONE-A-172.10-UDP-PASS table-name Egress-Table-1-0 vrf VRF-1 metadata 10 dst-ip 172.10.0.0/8 proto 17 action none
```

To allow TCP traffic from zone A to the destination IP address 172.10.0.0/8 and the reverse, use the set of following commands:

```
CLI (network-admin@switch1) > vflow-create name ZONE-A-172.10-TCP-PASS table-name Egress-Table-1-0 vrf VRF-1 metadata 10 dst-ip 172.10.0.0/8 proto 6 action none

CLI (network-admin@switch1) > vflow-create name 172.10-ZONE-A-TCP-PASS table-name Egress-Table-1-0 vrf VRF-1 metadata 11 src-ip 172.10.0.0/8 proto 6 action none
```

Configuring Distributed VRF-aware QoS Policies

You can use vFlows to set Differentiated Services Code Point (DSCP) and VLAN priority values for traffic in a VRF. For example, to set a DSCP value of 10 for traffic originating from 10.10.1.0/24, use the command:

```
CLI (network-admin@switch1) > vflow-create name flow5 scope local vrf VRF_2 src-ip 10.10.1.0 action set-dscp action-value 10
```

To view the details, use the `vflow-show` command:

```
CLI (network-admin@switch1) > vflow-show layout vertical
name:                   flow5
scope:                  local
type:                   pbr
src-ip:                 10.10.1.0
dst-ip:                 none
vrf:                    vrf_2
burst-size:             auto
precedence:             10
action:                 set-dscp
action-value:           10
```

```
metadata:                       none
enable:                         enable
table-name:                      System-L3-L4-PBR-1-0
```

To set a VLAN priority value of 5 for traffic from 10.10.1.1/24, use the command:

```
CLI (network-admin@switch1) > vflow-create name flow6 scope fabric vrf
VRF_3 src-ip 10.10.1.1/24 action set-vlan-pri action-value 5
```

Configuring Distributed VRF-aware vFlow-based Monitoring

You can copy packets to a port or CPU for traffic monitoring by using the action keywords `copy-to-cpu` and `copy-to-port.`

For example, to copy the packets with the source IP address of 10.1.1.1 in vrf10 to the CPU, use the command:

```
CLI (network-admin@switch1) > vflow-create name flow5 scope local vrf vrf10
src-ip 10.1.1.1 action copy-to-cpu
```

Use the `vflow-show` command to view the details:

```
CLI (network-admin@switch1) > vflow-show layout vertical
name:                           flow5
scope:                          local
type:                           pbr
src-ip:                         10.1.1.1
vrf:                            vrf10
burst-size:                     auto
precedence:                     default
action:                         copy-to-cpu
action-value:                   none
enable:                         enable
table-name:                     System-L3-L4-PBR-1-0
```

To copy packets with a source IP 10.1.1.2 in vrf10 to port 25, use the command:

```
CLI (network-admin@switch1) > vflow-create name flow7 scope local vrf vrf10
src-ip 10.1.1.2 action copy-to-port action-to-ports-value 25
```

For details on vFlows, see the *Configuring and Using vFlows* chapter.

Configuring Multicast Fabric VRFs

Note: Starting from NetVisor OS release 6.0.1, this feature is supported on NRU03 and NRU-S0301 platforms and on the Dell S4100 and S5200 Series.

Starting from NetVisor OS release 6.1.0, this feature is supported also on the following platforms:

- Dell Z9100-ON, S5048F
- Freedom F9532L-C/Edgecore AS7712-32X
- Freedom F9532-C/Edgecore AS7716-32X
- Freedom F9572L-V/Edgecore AS7312-54XS
- Freedom F9572-V/Edgecore AS7316-54XS
- Freedom F9532C-XL-R/Edgecore AS7716-32X

The CLI commands to configure and verify distributed multicast forwarding with Multicast Fabric VRFs are described below:

```
CLI (network-admin@switch) > vrf-multicast-show
enable: yes

CLI (network-admin@switch) > vrf-multicast-modify
```

`scope local|fabric`	Specify the scope - fabric or local
`disable`	Specify to disable flooding over tunnels in DM Forwarding
`enable`	Specify to enable flooding over tunnels in DM Forwarding

Enabling or disabling Multicast Fabric VRFs takes effect only after a reboot. For example, this command disables the feature:

```
CLI (network-admin@switch) > vrf-multicast-modify disable
System must be REBOOTED for this setting to take effect
```

The above command is analogous to IGMP Snooping's insofar as you can modify the feature's state with a local or fabric scope.

With Multicast Fabric VRFs enabled, to optimize certain network designs, in more recent NetVisor OS releases BUM traffic is *not* flooded to tunnels *by default*. However, you can use the modify command below to enable/disable flooding of BUM traffic on tunnels to cater to your network design's specific requirements. First check what is the current setting, and then modify it if needed:

```
CLI (network-admin@switch) > vrf-multicast-flood-show
enable: no

CLI (network-admin@switch) > vrf-multicast-flood-modify
scope    scope - fabric or local
disable  disable flooding over tunnels in DM Forwarding
enable   enable flooding over tunnels in DM Forwarding

CLI (network-admin@switch) > vrf-multicast-flood-modify enable
```

The `vrf-mroute-show` command, can be used to show the L3 entries dynamically created to implement L3 multicast forwarding:

```
CLI (network-admin@switch) > vrf-mroute-show

srcip     group    vxlan   vlan vrid hw-group-id send-vlan ports   flags
------- --------- ----   ---- ---- ----------- --------- -----  ------
0.0.0.0 224.1.1.1 10501   501  0     0x200098a     0          none   Active
0.0.0.0 224.1.1.1 10501   501  1     0x200098a     506        1      Active
```

This command can be used to check the list of multicast destination groups, incoming VLANs, outgoing ports and egress VLANs.

The above output shows (*, 224.1.1.1, 501) with port 1 as outgoing port and 506 as egress VLAN. Note that this kind of L3 entry will only contain routed ports in the egress VLAN(s) (instead, a separate L2 entry will bridge in the ingress VLAN).

The `igmp-show` command offers a high-level view of the dynamic behavior of the feature (in this example, port 15 is the *loopback port*):

```
CLI (network-admin@switch) > igmp-show

group-ip  node-ip   vnet bd vlan port       source  node-type expires(s)
--------- --------- ---- -- ---- ---------- ------- --------- ----------
<snip>
225.1.1.1 ::                10   1275068416 0.0.0.0 host      0
225.1.1.1 ::                10   15         0.0.0.0 host      0
225.1.1.1 40.1.1.5          30   19         0.0.0.0 host      279
```

The first line is a Layer 2 entry for group 225.1.1.1 that corresponds to a logical port (1275068416), namely, to a VXLAN tunnel to replicate the multicast traffic to (as it has at least a receiver in the same VNI-mapped VLAN). More than one tunnel may be listed in this position.

The second line is dynamically programmed by the software to send a copy of the multicast traffic for group 225.1.1.1 in VLAN 10 to a special loopback port (see below) so that a second (routing) pass after recirculation can be performed. This means that there is a source sending traffic to the group on the ingress VLAN, therefore a L3 lookup can happen for local routing.

The third line (in the show output above) shows IGMP membership on port 19 in egress VLAN 30 (which needs to be routed to).

The loopback port(s) used by Multicast Fabric VRFs are called `mlb-loopback-trunk`, which is an additional loopback trunk for L3 multicast traffic only. The `vxlan-loopback-trunk` instead continues to be used for unicast traffic. While `vxlan-loopback-trunk` is auto-created, `mlb-loopback-trunk` needs to be manually created by the user.

Furthermore, under certain circumstances it may be necessary to use the following commands to add, show or delete a static IGMP entry on a port/trunk:

```
CLI (network-admin@switch) > igmp-static-group-create group-ip 239.1.1.1
vlan 10 ports 14

CLI (network-admin@switch) > igmp-static-group-show
group-ip   bd vlan ports
---------- -- ---- -----
239.1.1.1     10   14

CLI (network-admin@switch) > igmp-static-group-delete group-ip 239.1.1.1
vlan 10
```

In this example, the commands are useful for interoperability purposes when you need to simulate an IGMP join on a certain port (in this case port 14, to which a multicast receiver is indirectly connected through a third-party network). Then, when the receiver no longer needs to receive the multicast traffic, the static IGMP entry can be deleted.

Configuring Multiple VXLAN Next Hops on a Single Egress Port

Prior to the Netvsior ONE 6.1.0 release, a hardware limitation on the Broadcom ASICs allowed NetVisor OS to configure only one egress port to next hop index (NHI) mapping per physical egress port or trunk port. This limitation obstructed the creation of multiple VXLAN tunnels in the hardware if the respective next hops resolved on the same physical port or trunk port.

From NetVisor OS version 6.1.0 onward, you can create multiple VXLAN tunnel next hops on the same egress trunk port or physical egress port due to a resolution in the hardware restriction.

This feature is available on the following Trident3, Trident2+, and Maverick platforms of Arista switches:

• S5248F-ON	• S4128F-ON	• AS5812-54X
• S5224F-ON	• S4128T-ON	• AS5812-54T
• S5232F-ON	• S4112F-ON	• F9372-X
• S4148-ON	• S4112T-ON	• F9372-T
• S4148F-ON	• NRU03	
• S4148T-ON	• NRU03-S0301	

For example, consider a basic sample topology (***Figure 8-18***) consisting of four switches interconnected through physical and trunk ports. Each switch is connected to other three switches using a trunk port or a physical port (trunk port 1/ports 5, 6 in the diagram). They are also inter-connected via a layer 2 network. To build a VXLAN-based fabric, you can configure a full mesh of VXLAN tunnels connecting all the switches to one another.

Prior to NetVisor OS version 6.1.0, the switches were unable to establish three separate VXLAN tunnels to other three switches because their next hops were all routed through the same trunk port. In other words, you could achieve only a one-to-one mapping of egress ports to VXLAN next hops.

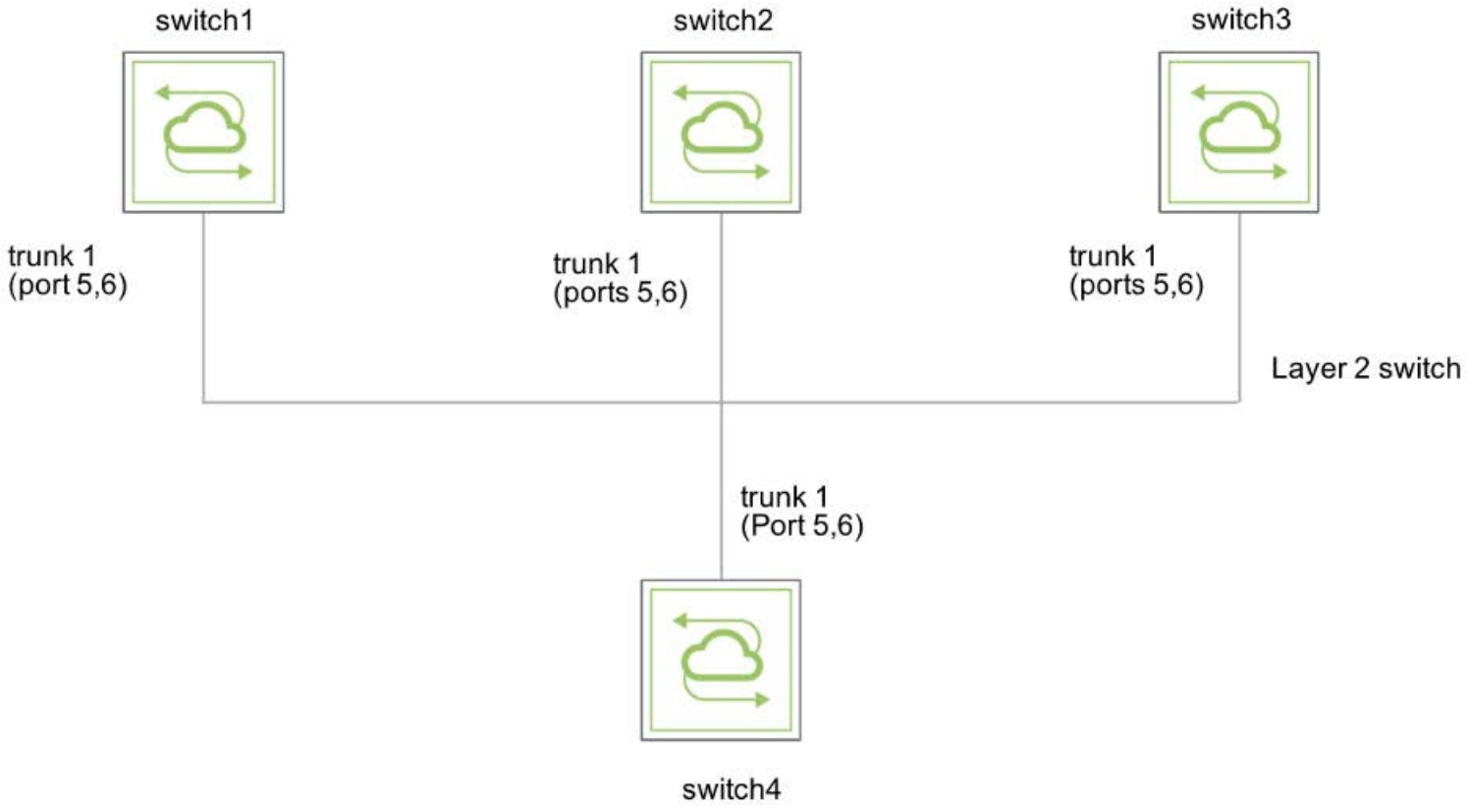


Figure 8-18 - Topology with Physical Egress Ports or Trunk Ports

With NetVisor OS 6.1.0, you can configure a one-to-many mapping of:

- Physical egress ports to VXLAN next hops, or
- Trunk egress ports to VXLAN next hops

To configure multiple VXLAN tunnels, use the command:

```
CLI (network-admin@switch4) > tunnel-create
```

`tunnel-create`	This command creates a tunnel between two switches.
`scope local|cluster`	Specify the scope.
`name <name-string>`	Enter the name of the tunnel you want to configure.
Specify any of the following options:	
`vrouter-name <vrouter name>`	Specify the name of the vRouter service
`local-ip <ip-address>`	Specify the local IP address.
`remote-ip <ip-address>`	Specify the remote host IP address.

As an example, in the above topology (***Figure 8-18***), configure three static VXLAN tunnels to connect switch4 to switch1, switch2, and switch3 by using the commands:

```
CLI (network-admin@switch4*) > tunnel-create scope local name tun20-9-3
vrouter-name vr-tucana local-ip 20.20.20.9 remote-ip 20.20.20.3

CLI (network-admin@switch4*) > tunnel-create scope local name tun50-9-3
vrouter-name vr-tucana local-ip 50.50.50.9 remote-ip 50.50.50.3

CLI (network-admin@switch4*) > tunnel-create scope local name tun80-9-3
vrouter-name vr-tucana local-ip 80.80.80.9 remote-ip 80.80.80.3
```

View the configuration details using the command:

```
CLI (network-admin@switch4*) > tunnel-show
```

```
scope name     type  vrouter-name local-ip   remote-ip router-if  next-hop   next-hop-mac      nexthop-vlan active state  bfd      bfd-state          route-info    ports auto-tunnel mac-learning
----- -------- ----- ------------ ---------- --------- ---------- ---------- ----------------- ------------ ------ ------ -------- ------------------ ------------- ----- ----------- ------------
local tun20-9-3 vxlan vr-tucana   20.20.20.9 20.20.20.3 eth0.20    20.20.20.3 66:0e:94:b7:e8:6b 20           yes    ok     disabled not-replicator-vtep 20.20.20.0/24 272   static      on
local tun50-9-3 vxlan vr-tucana   50.50.50.9 50.50.50.3 eth1.50    50.50.50.3 66:0e:94:b7:e8:6b 50           yes    ok     disabled not-replicator-vtep 50.50.50.0/24 272   static      on
local tun80-9-3 vxlan vr-tucana   80.80.80.9 80.80.80.3 eth0.80    80.80.80.3 66:0e:94:b7:e8:6b 80           yes    ok     disabled not-replicator-vtep 80.80.80.0/24 272   static      on
```

The output above shows that the VXLAN tunnels are created and are functional as in ***Figure 8-19***.

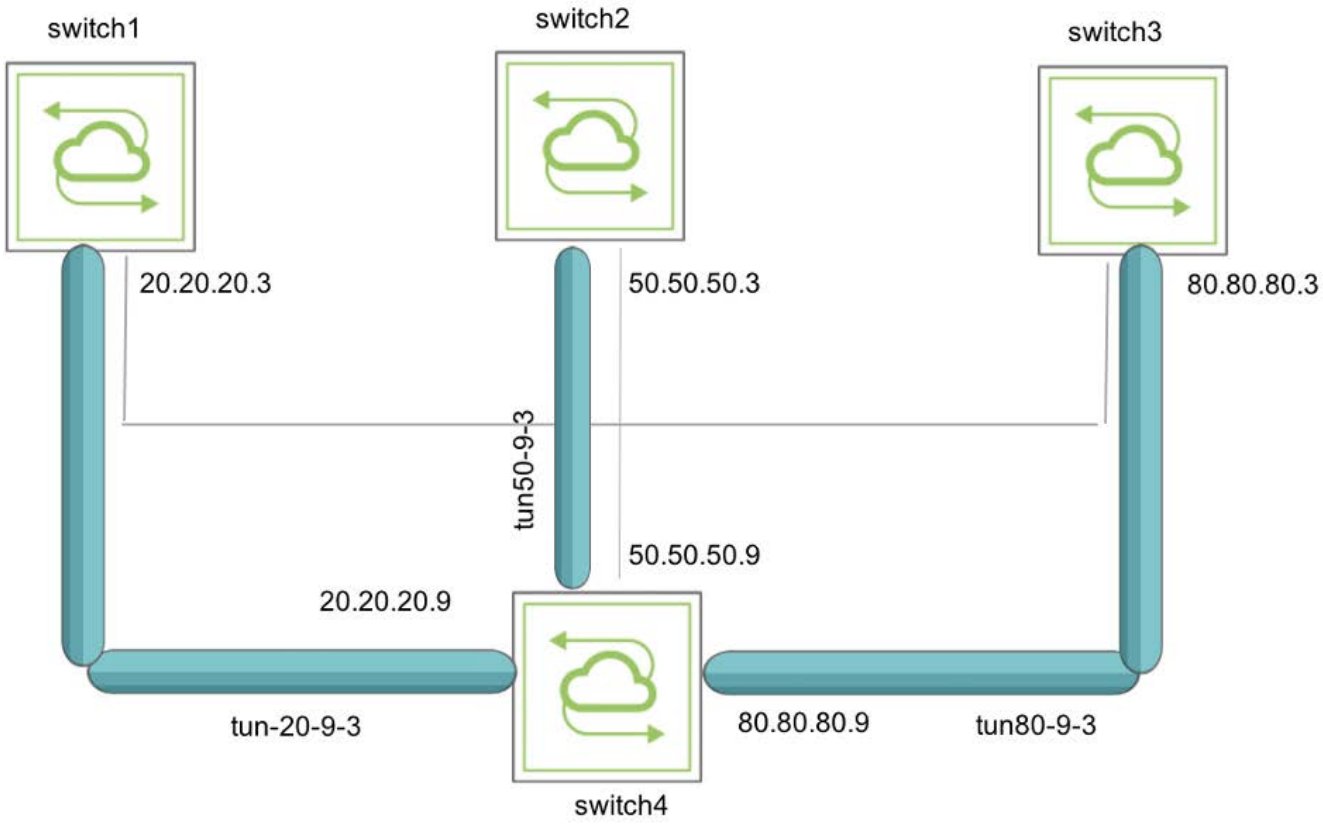


Figure 8-19 - Topology with VXLAN tunnels

To view the details of each tunnel, use the following commands:

```
CLI (network-admin@switch*) > vlan-show id 20

id type   auto-vxlan replicators scope description active stats ports       untagged-ports active-edge-ports
-- ------ ---------- ----------- ----- ----------- ------ ----- ----------  -------------- ----------------
20 public no         none        local vlan-20     yes    yes   0-1,125,272 none           0,272
```

```
CLI (network-admin@switch4*) > vlan-show id 50

id type   auto-vxlan replicators scope description active stats ports       untagged-ports active-edge-ports
-- ------ ---------- ----------- ----- ----------- ------ ----- ----------- -------------- ----------------
50 public no         none        local vlan-50     yes    yes   0-1,125,272 none           0,272

CLI (network-admin@switch*) > vlan-show id 80

id type   auto-vxlan replicators scope description active stats ports       untagged-ports active-edge-ports
-- ------ ---------- ----------- ----- ----------- ------ ----- ----------- -------------- ----------------
80 public no         none        local vlan-80     yes    yes   0-1,125,272 none           0,272
```

Supported Releases

Command/Parameter	NetVisor OS Version
`trunk-modify name vxlan-loopback-trunk`	Commands with parameters first supported in *version 2.4*
`tunnel-create`	Command added in *version 1.2*. The option, `cluster`, was added to scope and the parameters, `vrouter-name` and `peer-vrouter-name`, were added in *version 2.4*.
`tunnel-vxlan-add`	*In version 1.2 the command was introduced. In version 2.4 the parameter, vxlan, was added.*
`vtep-create, vtep-delete, vtep-show`	Commands introduced in *version 2.6.0.*
`vtep-vxlan-add, vtep-vxlan-remove, vtep-vxlan-show`	Commands introduced in *version 2.6.0*.
`vrf-create, vrf-delete, vrf-modify, vrf-show, subnet-create, subnet-delete, subnet-modify, subnet-show`	Commands introduced in *version 3.0.0*
`igmp-snooping-modify vxlan|no-vxlan`	Parameter introduced in *version 3.1.1*
`vrf-multicast-modify, vrf-mroute-show`	Commands introduced in *version 5.1.0*
`vle-create, subnet-delete, subnet-modify, subnet-show`	Commands introduced in *version 2.5.4*
`vlan-create vxlan-mode standard| transparent`	In *version 2.5.2* the parameter `vxlan-mode` was added.
`vlan-create auto-vxlan`	In version 6.0.0 the parameter `auto-vxlan` was added.
`mac-learning | no-mac-learning`	Parameter introduced in version 6.0.0
`system settings-modify riot-single-pass`	Command introduced in version 6.0.0
`system-settings-modify no-single-pass-flood|single-pass-flood`	Command introduced in version 6.0.0
`system-settings-modify no-single-pass-l2-known-multicast | single-pass-l2-known-multicast`	Command introduced in version 6.0.1
`isolated`	VTEP parameter introduced in version 6.0.1
`vsg-create|delete|modify, vsg-vrf-add|remove, vsg-network-add|remove, vsg-show, vsg-vrf-show, vsg-network-show`	Commands introduced in version 6.0.1

`type isolated, promiscuous`	vSG parameters introduced in version 6.0.1
`vsg-route-add, vsg-route-remove, vsg-route-show`	Commands introduced in version 6.1.0
`anycats-mac-for-forwarding/no-anycast-mac-for-forwarding`	VRF parameters introduced in version 6.1.0

Please also refer to the *Arista NetVisor OS Command Reference* document.

Related Documentation

Refer to these sections in the Configuration Guide:

- Configuring Layer 2 Features
- Configuring Layer 3 Features
- Setting up and Administering the Unified Cloud Fabric
- Configuring High Availability

for further information on concepts mentioned in this section (such as Layer 2, Layer 3, the fabric, clusters, vLAGs, VRRP, etc.)

Configuring Advanced Layer 2 Transport Services

This chapter provides information about the Advanced Layer 2 Transport Services on a NetVisor OS switch using the NetVisor OS command line interface (CLI).

Understanding Virtual Link Extension (vLE)

Arista Networks offers a highly flexible approach to software-defined networking (SDN), called *Adaptive Cloud Fabric*, which leverages a sophisticated distributed architecture in order to enable organizations to build scalable private and public clouds with the very desirable benefits of ease of management, reliability, security and performance.

With regard to ease of management and flexibility, the VXLAN technology described in the *Understanding VXLAN* chapter augments the Adaptive Cloud Fabric with a plethora of advanced capabilities. Among such features is the ability to natively transport traffic across the VXLAN fabric for end-to-end data center connectivity through overlay bridging and distributed routing.

Furthermore, in certain scenarios, network administrators have the requirement for the VXLAN fabric to be able to transport traffic *transparently* between two different ports as if interconnected through a *virtual pipe*: in other words, what comes in on one end goes out of the other end. This capability is often referred to as a '*pseudo-wire*' in the literature. Arista calls it a *virtual wire*.
Arista Virtual Link Extension (vLE) implements a transparent point-to-point virtual link over the VXLAN fabric to emulate the connectivity of an actual wire interconnecting any two fabric ports selected by the user, as shown in ***Figure 9-1*** below.

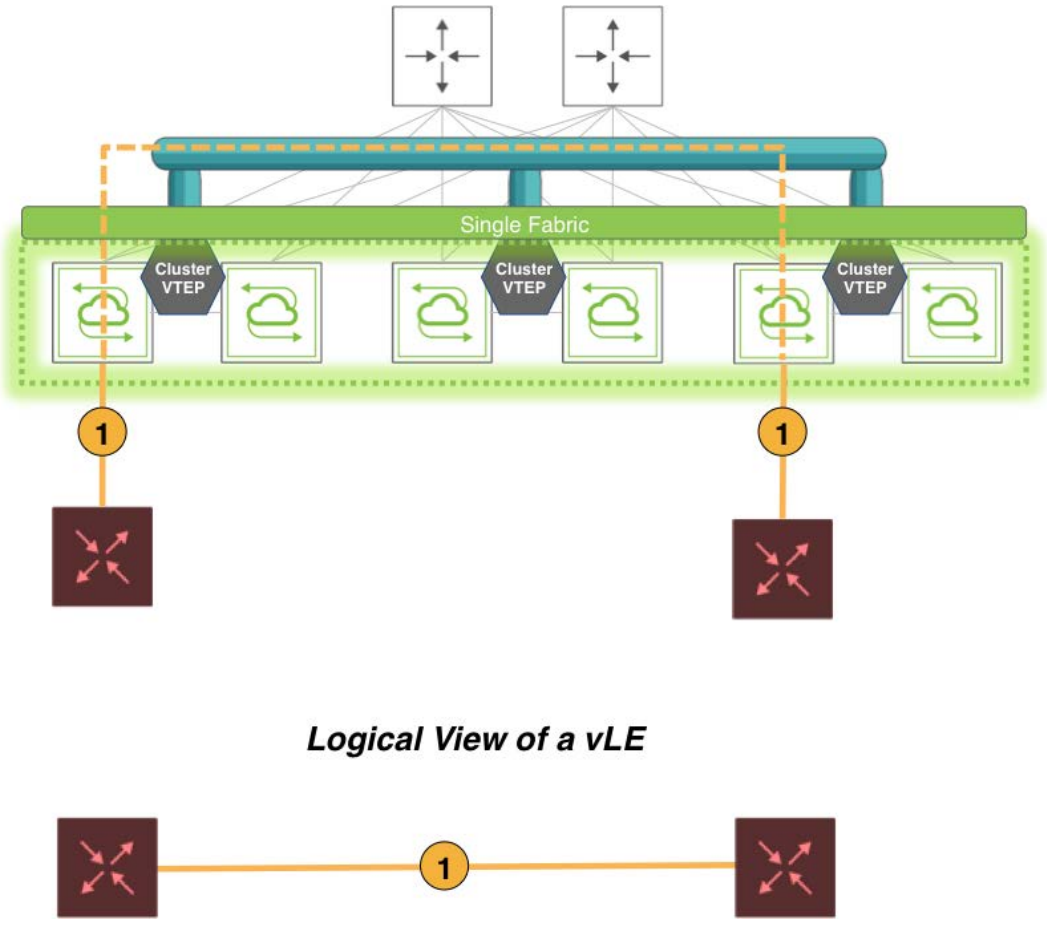


Figure 9-1: Virtual Link Extension over the VXLAN Fabric

The key characteristic of a vLE is its ability to take what comes in on the local end of the virtual pipe and send it to the remote end. That includes regular data plane traffic as well as Layer 2 PDUs.

Hence, one application of Arista's Virtual Link Extension feature is the remote monitoring of traffic between two fabric ports.

Another common use case is *lab automation*, i.e., the flexible interconnection of lab equipment through the fabric: with multiple vLEs the port interconnections can be dynamically (for example, programmatically) managed by the lab administrator through the creation of ad-hoc virtual pipes.

This type of use case is sometimes also referred to as '*IP Virtual Wire*' (as it runs on top of a VXLAN overlay) or '*Virtual Wire+*' to distinguish it from the regular (Layer 1) VirtualWire™ functionality (refer to the *Configuring VirtualWire Deployment Guide*). It is usually deployed with *error transparency* enabled (see below for more details).

From an implementation perspective, it should be noted that the Adaptive Cloud Fabric can transport multiple vLEs across a common high-performance VXLAN-based interconnect, thus enabling a very high degree of configuration flexibility and scalability. The main network constraint is the maximum bandwidth available on any of the shared physical links, which however can be scaled using higher speed Ethernet ports or trunks to avoid any chokepoints.

From a network design standpoint, in terms of ease of configuration, vLEs can use manual or automatic VXLAN tunnels to piggyback the mirrored traffic end-to-end and thus they enjoy the same wide-spanning benefits of any other VXLAN-based feature. The only constraint is that they cannot share the same VXLAN tunnels with other features.

Alternatively, the Arista NetVisor UNUM management platform can be leveraged to very effectively perform the provisioning of the entire vLE deployment in very few steps. First, a Layer 3 fabric can be deployed from a UNUM Layer-3 Playbook. Then, *Manager --> Services --> Manage vLE* can be used to create and manage one or more vLEs through user-friendly configuration forms. (Refer to the UNUM Fabric Manager documentation for more details.)

Finally, programmatic creation and management of vLEs can be performed through REST APIs.

About Virtual Link Extension State Tracking

Just as in the case of physical back-to-back Ethernet connections, it is usually desirable that if one end of a vLE goes down the peer end mirrors that state change.

This behavior is called *link state tracking*. With such symmetric behavior, link state changes can be propagated end-to-end and hence network nodes or endpoints are able to react to them.

In other words, when a vLE is created with tracking enabled between two physical ports of two switches, the vLE remains up as long as both physical ports are in the link up state.

When a vLE is created with tracking enabled between two trunk ports (a.k.a. port channels), the vLE stays up as long as at least one member port in the trunk is up and the remote trunk is also up. When the last trunk member goes down, the vLE is brought down. (Note that when you configure vLE tracking on a trunk port, you cannot configure tracking on individual trunk members.)

In the CLI link state tracking can be enabled/disabled over each vLE individually through the `tracking|no tracking` option. (In fact, having tracking enabled may not be required in all cases, such as for troubleshooting purposes.) Similarly, in the Arista NetVisor UNUM management platform the `tracking|no tracking` option corresponds to a true/false configuration checkbox to be ticked by the user.

For more CLI configuration details refer to the *Configuring Virtual Link Extension State Tracking* section below. Refer to the UNUM Fabric Manager documentation for the specific pane and dashboard layout details for the link state tracking configuration checkbox.

The `node-1-port` and `node-2-port` are the names of the two vLE endpoints whose link state can be tracked. When `node-1-port` goes down the `vle-show` command can be used to check and display the overall vLE state, which is `local-down`. When `node-2-port` goes down, instead, the `vle-show` command displays the `remote-down` state.

If needed, as part of the vLE configuration, it is also possible to tweak the timing to be used by the tracking logic to detect a port down state and to trigger a (virtual) link down event.

For more details on the configuration, refer to the following sections.

About Virtual Link Extension Between Two Ports of the Same Switch

In earlier versions of NetVisor OS, the vLE implementation supported only one vLE endpoint per node. With version 5.1.1, both vLE endpoints can be part of the same node. This configuration is referred to as *local vLE*.

A vLE can have two ports in the same node only if the corresponding vLE's VXLAN ID is not already mapped to a tunnel. Similarly, a tunnel can be mapped to a vLE's VXLAN ID if there is at most one local port in the vLE. In the case of auto-tunnels, a VXLAN ID can be mapped to a VTEP only if there is at most one local port in the node associated with that VTEP.

NetVisor OS allows the users to track the vLE port status when part of a local vLE. In such case, the `node-1` and `node-2` labels refer to the same switch. However, for consistency reasons, the `node-2-port` is still considered as '*remote port*' in the `vle-show` output, even though in actuality both the vLE ports are on the same switch.

Let us assume, for example, that port 30 (`node-2-port`) goes down, the vLE status is displayed as `remote-down` (see example below) so the user can understand exactly which port is causing the vLE to be down. Instead, if port 22 (`node-1-port`) is down, then the vLE status is displayed as `local-down`.

```
CLI (network-admin@switch) > vle-show layout vertical
name:              vle1
vnet:
node-1:            vanquish1
node-2:            vanquish1
node-1-port:       22
node-2-port:       30
status:            remote-down
tracking:          enabled
ports-state:       override
create-time:       09:44:56
```

Use the `port-show` command to check if the `node-2-port` status is `down` or `vle-wait`. The `vle-wait` state indicates that the port is brought down due to remote port being down

About Virtual Link Extension Redundancy with Clusters

In network designs using pseudo-wires, link redundancy can be achieved by creating trunks using (at least)

pairs of pseudo-wires as shown below. Such trunks originate and terminate on the *end devices*.

That can be conveniently achieved in fabric designs that leverage pairs of leaf switches (called *clusters* in Arista parlance) to provide redundancy for Layer 2 and Layer 3 last-hop traffic: as shown in ***Figure 9-2*** below, when vLEs are configured, vLE redundancy can be implemented by setting up *one vLE per cluster switch* and *by leveraging port channeling* or *NIC teaming* (depicted with an ellipse in the diagram) on the end devices to implement end-to-end *vLE trunks*.

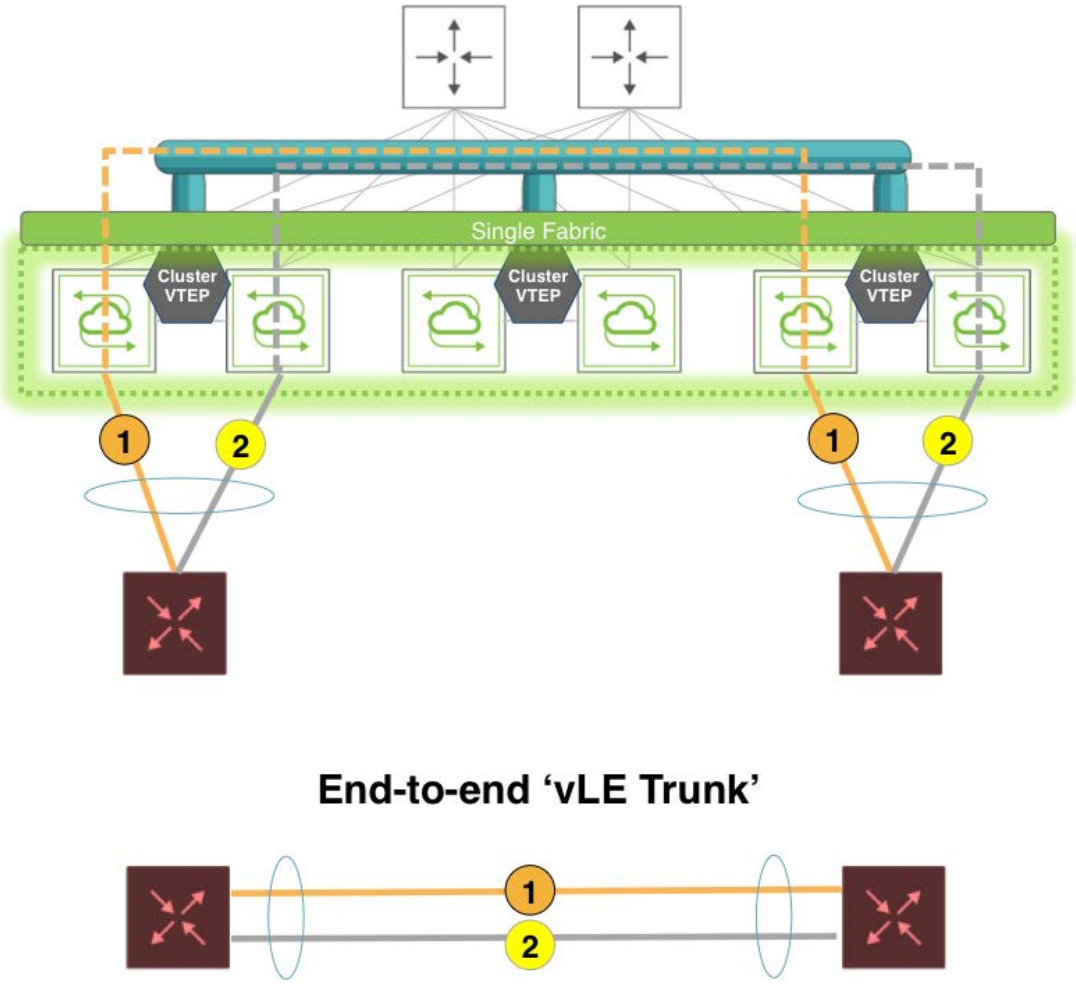


Figure 9-2: Virtual Link Extension End-to-end Redundancy

Leaf switches don't need to implement load balancing and failover for the vLE trunks as vLEs are in essence just end-to-end transparent pipes, which transport all the traffic and propagate port state changes through link state tracking. Hence, vLE-attached end devices can just use standard 802.3ad link bundling, static or dynamic with the LACP protocol, to manage the aggregation and the failover of the virtual wires.

vLE redundancy has the following characteristics:

- Failover between VTEPs used by vLEs is not required.
- Hence VTEPs associated with vLEs do not use a VRRP virtual IP address. They use a primary IP address, instead, or the real IP address of a dedicated interface.
- L2 PDUs such as LACP messages on vLE transparent VLANs are not sent to a fabric node's management CPU, as they need to be passed transparently through the virtual wire.

In the example topology shown above in Figure 9-2, two generic switches are connected respectively to two cluster leaf nodes. There is no vLAG on such nodes as it is not required. Both generic switches are configured with the LACP protocol on their trunk links to run connectivity checks and to form the trunk dynamically. Hence, they are directly in charge of link redundancy and failover management, whereas the leaf nodes are responsible for *transparent end-to-end traffic transport*.

About Virtual Link Extension Error Transparency

In lab automation deployments employing vLE it is usually required to not only transport valid data plane frames and PDUs but also to include potential malformed frames that are received on a vLE port.

Starting from NetVisor OS release 5.1.1, with the optional configuration for *error transparency*, a vLE can be configured to transport also bad CRC frames as well as runts (i.e., undersized frames). Transported runt frames are padded with zeros up to a length of 64 bytes and their CRC is zeroed out, which can be used to recognize an undersized frame on the receiving end.

Error transparency is hardware-dependent (on newer ASICs) and thus works only on the following capable platforms:

- Dell S4100 Series
- Dell S5048
- Dell S5200 Series
- Dell Z9100
- Freedom F9372-X and F9372-T

In particular, on these platforms undersized frames with a *minimum* length of *50 bytes* are supported. Furthermore, full jumbo frame transport is also supported in NetVisor OS version 5.1.1 by upping the maximum supported MTU size to 9398 bytes.

Refer to the *Enabling Jumbo Frame Support* section of this Configuration Guides for command details.

Error transparency is enabled globally to minimize the configuration steps required in lab automation setups. However, since it disables CRC checks even on inter-switch links, it can be turned off on a per port basis wherever it's not required.

About Virtual Link Extension Traffic Analytics

NetVisor OS does not make copies of vLE-transported control frames (such as LLDP, etc.) to the switch management CPU. Any inner VLAN tag, if present, is also preserved. This is achieved by installing a system vFlow entry called *Virtual-Link-Extend* with the highest priority of 15 with no action specified so that control frames are not terminated and sent to CPU.

In addition, to support *vLE traffic analytics*, a few additional system vFlow entries (named *System-vLE-x*, where *x* can be *S* or *F* or *R*) are installed with the same priority as the Virtual-Link-Extend entry in order to copy TCP SYN/FIN/RST packets to the management CPU. This ensures that any SYN/FIN/RST packets carried by vLE can be used for *TCP flow analysis*.

The `vflow-system-show` command can be used to display such system entries used for transparency and flow analysis:

```
CLI (network-admin@switch) > switch * vflow-system-show name Virtual-Link-
Extend

switch  name                  scope type    precedence action enable table-name
------- ------------------  ----- ------  ---------- ------ ------ -------------------
switch  Virtual-Link-Extend local system 15           none   enable System-L1-L4-Tun-1-0
switch1 Virtual-Link-Extend local system 15           none   enable System-L1-L4-Tun-1-0
switch2 Virtual-Link-Extend local system 15           none   enable System-L1-L4-Tun-1-0

CLI (network-admin@switch) > connection-show layout vertical

vnet:     100
vlan:     100
vxlan:    10100
src-ip:   20.20.20.1
dst-ip:   20.20.20.2
dst-port: http
cur-state:    fin
syn-resends:  0
syn-ack-resends:   0
latency:  74.8us
obytes:   149
ibytes:   311
total-bytes:  460
age: 2h11m21s
```

Understanding Virtual Port Groups (vPG)

In addition to vLE, Arista Networks' NetVisor OS supports an additional and more flexible VXLAN-based technology called a Virtual Port Group (vPG). This powerful CLI construct can be used to define a set of ingress (or source) ports and of destination ports so that the hardware can transport traffic from the source ports to (multiple) desired destination ports, and (optionally) in the reverse direction too.

Forwarding is performed in hardware using vFlow policies. For more details on vFlow, please refer to the *Configuring vFlow* chapter.

When the vPG configuration forwards traffic from the source ports to the destination ports only, it is referred to as a unidirectional vPG topology (as the monitored traffic flows in one direction only).

The primary use case for this configuration is Arista's Network Packet Broker. For more details on this use case, please refer to the *Understanding and Configuring Network Packet Broker* sections. These sections provide a full configuration walk-through to set up vPGs and vFlow policies to establish a secondary monitoring network for production traffic.

On the other hand, when the vPG configuration forwards traffic bidirectionally between source ports and destination ports, it is referred to as a bidirectional vPG topology: it can be useful to establish a sophisticated set of virtual wires between network nodes for bidirectional data exchange (for example, for lab automation use cases).

About Virtual Port Group State Tracking

In bidirectional vPG topologies (for example, for lab automation purposes) it is often required to mirror the state of one end of the connection to the other end.

Starting from NetVisor OS release 7.0.1, the bidirectional vPG functionality leverages a BFD-based state tracking logic to achieve state tracking across VXLAN tunnels. In other words, when enabled this tracking logic makes sure that, if a port on a local vPG goes down, the corresponding remote vPG's port is also brought down, and vice versa. This tracking functionality can be particularly handy in lab automation use cases.

The common tracking logic is called the vBT (vLE/vPG BFD Tracking) module. For vPGs this logic is currently limited to intra-fabric tracking.
Whenever a local port (which is a part of a vPG with tracking enabled) goes down, the vBT logic uses BFD to propagate the change of state to the remote endpoint which then brings its corresponding port down and puts it into `vbt-wait` state. Subsequently, whenever that local port comes back up, the state change is again propagated and the remote endpoint is brought back up too and removed from the `vbt-wait` state.

Note that unidirectional vPGs don't support this bidirectional tracking logic.
Moreover, only point-to-point bidirectional vPGs support state tracking: that means that each bidirectional vPG can have only one port in one node for tracking to be enabled.

Refer to the *Configuring Virtual Port Group State Tracking* section below for the command syntax details.

Understanding VXLAN-based Bridge Domains for IEEE 802.1Q and QinQ Internet-working

With the IEEE 802.1Q standard a network design is constrained to 4094 possible VLAN numbers. When more Layer 2 identifiers are needed, and a hierarchy of network entities can be established, the *VLAN stacking technology* (informally also called *QinQ*, after the name of Cisco's original implementation, also known as 802.1Q Tunneling) can be utilized to scale up to 16 million (4094 x 4094) identifiers using *two* (*concatenated*, i.e., '*stacked*') *VLAN tags instead of one*.

The IEEE organization has standardized the VLAN stacking technology with the *Provider Bridges* standard, also known as *IEEE 802.1ad*, which was later incorporated into the IEEE 802.1Q-2011 revision of the LAN/MAN standard.

IEEE 802.1ad refers to providers of services, such as Transparent LAN Services in Metro networks. There are also a number of data center designs that can tap the scalability of the VLAN stacking technology: for example data center networks in which one tag (the so-called *outer tag*) is used to identify a data center customer, while the second tag (the so-called *inner tag*) is used to identify a customer service.

With this technology it is possible to identify up to 16 M services for up to 4K customers. The double-tagged traffic is oftentimes handed off to a provider (for example a cloud exchange provider) which can terminate it for end-to-end connectivity. Hence, this scheme can be used to deploy *hybrid cloud designs* in which private clouds integrate with external clouds.

In practice in such designs QinQ (i.e., VLAN stacking) augments the basic 802.1Q capabilities by massively scaling the number of usable network identifiers organized in a hierarchy.

Arista combines this technology with the VXLAN-based Unified Cloud Fabric to yield an even higher degree of scalability and flexibility. In fact, VXLAN IDs (a.k.a. VNIs) are 24-bit long (vs. 12-bits of VLAN IDs) and therefore are a perfect match for double-tagging Layer 2 network transports.

In addition, VLAN ID to VXLAN ID mappings enable a great degree of flexibility for network designers that Arista substantiates with a new type of configuration object called *VXLAN-based bridge domain* (BD).

VXLAN-based bridge domains will be supported in a future release of NetVisor OS: they can be mapped to single or multiple 802.1Q VLANs as well as to dual IEEE 802.1ad VLAN tags, covering all possible design requirements.

The different available configuration models for VXLAN-based bridge domains are:

- *Single tag mapping*, in which an outer VLAN tag (for example, a customer ID) is mapped to a VNI on an IEEE 802.1ad port and the inner VLAN tags are preserved inside the VXLAN encapsulation. This type of mapping can be used on customer facing 'QinQ access' interfaces.
- *Double tag mapping*, in which an outer VLAN + inner VLAN tag pair is mapped to a VNI on an IEEE 802.1ad port (traffic is received double-tagged in ingress and is marked with two tags in egress after VXLAN decapsulation). This type of mapping can be used on multi-VLAN ports (sometimes called *QinQ trunks*) facing for example an external cloud provider.
- *Single 802.1Q tag mapping*, in which a single 802.1Q VLAN (or multiple 802.1Q VLANs) are mapped to a common VNI (for example, for inter-DC communication within the same customer's private cloud network).

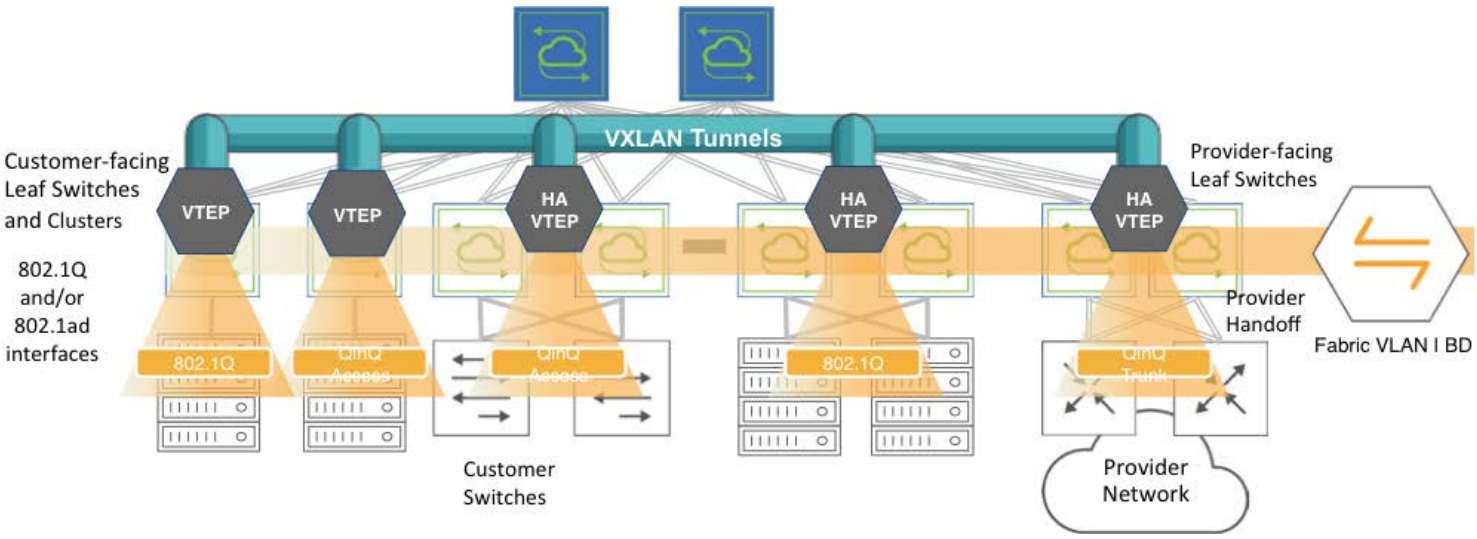


Figure 9-3: Hierarchical Fabric Structure Using Bridge Domains for Intra- and Inter-DC Connectivity

The various configuration models of bridge domains yield an unprecedented level of flexibility in terms of advanced Layer 2 transport services, with the ability of re-using VLANs across tenants, supporting QinQ hierarchies and handoff links, as well as aggregating multiple VLANs in the same overlay construct (thus supporting high-scale L2 tenant services).

Configuring Virtual Link Extension

The vLE configuration revolves around the creation and transport over VXLAN of one or more transparent VLANs using the transparent option of the `vlan-create vxlan-mode standard|transparent` command. To explain how that is implemented, let's first refer to ***Figure 9-4*** below which shows an example with port 10 and 17 as ingress/egress vLE ports on two fabric leaf switches that are supposed to be connected transparently through a vLE pseudo-wire:

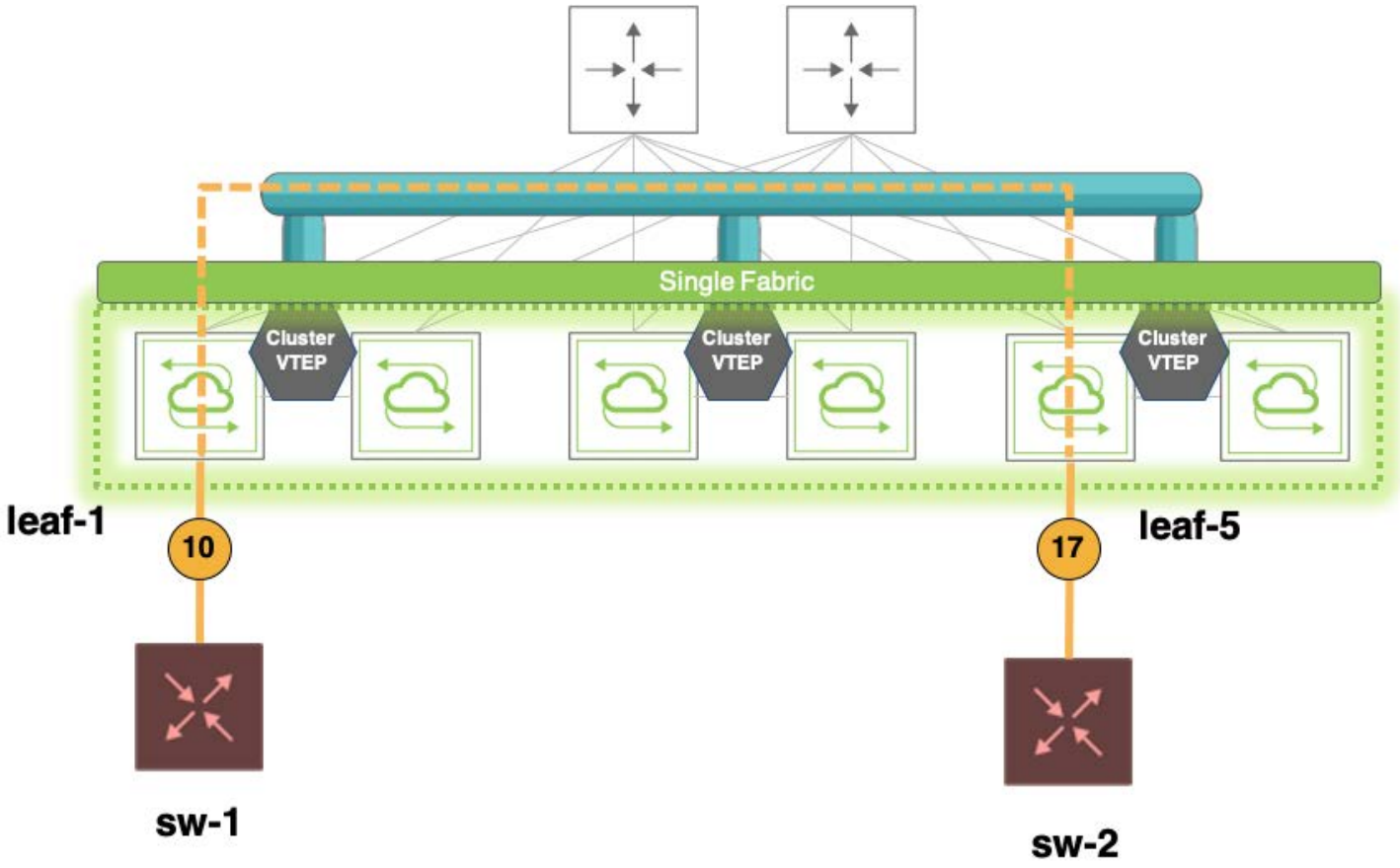


Figure 9-4. Two Ports Interconnected Through a vLE Pseudo-wire

Establishing such vLE pseudo-wire transport requires multiple configuration steps:

1. Configuration of jumbo frames on all the required ports (ingress and egress ports, as well as all inter-switch ports that are used to transport vLE traffic)
2. Creation of Layer 3 VLANs (or of routed ports) that are used as tunnel end points (with an MTU configuration suitable to route jumbo frames)
3. Set up of dedicated VXLAN connections. Typically, manual tunnels are created and set aside for vLE use.
4. Creation of a VXLAN-mapped *transparent VLAN* to transport the traffic over vLE.
5. Addition of the transparent VLAN's VXLAN ID to the allocated tunnels.

Let us look at one configuration example for each of the above steps.

Step 1 requires adding the jumbo option as part of the port configuration with the following command(s):

```
CLI (network-admin@leaf-1) > port-config-modify port 10 jumbo
```

In this example, as shown in the figure above, port 10 on leaf-1 is a vLE port. Jumbo (i.e., oversized) frame support should be enabled if mirroring of ingress frames with a (typical) MTU of 9216 bytes is required.

This option is usually already configured end-to-end on inter-switch links if a VXLAN transport is used in the fabric (refer to the *Configuring VXLAN* section).

Otherwise, the following commands can be used to enable the transport of oversized frames on inter-switch links (individual ports or trunks):

```
CLI (network-admin@leaf-1) > port-config-modify port <inter-switch port-
list> jumbo

CLI (network-admin@leaf-1) > trunk-modify name <trunkname> jumbo
```

Step 2 requires the creation of the *Layer 3 routing endpoints*, in this case VLAN interfaces (a.k.a. SVIs), on the switches where the ingress and egress vLE ports are located:

```
CLI (network-admin@leaf-1)> vlan-create id 1021 scope local description
VLE-VTEP ports none

CLI (network-admin@leaf-1)> vrouter-interface-add vrouter-name Leaf-1 vlan
1021 ip 10.21.1.1/30 mtu 9398

CLI (network-admin@leaf-5)> vlan-create id 1021 scope local description
VLE-VTEP ports none

CLI (network-admin@leaf-5)> vrouter-interface-add vrouter-name Leaf-5 vlan
1021 ip 10.21.7.1/30 mtu 9398
```

Then the VLAN(s) can be checked with the following command:

```
CLI (network-admin@leaf-1)> vlan-show id 1021

switch     id   scope description active
---------- ---- ------ ----------- -----
leaf-1     1021 local VLE-VTEP    yes
leaf-5     1021 local VLE-VTEP    yes
```

Step 3 involves the creation of the *VXLAN tunnels*:

```
CLI (network-admin@leaf-1)> tunnel-create name VLE_L1_to_L5 scope local local-ip 10.21.1.1
remote-ip 10.21.7.1 vrouter-name Leaf-1

CLI (network-admin@leaf-5)> tunnel-create name VLE_L5_to_L1 scope local local-ip 10.21.7.1
remote-ip 10.21.1.1 vrouter-name Leaf-5
```

The tunnel creation can be checked with the following commands:

```
CLI (network-admin@leaf-1) > tunnel-show local-ip 10.21.1.1 format
switch,scope,name,type,vrouter-name,local-ip,remote-ip

switch     scope name         type  vrouter-name local-ip  remote-ip
---------- ----- ------------ ----- ------------ --------- ---------
leaf-1     local VLE_L1_to_L5 vxlan Leaf-1       10.21.1.1 10.21.7.1
```

```
CLI (network-admin@leaf-5) > tunnel-show local-ip 10.21.7.1 format
switch,scope,name,type,vrouter-name,local-ip,remote-ip

switch     scope name         type  vrouter-name local-ip  remote-ip
---------- ----- ------------ ----- ------------ --------- ---------
leaf-5     local VLE_L5_to_L1 vxlan Leaf-5       10.21.7.1 10.21.1.1
```

Step 4 requires the creation of a *transparent VLAN* per vLE:

```
CLI (network-admin@leaf-1)> vlan-create id 3001 scope local description VLE-1 vxlan-mode
transparent vxlan 3001000 ports 10
CLI (network-admin@leaf-5)> vlan-create id 3001 scope local description VLE-1 vxlan-mode
transparent vxlan 3001000 ports 17
```

The VLAN's creation can be checked with the following command:

```
CLI (network-admin@leaf-1)> vlan-show id 3001 format
switch,id,type,vxlan,scope,description,active,ports

switch     id   type   vxlan   scope description active ports
---------- ---- ------ ------- ----- ----------- ------ -----
leaf-1     3001 public 3001000 local VLE-1       yes    10
leaf-5     3001 public 3001000 local VLE-1       yes    17
```

Finally, in *step 5* the above VXLAN ID(s) of the transparent VLAN(s) are mapped to the tunnel(s):

```
CLI (network-admin@leaf-1)> tunnel-vxlan-add name VLE_L1_to_L5 vxlan 3001000
CLI (network-admin@leaf-5)> tunnel-vxlan-add name VLE_L5_to_L1 vxlan 3001000
```

The VXLAN tunnels and their mappings can be checked with the following command:

```
CLI (network-admin@leaf-1)> tunnel-vxlan-show vxlan 3001000

switch     name         vxlan
---------- ------------ -------
leaf-1     VLE_L1_to_L5 3001000
leaf-5       VLE_L5_to_L1 3001000
```

Note: Starting from NetVisor OS release 6.1.1, to prevent the misconfiguration of a vLE port which may get assigned to another VLAN by mistake, a new configuration check was added. The restriction is that a port assigned to a regular VLAN cannot be assigned to a vLE VLAN (i.e., in transparent mode) and vice versa.

Let's suppose VLAN 1 (the default VLAN) and VLAN 100 are assigned to port 1, as shown below:

```
CLI (network-admin@switch) > port-vlan-show ports 1

switch port vlans untagged-vlan description active-vlans
------ ---- ----- ------------ ----------- ------------
switch 1    1,100 1                        none
```

An attempt to assign this port to a vLE VLAN (VLAN 555 in transparent mode) is rejected:

```
CLI (network-admin@switch) > vlan-create id 555 scope local ports 1 vxlan-mode transparent
vxlan 666666
vlan-create: vle port 1 cannot be member of other vlans
```

If you want to unassign the port and then reassign it to a vLE VLAN, you can go through a two-step process. First, remove the port from the specific VLAN (VLAN 100 in the above example, or from `vlans all`):

```
CLI (network-admin@switch) > port-vlan-remove port 1 vlans 100

CLI (network-admin@switch) > port-vlan-show ports 1
switch port vlans untagged-vlan description active-vlans
------ ---- ----- ------------ ----------- ------------
switch 1    1     1                        none
```

Then you can assign it to a vLE VLAN like so:

```
CLI (network-admin@switch) > vlan-create id 555 scope local ports 1 vxlan-mode transparent
vxlan 666666
Vlans 555 created

CLI (network-admin@switch) > port-vlan-show ports 1
switch port vlans untagged-vlan description active-vlans
------ ---- ----- ------------ ----------- ------------
switch 1    555   555                      none
```

Configuring Virtual Link Extension State Tracking

In addition to the above configuration steps, it is often common to also turn on state tracking in order to make sure that link state changes are mirrored:

```
CLI (network-admin@leaf-1)> vle-create name vLE-1 node-1 leaf-1 node-1-port
10 node-2 leaf-5 node-2-port 17 tracking
```

This configuration step requires specifying the two vLE nodes (leaf-1 and leaf-5) as well as their respective ports in addition to the tracking option by following this syntax:

```
CLI (network-admin@leaf-1)> vle-create
```

`vle-create`	Create vLE configuration.
`name name-string`	Specify the vLE name.
`vnet` *`vnet-name`*	Specify the vNET assigned to this vLE.
`node-1` *`fabric-node name`*	Specify the vLE node 1 name.
`node-2` *`fabric-node name`*	Specify the vLE node 2 name.
`node-1-port` *`node-1-port-number`*	Specify the vLE node 1 port.
`node-2-port` *`node-2-port-number`*	Specify the vLE node 2 port.
`node-1-port-desc` *`node-1-port-desc-string`*	Specify a description for the node 1 port.
`node-2-port-desc` *`node-2-port-desc-string`*	Specify a description for the node 2 port.
`[tracking|no tracking]`	Enable or disable tracking between vLE ports.
`ports-state override|enable|disable`	Specify the ports admin control state.

The link state tracking configuration can be verified with the `vle-show` command:

```
CLI (network-admin@leaf-1) > vle-show layout vertical
name:              vLE-1
vnet:
node-1:            leaf-1
node-2:            leaf-5
node-1-port:       10
node-2-port:       17
node-1-port-desc:
node-2-port-desc:
status:            up
tracking:          enabled
ports-state:       override
create-time:       08:47:56
```

Furthermore, since a vLE transparently transmits PDUs over to the other end, it is possible to test state tracking by leveraging for example the LLDP protocol messages exchanged between two test switches

(sw-1 and sw-2 in the figure above) connected to the vLE ports 10 and 17.

The two test switches exchange LLDP messages by default, so the LLDP peer relationship can just be shown with the command:

```
CLI (network-admin@sw-1) > lldp-show

switch local-port bezel-port chassis-id port-id port-desc          sys-name
------ ---------- ---------- ---------- ------- ----------------- --------
sw-1   1          1          0b0012c0   1       PN Switch Port(1) sw-2
```

Then, if a vLE port is disabled (in this case, port 10), the peer relationship is lost and `lldp-show` does not display a neighbor anymore. The test switch's port also goes down thanks to vLE state tracking. Once the port is re-enabled, the neighboring relationship is re-established:

```
CLI (network-admin@leaf-5) > port-config-modify port 17 disable

CLI (network-admin@sw-1) > port-phy-show port 1

switch port bezel-port state speed eth-mode    max-frame learning def-vlan
------ ---- ---------- ----- ----- ---------- --------- -------- --------
sw-1   1    1          down  10000 10Gbase-cr 1540      off      0

CLI (network-admin@sw-1) > lldp-show

CLI (network-admin@leaf-5) > port-config-modify port 17 enable

CLI (network-admin@leaf-5) > port-phy-show port 17

port bezel-port state speed eth-mode max-frame learning def-vlan
---- ---------- ----- ----- -------- --------- -------- --------
17   17         up    10000 10G-SFI   1540       off      0

CLI (network-admin@sw-1) > port-phy-show port 1

switch   port bezel-port state speed eth-mode    max-frame learning def-vlan
-------- ---- ---------- ----- ----- ---------- --------- -------- --------
sw-1     1    1          up    10000 10Gbase-cr  1540      off      1

CLI (network-admin@sw-1) > lldp-show

switch  local-port bezel-port chassis-id port-id port-desc         sys-name
------- ---------- ---------- ---------- ------- ----------------- --------
sw-1    1          1          0b0012c0   1       PN Switch Port(1) sw-2
```

To enable or disable tracking between existing vLE ports, use the `vle-modify` command:

```
CLI (network-admin@switch) > vle-modify name name-string tracking|no
tracking
```

`vle-modify`	Modify virtual link extension tracking
`name name-string`	The vLE name to be modified
`tracking|no tracking`	Enable or disable tracking between vLE ports

To delete a state tracking configuration, use the vle-delete command:

```
CLI (network-admin@switch) > vle-delete name name-string
```

Configuring Virtual Link Extension in vNETs

Starting from NetVisor OS version 6.1.0, you can configure a vLE on vNET managed ports. This enhancement enables you to separate the management of the feature on a per tenant basis.

To configure a vLE in a vNET, follow the steps from step 1 to step 4 discussed in the Configuring Virtual Link Extension section. You can then use the vle-create command to configure the vNET and enable state tracking. For example, as an extension of step 4, you can assign the vNET vnet1 to the vLE vLE-1 and enable state tracking by using the command:

```
CLI (network-admin@switch) > vle-create name vLE-1 vnet vnet1 node-1 leaf-1
node-1-port 10 node-2 leaf2 node-2-port 17 tracking
```

Note: vLE creation is not allowed on vNET shared ports.

Use the vle-show command to display the configuration.

Use the `vle-show` command to display the configuration.

```
CLI (network-admin@switch) > vle-show
```

`vle-show`	Display vLE configuration.
`name name-string`	The vLE name.
`id`	The ID assigned to the vLE.
`vnet` *`vnet-name`*	The vNET assigned to this vLE.
`node-1` *`fabric-node-name`*	vLE node 1 name.
`node-2` *`fabric-node-name`*	vLE node 2 name.
`node-1-port` *`node-1-port-number`*	vLE node 1 port.
`node-2-port` *`node-2-port-number`*	vLE node 2 port.
`node-1-port-desc` *`node-1-port-desc-string`*	Description for the node 1 port.
`node-2-port-desc` *`node-2-port-desc-string`*	Description for the node 2 port.
`status unknown|remote-down|local-down|up|down|disabled`	The vLE status.
`[tracking|no tracking]`	Enable or disable tracking between vLE ports.
`ports-state override|enable|disable`	Ports admin control state.
`create-time` *`date/time: yyyy-mm-ddTHH:mm:ss`*	Time at which the vLE was created.
`elapsed-time` *`duration: #d#h#m#s`*	Time duration since the vLE was created.

up-time *duration: #d#h#m#s*	Time since the last instance at which the vLE came online.

```
CLI (network-admin@leaf-1) > vle-show format all layout vertical
name:             vle1
id:               900ad7:0
vnet:             vnet1
node-1:           leaf-1
node-2:           leaf-2
node-1-port:      10
node-2-port:      17
node-1-port-desc:
node-2-port-desc:
status:           up
tracking:         enabled
ports-state:      override
create-time:      09:44:56
elapsed-time:     11m37s
up-time:          0s
```

Configuring Virtual Link Extension Error Transparency

In order to be able to preserve bad CRC and undersized packets as part of the vLE transport, a new fabric-wide *error transparency* configuration option has been added:

```
CLI (network-admin@leaf-1) > vle-transparent-modify mode enable

CLI (network-admin@leaf-1) > vle-transparent-show
fabric: Transparent VLE mode is enabled

CLI (network-admin@leaf-1) > vle-transparent-modify mode disable

CLI (network-admin@leaf-1) > vle-transparent-show
fabric: Transparent VLE mode is disabled
```

However, since disabling CRC checks is not necessarily desirable on all links, some ports can be excluded (for example certain edge ports and inter-switch links) with a local scope command:

```
CLI (network-admin@leaf-1) > vle-transparent-modify mode enable

CLI (network-admin@leaf-1) > vle-transparent-port-remove port 18,45

CLI (network-admin@leaf-1) > vle-transparent-port-show
switch: leaf-1
leaf-1: ports removed from VLE transparent mode: 18,45

CLI (network-admin@leaf-1) > vle-transparent-port-add port 18,45

CLI (network-admin@leaf-1) > vle-transparent-port-show
switch: leaf-1
leaf-1: ports removed from VLE transparent mode: none
```

> **Informational note**: Transparency exclusion should be applied to ports that are not in the end-to-end vLE transport path through the fabric, which needs to be based on supported switches that are capable of error transparency to preserve bad frames.

Configuring Virtual Link Extension Redundancy with Clusters

The configuration of redundant vLE pseudo-wires requires the replication of the configuration steps described in the above sections. In addition, the endpoints (for example two switches as in the figure below) need to have link bundling enabled (static or dynamic LACP-based negotiation) in order to manage end-to-end path redundancy.

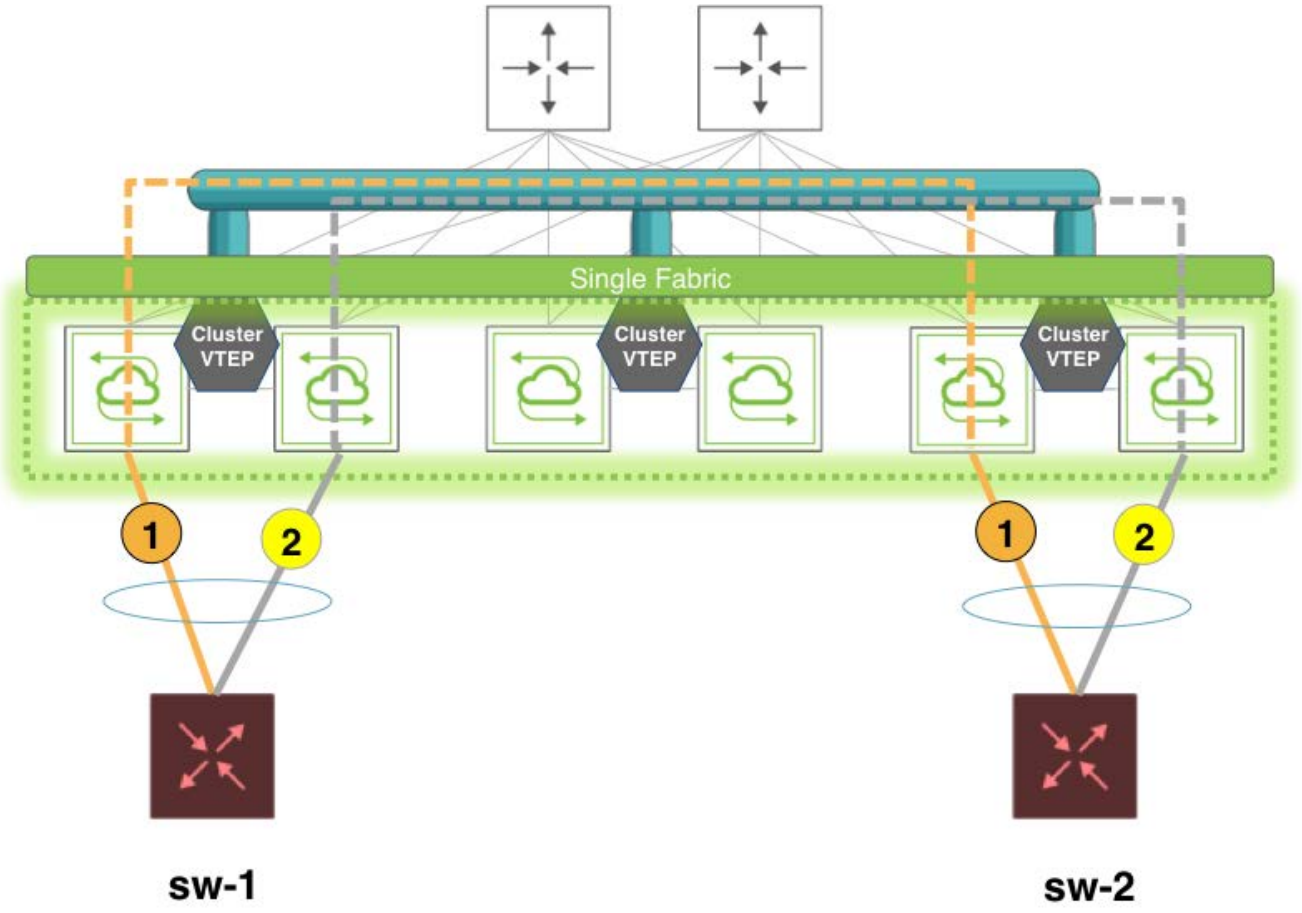


Figure 9-5: Virtual Link Extension End-to-end Redundancy

In this example, let us assume that both test switches, `sw-1` and `sw-2`, are configured with the LACP active negotiation option.

Then, every vLE configuration step needs to be replicated twice to create VLE-1 and VLE-2 (based on two transparent VLANs and four VTEPs):

1. Creation of Layer 3 VLANs that are used as tunnel end points
2. Set up of dedicated VXLAN connections
3. Creation of VXLAN-mapped *transparent VLANs*
4. Addition of the transparent VLAN's VXLAN ID to the allocated tunnels.

The distinctive configuration step is the creation of the VTEPs (step 2 above): VTEP HA is not required and not supported in case of vLE redundancy (namely, the virtual IPs cannot be used). The four VTEPs can be created instead with each cluster member's primary IP address like so:

```
CLI (network-admin@leaf-1) > tunnel-create scope local name VTEP1 vrouter-name vr-s1 local-ip
10.10.11.1 remote-ip 10.10.15.1
```

```
CLI (network-admin@leaf-2) > tunnel-create scope local name VTEP2 vrouter-name vr-s2 local-ip
20.20.22.1 remote-ip 20.20.26.1

CLI (network-admin@leaf-5) > tunnel-create scope local name VTEP3 vrouter-name vr-s3 local-ip
10.10.15.1 remote-ip 10.10.11.1

CLI (network-admin@leaf-6) > tunnel-create scope local name VTEP4 vrouter-name vr-s3 local-ip
20.20.26.1 remote-ip 20.20.22.1
```

Note that `10.10.11.1` and `10.10.15.1` are primary IP addresses used for VLE-1 on leaf-1 and leaf-5 respectively while `20.20.22.1` and `20.20.26.1` are primary IP addresses used for VLE-2 on leaf-2 and leaf-6.
The remaining four steps are the same as those described in the previous sections, replicated for both VLE-1 and VLE-2.

vLE tracking is also greatly beneficial in order to implement redundancy on NetVisor OS nodes as it propagates link state changes and helps client side's LACP with timely link up/link down detection.

Showing Virtual Link Extension Between Ports on the Same Switch

In certain network configurations, both vLE ports should be part of the same node. This case is supported by NetVisor OS and is referred to as *local vLE*.

In the example below `node-1` and `node-2` refer to the same switch in the local vLE, in which port 30 is displayed as remote port in the `vle-show` output, even though both the vLE ports are on the same node.

Port 30 went down, therefore the vLE status is shown as `remote-down` (see command output below) so you can understand exactly which port caused the vLE to be down:

```
CLI (network-admin@switch) > vle-show layout vertical
name:         vle1
vnet:
node-1:       vanquish1
node-2:       vanquish1
node-1-port:  22
node-2-port:  30
status:       remote-down
tracking:     enabled
ports-state:  override
create-time:  09:44:56
```

You can use the `port-show` command to check if the `node-1-port` status is `down` or `vle-wait`. The `vle-wait` state indicates that the port was brought down due to the remote port being down.

Configuring Keep-Alive Timeout for Virtual Link Extension

As part of the vLE link state tracking logic each node hosting a vLE endpoint sends *fabric fast keepalive messages* every second to its peer node that hosts the other endpoint.

A node considers its peer node down if a keepalive is not received from the remote node within 3 seconds (which is the default *vLE tracking timeout*). If that happens, the local port is brought down and the vLE state reflects both ports' down state.

However, in some deployments (for example with frequent link flaps), a three second timeout may not be enough to avoid false positives. Therefore, it can be modified with the following command:

```
CLI (network-admin@switch) > system-settings-modify vle-tracking-timeout seconds
```

You can configure a value *from 3 to 30 seconds*. (Note that the fast keepalive transmission frequency remains one second. Only the timeout value is adjusted to the configured value.) The configured value can be checked with the `system-settings-show` command.

> **Note:** The default 3 second keepalive timeout is automatically configured when vLE link state tracking is enabled. When it is disabled, the standard fabric message timeout interval is 21 seconds as shown for the three leaf switches below that don't use tracking:

```
CLI (network-admin@switch) > fabric-node-show format name,keepalive-timeout

name           keepalive-timeout
-------------- -----------------
leaf-1         21
leaf-2         21
leaf-3         21
switch         3
```

```
CLI (network-admin@switch) > system-settings-show format vle-tracking-timeout
vle-tracking-timeout: 3
```

When tracking gets disabled, the fabric keepalive timeout goes back to 21 seconds:

```
CLI (network-admin@switch) > vle-modify name vLE no-tracking

CLI (network-admin@switch) > fabric-node-show format name,keepalive-timeout

name           keepalive-timeout
-------------- -----------------
leaf-1         21
leaf-2         21
leaf-3         21
switch         21

CLI (network-admin@switch) > system-settings-show format vle-tracking-timeout,
vle-tracking-timeout: 3
```

When it is necessary to change the administrative vLE keepalive timeout value, the fabric keepalive

timeout reflects the new value when tracking is enabled, as shown below:

```
CLI (network-admin@switch) > system-settings-modify vle-tracking-timeout 30

CLI (network-admin@switch) > fabric-node-show format name,keepalive-timeout

name           keepalive-timeout
-------------- -----------------
leaf-1         21
leaf-2         21
leaf-3         21
switch         30
```

Enabling and Disabling a Virtual Link Extension End-to-End

At times, whether in the CLI or in the Arista NetVisor UNUM GUI, it can be convenient to manage vLE ports in *pairs* as a single entity (instead of doing so individually). Starting from NetVisor OS release 6.0.0, a new end-to-end vLE configuration option can be used to enable or disable both vLE ports at the same time as part of the `vle-create` command.

The new `ports-state` option can have three values:

- `override` is the default setting which means that, as usual, each individual port state configuration takes precedence
- `disable` means that both vLE ports get disabled together and this option takes precedence over each individual port state
- `enable` means that both vLE ports get enabled and this option takes precedence over each individual port state.

For example:

```
CLI (network-admin@swi01) > vle-create name vle-test node-1 swi01 node-2
swi02 node-1-port 13 node-2-port 13 no-tracking ports-state disable
```

> Note: If `node-1` and `node-2` are the same switch, then it becomes a local vLE configuration.

The `vle-modify` command can be used to further change the end-to-end vLE configuration after the initial `vle-create`.

For example, after setting `ports-state` to disable in `vle-create`, the user can revert to the default behavior in which local port configuration takes prececence with the following command:

```
CLI (network-admin@swi01) > vle-modify name vle-test ports-state override
```

In case of responsibility conflict in the CLI a message is printed out to explain what is taking precedence and to suggest a possible course of action.

For example, when a user tries to disable a vLE port when `ports-state` is set to `enable`, the following message is printed:

```
CLI (network-admin@swi01) > vle-show layout vertical
name:          vle-test
vnet:
node-1:        swi01
node-2:        swi02
node-1-port:   13
node-2-port:   13
status:        up
tracking:      enabled
ports-state:   enable
create-time:   08:47:56

CLI (network-admin@swi01) > port-config-modify port 13 disable
port-config-modify: could not disable port 13 as it is VLE vle-test administratively enabled
port. Use the following cli to override: vle-modify name vle-test ports-state override
```

Similarly, when a user tries to enable a vLE port when `ports-state` is `disable`:

```
CLI (network-admin@swi01) > vle-show layout vertical
name:          vle-test
vnet:
node-1:        swi01
node-2:        swi02
node-1-port:   13
node-2-port:   13
status:        disabled
tracking:      enabled
ports-state:   disable
create-time:   08:47:56

CLI (network-admin@swi01) > port-config-modify port 13 enable
port-config-modify: could not enable port 13 as it is VLE vle-test
administratively disabled port. Use the following cli to override: vle-
modify name vle-test ports-state override
```

Even when the user configuration intent is the same (both in the local configuration and the vLE configuration), but there is still a responsibility conflict, a message is also printed out. For example, when a user tries to disable a vLE port when `ports-state` is `disable`:

```
CLI (network-admin@swi01) > vle-show layout vertical
name:         vle-test
vnet:
node-1:       swi01
node-2:       swi02
node-1-port:  13
node-2-port:  13
status:       disabled
tracking:     enabled
ports-state:  disable
create-time:  08:47:56

CLI (network-admin@swi01) > port-config-modify port 13 disable
port-config-modify: could not disable port 13 as it is VLE vle-test administratively disabled
port. Use the following cli to override: vle-modify name vle-test ports-state override
```

Or when a user tries to enable a vLE port when `ports-state` is `enable`:

```
CLI (network-admin@swi01) > vle-show layout vertical
name:         vle-test
vnet:
node-1:       swi01
node-2:       swi02
node-1-port:  13
node-2-port:  13
status:       up
tracking:     enabled
ports-state:  enable
create-time:  08:47:56

CLI (network-admin@swi01) > port-config-modify port 13 enable
port-config-modify: could not enable port 13 as it is VLE vle-test administratively enabled
port. Use the following cli to override: vle-modify name vle-test ports-state override
```

Saving Virtual Link Extension Topology Configurations

In lab automation deployments in which Arista Networks switches are used with the vLE feature, users require easy repeatability when the same topology is to be deployed. For example, there is a need to be able to configure different vLE connections and then save them to be re-created when required.

Before NetVisor OS release 7.0.0, the software only supported saving and replaying individual configurations by using the `switch-config-export` and `switch-config-import` commands. However, this process is not optimal and convenient for vLE setups because it saves entire switch configurations and restarts the devices.

Starting from release 7.0.0, a new vLE topology feature can be used to automate the creation of vLEs, their associated transparent VLANs and the associated VXLAN IDs on the tunnels, in a convenient and user-friendly fashion. This new functionality introduces the `topology-vle-add/-remove/-modify/-show` commands to store and create the vLEs so that users can easily save and replay their topologies without having to restart the devices.

The feature's logic checks for the presence of the *static* VXLAN tunnel(s) that is/are needed between the vLE nodes: a warning is displayed in case they are not active, and the automation does not proceed. On the other hand, if the tunnel(s) required for vLE creation is/are present, then the automation process adds the required VXLAN ID to each tunnel and the vLE setup is created.

The feature's logic uses automatic assignment for parameters that are not specified and uses a dry run process to verify potential conflicts. The software checks:

1. Ports for conflicts during vLE creation
2. VLAN and VXLAN IDs to see if they are already in use
3. The tunnels required for the vLE functionality

Note: *Auto-tunnels* are not supported. Only static ones are.

Note: Only single fabric deployments are supported.

The feature includes the following commands:

- `topology-create name <name>` is used to create a named topology entity.

Example:

```
CLI (network-admin@switch) > topology-create name topo1
```

- `topology-delete name <name>` deletes the topology configuration.

Example:

```
CLI (network-admin@switch) > topology-delete name topo1
```

- `topology-show` displays all the configured topologies and their enabled state.

Example:

```
CLI (network-admin@switch) > topology-show
name  enable
----- ------
topo1 yes
topo2 no
```

- `topology-modify enable | disable` creates or deletes the vLEs.

`enable` creates the vLEs and the associated VLANs between the node ports
`disable` deletes the vLEs and the associated VLANs between the node ports

Example:

```
CLI (network-admin@leaf-1) > topology-modify name topo1 enable

CLI (network-admin@leaf-1) > vlan-show format switch,id,vxlan,auto-
vxlan,vxlanmode,description,active,ports,untagged-ports,active-edge-ports

switch id   vxlan  auto-vxlan vxlan-mode  description          active ports untagged-ports active-edge-ports
------ ---- ------ ---------- ----------- -------------------- ------ ----- -------------- -----------------
leaf-1 3001 300100 no         transparent topo-topo1-vlan-2750 yes    14    14             none
leaf-5 3001 300100 no         transparent topo-topo1-vlan-2750 yes    12    12             none

CLI (network-admin@leaf-1) > tunnel-vxlan-show

switch name         vxlan
------ ------------ ------
leaf-1 VLE_L5_to_L1 300100
leaf-5 VLE_L1_to_L5 300100

CLI (network-admin@leaf-1) > vle-show format name,node-1,node-2,node-1-port,node-2-
port,status,tracking,ports-state,topology

name      node-1 node-2 node-1-port node-2-port status tracking ports-state topology
--------- ------ ------ ----------- ----------- ------ -------- ----------- --------
topo1-vle leaf-1 leaf-5 10          17          up     enabled  override    topo1
```

- `topology-import` imports existing vLEs into a topology and stores the endpoints' information (switch name and port).

> **Note**: After importing an existing vLE connection into a topology, you must delete it before enabling the topology, because the topology setup process needs to recreate the vLE connection from scratch.

Example of successful imports:

```
CLI (network-admin@leaf-1) > vle-show format name,node-1,node-2,node-1-port,node-2-
port,status,tracking,ports-state,topology

name           node-1 node-2 node-1-port node-2-port status tracking ports-state topology
-------------- ------ ------ ----------- ----------- ------ -------- ----------- --------
DUT2-to-DUT3 leaf2  leaf3  11          19          up     enabled  override

CLI (network-admin@leaf-1) > topology-import name topo1 vle_name DUT2-to-DUT3
```

```
CLI (network-admin@leaf-1) > topology-vle-show

name  vle-name     node1 node2 node-1-port node-2-port vlan-id vxlan
----- ------------ ----- ----- ----------- ----------- ------- ---------
topo1 DUT2-to-DUT3 leaf2 leaf3 11          19          100     100000
```

Example of import with conflict when the topology is enabled:

```
CLI (network-admin@leaf-1) > topology-import name topo2 type vle vle_name vle1

CLI (network-admin@leaf-1) > topology-vle-show

name  vle-name node1 node2   node-1-port node-2-port vlan vxlan
----- -------- ------ ------ ----------- ----------- ---- --------
topo1 vle4     leaf-0 leaf-1 13          16          2752 12500003
topo1 vle2     leaf-0 leaf-1 12          15          2753 12500004
topo2 vle1     leaf-0 leaf-0 13          14          2754 12500005

CLI (network-admin@leaf-1) > topology-modify name topo2 enable
topology-modify: vle conflict, please delete vle vle1 or modify node-1-port of topology vle
vle1
```

- `topology-vle-add` specifies the vLE name to be added, the node and the node ports between which the vLE needs to be set up. VLAN ID and VXLAN ID are optional. IDs are taken from the auto VXLAN range when not specified.

Example:

```
CLI (network-admin@leaf-1)> tunnel-create name VLE_L1_to_L5 scope local local-ip 10.21.1.1
remote-ip 10.21.7.1 vrouter-name vr1

CLI (network-admin@leaf-5)> tunnel-create name VLE_L5_to_L1 scope local local-ip 10.21.7.1
remote-ip 10.21.1.1 vrouter-name vr2

CLI (network-admin@leaf-1) > tunnel-show local-ip 10.21.1.1 format
switch,scope,name,type,vrouter-name,local-ip,remote-ip

switch scope name         type  vrouter-name local-ip  remote-ip
------ ----- ------------ ----- ------------ --------- ---------
leaf-1 local VLE_L1_to_L5 vxlan Leaf-1       10.21.1.1 10.21.7.1

CLI (network-admin@leaf-5) > tunnel-show local-ip 10.21.7.1 format
switch,scope,name,type,vrouter-name,local-ip,remote-ip

switch scope name         type  vrouter-name local-ip  remote-ip
------ ----- ------------ ----- ------------ --------- ---------
leaf-5 local VLE_L5_to_L1 vxlan Leaf-5       10.21.7.1 10.21.1.1

CLI (network-admin@leaf-1) > topology-vle-add name topo1 vle-name DUT1-to-DUT5 node-1 leaf-1
node-1-port 10 node-2 leaf-5 node-2-port 17 [vlan 3001] [vxlan 300100]
```

- `topology-vle-remove` removes a VLE from a topology. To succeed, it requires the topology to be disabled first.

Example of successful removal:

```
CLI (network-admin@leaf-1) > topology-vle-show
```

```
name  vle-name node1 node2   node-1-port node-2-port vlan vxlan
----- -------- ------ ------ ----------- ----------- ---- --------
topo1 vle1     leaf-0 leaf-1 12          14          2750 12500000
topo1 vle4     leaf-0 leaf-1 13          16          2752 12500003

CLI (network-admin@leaf-1) > topology-vle-remove name topo1 vle-name vle1

CLI (network-admin@leaf-1) > topology-vle-show

name  vle-name node1 node2  node-1-port node-2-port vlan vxlan
----- -------- ----- ------ ----------- ----------- ---- --------
topo1 vle4     leaf-0 leaf-1 13         16          2752 12500003
```

Example of unsuccessful removal:

```
CLI (network-admin@leaf-1) > topology-show

name  enable
-----  ------
topo1 yes

CLI (network-admin@leaf-1) > topology-vle-remove name topo1 vle-name vle2
topology-vle-remove: topology topo1 is enabled, please disable it before
removing a vle connection
```

- `topology-vle-show` displays the list of vLEs that have been added, the corresponding node names and ports, as well as the VLAN IDs and VXLAN IDs.

Example:

```
CLI (network-admin@leaf-1) > topology-vle-show

name  vle-name     node1 node2 node-1-port node-2-port vlan vxlan
----- ------------ ----- ----- ----------- ----------- ---- ------
topo1 DUT1-to-DUT5 leaf1 leaf5 10          17          3001 300100
```

- `topology-vle-modify` modifies a vLE's parameters.

Example of node port modification:

```
CLI (network-admin@leaf-1) > topology-vle-show

name vle-name  node1 node2   node-1-port node-2-port vlan vxlan
----- -------- ------ ------ ----------- ----------- ---- --------
topo1 vle4     leaf-0 leaf-1 13          16          2752 12500003
topo1 vle2     leaf-0 leaf-1 12          15          2753 12500004

CLI (network-admin@leaf-1) > topology-vle-modify name topo1 vle-name vle2 node-2-port 17

CLI (network-admin@leaf-1) > topology-vle-show

name  vle-name node1  node2  node-1-port node-2-port vlan vxlan
----- -------- ------ ------ ----------- ----------- ---- --------
topo1 vle4     leaf-0 leaf-1 13          17          2752 12500003
topo1 vle2     leaf-0 leaf-1 12          15          2753 12500004
```

Example of VLAN modification:

```
CLI (network-admin@leaf-1) > topology-vle-show

name  vle-name node1  node2 node-1-port node-2-port  vlan vxlan
----- -------- ------ ------ ----------- ----------- ---- --------
topo1 vle4     leaf-0 leaf-1 13          16          2752 12500003
topo1 vle2     leaf-0 leaf-1 12          15          2753 12500004

CLI (network-admin@leaf-1) > topology-vle-modify name topo1 vle-name vle2 vlan 1234

CLI (network-admin@leaf-1) > topology-vle-show

name  vle-name node1  node2 node-1-port node-2-port  vlan vxlan
----- -------- ------ ------ ----------- ----------- ---- --------
topo1 vle4     leaf-0 leaf-1 13          17          2752 12500003
topo1 vle2     leaf-0 leaf-1 12          15          1234 12500004
```

The feature's dry run logic can catch conflicts when they occur and notify the user, as shown in the examples below:

```
CLI (network-admin@leaf-1) > topology-modify name topo2 enable
topology-modify: vle conflict, please disable topology topo1 or modify node-1-port of topology
vle vle1
```

In this case, when topo2 is applied, it conflicts with the vLE of another topology.

```
CLI (network-admin@leaf-1) > topology-modify name topo2 enable
topology-modify: vle conflict, please delete vle vle1 or modify node-1-port of topology vle
vle1
```

In the case above there is a conflict with an existing vLE, which needs to be deleted first.

```
CLI (network-admin@leaf-1) > tunnel-delete name VLE_L5_to_L1

CLI (network-admin@leaf-1) > topology-modify name topo1 enable
topology-modify: tunnels do not exist between the switches leaf-0 and leaf-1 for vle vle1
creation or tunnels between these switches have vxlan of standard mode.Please create new
tunnels as per vle config docs
```

In the case above the required static VXLAN tunnels don't exist.

```
CLI (network-admin@leaf3*) > vlan-show ports 12 format id,vxlan,autovxlan,
scope,description,active,ports

id  vxlan    auto-vxlan scope description  active ports
--- -------- ---------- ------ ----------- ------ -------------------------
100 12500000 yes        fabric vlan-100    yes    0-12,14-48,50-52,54-72,397

CLI (network-admin@leaf3) > topology-vle-show

name  vle-name node1 node2 node-1-port node-2-port vlan vxlan
----- -------- ----- ----- ----------- ----------- ---- -----
topo1 vle1     leaf3 leaf4 12          13          2625 12345

CLI (network-admin@leaf3*) > topology-modify name topo1 enable
topology-modify: node_1_port 12 of topology vle vle1 has vlan 100 already created, please
delete it or use a different port using topology-vle-modify
```

In the case above a VLAN ID is already present on a port when the vLE creation takes place.

```
CLI (network-admin@leaf3*) > vlan-show vxlan 12345

id   type   vxlan auto-vxlan vxlan-hybrid-mode replicators scope description active
---- ------ ----- ---------- ----------------- ----------- ----- ----------- ------
1234 public 12345 no         standard          none        local vlan-1234   yes

CLI (network-admin@leaf3*) > topology-modify name topo1 enable
topology-modify: vxlan 12345 is in use, please use a different vxlan for vle vle1 using
topology-vle-modify
```

In the case above a VXLAN ID to be added to a tunnel is already active.

```
CLI (network-admin@leaf3*) > vpg-show

scope  name type  ports
------ ---- ----- -----
fabric vpg2 source 12

CLI (network-admin@leaf3*) > topology-vle-show

name  vle-name node1 node2 node-1-port node-2-port vlan vxlan
----- -------- ----- ----- ----------- ----------- ---- -----
topo1 vle1     leaf3 leaf4 12          13          2625 12345

CLI (network-admin@leaf3*) > topology-modify name topo1 enable
topology-modify: vle vle1 of topology has port conflict. Port 12 is already configured for vpg
```

In this case, a port that is needed for vLE creation is already in use by a vPG.

Configuring VXLAN-based Bridge Domains for 802.1Q and QinQ Internetworking

Bridge domains' configuration process is based upon the deployment of a VXLAN transport as described in the *Configuring VXLAN* chapter of this guide.

Subsequently, switch ports can be selectively added to a BD to accept either 802.1Q frames or double-tagged 802.1ad frames based on the connected device(s) and/or network(s), as depicted in the ***Figure 9-6***.

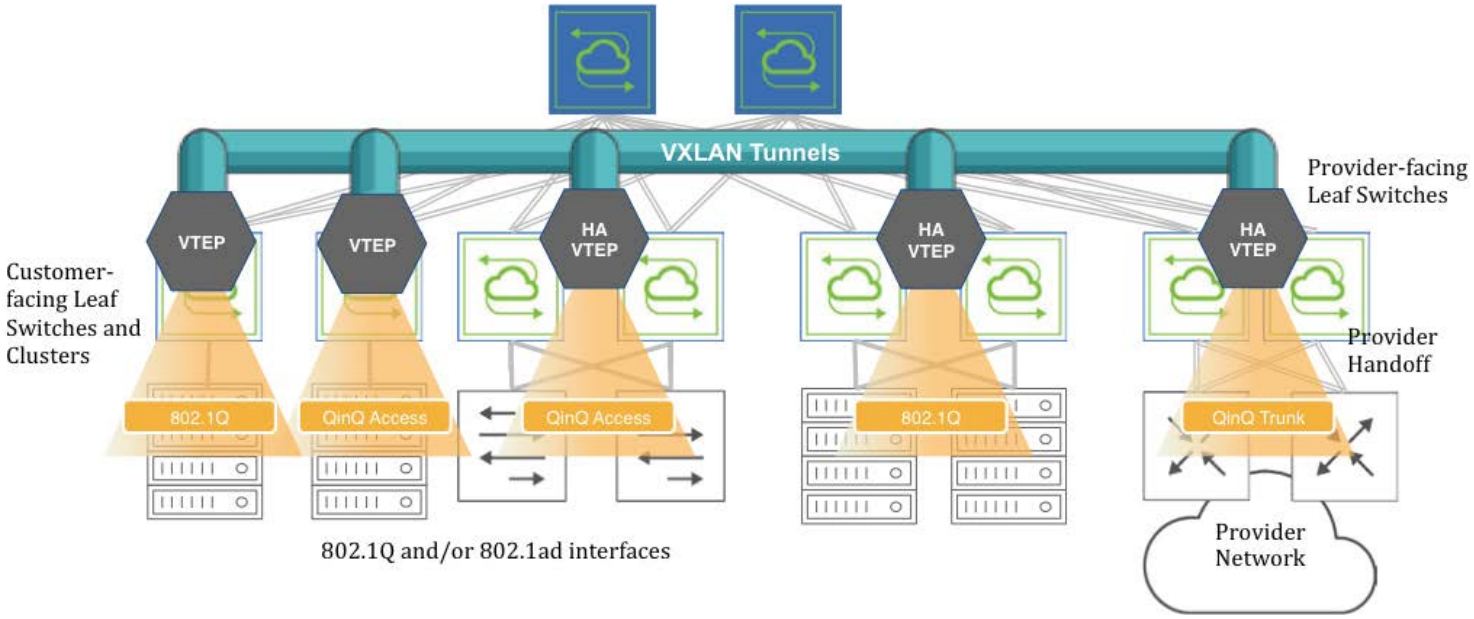


Figure 9-6: VXLAN-based BD Internetworking

Switch ports can be connected to either servers/hosts or switches/provider gateways as shown above.

In case of interconnection with an external cloud provider's gateway devices, as shown on the right hand side of ***Figure 9-6***, the connectivity is based on double-tagged 802.1ad frames, as it is required to preserve all 16M (4094 x 4094) possible combinations of user IDs and service IDs.

On the other hand, connectivity to servers can be based on a single 802.1Q VLAN tag. This case is useful in private cloud networks for direct server-to-server connectivity (for example a Web services node that needs to communicate with a database node).

Alternatively, server NICs can be configured with both an outer VLAN and an inner VLAN ID. In such case, the inner VLAN represents for example the service implemented on such server. Certain network designs employ this double-tagging scheme to be able to scale the number of supported services to 16M. In this case, switch ports have to be configured in 802.1ad mode to preserve the inner tag.

The 802.1Q ports can also be used to connect to customer switches and traffic can be double tagged by the fabric leaf nodes themselves.

Note: BD ports can be configured in 802.1Q mode or in 802.1ad/QinQ mode but not in both on the same switch for the same BD. Starting from NetVisor OS release 6.1.1, in `remove-tags` mode, this restriction

has been lifted (see the following sections for more details).

Note: Starting from NetVisor OS release 7.0.1, it is possible to configure BD ports in 802.1Q mode and in 802.1ad/QinQ mode on the same switch.

Before adding ports to any of the aforementioned scenarios, the first basic configuration step is the creation of a VXLAN-based bridge domain entity with the command:

```
CLI (network-admin@leaf-5) > bridge-domain-create
```

`bridge-domain-create`	Create a bridge domain.
`name` *`name-string`*	Specify the name for the bridge domain.
`scope` *`local`*`|`*`cluster`*`|`*`fabric`*	Specify the bridge domain scope.
Specify any of the following options:	
`vnet` *`vnet-name`*	Specify the vNET for this bridge domain.
`vxlan` *`0..16777215`*	Specify the VXLAN identifier for the tunnel.
`auto-vxlan|no-auto-vxlan`	Specify the options to automatically assign VXLAN and/or assign to VTEPs.
`description` *`description-string`*	Add a bridge domain description.
`rsvd-vlan` *`1..4093`*	Specify the fabric reserved VLAN for cluster switches for bridge domain.
`local-rsvd-vlan` *`1..4093`*	Specify the local reserved VLAN for cluster switches for bridge domain.
`mac-learning|no-mac-learning`	Specify if you want to enable or disable MAC address learning.

```
CLI (network-admin@leaf-5) > bridge-domain-create name bd1000 scope fabric
vxlan 10000 rsvd-vlan 4002
```

A bridge domain's scope can be `fabric`, `cluster`, or `local`.

The `bridge-domain-create` command uses a manually assigned VNI to uniquely identify the bridge domain at the hardware level.
Moreover, with `fabric` and `cluster` scope, it must set aside a VLAN ID as reserved for supporting *switch clusters*:

```
CLI (network-admin@leaf-5) > bridge-domain-create name bd1000 scope fabric
vxlan 10000 rsvd-vlan 4002
```

This command reserves VLAN 4002 on all *cluster switches* in the fabric for BD's internal use, so it can no longer be configured as a regular VLAN number. (The same requirement applies to `bridge-domain-create scope cluster` commands).

If for a particular pair of cluster switches the reserved VLAN has to be modified, on any one of the two switches you can use the following command:

```
CLI (network-admin@leaf-5) > bridge-domain-modify name bd1000 local-rsvd-vlan 4006
```

Note that the above command requires both cluster switches to be rebooted to take effect. Before rebooting, both VLAN 4002 and 4006 are no longer available but only 4006 is reserved on the cluster switches after reboot.

Also note that in a cluster both switch members are required to use the same mode for a port, either QinQ or 802.1Q, for the same BD.

Bridge domains can also be created with a textual description to provide more context like so:

```
CLI (network-admin@leaf-5) > bridge-domain-create name bd1000 scope fabric vxlan 10000 description business rsvd-vlan 4002

CLI (network-admin@leaf-5) > bridge-domain-show

name   scope  vxlan auto-vxlan description ports cluster-name
------ ------ ----- ---------- ----------- ---- ------------
bd1000 fabric 10000 no         business    13
```

Starting from NetVisor OS release 6.0.0 the `auto-vxlan` parameter is supported as an option of the BD creation process.

This new parameter can be used either in combination with an explicit VNI value or implicitly without specifying it. In both cases any created BD or VNI mapping is automatically added to all the existing VTEP connections. Additionally, in the latter case, the VNI value is automatically picked and assigned out of a predefined range (see also the examples below) when `scope fabric` is used.

The next configuration step is to set up the fabric overlay either with manual tunnel creation or with VTEP object configuration. The latter way is *preferred*.

In order to manually create a VXLAN tunnel, say, on cluster member `leaf-5` in the ***Figure 9-6***, the following commands can be used:

```
CLI (network-admin@leaf-5) > tunnel-create scope cluster name tunnel-5 vrouter-name vr5 peer-vrouter-name vr1 local-ip 20.20.25.1 remote-ip 20.20.27.1

CLI (network-admin@leaf-5) > trunk-modify name vxlan-loopback-trunk ports 17 jumbo
```

The same steps are required on the other leaf switches to deploy an overlay comprising a full mesh of VXLAN-based interconnections.

Alternatively, VTEP objects can be configured so that a full mesh of VXLAN interconnections is brought up automatically without requiring manual configuration.

This is the command required to create a VTEP object on a non-clustered switch:

```
CLI (network-admin@leaf-1) > vtep-create name vtep1 vrouter-name vr1 ip
20.20.25.1
```

which can be used to create each fabric termination point (vtep1, vtep2, etc. on each leaf switch).

In case of clustered switches using a common VIP, the following command pairs should be used:

```
CLI (network-admin@leaf-5) > vtep-create name vtep5 vrouter-name vr5 ip
103.103.103.250  virtual-ip  103.103.103.10

CLI (network-admin@leaf-6) > vtep-create name vtep6 vrouter-name vr6 ip
103.103.103.251  virtual-ip  103.103.103.10
```

The next configuration step is to associate the BD's VXLAN ID (e.g., `10000`) to the newly created tunnel(s) or VTEP objects. For example, this command implements the association with a tunnel:

```
CLI (network-admin@leaf-1) > tunnel-vxlan-add name tunnel-1 vxlan 10000
```

whereas this command implements the association with a VTEP object:

```
CLI (network-admin@leaf-1) > vtep-vxlan-add name vtep1 vxlan 10000
```

As mentioned above, starting from NetVisor OS release 6.0.0, as an alternative to the manual assignment, you can automatically assign a certain user-defined BD or VNI mapping to *all tunnels* with the following command:

```
CLI (network-admin@leaf-5) > bridge-domain-create name bd5001234 scope
fabric vxlan 5001234 auto-vxlan
```

Instead, if you want the software to automatically pick and assign the VNI, you can simply enter the following shorter command that requires `scope` to be `fabric`:

```
CLI (network-admin@leaf-5) > bridge-domain-create name BD-VNI-SELF scope
fabric auto-vxlan
```

The range used for automatic VNI assignment can be controlled and modified, if needed, with the `vtep-auto-vxlan-show` and `vtep-auto-vxlan-modify` commands.

Once the VXLAN overlay deployment is complete and the BD's VXLAN ID is (manually or automatically) associated to it, then it is possible to start adding switch ports.

> **Informational note**: Before NetVisor OS release 6.1.1., when a port is assigned to a BD (as in the examples below) it cannot be associated with a regular VLAN configured with the `vlan-create` command, and vice versa. The restriction has been lifted in release 6.1.1.

In order to add a port connected to the cloud provider network to the BD, you can use the following command that explicitly maps both inner and outer VLANs to the BD:

```
CLI (network-admin@leaf-7) > bridge-domain-port-add name bd1000 port 13
outer-vlan 13 inner-vlan 100
```

This configuration also means that a frame's MAC lookup in the Layer 2 table is performed including both

the outer VLAN and the inner VLAN.

The same configuration can be used to add a port connected to an 802.1ad-configured server to the BD:

```
CLI (network-admin@leaf-2) > bridge-domain-port-add name bd1000 port 11
outer-vlan 13 inner-vlan 100
```

In order to add a port connected to an 802.1Q network to the BD, you can use the following command that explicitly maps only the outer VLAN to the BD (and preserves the inner VLAN):

```
CLI (network-admin@leaf-2) > bridge-domain-port-add name bd1000 port 15
outer-vlan 19
```

This configuration also means that a frame's MAC lookup in the Layer 2 table is performed including only the outer VLAN while the inner VLAN is ignored.

In order to add a port connected to an 802.1Q server to the BD, you can use the following command that explicitly associates one or more VLANs on the same switch port to the same BD:

```
CLI (network-admin@leaf-1) > bridge-domain-port-add name bd2100 port 10
vlans 210,310
```

> **Note**: The multiple VLAN IDs on the same port can be part of the same bridge domain. However, VLAN IDs added to a BD cannot be reused with another BD on the same port. Also note that MAC addresses associated with these two VLAN IDs must be unique within the BD.

Finally, you should also configure all cluster ports/trunks and ports in QinQ mode to work with the 802.1ad TPID (i.e., 0x88A8) with the following command:

```
CLI (network-admin@switch) > port-config-modify port <port_number> allowed-
tpid q-in-q
```

Port removal from a BD and BD deletion follow the reverse order compared to the addition process described in the above steps.
Ports have to be removed first with the command:

```
CLI (network-admin@leaf-2) > bridge-domain-port-remove name bd1000 port 15
```

Then a BD's VXLAN ID may be manually removed from the associated tunnel(s):

```
CLI (network-admin@leaf-2) > tunnel-vxlan-remove name tunnel-2 vxlan 10000
```

or from the corresponding VTEP objects like so:

```
CLI (network-admin@leaf-2) > vtep-vxlan-remove name vtep2 vxlan 10000
```

Finally, the BD can be deleted with the following command:

```
CLI (network-admin@leaf-2) > bridge-domain-delete name bd1000
```

Starting from NetVisor OS release 6.0.0, when a BD is deleted, if it was previously created with the `auto-vxlan` option, the BD or VNI mapping gets removed from all the tunnels automatically, so you don't need to

do it manually.

From NetVisor OS version 6.1.0, the `vxlan-stats-show` command displays the statistics for VXLANs associated with bridge domains.

For example, to view the VXLAN statistics for the bridge domain BD100, use the following command:

```
CLI (network-admin@switch) > vxlan-stats-show bd BD100 layout vertical
time:         03:23:19
vnid:         12591499
vxlan-name:
vnet:
bd:           BD100
ibytes:       7.60T
ibits:        66.8T
ipkts:        3.59G
idrops-pkts:  0
obytes:       7.54T
obits:        66.5T
opkts:        3.59G
odrops-pkts:  0
```

Configuring Outer TPID for Double-tagged Packets

By default a QinQ bridge domain port is configured to use the 802.1ad TPID (i.e., 0x88A8) in the outer tag.

Starting from NetVisor OS release 7.0.1 it is possible to configure the TPID to be 0x9100 or 0x8100, instead, to be compatible with other QinQ implementations (for example, Cisco Systems' original QinQ).

On the NRU03, NRU-S0301, AS7726-32X/F9432-C, AS7326-56X/F9480-V, AS5835-54X/F9460-X, AS5835-54T/F9460-T platforms and on the Dell S5200 Series it is possible to configure the TPID on a per switch port basis, like so:

```
CLI (network-admin@switch) > port-tpid-modify ports <port-list> outer-tpid
<tpid-value>
```

Possible TPID values are:

- `dot1q` (0x8100)
- `q-in-q` (0x88A8)
- `q-in-q-old` (0x9100)
- `none` (default)

When `outer-tpid` is configured as `none` (default), double tagged packets use the standard 802.1ad encapsulation with the outer TPID equal to 0x88A8 (and the inner TPID equal to 0x8100).

Here is an example of configuration of the TPID to be 0x9100:

```
CLI (network-admin@switch) > port-tpid-show ports 121,123,32
switch ports outer-tpid
------ ----- ----------
```

```
switch 121   q-in-q
switch 123   q-in-q
test4  32    q-in-q

CLI (network-admin@switch) > port-tpid-modify ports 123,121 outer-tpid q-in-q-old

CLI (network-admin@switch) > port-tpid-show ports 121,123,32
switch ports outer-tpid
------ ----- ----------
switch 121   q-in-q-old
switch 123   q-in-q-old
test4  32    q-in-q
```

On the other hand, on platforms that are **not** NRU03, NRU-S0301, AS7726-32X/F9432-C, AS7326-56X/F9480-V, AS5835-54X/F9460-X, AS5835-54T/F9460-T and the Dell S5200 Series it is possible to configure the TPID on a per bridge domain port basis like so:

```
CLI (network-admin@switch) > bridge-domain-port-add name <bd-name> port <port> outer-vlan <vlan-id> outer-tpid <tpid-value> [inner-vlan <vlan-id>]
```

For example:

```
CLI (network-admin@switch*) > bridge-domain-port-add name bd1 port 11 outer-vlan 2001 inner-vlan 201 outer-tpid q-in-q-old
```

```
CLI (network-admin@switch*) > bridge-domain-port-show
name port outer-vlan single-bum-domain vlans inner-vlan l2-learning outer-tpid
---- ---- ---------- ----------------- ----- ---------- ----------- ----------
bd1  11   2001       false                 201        none        q-in-q-old
```

> **Note:** TPID configuration on a per bridge domain port basis is not supported for bridge domains when `vxlan-inner-packet` mode is set to transparent.

Configuring Bridge Domains in vNETs

NetVisor OS version 6.1.0 supports the configuration of bridge domains in vNETs. This enhancement enables vNET users to configure bridge domains on vNET ports so that they can be managed on a per tenant basis.

To configure bridge domains in a vNET, you should first specify the number of allowed bridge domains in the vNET when creating it. Also, you must specify the `vlan-type` as `private,` because the default `vlan-type` is `public` but bridge domains are supported only in global or private vNETs. For example:

```
CLI (network-admin@switch) vnet-create name vnet1 scope fabric vlan-type private vlans 300-400 num-bridge-domains  3 vxlans 10100 vxlan-end 10200
```

You can now create a bridge domain in `vnet1` by using the command:

```
CLI (network-admin@switch) bridge-domain-create name bd1 scope fabric vxlan 10100 vnet vnet1
```

View the bridge domain configuration by using the command:

```
CLI (network-admin@switch) > bridge-domain-show layout vertical
name:                bd1
vnet:                vnet1
scope:               fabric
vxlan:               10100
auto-vxlan:          no
rsvd-vlan:           2833
vxlan-inner-packet:  auto
qinq_rsvd_vlan:      2833
mac-learning:        on
```

Note:You cannot specify a reserved VLAN for a bridge domain in a vNET configuration as the reserved VLAN is automatically selected from the `vtep-auto-vlan` range. Since this VLAN range is common to all vNETs, you cannot create bridge domains in a vNET if this pool is exhausted.

The `vtep-auto-vlan` range has 500 VLANs allocated by default:

```
CLI (network-admin@switch) > vtep-auto-vlan-show
vlans:    2500-2999
```

You can widen this range, for example, by 11 VLANs by using the command:

```
CLI (network-admin@switch) > vtep-auto-vlan-modify vlans 2500 vlan-end 3010
```

`vtep-auto-vlan-modify`	Modify the VLAN ID range for automatic assignment to VTEPs at the fabric level.
`vlans` *`0..4095`*	Specify the starting VLAN ID for `vtep-auto-vlan` range.
`vlan-end` *`0..4095`*	Specify the ending VLAN ID for `vtep-auto-vlan` range.

Note: The `vtep-auto-vlan` reserved range and the VLAN range manually allocated to each vNET must be mutually exclusive.

Add ports to `bd1` by using the `bridge-domain-port-add` command:

```
CLI (network-admin@switch) > bridge-domain-port-add name bd1 port 33 vlans
100
```

Note: You can only add vNET managed ports to the bridge domain. However, there is no restriction on assigning VLANs that are not part of the vNET to the bridge domain.

View the configuration using the command:

```
CLI (network-admin@switch) > bridge-domain-port-show
switch name port vlans   l2-learning
------ ---- ---- ------- -----------
```

```
switch bd1   33    100      none
```

To complete the configuration, as demonstrated earlier, associate the bridge domain's VXLAN ID with a tunnel by using the command:

```
CLI (network-admin@switch) > tunnel-vxlan-add name t1 vxlan 10100
```

You can also associate the bridge domain's VXLAN ID with a VTEP by using the `vtep-vxlan-add` command as shown earlier.

Configuring VLAN Ranges On Bridge Domain Ports

Starting from release 7.0.0 NetVisor OS supports the configuration of VLAN ranges on a bridge domain port.

A VLAN range can be used to minimize the amount of flooded (Broadcast/Unicast /Multicast, BUM in short) traffic that goes out of a bridge domain port: for each VLAN range only one copy of this kind of traffic is transmitted (instead of N copies, with N equal to the number of VLAN numbers in the range).

Referring to ***Figure 10-6*** above, for example an 802.1Q BD port on a switch can send traffic to a QinQ BD port on a different switch (over the VXLAN transport): a QinQ BD port (with the `outer-vlan` parameter configured) and a BD port with a VLAN range can be configured across a VXLAN-tunnel to provide the desired end-to-end 802.1Q tunnel functionality.

NetVisor OS supports this feature on the following platforms:

- F9480-V (AS7326-56X), F9460-X (AS5835-54X), F9460-T (AS5835-54T)
- Dell S5200 Series
- NRU03, and NRU-S0301.

> **Note**: This feature is supported only in auto BD mode.

> **Note**: VLAN translation and the `untagged-port-vlan` configuration are not supported in conjunction with this feature.

A `vlan-range` corresponds to a compact grouping of VLAN numbers described by two values: the lowest and the highest number in the range. The command syntax is:

```
CLI (network-admin@switch) > bridge-domain-port-add name <name> port <port>
vlan-range <range>
```

You can configure only one range on a port for a bridge domain (the maximum number is 8 ranges per port with different bridge domains; the limit is hardware dependent), as shown in this example:

```
CLI (network-admin@switch) > bridge-domain-port-add name test port 12 vlan-
range 15-16
```

If you try to configure more than what the hardware supports, you will get an error message:

```
CLI (network-admin@switch) > bridge-domain-port-add name test port 12 vlan-
```

```
range  12,14,15-16,20,23,25,27-29,30,32,35,38
bridge-domain-port-add: Too many ranges, limit to 1
```

A VLAN range can include all VLANs (1-4092) like so:

```
CLI (network-admin@switch) > bridge-domain-port-add name bd-1 port 10 vlan-
range all

CLI (network-admin@switch) > bridge-domain-port-show

switch name port vlans   single-bum-domain l2-learning
------ ---- ---- ------ ----------------- -----------
switch bd-1 10   1-4092 true              none
```

On the other hand, a traditional range can be created by using this syntax:

```
CLI (network-admin@switch) > bridge-domain-port-add name bd-1 port 12 vlans
10-12
```

In this case, three copies of any flooded traffic with three VLAN numbers (10, 11 and 12) will be generated by the hardware.

You can distinguish traditional ranges from `vlan-range` groupings in the bridge-domain-port-show output by looking at the single-bum-domain column: when it's true, it's a vlan-range:

```
CLI (network-admin@DUT) > bridge-domain-port-show

switch name port vlans single-bum-domain l2-learning
------ ---- ---- ----- ---------------- -----------
switch bd-1 10   10-12 true             none
switch bd-1 12   10-12 false            none
```

Note: You cannot change a `vlan-range` with the `bridge-domain-port-modify` command: you need to delete the port first and then re-add it with the new `vlan-range`.

Note: To work on clusters the vlan-range feature requires the cluster over Layer 3 feature to be enabled. Refer to the Configuring High Availability chapter for more details.

Configuring Transparent and Remove Tags Modes for Bridge Domain

Starting from NetVisor OS release 6.1.0, a new VXLAN transport mode, `transparent` has been added to the bridge domain configuration. This new mode can be used when end-to-end VLAN tag and CoS field (priority bits) transparency (i.e., preservation) is required.

Starting from the same release, there is an additional `remove-tags` mode, which can be used when 802.1Q tag preservation is not required. The typical use case is when an untagged host needs to talk to a VLAN-tagged one or with one configured with two tags.

> **Note**: In this mode `outer-vlan` is not supported as standalone parameter and produces an error if used in the configuration of bridge domain ports. The user can instead specify both the `outer-vlan` and `inner-vlan` parameters in the port configuration.

Furthermore, starting from NetVisor OS release 6.1.1 the `remove-tags` mode has been extended to support all port types with the same bridge domain (with the above exception of `outer-vlan` mode): untagged, tagged or double-tagged.

In this mode, when a packet is received on a port, the hardware removes all the VLAN tags flagging it as "untagged" and keeping it untagged also during VXLAN encapsulation. Then, after being decapsulated and when going out of the egress port, zero, one or two tags will be added to the packet depending on the egress port type. In this mode, CoS field (priority bits) configuration has been added too, as discussed below.

> **Note**: Supporting multiple BD modes provides flexibility for numerous use cases, however BD modes *must be configured consistently across bridge domain end points*. In other words, a BD end point configured with local scope in `remove-tags` mode will not talk to another end point on a different switch configured with local scope in, say, `auto` mode.

First off, use the `vxlan-inner-packet` parameter to select a mode or modify it:

```
CLI (network-admin@switch) > bridge-domain-create name <name> scope <scope>
vxlan <vxlan-id> vxlan-inner-packet  auto|remove-tags|transparent

CLI (network-admin@switch) > bridge-domain-modify name <bridge-domain-name>
vxlan-inner-packet  auto|remove-tags
```

`Auto` is the implicit default transport mode, which represents the pre-6.1.0 mode of operation. In the VXLAN transport it removes the outer VLAN tag for 802.1ad-tagged packets and retains the VLAN tag for 802.1Q-tagged packets.

Instead, the new `transparent` and `remove-tags` modes must be configured explicitly by the user to be applied to the VXLAN transport.

When a bridge domain is created in transparent mode there are some important configuration limitations/requirements to keep in mind:

- Transparent mode is supported only with ports configured with the `vlans` or `outer-vlan` parameter, but **not** with both for the same bridge domain.

- The `bridge-domain-modify` command is not supported. Instead, you must delete the BD followed by adding the BD again by using the commands, `bridge-domain-delete`, followed by `bridge-domain-add`.
- The `bridge-domain-port-modify` command is not supported. Instead, use `bridge-domain-port-remove` followed by `bridge-domain-port-add` commands.
- Untagged ports are not supported.
- For 802.1ad-tagged (QinQ) ports, use `outer-vlan <vlan-id>` parameter only. The `inner-vlan` parameter is not supported while using the `bridge-domain-port-add` command.
- On all the ports of a transparent bridge domain the user needs to configure a common global VLAN, which needs to be set aside specifically for that use in the fabric.
- The user does not need to configure a special `rsvd-vlan` on clusters. NetVisor OS takes care of that automatically.
- `Local` scope is not supported.
- The user needs to add both `vlan` and `q-in-q` TPIDs on cluster links.
- NetVisor OS supports both the parameters: `mac-learning|no-mac-learning` to enable or disable MAC learning on a bridge domain. However, we recommend enabling MAC learning. If you disable MAC learning by using the parameter, `no-mac-learning`, then it may lead to unnecessary flooding including to *cluster* and *CPU port*.
- The `port congestion` displayed in the `port-stats-show port 0` (CPU port) reflects packets dropped by the hardware *rate-limiter* that exceeds the configured Q threshold to protect the CPU. This includes flooded (BUM) traffic.

Here is an example of transparent bridge domain creation:

```
CLI (network-admin@switch) bridge-domain-create name transparent-bd scope
fabric vxlan-inner-packet transparent vxlan 10000

CLI (network-admin@switch) > bridge-domain-show

Switch  name           scope  vxlan auto-vxlan vxlan-inner-packet mac-learning
------- -------------- ------ ----- ---------- ------------------ ------------
switch  transparent-bd fabric 10000  no         transparent        on
```

Once the bridge domain is created, you can assign a port configured with a regular 802.1Q VLAN (say, 100) to the transparent bridge domain like so:

```
CLI (network-admin@switch) > bridge-domain-port-add name transparent-bd
port 10 vlans 100
```

Alternatively, if a port needs to receive 802.1ad double-tagged traffic (with outer VLAN 100), you can add it to the transparent bridge domain like so:

```
CLI (network-admin@switch) > bridge-domain-port-add name transparent-bd
port 11 outer-vlan 100
```

In the above examples the common global VLAN that gets dedicated to the transparent bridge domain is VLAN 100. The user has to make sure that the same VLAN is used consistently on each transparent bridge domain port. Also, as noted in the limitation list above, ports configured with the `vlans` parameter and ports configured with the `outer-vlan` parameter are *not* allowed to be used in the same transparent bridge domain.

An important requirement with transparent bridge domains is that cluster links (for example, switch ports 8

and 9 on a cluster pair) be manually configured to carry both 802.1Q and 802.1ad TPIDs (namely, 0x8100 and 0x88a8) like so:

```
CLI (network-admin@switch) > port-config-modify ports 8,9 allowed-tpid vlan,q-in-q
```

To view the statistics (byte, packet, unicast, multicast, broadcast, input, output, drops) use the following commands:

- To view per tunnel statistics, use the `tunnel-stats-show` command.
- To view the per VNI (tied to VLAN or bridge-domain) statistics, use the `vxlan-stats-show` command.
- To view the per port statistics , use the `port-stats-show` command.
- To view the per port per CoS statistics, use the `port-cos-stats-show` command.

Use the `show-diff-interval <interval-in-secs>` formatting option with the above commands to monitor the real-time statistics every `<interval-in-secs>` seconds. For example,

```
CLI (network-admin@switch) > tunnel-stats-show show-diff-interval 2
```

time	tunnel-name	ibits	iUpkts	iBpkts	iMpkts	iCongDrops	ierrs	obits	oUpkts	oBpkts	oMpkts	oCongDrops	oerrs	mtu-errs
16:21:17	VLE-TUNNEL	0	0	0	0	0	0	40.8M	8.01K	0	0	0	0	0
16:21:17	auto-tunnel-11.0.0.1_14.0.0.1	9.27M	0	0	10.9K	0	0	0	0	0	0	0	0	0
16:21:17	auto-tunnel-11.0.0.1_13.0.0.1	31.9M	0	0	37.4K	0	0	0	0	0	0	0	0	0
16:21:17	auto-tunnel-11.0.0.1_16.0.0.1	6.85M	0	0	8.04K	0	0	0	0	0	0	0	0	0
16:21:17	auto-tunnel-11.0.0.1_15.0.0.1	27.5K	0	0	33	0	0	0	0	0	0	0	0	0
time	tunnel-name	ibits	iUpkts	iBpkts	iMpkts	iCongDrops	ierrs	obits	oUpkts	oBpkts	oMpkts	oCongDrops	oerrs	mtu-errs
16:21:19	VLE-TUNNEL	0	0	0	0	0	0	0	0	0	0	0	0	0
16:21:19	auto-tunnel-11.0.0.1_14.0.0.1	0	0	0	0	0	0	0	0	0	0	0	0	0
16:21:19	auto-tunnel-11.0.0.1_13.0.0.1	0	0	0	0	0	0	0	0	0	0	0	0	0
16:21:19	auto-tunnel-11.0.0.1_16.0.0.1	0	0	0	0	0	0	0	0	0	0	0	0	0
16:21:19	auto-tunnel-11.0.0.1_15.0.0.1	0	0	0	0	0	0	0	0	0	0	0	0	0

Use the `format all` option to view detailed statistics.

Starting from NetVisor version 6.1.1, the `remove-tags` mode supports all port types at the same time on the same physical interface. You can use different bridge domains or even the same bridge domain configured in all modes on the same port.

> **Note**: As with `transparent` mode, the `bridge-domain-port-modify` command is not supported. Instead, use `bridge-domain-port-remove` followed by the `bridge-domain-port-add` command to change port mode.

To configure `remove-tags` mode use the command:

```
CLI (network-admin@switch) > bridge-domain-create name <new-name> scope <scope> vxlan <vxlan-id> vxlan-inner-packet remove-tags
```

Or, for an existing bridge domain, use the command:

```
CLI (network-admin@switch) > bridge-domain-modify name <bridge-domain-name> vxlan-inner-packet remove-tags
```

For example, you can configure multiple port modes on bridge domain BD2 in `remove-tags` mode like so:

```
CLI (network-admin@switch) > bridge-domain-show

name scope  ports vxlan-inner-packet mac-learning l2-tunneling
---- ------ ----- ------------------ ------------ ------------
BD1  local        auto               on           none
BD2  local  20    remove-tags        on           none

CLI (network-admin@switch) > bridge-domain-port-add name BD2 port 20
untagged-port-vlan 200
CLI (network-admin@switch) > bridge-domain-port-add name BD2 port 20 vlans
300
```

The result of the port configuration can be displayed with the command:

```
CLI (network-admin@switch) > bridge-domain-port-show

name port vlans untagged-port-vlan l2-learning
---- ---- ----- ------------------ -----------
BD2  20         200                none
BD2  20   300                      none
```

You can also configure different BDs on the same port in different modes, for example like so:

```
CLI (network-admin@switch) > bridge-domain-show

Switch name scope vxlan-inner-packet mac-learning l2-tunneling
------ ---- ----- ------------------ ------------ ------------
switch bd1  local remove-tags        on           none
switch bd2  local remove-tags        on           none
switch bd3  local remove-tags        on           none

CLI (network-admin@switch) > bridge-domain-port-show

switch name port outer-vlan vlans untagged-port-vlan inner-vlan l2-learning
------ ---- ---- ---------- ----- ------------------ ---------- -----------
switch bd1  30                    500                           none
switch bd2  30             300                                  none
switch bd3  30   2000                              1000       none
```

It is also possible to configure multiple VLANs on the same port for the same bridge domain:

```
CLI (network-admin@switch) > bridge-domain-port-show

name port outer-vlan vlans untagged-port-vlan inner-vlan l2-learning
---- ---- ---------- ----- ------------------ ---------- -----------
BD1  25             103                                 none
BD1  25             104                                 none
```

And it is possible to configure multiple combinations of inner and outer VLANs on the same port for the same bridge domain, as shown below:

```
CLI (network-admin@switch) > bridge-domain-port-show
```

```
switch name port outer-vlan vlans untagged-port-vlan inner-vlan l2-learning
------ ---- ---- ---------- ----- ------------------ ---------- -----------
switch BD1  9    2000                                 1000       none
switch BD1  9    2001                                 1001       none
switch BD1  9    2002                                 1002       none
```

Furthermore, it is possible to configure the egress CoS field value (priority bits) to include in the packet tag by using one of the following commands depending on the egress port type:

- Double-tagged traffic port:

```
CLI (network-admin@switch) > bridge-domain-port-add name bd1 port 10 outer-vlan 10 outer-cos 7 inner-vlan 100 inner-cos 5
```

- Single-tagged traffic port:

```
CLI (network-admin@switch) > bridge-domain-port-add name bd1 port 20 vlans 200 cos 3
```

Note: The `bridge-domain-port-modify` command cannot be used to change CoS values.

Configuring a Bridge Domain and a VLAN on the Same Port

Before NetVisor OS release 6.1.1, a bridge domain and a VLAN were not allowed to be configured on the same port. They were mutually exclusive.

A port can be added to a bridge domain with the `bridge-domain-create` plus `bridge-domain-port-add` command sequence. On the other hand, a port can be added to a VLAN with the `vlan-create` plus `vlan-port-add` command sequence.

In the latter case above, when a VLAN was already configured on a port (say, port 20), attempting to configure a bridge domain on the same port would have produced the following error:

```
vlan/vxlan is already configured on port 20, remove it to configure Q-in-Q
```

Vice versa, on a port (say, port 21) already configured with a bridge domain (with the first sequence described above), attempting to configure a VLAN on the same port would have produced the following error:

```
Ignoring port (21) addition to vlan 10 since these port are part of Q-in-Q
```

Starting from NetVisor OS release 6.1.1, these checks (and the corresponding errors) are removed as it is now possible to configure both a VLAN and a BD on the same port.

However, there are still some *restrictions* that must be considered:

- A bridge domain and a VLAN are not allowed to share the same (port, VLAN ID) value pair. So, for example, if (port 20, VLAN 100) is already part of `BD1`, then the VLAN configuration cannot use the same (port 20, VLAN 100) pair. If a conflicting pair is used by mistake in the configuration, it will be rejected.
- Since a port can only have one untagged VLAN, it can either be used by a bridge domain (in untagged mode) or as a port's native VLAN. The same VLAN ID cannot be used in both configurations. Therefore, a conflicting configuration will be rejected.
- If vLE is configured on a port, the same port cannot be used for VLAN or BD configuration.
- VLAN and bridge domain configuration cannot be done on a port that is configured as a Network Packet Broker (NPB) port.

Configuring Packet Bridging Between Different QinQ S-TAG/C-TAG Pairs on the Same Bridge Domain Port

Starting from NetVisor OS release 7.0.2, an enhancement is implemented to support packet bridging between bridge domain logical ports, i.e., between different QinQ S-TAG/C-TAG pairs on the same physical port for a given bridge domain. This capability allows configurations to use different (S-TAG, C-TAG) BD pairs on the same port and to enable connected endpoints to communicate with each other (even if using different TAG pairs). Ports that support multiple TAG pair combinations (currently up to two) can be host ports as well as redundant trunk or vLAG ports (as shown in the figures below).

Note: This capability requires newer forwarding ASICs for the rewriting of both the S-TAG and C-TAG packet fields while performing packet bridging. So it is supported only on the NRU03, NRU-S0301, AS7726-32X/F9432-C, AS7326-56X/F9480-V, AS5835-54X/F9460-X, AS5835-54T/F9460-T platforms and on the Dell S5200 Series.

Note: For QinQ ports the default TPID is 0x88a8.

To add up to two TAG pairs (i.e., inner and outer VLAN ID pairs), you can use the command sequence shown in the following example:

```
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17 outer-vlan 106 inner-
vlan 41
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17 outer-vlan 107 inner-
vlan 42
```

Trying to add an additional pair to the same port and to the same BD yields an error (since the limit has been reached):

```
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17 outer-vlan 108 inner-
vlan 43
bridge-domain-port-add: cannot add BD port as max Q-in-Q BD port number 2 exceeded for bridge-
domain BD1 on port 17
```

The configuration for BD1 can be displayed like so:

```
CLI (network-admin@switch*) > bridge-domain-port-show name BD1
name     port outer-vlan single-bum-domain inner-vlan l2-learning
------- ---- ---------- ----------------- ---------- -----------
BD1      17   900        false             880        none
BD1      17   901        false             881        none
```

To remove the BD port configuration:

```
CLI (network-admin@switch*) > bridge-domain-port-remove name BD1 port 17
```

Note: This feature is supported with the auto or remove-tags mode of bridge domains. Transparent mode configuration is rejected in conjunction with this feature, as it is not applicable.

```
CLI (network-admin@switch*) > bridge-domain-create name BD1 scope fabric
```

```
vxlan 300900 rsvd-vlan 4009 vxlan-inner-packet  transparent

CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 1060 inner-vlan 410
bridge-domain-port-add: bridge-domain BD1 in TRANSPARENT mode, does not
support Untagged or QinQ port with both tags
```

Applicable Network Topologies

This feature's main use case is to interconnect different endpoints (for example VMs on the same server) configured with different S-TAGs and C-TAGs. The endpoints are configured in the same subnet and can communicate at Layer 2 by bridging traffic between them.

A server may be connected to the leaf switch with a single link as shown below:

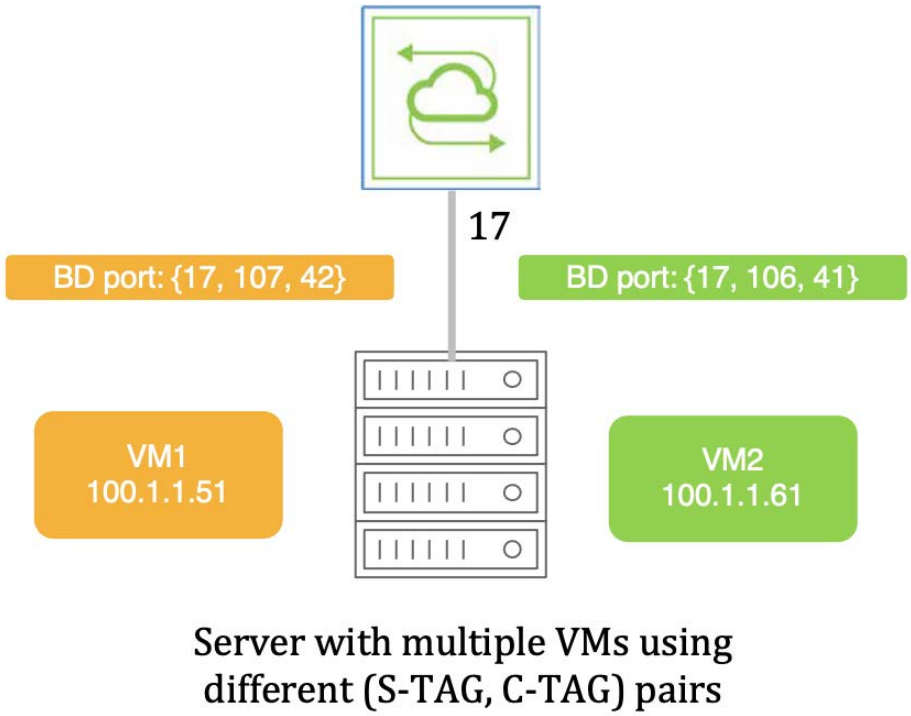


Figure 9-7: Single-homed Host Connected to a BD with Multiple (S-TAG, C-TAG) Pairs

Two endpoints within the same subnet can communicate using the following configuration:

```
CLI (network-admin@switch) > bridge-bridge-domain-create name BD1 scope fabric vxlan 300300
rsvd-vlan 4002

CLI (network-admin@switch) > bridge-domain-port-add name BD1 port 17 outer-vlan 106 inner-vlan
41

CLI (network-admin@switch) > bridge-domain-port-add name BD1 port 17 outer-vlan 107 inner-vlan
42
```

The inter-endpoint unicast and BUM traffic is bridged within BD1 and is tagged with the (106, 41) and (107, 42) TAG pairs accordingly based on the destination:

```
CLI (network-admin@switch) > l2-table-show bd BD1

mac               bd   vlan inner-vlan vxlan  ip          ports state  status
----------------- ---- ---- ---------- ------ ---------- ----- ------ ------
00:12:c0:80:36:a1 BD1  107  42         300300 100.1.1.61 17    active host
00:12:c0:80:33:1e BD1  106  41         300300 100.1.1.51 17    active host
```

Instead of a single link, a multi-link trunk can be used too for single-homed devices and can be configured like so:

```
CLI (network-admin@switch*) > trunk-show name trunk1 format
name,ports,hash-mode,enable,lacp-mode,lacp-timeout,status

trunk-id name   ports hash-mode enable lacp-mode lacp-timeout status
-------- ------ ----- --------- ------ --------- ------------ ------------------------------
272      trunk1 1,125 enhanced  on     off       slow         up,PN-switch,PN-other,STP-BPDUs

CLI (network-admin@switch*) > bridge-domain-create name BD1 scope fabric
vxlan 12000004 auto-vxlan rsvd-vlan 4019

CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 272
outer-vlan 510 inner-vlan 61
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 272
outer-vlan 610 inner-vlan 51
CLI (network-admin@switch*) > l2-table-show bd BD1
mac               bd  vlan inner-vlan vxlan    ip          ports state  status
----------------- --- ---- ---------- -------- ---------- ----- ------ ------
00:12:c0:80:33:1e BD1 510  61         12000004 109.1.1.51 1,125 active host
00:12:c0:80:36:a1 BD1 610  51         12000004 109.1.1.61 1,125 active host
```

For redundancy purposes, a switch cluster pair can be configured with two QinQ ports as a vLAG for the same BD, as shown below, to connect to a dual-homed server:

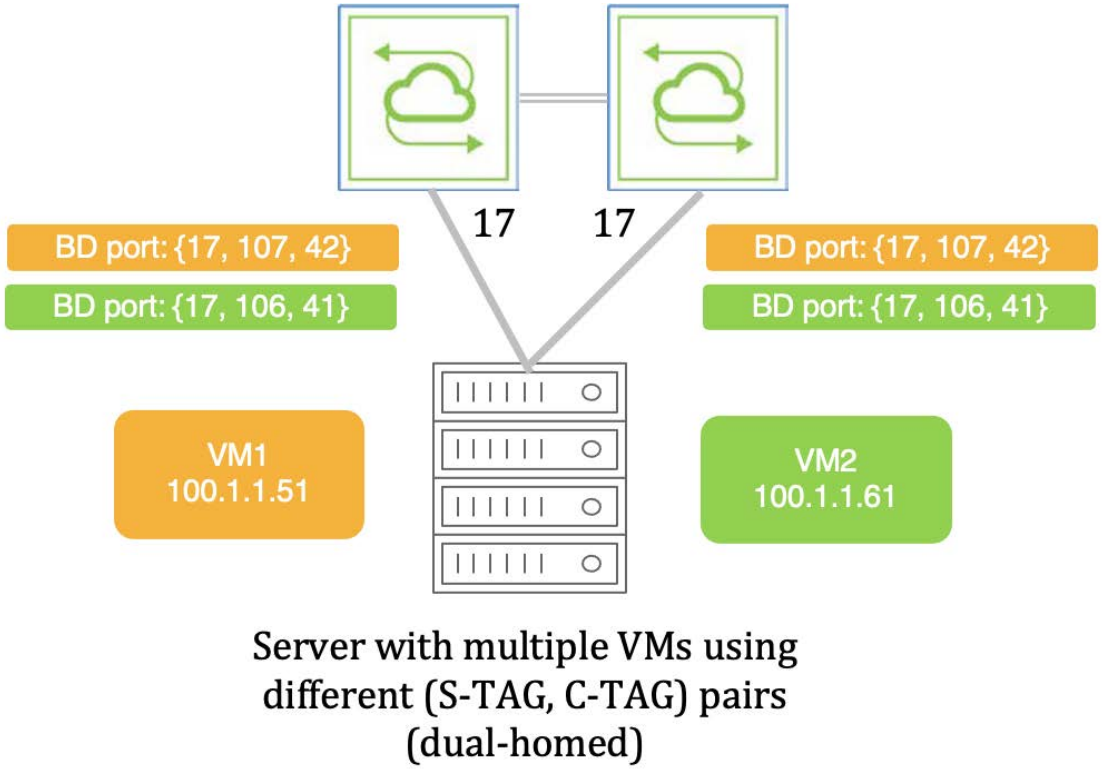


Figure 9-8: Dual-homed Host Connected to a BD with Multiple (S-TAG, C-TAG) Pairs

Note: This BD configuration works also with Cluster-over-Layer 3.

Alternatively, instead of a server, a switch can also be connected to the vLAG member ports as shown below:

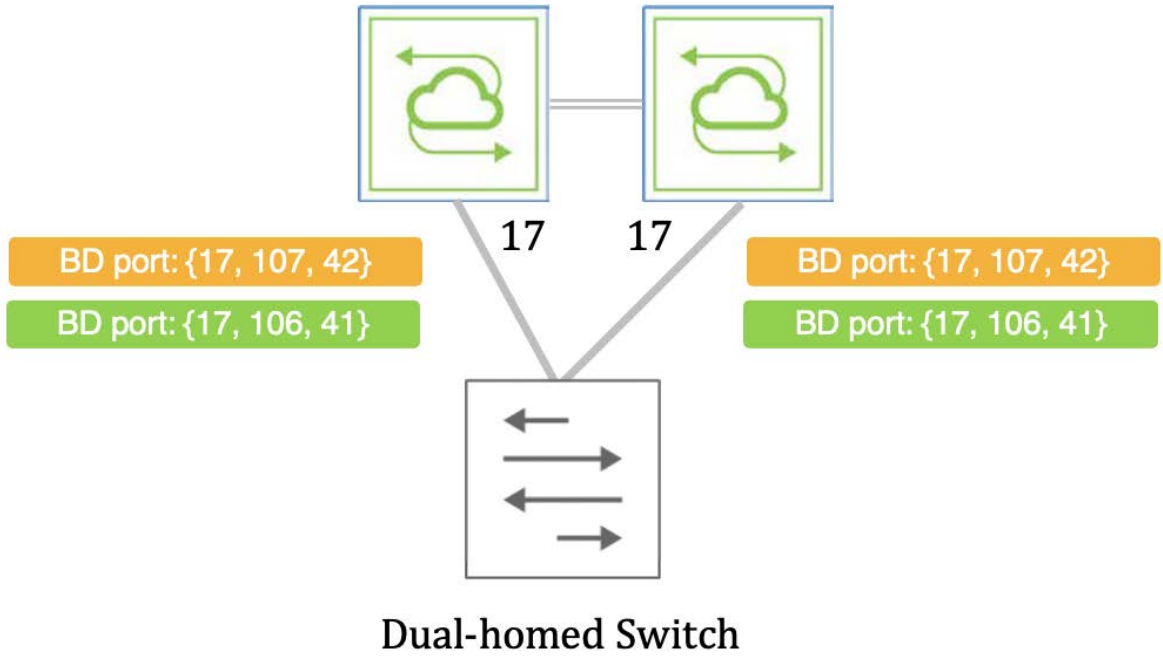


Figure 9-9: Dual-homed Switch Connected to a BD with Multiple (S-TAG, C-TAG) Pairs

Supported TAG Pairings on a Port for a Given BD

Not all TAG pairing schemes are supported:

1) Using different S-TAGs and C-TAGs in the pairings is supported as shown in this example:

```
CLI (network-admin@switch*) > bridge-domain-create name BD1 scope fabric
vxlan 12000001 auto-vxlan rsvd-vlan 4002
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 106 inner-vlan 41
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 107 inner-vlan 42
CLI (network-admin@switch*) > l2-table-show bd BD1
mac               bd  vlan inner-vlan vxlan    ip          ports state  status
----------------- --- ---- ---------- -------- ----------- ----- ------ ------
00:12:c0:80:33:1e BD1 106  41         12000001 100.1.1.51  17    active host
00:12:c0:80:36:a1 BD1 107  42         12000001 100.1.1.61  17    active host
```

2) Using the same CTAG with different S-TAGs in the pairings is supported:

```
CLI (network-admin@switch*) > bridge-domain-create name BD1 scope fabric
vxlan 12000002 auto-vxlan rsvd-vlan 4003

CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 206 inner-vlan 32
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 207 inner-vlan 32
CLI (network-admin@switch*) > l2-table-show bd BD1
mac               bd  vlan inner-vlan vxlan    ip          ports state  status
----------------- --- ---- ---------- -------- ----------- ----- ------ ------
00:12:c0:80:36:a1 BD1 207  32         12000002 102.1.1.61  17    active host
00:12:c0:80:33:1e BD1 206  32         12000002 102.1.1.51  17    active host
```

3) On the other hand, reusing the same S-TAG is not a valid configuration:

```
CLI (network-admin@switch*) > bridge-domain-create name BD1 scope fabric
auto-vxlan rsvd-vlan 4009

CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 306 inner-vlan 33
```

Trying to add another BD port with same S-TAG/outer VLAN ID is rejected (regardless of the chosen inner VLAN ID):

```
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 306 inner-vlan 35
bridge-domain-port-add: out-vlan 306 for BD port already configured for
bridge-domain BD1 on port 17

CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 306 inner-vlan 33
bridge-domain-port-add: out-vlan 306 for BD port already configured for
bridge-domain BD1 on port 17
```

Configuration of Single and Double TAG on the Same Port

It is also possible to configure on the same port a combination of a single C-TAG and a dual (S-TAG, C-TAG) pairing for the same BD.

In this case, the BD mode needs to be vxlan-inner-packet remove-tags. In other words, in auto or transparent mode the configuration is rejected with an error.

Here is an example of configuration of single and double TAG on the same port:

```
CLI (network-admin@switch*) > bridge-domain-create name BD1 scope fabric
auto-vxlan rsvd-vlan 4005 vxlan-inner-packet  remove-tags

CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 109 inner-vlan 45
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17 vlans
81

CLI (network-admin@switch*) > l2-table-show bd BD1
mac               bd  vlan inner-vlan vxlan    ip         ports state  status
----------------- --- ---- ---------- -------- ---------- ----- ------ ------
00:12:c0:80:33:1e BD1 109  45         12000000 200.1.1.51 17    active host
00:12:c0:80:36:a1 BD1 81              12000000 200.1.1.61 17    active host
```

Outer TPID Setting

The standard TPID value for IEEE 802.1ad is 0x88a8 (default setting). In some cases, though, another TPID may be needed (such as 0x8100, called dot1q). Such configuration change means that all the BD ports with QinQ configuration can use 0x8100 as outer TPID. This is possible using port-level configuration like so:

```
CLI (network-admin@switch*) > port-tpid-modify ports 17 outer-tpid  dot1q

CLI (network-admin@switch*) > bridge-domain-create name BD1 scope fabric
vxlan 12000004 auto-vxlan rsvd-vlan 4019
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 510 inner-vlan 61
CLI (network-admin@switch*) > bridge-domain-port-add name BD1 port 17
outer-vlan 610 inner-vlan 51
```

Configuring Layer 2 Protocol Tunneling

Starting from NetVisor OS release 6.1.1, PDU Transparency (a.k.a. Layer 2 Protocol Tunneling) is supported on bridge domains by leveraging hardware forwarding capabilities. With this feature certain protocol packets (i.e., PDUs) are no longer terminated on the ingress switch port and then sent to the CPU for processing. They are instead tunneled over a bridge domain. Currently the feature supports STP, LLDP and LACP PDUs.

Layer 2 Protocol Tunneling can be enabled during or after bridge domain creation with the new `l2-tunneling` keyword. When a bridge domain is created:

```
CLI (network-admin@switch) > bridge-domain-create name <name> scope <scope> vxlan <vxlan-id> vnet <vnet-id> l2-tunneling {stp | lacp | lldp | all | none}
```

`none` is the default value when the `l2-tunneling` keyword is not used.

Layer 2 Protocol Tunneling configuration can be modified with the following command:

```
CLI (network-admin@switch) > bride-domain-modify name <name> l2-tunneling {stp | lacp | lldp | all | none}
```

The `bridge-domain-show` command includes a new column to display the Layer 2 Protocol Tunneling configuration:

```
CLI (network-admin@switch) > bridge-domain-show format name,vxlan,auto-vxlan,rsvd-vlan,local-rsvd-vlan,qinq_rsvd_vlan,mac-learning,L2-tunneling,

switch name vxlan  auto-vxlan rsvd-vlan local-rsvd-vlan qinq_rsvd_vlan mac-learning L2-tunneling
------ ---- ------ ---------- --------- --------------- -------------- ------------ ------------
switch bd1  100101 no         1001      1001                           off          all
switch bd2  100102 no         1002      1002                           on           stp,lldp
switch bd3  100103 no         1003      1003                           on           none
```

Note: Bridge domain PDU transparency leverages the device's hardware capabilities and therefore requires the configuration of a VXLAN Loopback Trunk, even when single pass mode is used.

Disabling MAC Address Learning on Bridge Domains

In certain bridge domains, the MAC address learning function can be disabled in order to achieve better scalability, or because it is not strictly required (for example on point-to-point connections).

For these cases, starting from NetVisor OS release 6.0.0, it is possible to disable MAC address learning as part of a bridge domain configuration. For this purpose, a new configuration option is added, `mac-learning | no-mac-learning`, to the following commands:

```
bridge-domain-create scope local mac-learning | no-mac-learning
bridge-domain-modify scope local mac-learning | no-mac-learning
```

By default, MAC address learning is *enabled* at the bridge domain level.

Moreover, to support more granular per-port configurability, a new option, `l2-learning enable | disable | none`, is also added to the `bridge-domain-port-add` and `bridge-domain-port-modify` commands.

The default configuration is `l2-learning none`. In such case, at the port level the MAC address learning state is directly derived from the one of the encompassing bridge domain.

On the other hand, the `enable` and `disable` options can be used to take precedence over the per-bridge domain configuration to achieve port-level granularity.

The `bridge-domain-show` and `bridge-domain-port-show` commands are extended to display the `mac-learning/l2-learning` state.

For example, to disable MAC address learning for an entire bridge domain at creation, use the command:

```
CLI (network-admin@switch) > bridge-domain-create name bd1 scope local
vxlan 100101 no-mac-learning

CLI (network-admin@switch) > bridge-domain-show

switch name scope vxlan auto-vxlan rsvd-vlan ports mac-learning
------ ---- ----- ------ ---------- -------- ----  ------------
switch bd1  local 100101 no                       off
```

To reenable MAC address learning only on port 17, while leaving it disabled on port 21, you can use the commands:

```
CLI (network-admin@switch) > bridge-domain-port-add name bd1 port
17 vlans 100 l2-learning enable

CLI (network-admin@switch) > bridge-domain-port-add name bd1 port 21 vlans
200

CLI (network-admin@switch) > bridge-domain-port-show name bd1

switch   name port vlans l2-learning
------- ---- ---- ----- -----------
switch  bd1   17  100    enable
```

```
switch  bd1   21  200    none
```

To re-enable MAC address learning on the entire bridge domain:

```
CLI (network-admin@switch) > bridge-domain-modify name bd1 mac-learning

CLI (network-admin@switch) > bridge-domain-show name bd1

switch name  scope vxlan auto-vxlan ports mac-learning
------ ----- ----- ----- ---------- ----- ------------
switch  bd1  local 100101   no      17,21   on

CLI (network-admin@switch) > bridge-domain-port-show name bd1

switch name   port vlans l2-learning
------ -----  ---- -----  -----------
switch bd1    17   100    enable
switch bd1    21   200    none
```

Then to revert to the default per-port configuration on port 17 you can use:

```
CLI (network-admin@switch) > bridge-domain-port-modify name bd1 port 17 l2-
learning none
```

Displaying vLE Statistics

The vLE statistics information is useful for understanding the data traffic through a vLE. This information includes the number of incoming and outgoing packets (classified as unicast, multicast, and broadcast), bytes dropped, bytes discarded, Head-End-Replicated (HER) packets, and errors. Use the `vle-stats-show` command to display vLE statistics on each vLE port.

```
CLI (network-admin@switch1) > vle-stats-show
```

`vle-stats-show`	Displays port statistics.
`time date/time: yyyy-mm-ddThh:mm:ss`	Specify a time for the packet count statistics using the timestamp format yyyy-mm-ddThh:mm:ss.
`start-time` *`start-time`*	Specify a start time for the packet count statistics using the timestamp format yyyy-mm-ddThh:mm:ss.
`end-time` *`end-time`*	Specify an end time for the packet count statistics using the timestamp format YYYY-MM-DDTHH:MM:SS
`duration` *`duration`*	Specify the duration in seconds.
`interval duration: #d#h#m#s`	Specify the interval between statistics collection.
`since-start|no-since-start`	Specify if the statistics are collected from the start of collecting statistics.
`older-than duration: #d#h#m#s`	Specify if the statistics are older than the duration in days, hours, minutes, and seconds.
`within-last duration: #d#h#m#s`	Specify if the statistics are within the duration in days, hours, minutes, and seconds.
`vle-name` *`vle-name-string`*	Specify the vLE name.
`description` *`description`*	Displays the vLE description.
`counter` *`counter-number`*	Displays the counter number.
`ibytes ibytes-number`	Displays the incoming number of bytes.
`ibits` *`ibits-number`*	Displays the number of incoming bits.
`iUpkts` *`iUpkts-number`*	Displays the number of incoming unicast packets.
`iBpkts` *`iBpkts-number`*	Displays the number of incoming broadcast packets.
`iMpkts` *`iMpkts-number`*	Displays the number of incoming multicast packets.
`iPauseFs` *`iPauseFs-number`*	Displays the number of incoming pause frames.
`iCongDrops` *`iCongDrops-number`*	Displays the number of incoming packets dropped due to congestion.

idiscards *idiscards-number*	Displays the number of incoming packets discarded.
ierrs *ierrs-number*	Displays the number of incoming packets with errors.
obytes *obytes-number*	Displays the number of outgoing bytes.
oUpkts *oUpkts-number*	Displays the number of outgoing unicast packets.
oBpkts *oBpkts-number*	Displays the number of outgoing broadcast packets.
oMpkts *oMpkts-number*	Displays the number of outgoing multicast packets.
oPauseFs *oPauseFs-number*	Displays the number of outgoing pause frames.
oCongDrops *oCongDrops*-number	Displays the number of outgoing packets dropped due to congestion.
odiscards *odiscards-number*	Displays the number of outgoing discarded packets.
oerrs *oerrs-number*	Displays the number of outgoing packets with errors.
mtu-errs *mtu-errs-number*	Displays the number of MTU errors.
HER-packets *HER-pts-number*	Displays the number of Head End Replicated (HER) packets.
HER-bytes *HER-bytes-number*	Displays the number of Head End Replicated (HER) bytes.

Consider for example, a vLE represented by the following `vle-show` output:

```
CLI (network-admin@switch1) > vle-show layout vertical
name:          vle1
vnet:
node-1:        switch1
node-2:        switch2
node-1-port:   12
node-2-port:   23
status:        up
tracking:      enabled
ports-state:   override
create-time:   08:47:56
```

View the statistics for this vLE by using the `vle-stats-show` command:

```
CLI (network-admin@ara00) > vle-stats-show layout vertical
switch:       switch1
time:         06:14:49
vle-name:     vle1
port:         12
description:
```

```
ibits:        349M
iUpkts:       184K
iBpkts:       0
iMpkts:       0
iCongDrops:   0
ierrs:        0
obits:        63.3M
oUpkts:       56.2K
oBpkts:       0
oMpkts:       0
oCongDrops:   0
oerrs:        0
mtu-errs:     0
switch:       switch2
time:         06:14:49
vle-name:     vle1
port:         23
description:
ibits:        108M
iUpkts:       0
iBpkts:       0
iMpkts:       141K
iCongDrops:   0
ierrs:        0
obits:        63.3M
oUpkts:       0
oBpkts:       0
oMpkts:       56.3K
oCongDrops:   0
oerrs:        0
mtu-errs:     0
```

You can clear the statistics on the local switch by using the command `vle-stats-clear`.

Troubleshooting vLE

To troubleshoot vLE link state tracking, refer to the examples in the *Configuring Virtual Link Extension State Tracking* section. In particular, for both individual vLE pseudo-wires and redundant vLE trunks protocol messages can be leveraged to verify end-to-end connectivity (or lack thereof). LLDP messages (and LACP messages in case of trunks) show the neighboring port ID(s) and reflect the up/down connectivity status of the pseudo-wire(s). Therefore they are very valuable troubleshooting tools.

With vLE link state tracking enabled, a vLE pseudo-wire can go down (i.e., in vle-wait state) when the remote end of the vLE goes down. However, in some cases, it may be possible that a “false positive” event occurs in which the local vLE port goes down even if the remote end is *physically up*. This could happen if the vLE tracking logic incorrectly believes the remote end to be unreachable. In such case, you can disable vLE tracking, which should bring the local port back up and allow you to check if the traffic is flowing through the vLE pseudo-wire correctly. Increasing the vLE tracking timeout can help to avoid false positives by giving more leeway to the link state tracking software, especially when experiencing frequent link flaps or CPU overload conditions (which could affect the tracking logic negatively).

Configuring Virtual Port Group State Tracking

Starting from release 7.0.1, NetVisor OS leverages a BFD-based state tracking logic to support mirroring the port state across VXLAN connections for interconnected virtual Port Group (vPG). BFD is used automatically when the connection is established and so does not require any additional configuration. This logic is shared with vLE.

> **Note:** The vPG state tracking is supported only with bidirectional point-to-point vPG.

When configuring vSGs, state tracking can be enabled when creating new vFlow policies using the `vflow-create` command, or by using the `vflow-modify` command on existing policies.

A new tracking configuration parameter has been added to enable or disable vPG state tracking. A new tracking-status column shows the vFlow-related state of the vPGs, which can be one of the following values:

- **up:** the ports of both the local and remote vPGs of the vFlow are up.
- **local-down**: the port of the local vPG is down.
- **remote-down**: the port of the remote vPG is down.
- **down**: from the perspective of a node on which neither of the vPGs are local, it means that the ports of both vPGs are down.
- **disabled**: state tracking is disabled for a vFlow entry.
- **unknown**: the status could not be determined (which is possible during corner cases, for example due to errors).

For instance, the unknown status may be displayed when `vflow-show` is executed on a switch that is not a vPG state tracking endpoint and there is a communication error. Or when a user executes the `vpg-port-remove` command on a vPG that is being tracked: in such case, the status will be unknown until a port is added back.

Tracking is disabled by default. When in disable state, the tracking-status column shows disabled.

You can enable state tracking during vFlow creation like so:

```
CLI (network-admin@switch) > vflow-create name bivf scope fabric bidir-vpg-1 bi1 bidir-vpg-2 bi2 tracking enable
```

Then you can show the administrative and operational statuses like so:

```
CLI (network-admin@switch) > vflow-show format type,in-port,bidir-vpg-1,bidir-vpg-2,tracking,tracking-status,enable,table-name

name type  in-port bidir-vpg-1 bidir-vpg-2 tracking tracking-status enable table-name
---- ----- ------- ----------- ----------- -------- --------------- ------ --------------------
bivf vflow 25      bi1         bi2         enable   up              enable System-VCAP-table-1-0
```

You can enable/disable state tracking during vFlow modification like so:

```
CLI (network-admin@switch) > vflow-create name bivf scope fabric bidir-vpg-1 bi1 bidir-vpg-2 bi2
```

```
CLI (network-admin@switch) > vflow-modify name bivf tracking enable

CLI (network-admin@switch) > vflow-show format type,in-port,bidir-vpg-
1,bidir-vpg-2,tracking,tracking-status,enable,table-name

name type  in-port bidir-vpg-1 bidir-vpg-2 tracking tracking-status enable table-name
---- ----- ------- ----------- ----------- -------- --------------- ------ --------------------
bivf vflow 25      bi1         bi2         enable   up              enable System-VCAP-table-1-0

CLI (network-admin@switch) > vflow-modify name bivf tracking disable

CLI (network-admin@switch) > vflow-show format type,in-port,bidir-vpg-
1,bidir-vpg-2,tracking,tracking-status,enable,table-name

name type  in-port bidir-vpg-1 bidir-vpg-2 tracking tracking-status enable table-name
---- ----- ------- ----------- ----------- -------- --------------- ------ --------------------
bivf vflow 25      bi1         bi2         disable  disabled        enable System-VCAP-table-1-0
```

Below is a complete example of the behavior for the state tracking feature along with the port state changes:

a) The first configuration step is vPG creation with type bidirectional on two nodes (local and remote):

```
CLI (network-admin@switch2) > vpg-create name bi1 type bidirectional ports
25

CLI (network-admin@switch1) > vpg-create name bi2 type bidirectional ports
25
```

b) The subsequent configuration step is vFlow creation:

```
CLI (network-admin@switch2) > vflow-create name bivf scope fabric bidir-
vpg-1 bi1 bidir-vpg-2 bi2 tracking enable

CLI (network-admin@switch2) > vflow-show format type,in-port,bidir-vpg-
1,bidir-vpg-2,tracking,tracking-status,enable,table-name

name type  in-port bidir-vpg-1 bidir-vpg-2 tracking tracking-status enable table-name
---- ----- ------- ----------- ----------- -------- --------------- ------ --------------------
bivf vflow 25      bi1         bi2         enable   up              enable System-VCAP-table-1-0

CLI (network-admin@switch2) > switch-local port-show port 25

switch  port status               config
-------  ----  -------------------  ------
switch2  25    up,host,LLDP,vlan-up  fd,10g

CLI (network-admin@switch1) > switch-local port-show port 25
switch  port status               config
-------  ----  -------------------  ------
switch1  25    up,host,LLDP,vlan-up  fd,10g
```

c) When disabling the local vPG port, the remote port goes to `vbt-wait` state and on the local node the vFlow entry shows the tracking status to be `local-down`:

```
CLI (network-admin@switch2) > port-config-modify port 25 disable
```

```
CLI (network-admin@switch2) > switch-local port-show port 25
switch  port status   config
------- ---- -------- ------
switch2 25   disabled 10g

CLI (network-admin@switch2) > vflow-show format type,in-port,bidir-vpg-
1,bidir-vpg-2,tracking,tracking-status,enable,table-name

name type  in-port bidir-vpg-1 bidir-vpg-2 tracking tracking-status enable table-name
---- ----- ------- ----------- ----------- -------- --------------- ------ --------------------
bivf vflow 25      bi1         bi2         enable   local-down      enable System-VCAP-table-1-0

Meanwhile, at the remote end:

CLI (network-admin@switch1) > switch-local port-show port 25
switch  port status           config
------- ---- ---------------- ------
switch1 25   disabled,vbt-wait 10g
```

d) After re-enabling the local port, the remote port comes out of vbt-wait state and the vFlow entry shows the tracking status to be up:

```
CLI (network-admin@switch2) > port-config-modify port 25 enable

CLI (network-admin@switch2) > switch-local port-show port 25
switch  port status              config
------- ---- ------------------- ------
switch2 25   up,host,LLDP,vlan-up fd,10g

CLI (network-admin@switch2) > vflow-show format type,in-port,bidir-vpg-
1,bidir-vpg-2,tracking,tracking-status,enable,table-name

name type  in-port bidir-vpg-1 bidir-vpg-2 tracking tracking-status enable table-name
---- ----- ------- ----------- ----------- -------- --------------- ------ --------------------
bivf vflow 25      bi1         bi2         enable   up              enable System-VCAP-table-1-0
```

Meanwhile, at the remote end the port is back up:

```
CLI (network-admin@switch1) > switch-local port-show port 25
switch  port status                config
------- ---- --------------------- ------
switch1 25   up,host,LLDP,vlan-up  fd,10g
```

Guidelines and Limitations

This is a list of guidelines and limitations to keep in mind when configuring bridge domains:

- Currently on bridge domain ports in `auto` mode the QoS default behavior is as follows:
    - Incoming 802.1Q frames' CoS value (a.k.a. PCP) is preserved along with the VLAN tag when transported in the VXLAN payload
    - Incoming 802.1ad frames' CoS value from the inner tag is preserved along with the inner VLAN tag when transported in the VXLAN payload
    - For outgoing 802.1Q frames after VXLAN decapsulation *CoS is set to value 0*
    - For outgoing 802.1ad frames after VXLAN decapsulation the CoS value carried in the VXLAN packet is preserved and becomes the CoS value of the inner tag. The outer tag instead *carries a CoS value of 0*.

- Before NetVisor OS release 6.0.0 on the Dell S5232-ON and Dell S5248-ON switch models the QinQ and untagged modes were not supported, while the 802.1Q mode is allowed. Starting from release 6.0.0 support for QinQ and untagged mode was added (however, an untagged host can only communicate to another untagged host in a bridge domain).

- On ports added to a BD, the STP protocol and the IGMP snooping features are not supported.

- Routing on BDs is not supported.

- Before NetVisor OS release 6.1.0 vNETs were not supported with BDs.

- The LLDP and LACP protocols are supported on BD ports configured in QinQ mode. The trunking and vLAG functionalities are supported too.

- The optimized ARP and optimized ND features cannot coexist with BDs. They should be set to off in the `system-setting-show` output. If the user attempts to enable the feature, a conflict message is printed as below:

```
CLI (network-admin@switch) > system-settings-modify optimize-arps
system-settings-modify: Cannot turn on optimized ARP as bridge-domain Q-in-
Q is configured
```

When optimized ARP/ND is already configured before a port is added to a BD, a similar conflict message is printed:

```
CLI (network-admin@switch) > bridge-domain-port-add name bd1000 port 18
outer-vlan 18 inner-vlan 100
bridge-domain-port-add: Cannot add port in Q-in-Q mode to bridge-domain as
optimized ARP or ND configured
```

- The flowtrace function is not supported with VXLAN and BDs.

- The VLAN ID is not shown in the `l2-table-show` output when a L2 entry is associated to a cluster tunnel:

```
CLI (network-admin@leaf1) > l2-table-show bd bd1 format
```

```
switch,mac,bd,vlan,vxlan,inrf,ports,tunnel
```

```
switch           mac               bd  vlan vxlan  intf ports  tunnel
---------------- ----------------- --- ---- ------ ---- ----- ---------------------------------
leaf1            00:12:c0:80:1c:09 bd1 100  101001 274  82
ursa-scale-leaf2 00:12:c0:80:1c:09 bd1 100  101001 274  82
ursa-scale-leaf1 00:12:c0:80:40:52 bd1      101001            auto-tunnel-10.40.40.1_10.30.40.1
ursa-scale-leaf2 00:12:c0:80:40:52 bd1      101001            auto-tunnel-10.40.40.1_10.30.40.1
ursa-scale-leaf4 00:12:c0:80:40:52 bd1 100  101001 81   81
```

- When configuring vLE redundancy, if LACP is used for end-to-end connectivity checks (which is usually preferable), a vLE trunk can be set up only with multiple individual virtual pipes using separate VTEPs (e.g., 8 in case of a 4-way vLE trunk). When using static trunks, instead, it is also possible (but generally less desirable) to set up, say, a 4-way vLE trunk with 2 2-way trunks as endpoints (which equates to a 4 x 2-port trunk configuration on leaf switches and a 4-way trunk configuration on end devices).

- For vLE endpoints only use physical IP (PIP) addresses. Do not use VRRP virtual IP (VIP) addresses.

- With the `allowed-tpid qinq` (i.e., 802.1ad TPID) configuration on a BD port configured in 802.1Q mode, MAC address learning includes both outer and inner VLAN of a double-tagged 802.1ad frame. Instead, for a single-tagged 802.1Q frame MAC address learning is based on just one VLAN. Both types of MAC address entries can be seen with the `l2-table-show` command. The hardware associates a BD and a VNI to a packet based on the port type. For BD ports configured in 802.1Q mode, the hardware lookup does not use the inner VLAN, it uses only the outer VLAN. So for bridge domain-related troubleshooting, use the `l2-table-show` command to focus on the MAC address+VNI pairs, rather than looking at the outer and/or inner VLAN, as the hardware lookup for Layer 2 entries is based on MAC address and VNI value.

Supported Releases

Command/Parameter	NetVisor OS Version
`vxlan-mode standard|transparent`	Parameter added in version 2.5.2
`vle-create, vle-delete, vle-modify, vle-show`	Command added in version 2.5.4
`vle-tracking-timeout`	Parameter added in version 3.0.0
`vle-transparent-modify, vle-transparent-show, vle-transparent-port-remove, vle-transparent-port-show, vle-transparent-port-add`	Commands introduced in version *5.1.1*
`port-config-modify allowed-tpid q-in-q`	Parameter introduced in version 5.2.0
`bridge-domain-create, bridge-domain-rsvd-vlan-add, bridge-domain-port-add`	Commands introduced in version 5.2.0
`l2-table-show bd bd_name`	Parameter introduced in version 5.2.0
`ports-state`	Parameter introduced in version 6.0.0
`auto-vxlan`	Parameter introduced in version 6.0.0
`mac-learning|no-mac-learning l2-learning enable|disable|none`	Parameters introduced in version 6.0.0
`vnet, create-time, elapsed-time, up-time`	Parameters introduced in version 6.1.0
`vle-stats-show`	Command introduced in version 6.1.0
`vxlan-inner-packet transparent|auto|remove-tags`	Parameters added in version 6.1.0
`outer-cos, inner-cos, cos`	Parameters added in version 6.1.1
`l2-tunneling`	Parameters added in version 6.1.1
`vlan-range`	Parameters added in version 7.0.0
`topology-create, topology-delete,topology-modify, topology-import,topology-show, topology-vle-add,topology-vle-remove, topologyvle-show, topology-vle-modify`	Commands introduced in version 7.0.0
`outer-tpid`	Parameter added in version 7.0.1

Please also refer to the Arista NetVisor OS Command Reference document.

Related Documentation

For further information on concepts mentioned in this section (such as Layer 2, Layer 3, etc.), refer to these sections of the Configuration Guide:

- Configuring Switch Ports
- Configuring Layer 2 Features
- Configuring Layer 3 Features
- Configuring and Administering the Unified Cloud Fabric
- Configuring High Availability
- Configuring VXLAN
- Configuring vNETs

Configuring VXLAN Ethernet VPN (EVPN)

This chapter provides information about the configuration of the Ethernet VPN feature on a NetVisor OS switch using the NetVisor OS command line interface (CLI).

Understanding EVPN

Ethernet VPN (EVPN) is a standard technology that was created to overcome some of the limitations of a popular MPLS-based technology called *Virtual Private LAN Service (VPLS)*, as specified in the respective IETF RFCs for MAN/WAN use. In particular, for example for data center deployments, it became imperative to address certain *VPLS limitations* in areas such as multihoming and redundancy, provisioning simplicity, flow-based load balancing, and multipathing.

Hence, as an evolution of VPLS, EVPN was born as a multi-protocol (MP) BGP- and MPLS-based solution in RFC 7432 (later updated by RFC 8584) to implement the requirements specified in RFC 7209.

The initial implementation of EVPN was intended to leverage the benefits of an MPLS Label Switched Path (LSP) infrastructure, such ss fast reroute, resiliency, etc. Alternatively, the EVPN RFC includes support also for an IP or IP/GRE (Generic Routing Encapsulation) tunneling infrastructure.

As a further evolution, in RFC 8365 EVPN was expanded to support various other encapsulations including a VXLAN-based transport (called overlay encapsulation type 8).
By that time VXLAN had become the prevalent transport in the data center to implement virtual Layer 2 bridged connectivity between fabric nodes. So EVPN could then be used as an MP BGP-based control plane for VXLAN, with support for specialized Network Layer Reachability Information (NLRI) to communicate both Layer 2 and Layer 3 information for forwarding and tunneling purposes.

In EVPN parlance, VXLAN is a Network Virtualization Overlay (NVO) data plane encapsulation solution with its own identifiers, the VNIs (also called NVO instance identifiers). VNIs can be mapped one-to-one or many-to-one to EVPN instances (EVIs). VNIs can be globally unique identifiers (in a typical use case), but the EVPN RFC also includes support for the case when they are used as locally significant values (to mimic MPLS labels).

In a VXLAN network using EVPN, a VTEP node is called a Network Virtualization Edge (NVE) or Provider Edge (PE) node. An NVE node may be a Top-of-Rack (ToR) switch or a virtual machine, but—as we will see below—it can also be a border node.

For more details on EVPN, refer to the following IETF standards and drafts:

- RFC 7209 "Requirements for Ethernet VPN (EVPN)"
- RFC 7432 "BGP MPLS-Based Ethernet VPN"
- RFC 8365 "A Network Virtualization Overlay Solution Using Ethernet VPN (EVPN)"
- The draft-ietf-bess-evpn-prefix-advertisement-11, “IP Prefix Advertisement in EVPN”

Understanding NetVisor VXLAN EVPN Technology

Arista Networks' *Unified Cloud Fabric* offers a highly flexible approach to software-defined networking (SDN) with native support for the VXLAN transport with an optimized control plane.

As described in the *Configuring VXLAN* chapter earlier, the unified cloud fabric uses the VXLAN overlay technology for *intra-fabric connectivity*. It supports the creation of static VXLAN-based tunnels using the `tunnel-create` command. It also supports automatic tunnel creation through the *VTEP object* construct and its associated commands.

VTEP objects are fabric-scoped, making all the nodes aware of each other's VTEPs and their VNIs. Furthermore, fabric-wide Layer 2 and Layer 3 reachability information is propagated to each node inside a fabric through vPort data synchronization. Remote reachability data is leveraged to avoid the need to *flood and learn* packets. Hence, traffic forwarding is optimized.
Multi-tenancy is supported too by using highly scalable hardware-based VRF segmentation.

In this context, the EVPN technology's BGP-based control plane offers a unique opportunity to further augment the fabric's flexibility and provisioning simplicity in complex multi-pod data center network designs.

As we will describe in the following sections, the Arista Unified Cloud Fabric has been extended to integrate the EVPN standard as a means to improve the fabric's scalability and interconnection options between data center pods, as well as to support *multi-vendor interoperability*.

Starting from NetVisor OS release 6.1.0, Arista Networks implemented the EVPN technology to interconnect multiple fabric pods and to extend the VXLAN overlay across them, supporting both IPv4 and IPv6. EVPN's powerful control plane enables network architects to build larger scale multi-pod designs, where each pod can comprise up to 32 leaf nodes. It also allows an Arista pod to interoperate with pods using third party vendors' nodes.

In this implementation, VTEP objects, subnets, VRFs are naturally extended to communicate across pods. To do that, NetVisor OS leverages EVPN in a *pragmatic, integrated and standards-compatible* way to implement a BGP-based control plane that enables multi-hop VXLAN-based forwarding across pods using a number of key fabric enhancements, described in the following.

About EVPN Border Gateways

In Arista's fully interoperable implementation, EVPN's message exchange is supported *only* on specially designated *border nodes*, deployed individually or in redundant pairs (i.e., in clusters running VRRP). Such border nodes are commonly called *Border Gateways (BGW)*, as they implement the *EVPN Border Gateway functions* (which we will describe in detail in the following sections). In essence, they are in charge of exchanging messages with external EVPN-capable nodes (Arista or other vendors' devices) and of propagating and translating external EVPN routes, where needed, within each pod, as depicted in the figure below.

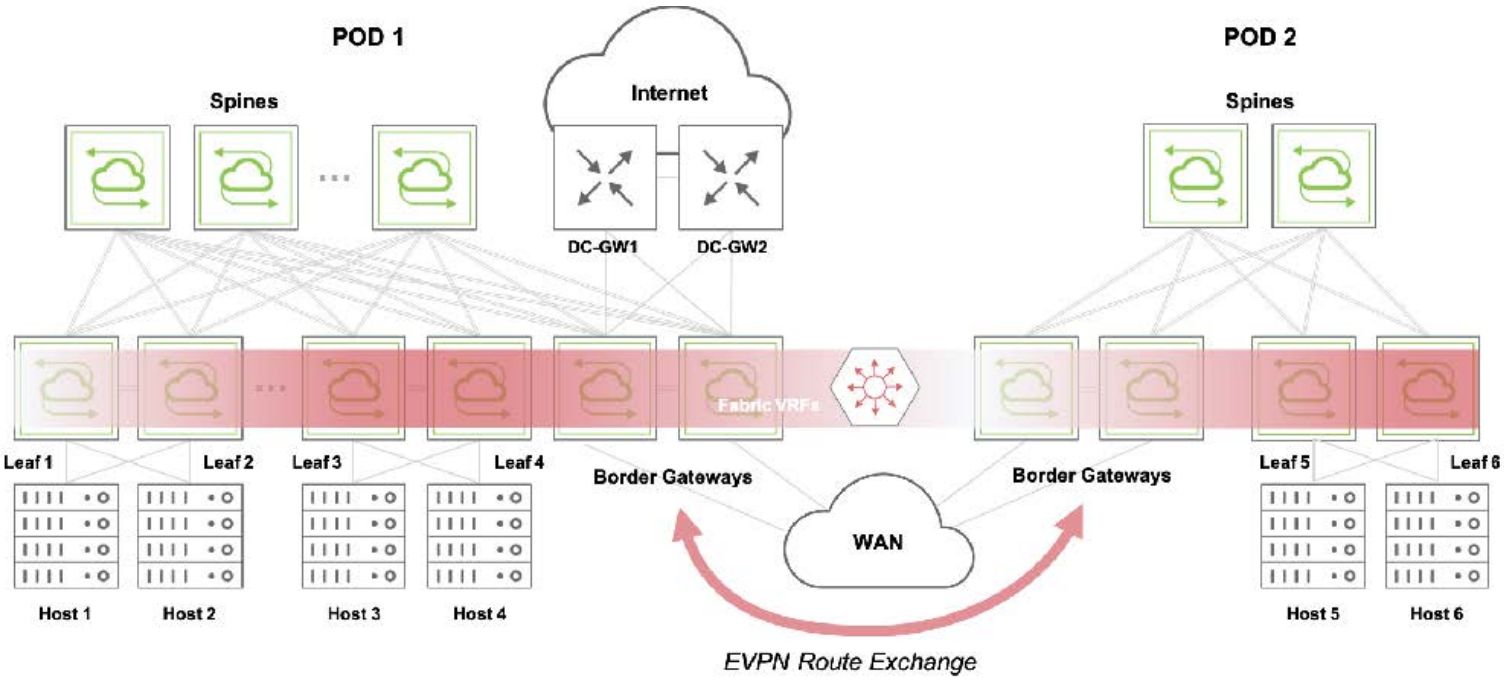


Figure 10-1: Border Gateways and EVPN Route Exchange

Border gateway nodes are identified by a special vRouter configuration that includes two steps: the vRouter must be configured with the `evpn-border` parameter and must have at least one BGP neighbor set to `l2vpn-evpn`, as described in more detail in the configuration section below.

Note: The VXLAN EVPN border gateway functions are supported only on the Dell S5200 Series.

Note: As of NetVisor OS release 6.1.0, only leaf nodes can be designated as border gateways.

About Border Gateway Functions

EVPN border gateways implement a number of key control plane and data plane functions that guarantee multi-fabric interconnection and interoperability.

By leveraging them, NetVisor OS can seamlessly extend VTEP objects, VNIs, subnets and VRFs across fabrics by simply leveraging the EVPN control-plane capabilities. Such capabilities are based on special *route types* specified by the standards.

NetVisor OS supports the following EVPN route types:

- **Type 2 routes**, i.e., *MAC/IP advertisement routes*, as defined in RFC 7432. They are required to propagate MAC learning information (and optionally associated IP addresses) for device reachability purposes over a so-called *Layer 2 VNI (L2VNI)*. Additionally, MAC/IP advertisements are used to propagate host route information.

- **Type 3 routes**, i.e., *Inclusive Multicast Ethernet Tag* routes, as defined in RFC7432. They are required for Broadcast, Unknown Unicast and Multicast (BUM) traffic delivery across EVPN networks. They are used to propagate VTEP and VNI information.

- **Type 5 routes**, i.e., *IP Prefix routes*, as defined in the *EVPN Prefix Advertisement* Internet Draft. This new type was added to advertise IP prefixes independently from MAC addresses (i.e., without relying on type 2 routes). They are used to advertise subnets and VRF information (the so-called L*ayer 3 VNIs*).

The above standard route types are used to implement various networking functions:

Type 2 Route Translation Function

For inter-fabric broadcast domain and Layer 2 learning extension, border gateways translate Type 2 routes received from their peers to NetVisor OS's vPort synchronization messages within the local fabric. Vice versa, they also translate local vPort synchronization messages to Type 2 routes and then send them over to their peers.

In other words, they act as 'language translators' because they import and export Layer 2 information to and from external EVPN nodes to keep both local and remote fabric nodes in sync.

Type 3 Route Distribution Function

To implement VXLAN-based connectivity, NetVisor OS supports special configuration constructs called VTEP *objects*, which automatically create a mesh of tunnels between each other to establish a fabric overlay network.

VTEPs can be configured within a fabric pod or can be *external to a pod* (in which case their location is called `host-external`). See the *Configuring VTEP Objects with Automatic Fabric Connections* section for more details.

Border gateways act as proxies for the communication of the local fabric with remote VTEP objects, for the inter-pod automatic tunnel creation and the VNI configuration.

They leverage EVPN T*ype 3 routes* to propagate VTEP and VNI information within BGP messages.

Such information is used with a `local scope` by border gateways to *automatically* create `host-external` VTEPs to *locally* represent the remote border gateways' VTEP objects. In other words, these special VTEPs are proxy representations of the neighbor VTEPs in the local border gateways (for an example, refer to the configuration section below).

The name of the Type 3 route-derived *external* VTEPs contains the `__evpn__` prefix string to make them easily identifiable (and also to distinguish them from manually created `host-external` VTEPs). The name also contains the (virtual) IP address of the corresponding remote VTEP object.

The VNI information exchanged in Type 3 routes is used to compare the VNIs configured on the local VTEPs with those configured on the remote ones. If there is at least one local VNI matching an external one, VXLAN tunnels are automatically established with the neighbor EVPN VTEPs from the local ones (see ***Figure 10-2*** below for a visual representation). The automatic tunnel naming scheme prefixes the `auto-tunnel-` string to the source and destination VTEP IPs. (For more details and examples, see the configuration section below.) VNIs are added and removed dynamically based on receipt of Type 3 routes.

In summary, once VTEP objects are configured by the user in the different pods, border gateways act as inter-fabric translators of Type 3 routes to and from NetVisor OS synchronization messages. This guarantees the full automation of the overlay creation process, not just within a single local pod but also across pods.

In addition, border gateways act as (individual or redundant) traffic forwarders for bridged traffic within the

same broadcast domains.

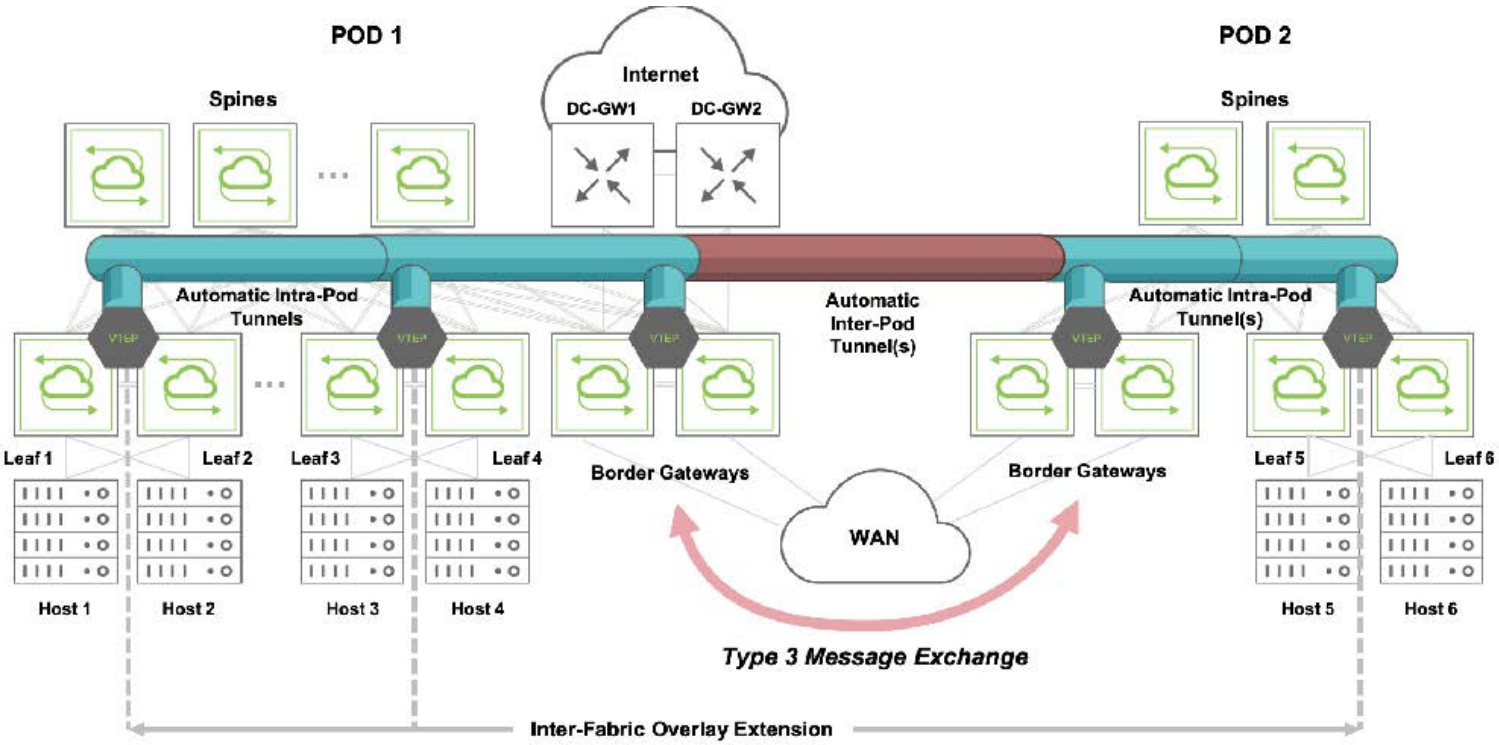


Figure 10-2: Inter-pod VXLAN Connections

Border gateways can also act as traffic forwarders for routed traffic, but that requires an additional key function as described next.

Type 5 Route Distribution Function

NetVisor OS supports the configuration of subnet objects and VRF instances as a way to perform distributed VXLAN routing in hardware with multitenancy, as described in the *Configuring Unicast Fabric VRFs with Anycast Gateway* section.

EVPN border gateways can be used to extend subnets and associated VRFs across fabrics. For that purpose, they employ type 2 and type 5 routes which they send to their remote gateway peers to notify them of remote subnets and hosts. As a consequence, the remote border gateways can install into their routing tables *prefix routes* and *host routes* corresponding to any active remote subnets and known hosts.

In addition, subnets are associated to VRFs for route table entry segregation purposes. With EVPN, these VRFs have to be uniquely identified as part of the inter-fabric synchronization process: this task is achieved by introducing a new type of VNI, called *Layer 3 VNI*, which is uniquely associated to each VRF and is exchanged through EVPN messages between border gateways.

About Inter-Pod VXLAN Routing with Subnet Objects

When deploying subnets in a multi-pod topology running EVPN, in Arista parlance subnets can be of three types:

- *local* (when present only in the local pod relative to a border gateway)
- *remote* (when located on a pod different from the local one)

- *stretched* (when present in two or more pods).

> **Note**: Having a stretched subnet does not necessarily mean that there are hosts in that subnet connected to all nodes. Hosts can be connected only to a subset of the nodes.

VXLAN routing between them is possible provided certain basic configuration rules are observed:

- A local subnet must be present on all leaf nodes: for ease of configuration, that can be achieved by using `fabric scope` in the subnet creation command.
- Subnets' L2 VNIs and prefix lengths must be configured consistently across all pods.
- VRFs' L3 VNIs must be configured consistently across all pods.
- As shown in ***Figure 10-3*** below, if needed, data center gateways should be connected to the border gateways for routing of North-South traffic.

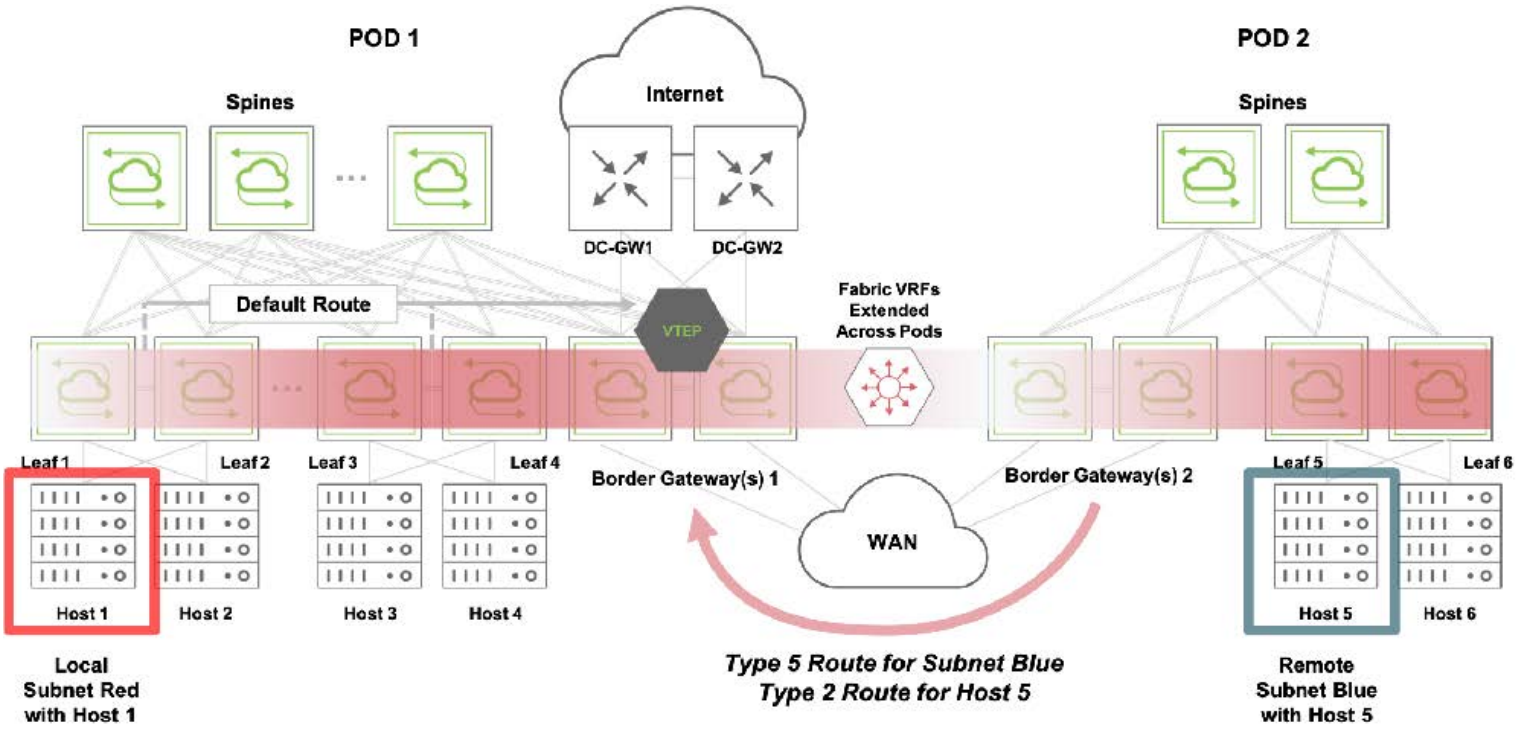


Figure 10-3: Control Plane for Inter-Pod VXLAN Routing (from Host 1 to Host 5)

In addition to the above configuration guidelines, some key automation is performed by the fabric when subnets are created: as depicted in ***Figure 10-3***, border gateways propagate local subnet information with Type 2 and Type 5 messages to their EVPN peers. Such peers then send special messages to their local fabric nodes (that are not border gateways) to automatically install a *default route* pointing to their border gateways' VTEP IP address. VXLAN traffic can then be routed to any remote pod (for example from Host 1 to Host 5, as depicted the figure above) using the *two hop model* described in the following section.

About Packet Flows for Inter-Fabric Forwarding

Let's analyze how forwarding is performed from a local subnet to a remote one, and then from a local subnet to a stretched one.

Local to Remote Subnet Forwarding

In this forwarding example, Host 1 in the local subnet (*Red*) needs to send a packet to Host 5 in a remote subnet (*Blue*), as already shown in ***Figure 10-3***.

In this scenario, as explained earlier, Border Gateways 2 (configured as a redundant pair) send Type 2 and Type 5 information to Border Gateways 1 regarding Host 5 and subnet Blue.

The fabric automation also makes sure that a default route points to Border Gateways 1's VTEP virtual address on leaf nodes 1, 2, 3 and 4. So when a packet needs to be sent from Host 1 to Host 5, a two hop forwarding process is performed as described in the next figure:

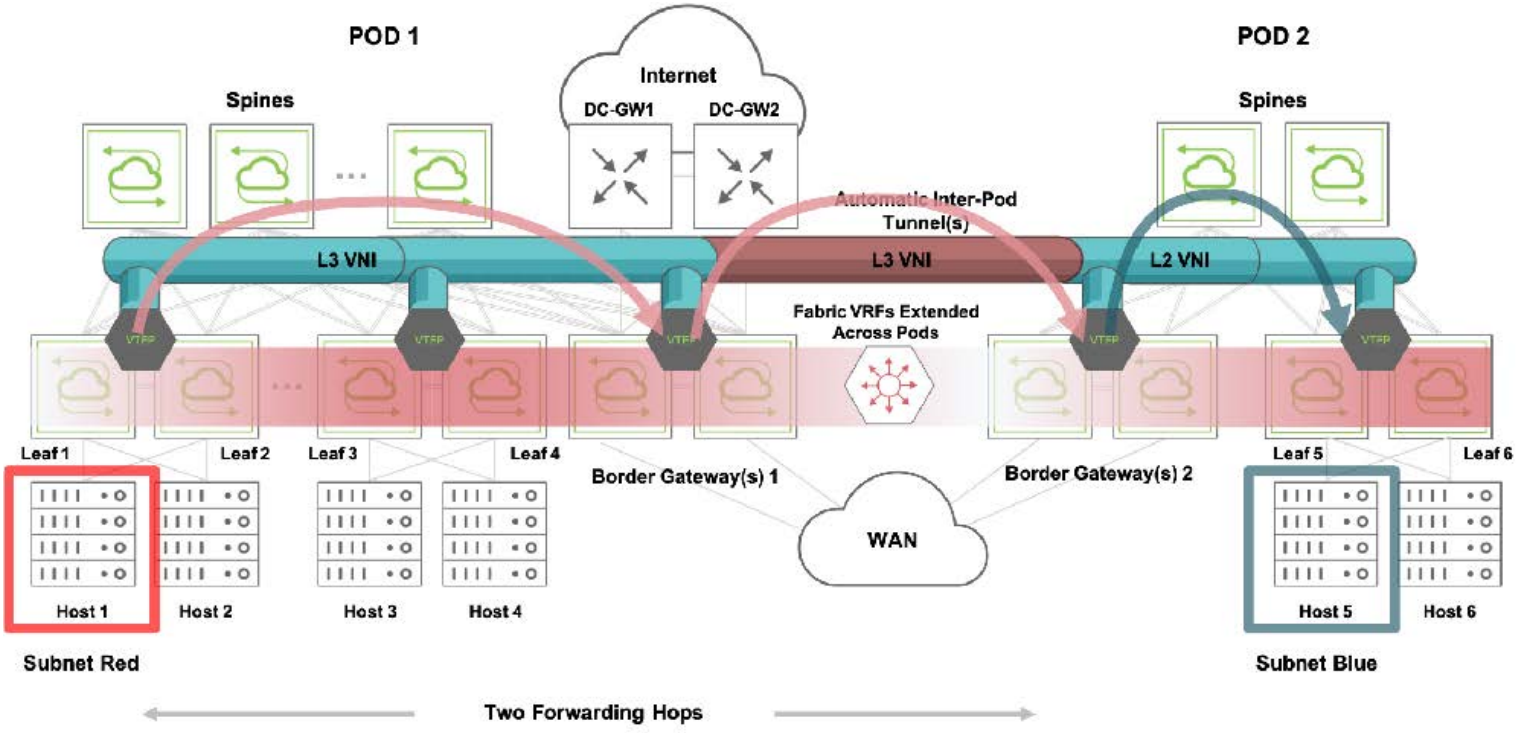


Figure 10-4: Two Hop VXLAN Routing

The packet from Host 1 is first routed to redundant Border Gateways 1 (per default route) over a VXLAN tunnel based on the VRF's L3 VNI.

Then, using the same L3 VNI, the packet is routed over a different tunnel to redundant Border Gateways 2 as the next hop. This intermediate tunnel is automatically created between the peer VTEPs of Border Gateways 1 and 2.

Lastly, when the packet reaches Border Gateways 2, the final routing from L3 VNI to subnet Blue is performed over a tunnel using subnet Blue's L2 VNI. In case the adjacency for Host 5 is not yet resolved, an ARP request is broadcast by the border gateway to all the devices in the subnet to learn about the destination address.

This multi-hop routing lookup model is also known as symmetric routing. The split horizon rule must be followed when forwarding VXLAN traffic in and out of tunnels to avoid traffic looping.

Local to Stretched Subnet Forwarding

When a subnet is present in two (or more) pods, it is called a *stretched subnet*: this case is handled differently from the above one, as shown in ***Figure 10-5*** below.

In this scenario, as seen earlier, Border Gateways 2 need to send a Type 5 route to Border Gateways 1 to advertise subnet Blue.

Moreover, since now the subnet exists in Pod 1 too (it's a *stretched subnet*), Border Gateways 1 also need to send a Type 5 route to Border Gateways 2. (In addition, subnet Red is advertised with a Type 5 route by Border Gateways 1 to Border Gateways 2.)

As soon as Host 5 is learned by its corresponding leaf node (Leaf 5), its MAC address and IP address information is sent by Border Gateways 2 to Border Gateways 1 with a Type 2 route. From there, the MAC/IP information is propagated to the rest of Pod 1 via *vPort updates*.

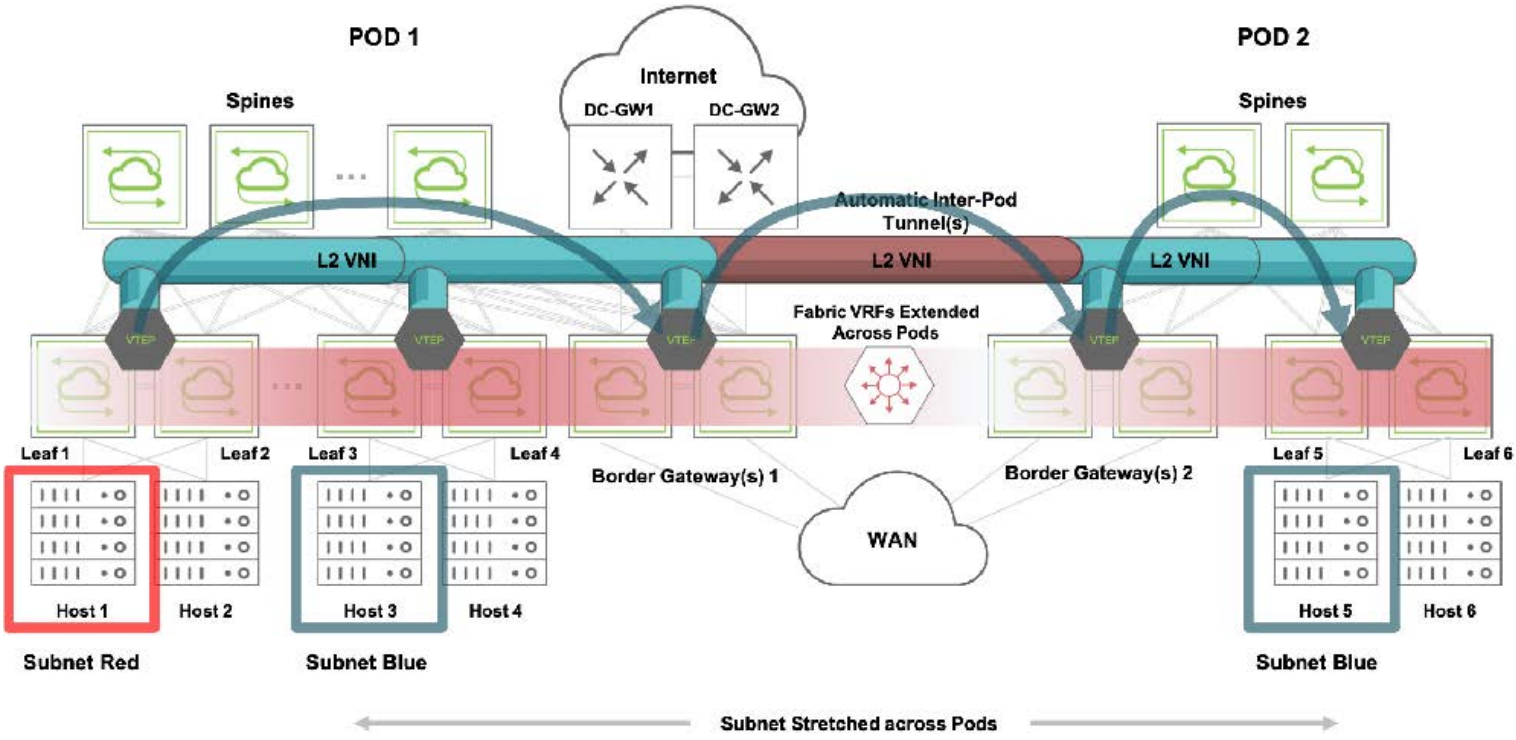


Figure 10-5: Local to Stretched Subnet Forwarding

As depicted in ***Figure 10-5*** above, in this scenario when a packet from Host 1 needs to be routed to Host 5, it is rewritten with the destination address of Host 5 and then sent over a tunnel using *subnet Blue's* L2 VNI from the ingress node's VTEP to the local fabric's Border Gateways 1. This is also referred to as the *asymmetric routing* model. Then, using the same L2 VNI, the packet is bridged over a different tunnel to redundant Border Gateways 2.

Lastly, when the packet reaches Border Gateways 2, the usual (and final) bridging step on subnet Blue is performed over a tunnel also using subnet Blue's L2 VNI.

In the reverse direction a response from Host 5 to Host 1 would follow the packet flow described in Figure 10-4 (non-stretched case).

Since two extra hops (using border gateways) are introduced, the model is also called the *two hop VXLAN forwarding model* for EVPN multi-pod integrated routing and bridging scenarios.

In a scenario with three or more (N) pods, it's possible for a subnet to be stretched only on two (or fewer than N) pods. So a *stretched subnet* can also be a *remote subnet* for at least one pod. When a host in that pod sends traffic to the remote stretched subnet, the destination border gateways in the remote pods *may not* have an adjacency resolved for the traffic destination. In such cases, as normal, the adjacency is resolved by the border gateways by broadcasting an ARP request throughout the stretched subnet (that is, *across multiple pods*) and by learning about the destination from the ARP response. Once a destination host is discovered through this mechanism, the packet flow is the same as in ***Figure 10-4***.

About Border Gateway High Availability

Any leaf node within a fabric can be made a border gateway, as long as it has BGP connectivity to EVPN nodes outside of the fabric.

A border gateway acts as a proxy for all the VTEPs in the fabric. As seen in the previous sections, it translates Type 2, Type 3, Type 5 routes to and from the EVPN peers. That means, in Arista parlance, that a border gateway *assumes ownership* of the translated vPort, VTEP, VNI and prefix/VRF route information gleaned from the external messages.

There can be multiple border gateways in a fabric to act as redundant nodes to proxy all of the above information. However, to avoid conflicts, it's important to keep *sole ownership* for that information within the fabric. For that purpose, a *primary border gateway* is elected.

From a control plane perspective, border gateway nodes in cluster pairs inherit the *master* or *backup state* from the VRRP interfaces configured for their VTEPs. A fabric node state change on the primary causes the secondary border node to become primary. Upon a border gateway state change triggered by a VRRP state change, the new primary node refreshes its routes without altering the BGP neighbor state.

Similarly, from a traffic forwarding perspective, VRRP is used for the selection of the master/backup roles and for the definition of a common virtual IP (VIP) address. BGP neighborship, instead, uses each node's physical IP (PIP) address.

The elected primary border gateway is in charge of injecting vPort/VTEP/VNI information learned from its EVPN peers. It can also deactivate such information as a consequence of a remote configuration change.

The secondary node is fully active too. It can advertise local VTEP/VNI information as part of the proxying process toward the external EVPN nodes. What a secondary node cannot do, however, is to inject (or deactivate) external vPort information into the fabric. That's reserved to the primary node, that is, to the *sole owner* of such translated/proxied information.

A secondary node also periodically monitors the primary node for any loss of connectivity or possible malfunction. If an issue is detected (for example, the primary border gateway goes offline unexpectedly), the secondary node takes over the role of primary node and maintains any learned vPort database information with its newly acquired ownership. This means that there is no disruption to already established tunnels and to the traffic that is traversing such tunnels to reach external VTEPs.

About the VXLAN Loopback Trunk with EVPN

VXLAN forwarding requires hardware functions like multiple lookups, encapsulation and decapsulation operations, packet replication, etc., even more so with EVPN's two hop forwarding model. This can be implemented using the VXLAN loopback trunk (i.e., the default option).

Starting from release 6.0.0, NetVisor OS supports single-pass VXLAN forwarding and flood as an alternative to using the VXLAN loopback trunk. With EVPN, customers using the VXLAN loopback trunk in prior releases can still use it for backward compatibility. On the other hand, single pass mode can be used on supported platforms for instance to optimize the two hop forwarding model on non-border gateway nodes.

> **Note**: As of NetVisor OS release 6.1.0, Dell S4100 and S5200 platforms can be used in single pass mode as non-border gateway nodes. Refer to the *Configuring the VXLAN Loopback Trunk* section in the *Configuring VXLAN* chapter for details on the configuration and platform support of single pass mode.

For nodes to be configured as border gateways and to use single pass forwarding, the node's hardware must be able to support both single pass mode and border gateway functions (for the latter refer to the platform list in the *About EVPN Border Gateways* section above).

About MAC Mobility with EVPN

A MAC address can be learned in one pod and then it can move to another pod, for example, when a virtual machine is migrated across locations. This is referred to as a 'MAC move'. NetVisor OS supports the standard RFC 7432's *MAC Mobility Extended Community* functionality to automatically deal with MAC moves between pods.

This specialized Extended Community is added by the EVPN control plane in MAC/IP Advertisement (i.e., Type 2) routes when a MAC move is detected. It comprises a list of fields, including flags and a sequence number. The sequence number is particularly important to ensure that border gateways retain the most recent MAC/IP Advertisement route when multiple updates occur for the same MAC address.

RFC 7432 also specifies how to deal with scenarios in which MAC moves are too frequent, for instance due to a mis-configuration. In these scenarios, it is possible that MAC address duplication occurs, and hence very frequent MAC moves are triggered for an extended period of time.

In order to perform MAC address duplicate detection per the RFC, the following configuration parameters are used in conjunction with a remediation action:

- Max moves (N MAC moves, with a default value of N = 5)
- Moves duration (M-second timer, with a default value of M = 180)

The RFC states that a node must start an M timer whenever it detects a MAC move: then, if it detects N MAC moves before the timer expires, it concludes that MAC address duplication has occurred. Consequently, the node must take a remediation action (such as alerting the operator) and must stop sending and processing any BGP MAC/IP Advertisement routes for that MAC address until a corrective action is taken.

As part of this corrective action, it's possible to configure a third parameter to stop the MAC address flapping for a configurable period of time. This additional parameter defaults to 180 seconds and is called:

- Freeze duration

For more details on the configuration of the above parameters, refer to the configuration section below.

(Instead, for intra-pod MAC move protection, refer to the Configuring Excessive MAC or IP Move Protection section of the *Configuration Guide*.)

About IGMP Snooping with EVPN

In case of an EVPN multi-pod network, basic IGMP Snooping support for VXLAN needs to be extended so that IGMP messages are propagated *across pods*. This is achieved as an extension of the local IGMP flooding process to forward the messages across the inter-pod automatic tunnels.

Whenever a multicast destination host sends an IGMP join or leave message for a multicast group, NetVisor OS running on a border gateway notices that the message originated from within the pod and so floods it across the automatic tunnels between the local pod and the peer ones. Then, when the flooded IGMP message arrives at the neighbor border gateways, they in turn flood it to all the local VTEPs over the intra-pod tunnels. This results in the IGMP messages propagating to all the nodes, both intra- and inter-pod, within a flood domain. This allows the distributed control plane to learn about all the (local or remote) hosts that want to join or leave a multicast group.

As a consequence, L2 data plane forwarding rules are applied in hardware to known multicast traffic so as to limit head-end replication only to the inter-pod tunnels over which at least one remote host has indicated interest in a multicast group.

Guidelines and Limitations

As discussed above, the user should follow these basic guidelines to avoid connectivity issues:

- A local subnet must be present on all leaf nodes within the pod, which can be achieved by using *fabric scope* in the subnet creation command. (See the *Configuring VXLAN* chapter for configuration details.)
- Subnets' L2 VNIs and prefix lengths must be configured consistently across all pods.
- VRFs' L3 VNIs must be configured consistently across all pods.
- Create *fabric scoped* VLANs associated to VNIs by using the `auto-vxlan <vni-id>` keyword to automatically map them to tunnels (no need to manually configure them with the `vtep-vxlan-add` command)
- For high availability enable the `evpn-border` option on both of the clustered border gateway nodes. Only one pair of border gateway nodes is supported in a fabric pod.
- Data center gateways should be connected to the border gateways for routing of North-South traffic.

In addition, the following limitations may apply:

- Multiple vNETs are not supported in conjunction with the EVPN border gateway configuration.
- Prior to NetVisor OS release 7.0.0. bridge domains and the vSG feature were only supported within a pod, i.e., they would not span across EVPN pods. Starting with release 7.0.0 these features are supported across multiple pods.
- EVPN's Route Distinguisher and Route Target fields are currently propagated but are not used for filtering or other purposes.

Configuring EVPN

The configuration of EVPN revolves around the selection of the border gateway nodes and the configuration of associated VNIs for the VRFs.

In addition, the fabric automation takes care of transparently performing various configurations, which can be verified with new show commands or with new parameters of existing commands, as shown in the examples below.

Configuring the Border Gateways

The first step involves configuring the vRouters on at least two nodes across two pods to make them border gateways. That is achieved with the `vrouter-create` command using the `evpn-border/no-evpn-border` parameter. In case of an existing vRouter, the `vrouter-modify` command can be used instead.

For example:

```
CLI (network-admin@switch) > vrouter-create name vRouter1 fabric-comm
enable location border-switch router-type hardware evpn-border bgp-as 5200
router-id  10.30.30.30
```

Then the `vrouter-show` command can be used to verify the current EVPN parameter configuration:

```
CLI (network-admin@switch) > vrouter-show name vRouter1 format all layout
vertical
id:                           c00029d:0
name:                         vRouter1
type:                         vrouter
scope:                        fabric
vnet:
location:                     switch
zone-id:                      c00029d:1
router-type:                  hardware
evpn-border:                  enable
```

The second step is for the user to configure an MP BGP neighbor with the `l2vpn-evpn` parameter to point to an external EVPN border gateway:

```
CLI (network-admin@switch) > vrouter-bgp-add vrouter-name vRouter1 neighbor
190.12.1.3 remote-as 65012 ebgp-multihop 3 update-source 190.11.1.3 multi-
protocol l2vpn-evpn
```

This configuration enables the use of the standard EVPN NLRI, which is carried using BGP *Multiprotocol* Extensions with an Address Family Identifier (AFI) of 25 (*L2VPN*) and a Subsequent Address Family Identifier (SAFI) of 70 (*EVPN*).

Configuring L3 VNIs

To uniquely identify VRF instances with EVPN it is necessary to specify a new `l3-vni` parameter during creation (or modification), like so:

```
CLI (network-admin@switch) > vlan-create id 1000 auto-vxlan 101000 scope
fabric ports none

CLI (network-admin@switch) > vrf-create name VRF1 l3-vni 101000
```

As explained in more detail in the *Configuring VXLAN* chapter, subnets can be mapped to VRFs with the `subnet-create` command, for example:

```
CLI (network-admin@switch) > vlan-create id 12 auto-vxlan 500012 scope
fabric

CLI (network-admin@switch) > subnet-create name subnet-vxlan-500012 scope
fabric vxlan 500012 network 172.10.2.0/24 anycast-gw-ip 172.10.2.1 vrf VRF1
```

L3 VNIs also enable a new forwarding mode (see the two hop model described above), which is used for inter-pod VXLAN routing.

Once configured, you can check the VRFs with the `vrf-show` command:

```
CLI (network-admin@switch) > vrf-show format name,vnet,anycast-mac,l3-vni,active,hw-router-
mac,hw-vrid,flags,enable

name vnet anycast-mac       l3-vni active hw-router-mac     hw-vrid flags  enable
---- ---- ----------------- ------ ------ ----------------- ------- ------ ------
VRF1 0:0  64:0e:94:40:00:02 101000 yes    66:0e:94:48:68:a7 1       subnet yes
```

> Note that the user doesn't need to confgure `vrf-gw` and `vrf-gw2` with EVPN except on border gateways connected to DC gateways for North South traffic. Traffic reaches the border gateways with the default route described in ***Figure 9-3*** above.

Configuring Route Maps for EVPN Filtering

Since EVPN is based on a BGP configuration between the border gateways, it's a natural extension (and general practice) to use route maps for filtering.

A user can create a route map using the match/set/permit/deny statements, and then apply it to the BGP EVPN configuration, inbound or outbound.

When a route map is applied inbound, it ensures that the match statement is applied to the routes that are learned from a BGP neighbor. When it is applied outbound, it ensures that the match statement is applied to the routes that are sent to a BGP neighbor.

Starting from release 7.0.0 NetVisor OS supports the configuration of route maps that support the following EVPN-related statements both in the inbound and the outbound direction:

- `match-evpn-default-route`
- `match-evpn-route-type`
- `match-evpn-vni`

- `match-evpn-rd`

These statements can match a default route (such as 0.0.0.0/0 or 0::0/0), a route type (Type 2, 3, 5 routes), a VNI (L2 or L3), and a route distinguisher (Type 2, 3, and 5 routes carry an RD field in them). In addition, it is possible to apply *IP prefix li*sts to the filters.

This enhancement extends the existing NetVisor OS `vrouter-route-map-add/-modify` commands, as described below:

```
CLI (network-admin@switch) > vrouter-route-map-add vrouter-name leaf1 name
map
action permit | deny
```

with the additional parameters:

```
match-evpn-default-route
match-evpn-route-type
match-evpn-vni
match-evpn-rd
```

> **Note**: This filtering capability must be configured on the EVPN border gateways. In redundant gateways, it must be enabled on both cluster nodes to ensure that the pair of EVPN border nodes filter routes symmetrically for the pod.

The following example filters Type 2 and Type 5 routes:

```
CLI (network-admin@switch) > vrouter-route-map-add vrouter-name vr1 name
type52
action deny seq 1 match-evpn-route-type  type-5

CLI (network-admin@switch) > vrouter-route-map-add vrouter-name vr1 name
type52
action deny seq 2 match-evpn-route-type  type-2

CLI (network-admin@switch) > vrouter-route-map-add vrouter-name vr1 name
type52
action permit seq 3
```

Note that the third closing command is required (as *permit all*) for the entire route map to work.

Next, for example, the route map can be applied in the outbound direction like so:

```
CLI (network-admin@switch) > vrouter-bgp-modify vrouter-name vr1 neighbor
190.12.1.2 route-map-out  type52
```

to stop advertising EVPN routes to the neighbor specified in the route map.

Additionally, it is also possible to filter the incoming routes from the matching neighbor like so:

```
CLI (network-admin@switch) > vrouter-bgp-modify vrouter-name vr1 neighbor
190.12.1.2 route-map-in  type52
```

The next example shows how to configure an IP filter list (`100`) and then apply it to a route map (`rmap1`) to filter EVPN routes in the outbound direction:

```
CLI (network-admin@switch) > vrouter-prefix-list-add vrouter-name vr1 name
100
seq 1 prefix 100.1.1.0/24 action permit

CLI (network-admin@switch) > vrouter-prefix-list-add vrouter-name vr1 name
100
seq 2 prefix 101.1.1.0/24 action deny

CLI (network-admin@switch) > vrouter-route-map-add vrouter-name vr1 name
rmap1
action permit match-prefix 100 match-evpn-route-type type-5 seq 1

CLI (network-admin@switch) > vrouter-bgp-modify vrouter-name vr1 neighbor
190.1.2.3 route-map-out 100

CLI (network-admin@switch) > vrouter-route-map-show format
vroutername,name,seq,action,match-prefix,match-evpn-route-type

vrouter-name  name   seq  action  match-prefix  match-evpn-route-type
------------  -----  ---  ------  ------------  --------------------
vr1           rmap1 10  deny   100           type-5
vr1            rmap1 20  permit none
```

On the neighboring node, route 101.1.1.0 is present, because it's not filtered by the prefix list 100, but 101.1.1.0 gets filtered as shown below:

```
CLI (network-admin@switch) > vrouter-evpn-bgp-routes-show route-type show-
interval 2 format
vrouter-name,vni,ip,route-type,next-hop,extended-community

vrouter-name vni    ip             route-type next-hop    extended-community
------------  -----  ------------  ----------  ----------  ----------------------------------------
vr4           13434 101.1.1.0/24 5            190.11.1.1 RT:65011:1003434 ET:8 Rmac:00:00:5e:00:01:64
vr4            13434 33.3.1.0/29  5            190.11.1.1 RT:65011:1003434 ET:8 Rmac:00:00:5e:00:01:64
```

Configuring Virtual Service Groups with EVPN

Starting from release 7.0.0 NetVisor OS supports the configuration of Virtual Service Groups (vSGs) not just in the local pod but also across EVPN pods. Refer to the *Configuring Virtual Service Groups* section in the *Configuring VXLAN* chapter for more details on the functionality.

vSGs can be used to import/export (i.e., 'leak') subnet prefixes between Unicast Fabric VRFs.
Starting from release 7.0.0, it is possible to decide which subnets/prefixes to import/export locally and which ones to import/export over an EVPN transport using Type-5 routes. The selection can be configured based on the new parameters discussed below and is needed on one EVPN pod only, from which the shared prefixes can be extended to the other pods.

By default, exported (i.e., 'leaked') networks remain local and are not extended over EVPN, so this behavior is consistent with prior releases. However, it can be changed when adding a VRF to a vSG with the `export-`

leaked-networks parameter:

```
CLI (network-admin@switch) > vsg-vrf-add vsg-name <vsg_name> vrf <vrf>
[export-leaked-networks evpn | none]
```

in which none is the default. For example, you can enable the extension over EVPN of the leaked networks/subnets associated with vrf3 like so:

```
CLI (network-admin@switch) > vsg-create name vsg-1

CLI (network-admin@switch) > vsg-vrf-add vsg-name vsg-1 vrf vrf3 [type
promiscuous] export-leaked-networks evpn

CLI (network-admin@switch*) > vsg-vrf-show

CLI (network-admin@switch*) > vsg-vrf-show

vsg-name vnet  vrf   type        export-leaked-networks
-------- ----- ----- ----------- ------------------------
vsg-1    0:0   vrf3  promiscuous evpn
```

The vsg-vrf-remove command would be used to remove a VRF from a vSG and consequently also stop any extension across pods, if enabled.

In a VRF that is extended over EVPN you can also decide which subnets to share with the subnet-create or subnet-modify commands. For that you need to use the evpn-export-leaked-networks parameter, which defaults to permit:

```
CLI (network-admin@switch) > subnet-create name <name> … [evpn-export-
leaked-networks permit | deny]
```

In other words, when the evpn-export-leaked-networks parameter is not specified leaked subnets are extended over EVPN by default (when export-leaked-networks is set to evpn). During creation or modification, you can elect not to extend a leaked subnet like so:

```
CLI (network-admin@switch) > subnet-create name vlan-235 vlan 235 vxlan
10235 vrf vrf3 network 13.1.5.0/24 anycast-gw-ip 13.1.5.254 network6
2002:13:1:5::/64 anycast-gw-ip6 2002:13:1:5::254 evpn-export-leaked-
networks deny

CLI (network-admin@switch*) > subnet-show name vlan-235 format
name,scope,vlan,vxlan,vrf,network,state,enable,evpn-export-leaked-subnets

name     scope  vlan vxlan  vrf  network     state enable evpn-export-leaked-subnets
-------- ------ ---- ------ ---- ----------- ----- ------ -------------------------
vlan-235 fabric 235  10235  vrf3 13.1.5.0/24 ok    yes    deny
```

CLI (network-admin@switch) > vsg-network-add vsg-name vsg-1 vrf vrf3 subnet vlan-235

Similarly, you can also decide to extend a specific network in the vsg-network-add/modify commands like so:

```
CLI (network-admin@switch) > vsg-network-add vsg-name vsg1 vrf vrf3 network
```

```
101.1.1.0/30
```

Configuring Bridge Domains with EVPN

Starting from release 7.0.0 NetVisor OS supports the configuration of bridge domains (along with VLANs) across multiple EVPN pods. (For more information on bridge domains, refer to the *Configuring Advanced Layer 2 Transport Services* chapter.)

The configuration of a VXLAN bridge domain (BD in short) does not change with the addition of the EVPN support: the main (and only) requirement for a bridge domain that is extended across EVPN pods is to use a *common VXLAN ID*. In other words, such ID has global significance and represents the bridge domain end-to-end. Note that this ID cannot be shared with VLAN-based VXLAN configurations.

A bridge domain supports a variety of modes (dot1q, q-in-q and untagged) as well as of tagging schemes (auto, remove-tags and transparent). All of them are transparently supported with EVPN as well.

> **Note**: Starting from release 7.0.0, NetVisor OS supports bridge domains over EVPN on the following platforms:
>
> - F9480-V (AS7326-56X), F9460-X (AS5835-54X), F9460-T (AS5835-54T)
> - Dell S4100 and S5200 Series
> - NRU03, and NRU-S0301

To create a BD in an EVPN pod, you can use the regular `bridge-domain-create` command (as also described in the respective chapter), for example like so:

```
CLI (network-admin@switch) > bridge-domain-create name bd-evpn1 scope
fabric vxlan 500500 rsvd-vlan 4030
```

To add a port in the pod to a BD, you can use for example the command:

```
CLI (network-admin@switch) > bridge-domain-port-add name bd-evpn1 port 81
vlans 55
```

With EVPN you can also create another BD (name) in another pod, for example like so:

```
CLI (network-admin@switch2) > bridge-domain-create name bd-evpn2 scope
fabric vxlan 500500 rsvd-vlan 4050
```

And add a port to it:

```
CLI (network-admin@switch2) > bridge-domain-port-add name bd-evpn2 port 31
vlans 500
```

Note that the the *BD names and VLAN numbers* in the two pods can even be different but the VXLAN ID *must be the same*. The common VXLAN ID represents a unique forwarding domain end-to-end. Obviously in many cases, for consistency's sake, it can be convenient to use the same BD name end-to-end. Also in

certain scenarios, when possible, using consistent VLAN numbers end-to-end can be desirable to simplify the network design.

You can check the BD configuration on a non-border gateway node:

```
CLI (network-admin@switch*) > l2-table-show bd bd-evpn1 format
mac,bd,vlan,vxlan,ip,ports,state,status,vtep-ip

mac               bd       vlan vxlan  ip           ports state       status vtep-ip
----------------- -------- ---- ------ ------------ ----- ----------- ------ -----------
00:12:c0:88:07:32 bd-evpn1      500500 205.205.1.64       tunnel,evpn        93.93.93.93
00:12:c0:80:33:1e bd-evpn1 55   500500 205.205.1.51 81    active      host
```

`vtep-ip 93.93.93.93` corresponds to the border gateway's VTEP:

```
CLI (network-admin@bgw*) > fabric-evpn-node-show

fabric-pod evpn-node version        vtep-ip     evpn-role
---------- --------- -------------- ----------- ---------
evpn-fab1  bgw       6.2.6020018465 93.93.93.93 primary
```

```
CLI (network-admin@bgw*) > vtep-show name vtep9

scope  name  location vrouter-name ip         virtual-ip  mac-learning
------ ----- -------- ------------ ---------- ----------- ------------
fabric vtep9 bgw      vr9          93.93.93.9 93.93.93.93 on
```

The BD extends to the border gateway (and beyond):

```
CLI (network-admin@bgw*) > l2-table-show bd bd-evpn1 format
mac,bd,vlan,vxlan,ip,ports,state,status,vtep-ip

mac               bd       vlan vxlan  ip           state       status vtep-ip
----------------- -------- ---- ------ ------------ ----------- ------ ----------
00:12:c0:88:07:32 bd-evpn1      500500 205.205.1.64 tunnel,evpn        22.22.22.2
00:12:c0:80:33:1e bd-evpn1 55   500500 205.205.1.51 tunnel      host
```

The BD is extended to Pod 2 too, as shown below on a non-border gateway node:

```
CLI (network-admin@switch2*) > l2-table-show bd bd-evpn2 format
mac,bd,vlan,vxlan,ports,state,status,vtep-ip

mac               bd       vlan vxlan  ports state                     status vtep-ip
----------------- -------- ---- ------ ----- ------------------------- ------ ----------
00:12:c0:88:07:32 bd-evpn2 500  500500 31    active                    host
00:12:c0:80:33:1e bd-evpn2      500500       tunnel,local-tunnel,evpn        22.22.22.2
```

This extension happens transparently as there is no BD-specific information in BGP:

```
CLI (network-admin@bgw*) > vrouter-evpn-bgp-routes-show format vrouter-
name,vni,mac,ip,route-type,extended-community

vrouter-name vni    mac               ip              route-type extended-community
------------ ------ ----------------- --------------- ---------- --------------------
vr9          500500 00:12:c0:88:07:32                 2          RT:5002:500500 ET:8
vr9          500500 00:12:c0:88:07:32 205.205.1.64/32 2          RT:5002:500500 ET:8
```

Note that, as with VLANs when configured with EVPN, a BD is enabled once the BD's VXLAN ID is added to the respective border gateway's VTEP and that same VXLAN ID is configured consistently on all the pods, as shown below on two nodes of different pods.

First make sure EVPN is enabled:

```
CLI (network-admin@bgw*) > vrouter-show name vr9 format evpn-border,

evpn-border
-----------
enable
```

Then you can check that the BD's VXLAN ID is correctly configured in Pod 1:

```
CLI (network-admin@bgw*) > vtep-vxlan-show name vtep9 vxlan 500500

name  vxlan  isolated
----- ------ --------
vtep9 500500 no
```

And also in Pod 2:

```
CLI (network-admin@bgw2*) > vtep-vxlan-show name vtep2 vxlan 500500

name  vxlan  isolated
----- ------ --------
vtep2 500500 no
```

Note: A hardware limitation is that you need to configure a BD port on the border gateway, regardless of whether there needs to be a host attached. So on the border gateway for BD configurations a port needs to be set aside anyway.

Configuring and Displaying MAC Mobility

MAC mobility is handled automatically by the EVPN control plane. However, it is important to deal with duplicate MAC address scenarios appropriately. Therefore, some special parameters are available to help with the remediation in such scenarios, as explained in the *About MAC Mobility with EVPN* section above.

You can (optionally) configure the three duplicate MAC address parameters on a per vRouter basis with the following command:

```
CLI (network-admin@switch) > vrouter-create name <vr-name> evpn-dup-addr-max-moves <count> evpn-dup-addr-moves-duration <seconds> evpn-dup-addr-freeze <seconds>
```

When not specified, the default values are:

- evpn-dup-addr-max-moves: 5
- evpn-dup-addr-moves-duration: 180
- evpn-dup-addr-freeze: 180

You can also modify those parameters with the `vrouter-modify` command.

For example, let's consider the case in which those three parameters are modified from the default values and are configured to 8, 401 and 301, respectively. You can display the new values with the following condensed command:

```
CLI (network-admin@switch*) > vrouter-show format name,evpn-border,evpn-dup-addr-max-moves,evpn-dup-addr-moves-duration,evpn-dup-addr-freeze,evpn-border

name     evpn-border evpn-dup-addr-max-moves(s) evpn-dup-addr-moves-duration(s) evpn-dup-addr-freeze
-------- ----------- -------------------------- ------------------------------ --------------------
vRouter1 enable      8                          401                            301
vRouter1 enable      8                          401                            301
```

In this case, if 8 MAC moves are detected in a 401 second time window, the duplicate MAC address entry is frozen for 301 seconds to facilitate the operator in the remediation. The frozen entry and the corresponding sequence number received before the 8th MAC move can be displayed with the following command:

```
CLI (network-admin@switch) > vrouter-evpn-duplicate-mac-show
switch       vrouter-name    host-mac          seq
------------ --------------- ----------------- ---
switch       vRouter1        00:12:c0:80:33:6a 7
```

This output will clear after *dup-addr-freeze* (180, by default) seconds have elapsed.

Furthermore, the total MAC move count can be periodically checked in the MM (MAC Move) field with this command:

```
CLI (network-admin@switch) > switch * vrouter-evpn-bgp-routes-show route-type 2 format vrouter-name,rd,vni,mac,route-type,next-hop,extended-community
```

```
switch  vrouter-name rd        vni    mac               route-type next-hop  path  extended-community
------- ------------ --------- ------ ----------------- ---------- --------- ----- ------------------------
switch  vr2          2.2.0.1:2 100100 2e:d7:27:b9:11:6d 2          20.0.12.1 66001 RT:465:100100 ET:8 MM:48
switch1 vr1          2.2.0.1:2 100100 2e:d7:27:b9:11:6d 2          20.0.12.1 66001 RT:465:100100 ET:8 MM:48
```

In addition, typically for troubleshooting purposes, you can see each MAC move being notified and logged on a node by using the following command and looking for the string `action MAC_MOVE`:

```
CLI (network-admin@switch) > vrouter-log-show vrouter-name vr2 protocol
evpnsnoop

log-message

---------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------
-----------------------------
---------------------------------------------------------------------------------------------------
-------------
<snip>
2021-05-21,05:24:26.075.:rs_msg.c:624:rs_msg_vport_update_event_cb   L2_UPDATE  mac:
2e:d7:27:b9:11:6d,   log_type l2-modify caller cluster-status: vxlan 100100, vlan 100
reason:modify,evpn-mac-move owner_flags 0x0, over_
tunnel 1791,  2e:d7:27:b9:11:6d action MAC_MOVE ip 10.0.100.30
<snip>
```

which shows the MAC move happening due to a certain host configured with a certain MAC and IP address pair. That information can be compared to the same command output obtained on the other node where the address duplication is also happening.

Configuring Optimized Allocation of EVPN Host Routes

Starting from NetVisor OS release 7.0.0 it is possible to leverage the host route hardware table to allocate IPv4 /32 and IPv6 /128 routes when advertised by EVPN. This is advantageous as the host route table provides ample additional allocation space (its size depends on the specific ASIC used by the switch).

To enable/disable the allocation in the host route table, you can use the following command:

```
CLI (network-admin@switch) > system-settings-modify no-l3-use-host-table |
l3-use-host-table
```

The functionality is disabled by default. When it gets enabled, a switch reboot is requested with the following output message:

```
"!!!! Please reboot the system for the new
l3-use-host-table setting to take effect correctly !!!!"
```

You can verify the current configuration with this command:

```
CLI (network-admin@switch) > system-settings-show format l3-use-host-table

l3-use-host-table: on
```

After being enabled and the switch being rebooted, a /32 IPv4 route or a /128 IPv6 route advertised by EVPN will get installed into the host table, for example as shown below:

```
CLI (network-admin@switch) > vrouter-evpn-bgp-routes-show | grep 2001

vrouter-name rd        vni    mac               ip              route-type next-hop path extended-community

------------ --------- ------ ----------------- --------------- ---------- -------- ----
-------------------------------------------------
vr3            11.1.1.2: 1005    00:12:c0:80:1b:e7 2001:20::10/128 2           11.1.1.2 101   RT:101:1005
RT:101:10020  ET:8  Rmac:66:0e:94:68:86:c2

CLI (network-admin@switch) > vrouter-fib-arps-show | grep 2001

vrid ip          if-id ports vnet bd vlan mac               egress-id flags
---- ----------- ----- ----- ---- -- ---- ----------------- --------- --------
0    2001:20::10 2     22             1005 66:0e:94:68:86:c2 100061    Trunk
```

Displaying VXLAN Features with EVPN

A multi-pod EVPN configuration requires the extension of the intra-pod features commonly used for single-fabric setups. Therefore, before proceeding, refer to the VXLAN commands described in the *Configuring VXLAN* chapter above for reference.

VTEPs are the VXLAN termination points: NetVisor OS provides a CLI construct called *VTEP object* to automatically set up a mesh of bidirectional connections between the end points.
Multi-pod EVPN designs comprise two or more pods in which at least one node per pod is designated as *border gateway*, as explained earlier. Border gateways are interconnected by EVPN-derived automatic VXLAN tunnels, as shown in dark red color in the figure below for POD 1 and POD 2.

In the network design in ***Figure 10-6***, for redundancy, two pairs of clustered leaf nodes are selected as border gateways (highlighted in yellow and green).

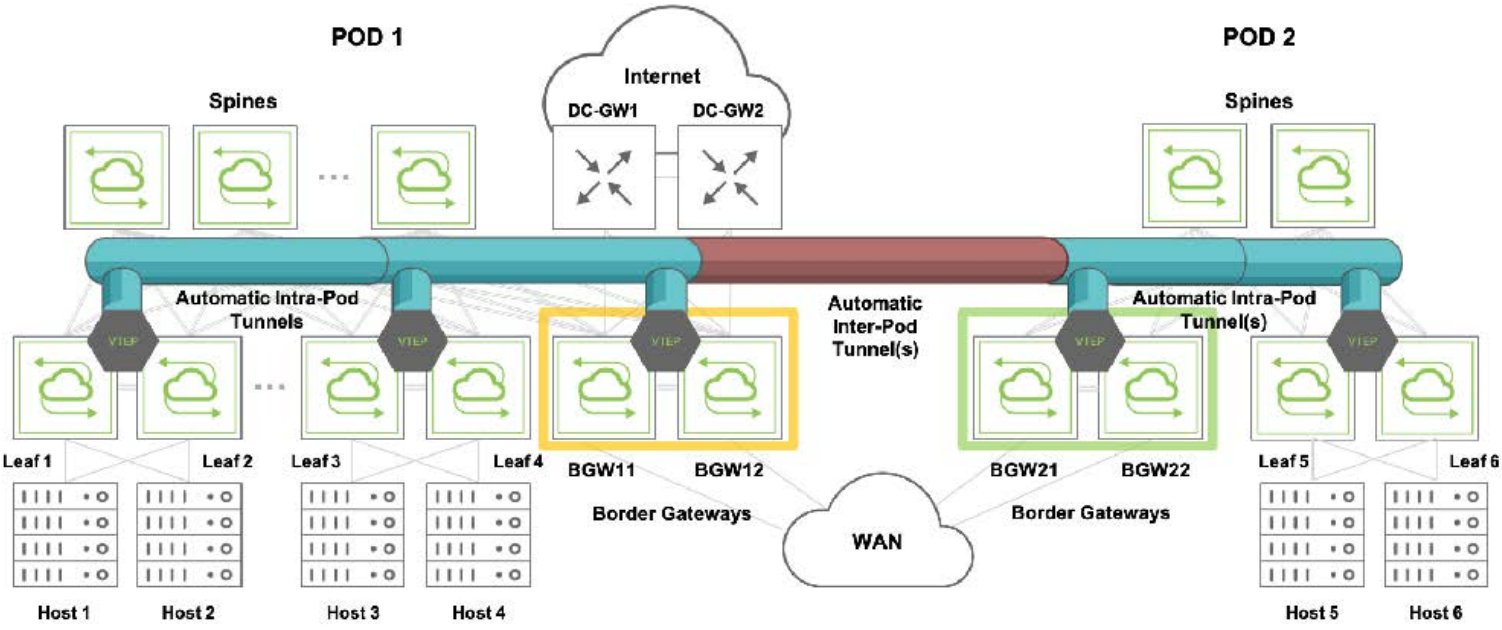


Figure 10-6: Border Gateways vs. Non-Border Gateways

Displaying VTEPs

VTEPs can be created normally in each pod on all the leaf node pairs to automatically set up the intra-pod tunnels (in blue color in the figure above).
However, when the border gateways are configured and actively exchanging EVPN messages a pair of automatic VTEP objects (and the corresponding tunnels) are created as displayed in the output below:

```
CLI (network-admin@bgw11) > vtep-show format
switch,scope,name,location,vrouter-name,ip,virtual-ip,

switch scope  name               location     vrouter-name    ip           virtual-ip
------ ------ ------------------- ------------ --------------- ------------ ------------
       fabric vtep-leaf1          leaf1        vrouter-leaf-1  172.16.128.1 172.16.128.3
```

```
       fabric vtep-leaf2          leaf2        vrouter-leaf-2  172.16.128.2 172.16.128.3
       fabric vtep-leaf3          leaf3        vrouter-leaf-3  172.16.129.1 172.16.129.3
       fabric vtep-leaf4          leaf4        vrouter-leaf-4  172.16.129.2 172.16.129.3
       fabric vtep-leaf5          leaf5        vrouter-leaf-5  172.16.130.1 172.16.130.3
       fabric vtep-leaf6          leaf6        vrouter-leaf-6  172.16.130.2 172.16.130.3
       fabric vtep-bgw11          bgw11        vrouter-bgw-11  172.16.133.1 172.16.133.3
       fabric vtep-bgw12          bgw12        vrouter-bgw-12  172.16.133.2 172.16.133.3
bgw11  local  __evpn__172.16.134.3 host-external               172.16.134.3 ::
bgw12  local  __evpn__172.16.134.3 host-external               172.16.134.3 ::
```

Each automatic VTEP is created with a local scope and the `host-external` location string on the two border gateways only. The name starts with the `__evpn__` string followed by the (virtual) IP address of the neighboring border gateway (pair).

EVPN VTEPs can be displayed with the following command and identified from their name and from the `vtep-source` column:

```
CLI (network-admin@bgw11*) > vtep-show format name,vtep-source
name                  vtep-source
--------------------  -----------
vtep-bgw11            config
__evpn__172.16.134.3  evpn
```

Displaying Automatic Tunnels

Tunnels are created automatically between border gateways for inter-pod communication. They can be identified from the `type` column in the following command:

```
CLI (network-admin@bgw11*) > tunnel-show format name,type,
name                                    type
--------------------------------------  ----------
auto-tunnel-190.11.1.1_65.1.1.10        vxlan
auto-tunnel-172.16.133.3_172.16.134.3   vxlan-evpn
```

Displaying EVPN Border Gateways

The role of border gateway, as configured by the user, can be shown with the expanded `vtep-show format all` command that includes the `evpn-border` column:

```
CLI (network-admin@bgw11) > vtep-show format all

switch scope  name                 location      vrouter-name    ip           virtual-ip   mac               description mac-learning evpn-border
------ ------ -------------------- ------------- --------------- ------------ ------------ ----------------- ----------- ------------ -----------
       fabric vtep-leaf1           leaf1         vrouter-leaf-1  172.16.128.1 172.16.128.3 00:00:5e:00:01:01             on           false
       fabric vtep-leaf2           leaf2         vrouter-leaf-2  172.16.128.2 172.16.128.3 00:00:5e:00:01:01             on           false
       fabric vtep-leaf3           leaf3         vrouter-leaf-3  172.16.129.1 172.16.129.3 00:00:5e:00:01:02             on           false
       fabric vtep-leaf4           leaf4         vrouter-leaf-4  172.16.129.2 172.16.129.3 00:00:5e:00:01:02             on           false
       fabric vtep-leaf5           leaf5         vrouter-leaf-5  172.16.130.1 172.16.130.3 00:00:5e:00:01:03             on           false
       fabric vtep-leaf6           leaf6         vrouter-leaf-6  172.16.130.2 172.16.130.3 00:00:5e:00:01:03             on           false
       fabric vtep-bgw11           bgw11         vrouter-bgw-11  172.16.133.1 172.16.133.3 00:00:5e:00:01:06             on           true
       fabric vtep-bgw12           bgw12         vrouter-bgw-12  172.16.133.2 172.16.133.3 00:00:5e:00:01:06             on           true
bgw11  local  __evpn__172.16.134.3 host-external                 172.16.134.3 ::           00:00:00:00:00:00             on           false
bgw12  local  __evpn__172.16.134.3 host-external                 172.16.134.3 ::           00:00:00:00:00:00             on           false
```

The role of a border gateway can also be verified by using the `vrouter-show` command; however, the basic command may not be sufficient. To obtain additional info, you may use the `format-all` option. Alternatively, you can use the `evpn-border` filter to display only the border gateway node(s) for example like so:

```
CLI (network-admin@bgw11) > vrouter-show format name,hw-router-mac,evpn-border

name            hw-router-mac     evpn-border
--------------- ----------------- -----------
vrouter-bgw-11  66:0e:94:14:f3:5b enable
vrouter-bgw-12  66:0e:94:22:d7:1c enable
vrouter-leaf-1  66:0e:94:ad:d4:41 disable
vrouter-leaf-2 66:0e:94:95:f9:74  disable
<snip>
```

The border gateway role is also reflected in the BGP configuration, as discussed above and as shown in this command output:

```
CLI (network-admin@bgw11*) > vrouter-bgp-show multi-protocol l2vpn-evpn

vrouter-name    neighbor     remote-as next-hop-self prepend replace-as ebgp-multihop update-source route-reflector-client override-capability soft-reconfig-inbound max-prefix-warn-only bfd multi-protocol weight default-originate neighbor-keepalive-interval(s) neighbor-holdtime(s) dup-addr-max-moves(s) dup-addr-moves-duration(s) dup-addr-freeze connect-retry-interval(s) send-community allowas-in allowas-in-origin as-override advertisement-interval(s) description
-------------- ------------ --------- ------------- ------- ---------- ------------- ------------- --------------------- -------------------
-------------------- ------------------- --- ------------- ------ ---------------- ---------------------------- -------------------
-------------------- ------------------------ -------------- ----------------------- ------------- ---------- ---------------- -----------
------------------------ -----------
vrouter-bgw-11  172.16.134.1 66611     no            yes     no         4            172.16.133.1  no                    no                   yes
           no                   no  l2vpn-evpn     none   no                60                             180                  5
  180                   180           15                         yes           no         no                no          0
  none
vrouter-bgw-11  172.16.134.2 66611     no            yes     no         4            172.16.133.1  no                    no                   yes
           no                   no  l2vpn-evpn     none   no                60                             180                  5
  180                   180           15                         yes           no         no                no          0
  none
```

The above verbose command output can be compacted, for example like so, to get an abridged overview of

the EVPN neighborship:

```
CLI (network-admin@bgw11*) > vrouter-bgp-show format ,neighbor,remote-
as,update-source,multi-protocol

vrouter-name   neighbor      remote-as update-source multi-protocol
-------------- ------------ --------- ------------- --------------
vrouter-bgw-11 172.16.0.161 64311                   ipv4-unicast
vrouter-bgw-11 172.16.2.161 64312                   ipv4-unicast
vrouter-bgw-11 172.16.4.161 64313                   ipv4-unicast
vrouter-bgw-11 172.16.6.161 64314                   ipv4-unicast
vrouter-bgw-11 172.16.133.2 64518                   ipv4-unicast
vrouter-bgw-11 101.1.1.1    60000                   ipv4-unicast
vrouter-bgw-11 172.16.134.1 66611     172.16.133.1  l2vpn-evpn
vrouter-bgw-11 172.16.134.2 66611     172.16.133.1  l2vpn-evpn
```

Displaying VNIs

VTEPs' VNI associations can be displayed with the `vtep-vxlan-show` command locally on the border gateway(s):

```
CLI (network-admin@bgw11*) > vtep-vxlan-show
name                 vxlan  isolated
-------------------- ------ --------
vtep-leaf1           10201  no
vtep-leaf1           103501 no
vtep-leaf1           10403  no
vtep-leaf2           10201  no
vtep-leaf2           103501 no
vtep-leaf2           10403  no
vtep-leaf3           10201  no
vtep-leaf3           103501 no
vtep-leaf3           10403  no
vtep-leaf4           10201  no
vtep-leaf4           103501 no
vtep-leaf4           10403  no
vtep-leaf5           10201  no
vtep-leaf5           103501 no
vtep-leaf5           10403  no
vtep-leaf6           10201  no
vtep-leaf6           103501 no
vtep-leaf6           10403  no
vtep-bgw11           10201  no
vtep-bgw11           103501 no
vtep-bgw11           10403  no
vtep-bgw12           10201  no
vtep-bgw12           103501 no
vtep-bgw12           10403  no
__evpn__172.16.134.3 10201  no
__evpn__172.16.134.3 103501 no
```

In the above example, note that VNIs 10201 and 103501 are present on all the leaf nodes including the border gateways' VTEPs and the automatically created `host-external` ones.

In other words, this command can be used to verify that the inter-pod VNIs are configured consistently on each pod's VTEPs and that, as a consequence, the common VNIs get added to the automatic inter-pod tunnels too.

Let's suppose the number of VNIs grows pretty large. Then, you can limit the output of the display for example to a particular EVPN VTEP of interest, like so:

```
CLI (network-admin@bgw11*) > vtep-vxlan-show name __evpn__172.16.134.3
name                  vxlan  isolated
--------------------  ------  --------
__evpn__172.16.134.3  10201   no
__evpn__172.16.134.3  10202   no
__evpn__172.16.134.3  10203   no
__evpn__172.16.134.3  10204   no
__evpn__172.16.134.3  10205   no
...
```

Displaying EVPN Route Types

In order to display the supported EVPN route types for VXLAN (encapsulation type 8) received by a border gateway node, you can use the following command:

```
CLI (network-admin@bgw11*) > vrouter-evpn-bgp-routes-show format vrouter-
name,rd,vni,network,route-type,next-hop,path,extended-community
```

```
vrouter-name   rd             vni    network                 route-type next-hop     path extended-community
-------------- -------------- ------ ----------------------- ---------- ------------ ---- ----------------------------------------
vrouter-bgw-11 0.0.0.0:2      103501 [24]:[10.2.3.0]         5          172.16.133.3      ET:8 RT:64518:103501 Rmac:00:00:5e:00:01:06
vrouter-bgw-11 0.0.0.0:2      103501 [24]:[10.2.4.0]         5          172.16.134.3      RT:1075:103501 ET:8 Rmac:00:00:5e:00:01:01
vrouter-bgw-11 0.0.0.0:2      103501 [24]:[10.2.4.0]         5          172.16.134.3      RT:1075:103501 ET:8 Rmac:00:00:5e:00:01:01
vrouter-bgw-11 192.168.0.11:3 10201  [48]:[00:3b:01:00:00:01] 2          172.16.133.3      ET:8 RT:64518:10201
vrouter-bgw-11 192.168.0.11:3 10201  [32]:[172.16.133.3]      3          172.16.133.3      ET:8 RT:64518:10201
vrouter-bgw-11 192.168.0.11:4 10403  [32]:[172.16.133.3]      3          172.16.133.3      ET:8 RT:64518:10403
vrouter-bgw-11 192.168.0.13:3 10404  [32]:[172.16.134.3]      3          172.16.134.3      RT:1075:10404 ET:8
vrouter-bgw-11 192.168.0.13:4 10201  [48]:[00:3d:01:00:00:01] 2          172.16.134.3      RT:1075:10201 ET:8
vrouter-bgw-11 192.168.0.13:4 10201  [32]:[172.16.134.3]      3          172.16.134.3      RT:1075:10201 ET:8
vrouter-bgw-11 192.168.0.14:3 10404  [32]:[172.16.134.3]      3          172.16.134.3      RT:1075:10404 ET:8
vrouter-bgw-11 192.168.0.14:4 10201  [48]:[00:3d:01:00:00:01] 2          172.16.134.3      RT:1075:10201 ET:8
vrouter-bgw-11 192.168.0.14:4 10201  [32]:[172.16.134.3]      3          172.16.134.3      RT:1075:10201 ET:8
```

Note that the route type is listed in a separate column. The EVPN route contents are shown in the `network` column (IP addresses for type 5, MAC addresses for type 2, etc.). Also note that EVPN Route Targets (RT) and Route Distinguishers (RD) are displayed. The RT field is used to transport the *VNI values*.
You can selectively display a single route type with these commands:

```
CLI (network-admin@bgw11*) > vrouter-evpn-bgp-routes-show route-type 2
format    vrouter-name,rd,vni,network,route-type,next-hop,path,extended-
community
```

```
vrouter-name   rd             vni   network                 route-type next-hop      path extended-community
-------------- -------------- ----- ----------------------- ---------- ------------ ----
-------------------
vrouter-bgw-11 192.168.0.11:3 10201 [48]:[00:3b:01:00:00:01] 2          172.16.133.3      ET:8
RT:64518:10201
vrouter-bgw-11 192.168.0.13:4 10201 [48]:[00:3d:01:00:00:01] 2          172.16.134.3      RT:1075:10201 ET:8
vrouter-bgw-11 192.168.0.14:4 10201 [48]:[00:3d:01:00:00:01] 2          172.16.134.3      RT:1075:10201 ET:8
```

```
CLI (network-admin@bgw11*) > vrouter-evpn-bgp-routes-show route-type 3
```

```
format    vrouter-name,rd,vni,network,route-type,next-hop,path,extended-
community

vrouter-name   rd              vni   network              route-type  next-hop      path  extended-community
-------------  --------------  -----  -------------------  ----------  ------------  ----  -------------------
vrouter-bgw-11 192.168.0.11:3 10201 [32]:[172.16.133.3] 3             172.16.133.3        ET:8 RT:64518:10201
vrouter-bgw-11 192.168.0.11:4 10403 [32]:[172.16.133.3] 3             172.16.133.3        ET:8 RT:64518:10403
vrouter-bgw-11 192.168.0.13:3 10404 [32]:[172.16.134.3] 3             172.16.134.3       RT:1075:10404 ET:8
vrouter-bgw-11 192.168.0.13:4 10201 [32]:[172.16.134.3] 3             172.16.134.3       RT:1075:10201 ET:8
vrouter-bgw-11 192.168.0.14:3 10404 [32]:[172.16.134.3] 3             172.16.134.3       RT:1075:10404 ET:8
vrouter-bgw-11 192.168.0.14:4 10201 [32]:[172.16.134.3] 3             172.16.134.3       RT:1075:10201 ET:8
```

```
CLI (network-admin@bgw11*) > vrouter-evpn-bgp-routes-show route-type 5
format    vrouter-name,rd,vni,network,route-type,next-hop,extended-community

vrouter-name   rd        vni    network         route-type next-hop     extended-community
-------------- --------- ------ --------------- ---------- ------------ ----------------------------------------
vrouter-bgw-11 0.0.0.0:2 103501 [24]:[10.2.3.0] 5          172.16.133.3 ET:8 RT:64518:103501 Rmac:00:00:5e:00:01:06
vrouter-bgw-11 0.0.0.0:2 103501 [24]:[10.2.4.0] 5          172.16.134.3 RT:1075:103501 ET:8 Rmac:00:00:5e:00:01:01
vrouter-bgw-11 0.0.0.0:2 103501 [24]:[10.2.4.0] 5          172.16.134.3 RT:1075:103501 ET:8 Rmac:00:00:5e:00:01:01
```

Displaying VRFs

VRF information, including the associated L3 VNIs, can be displayed with the command:

```
CLI (network-admin@bgw11*) > vrf-show
```

```
name       vnet scope  anycast-mac       vrf-gw vrf-gw2 vrf-gw-ip6 vrf-gw2-ip6 l3-vni active hw-router-mac     hw-vrid flags  enable description
---------- ---- ------ ----------------- ------ ------- ---------- ----------- ------ ------ ----------------- ------- ------ ------ -----------
vrf-pod1-1 0:0  fabric 64:0e:94:40:00:02 ::     ::      ::         ::          103501 yes    66:0e:94:13:bd:c8 1       subnet yes
```

Subnet information, including the associated VRF, can be displayed with the command:

```
CLI (network-admin@bgw11*) > subnet-show
```

```
name      scope   vlan vxlan vrf          network      anycast-gw-ip packet-relay forward-proto state enable
--------  ------  ---- -----  ----------  -----------  ------------  ------------  -------------  -----  ------
vlan-403 fabric 403   10403 vrf-pod1-1 10.2.3.0/24 10.2.3.1       disable       dhcp            ok     yes
```

Displaying Routing Information

To verify the RIB routes associated with EVPN, you can use either of the following commands:

```
CLI (network-admin@bgw11*) > vrouter-rib-routes-show vrid 1
```

```
vrid ip        prelen number-of-nexthops nexthop       flags                vnet bd vlan intf-ip  intf-id
---- --------  ------  ------------------ ------------ ------------------ ---- -- ---- --------  -------
1    10.2.4.0 24     1                  172.16.134.3 in-hw,evpn                 3501 ::       10
1    10.2.3.0 24     1                  10.2.3.1     in-hw,local-subnet         403  10.2.3.1 11
```

CLI (network-admin@bgw11*) > vrouter-rib-routes-show vrid 1 flags evpn

```
vrid ip        prelen number-of-nexthops nexthop       flags       vnet bd vlan intf-ip intf-id
---- --------  ------  ------------------ ------------ ----------  ---- -- ---- -------  -------
1    10.2.4.0 24     1                  172.16.134.3 in-hw,evpn         3501 ::      10
```

On non-border gateway nodes (such as Leaf 1), you can verify the presence of an automatically created default route pointing to the border gateway(s) with either of the following commands:

```
CLI (network-admin@leaf1*) > vrouter-rib-routes-show vrid 1
```

```
vrid ip        prelen number-of-nexthops nexthop       flags                vnet bd vlan intf-ip  intf-id
---- --------  ------  ------------------ ------------ ------------------ ---- -- ---- --------  -------
1    0.0.0.0  0      1                  172.16.133.3 in-hw,evpn                 3501 ::       8
1    10.2.3.0 24     1                  10.2.3.1     in-hw,local-subnet         403  10.2.3.1 9
```

```
CLI (network-admin@leaf1*) > vrouter-rib-routes-show vrid 1 flags evpn
```

```
vrid ip       prelen number-of-nexthops nexthop       flags       vnet bd vlan intf-ip intf-id
---- -------  ------  ------------------ ------------ ----------  ---- -- ---- -------  -------
1    0.0.0.0 0      1                  172.16.133.3 in-hw,evpn         3501 ::      8
```

In this case the default gateway points to a VIP address of a redundant border gateway pair, which can be displayed like so:

```
CLI (network-admin@bgw11*) > vrouter-interface-show is-vip true

vrouter-name   nic       ip             mac               vlan vlan-type nic-state is-vip vrrp-id vrrp-primary vrrp-state mtu  priority-tag
-------------- --------- -------------- ----------------- ---- --------- --------- ------ ------- ------------ ---------- ---- ------------
vrouter-bgw-11 eth21.3808 172.16.133.3/29 00:00:5e:00:01:06 3808 public    up        true   6       eth20.3808   master     9216 off
```

Displaying BGP Neighbor Information

To display BGP neighbors associated with EVPN, you can use the following command:

```
CLI (network-admin@bgw11*) > vrouter-evpn-bgp-neighbor-summary-show format
all

vrouter-name    neighbor      ver as     msg-rcvd msg-sent Tb/Ver InQ OutQ up/down   state         pfxrcd
--------------  ------------  --- -----  -------- -------- ------ --- ---- --------  -----------  ------
vrouter-bgw-11 172.16.134.1 4    65011 122       226       0       0    0     00:00:39 Connect      0
vrouter-bgw-11 172.16.134.2 4    65011 83        190       0       0    0     00:07:36 Established 4
```

Displaying vPort Information

On a non-border gateway node such as Leaf 1, you can verify the vPort database information associated with the EVPN control plane by using the following command:

```
CLI (network-admin@leaf1*) > vport-show state evpn

owner mac               vxlan  state vtep-ip
----- ----------------- ------ ----- ------------
leaf1 64:0e:94:40:00:02 10201  evpn  172.16.133.3
leaf1 64:0e:94:40:00:02 103501 evpn  172.16.133.3
```

Displaying Additional BGP Control Plane Information

EVPN uses BGP as control plane to exchange various kinds of information that are necessary for proper network functioning, such as VNI values, MAC/IP address pair entries (as found in ARP caches), and Layer 2 address information.

Note that the EVPN control plane in NetVisor OS is divided into two layers. The higher layer comprises the familiar NetVisor OS commands, which provide a holistic view on the network entities as described in the above sections. Instead, the lower layer (also known as FRR) provides a low-level 'raw' view on certain control plane functions.

For troubleshooting purposes, it can be useful to check that the upper layer and the lower layer information match.

For instance, it is possible to verify the list of active VNIs on a Border Gateway with the familiar 'high-level' `vtep-vxlan-show` command. For example, on `bgw21` (limited to `vtep-leaf2` for brevity's sake) the command output shows:

```
CLI (network-admin@bgw21*) > vtep-vxlan-show name vtep-leaf2 sort-asc
vxlan,
```

```
name       vxlan  isolated
---------- ------ --------
vtep-leaf2 10110  no
vtep-leaf2 10112  no
vtep-leaf2 10120  no
vtep-leaf2 10121  no
vtep-leaf2 10122  no
vtep-leaf2 10130  no
vtep-leaf2 10131  no
vtep-leaf2 10140  no
vtep-leaf2 10145  no
vtep-leaf2 10146  no
vtep-leaf2 10147  no
vtep-leaf2 10148  no
vtep-leaf2 10149  no
vtep-leaf2 10150  no
```

The same VNI information can be checked also in the low-level 'raw' control plane view with the `vrouter-evpn-bgp-vni-show` command, which additionally shows that the VNIs are transported in the RT values. The command also shows that NetVisor OS imports and exports all RTs from/to neighbor switches:

CLI (network-admin@bgw21*) > vrouter-evpn-bgp-vni-show sort-asc vni,

```
switch vrouter-name vni    type rd          import-rt    export-rt
------ ------------ ------ ---- ---------- ------------ ------------
bgw21  bgw21-vr     10110  2    2.4.0.1:15 65012:10110  65012:10110
bgw21  bgw21-vr     10112  2    2.4.0.1:27 65012:10112  65012:10112
bgw21  bgw21-vr     10120  2    2.4.0.1:30 65012:10120  65012:10120
bgw21  bgw21-vr     10121  2    2.4.0.1:35 65012:10121  65012:10121
bgw21  bgw21-vr     10122  2    2.4.0.1:39 65012:10122  65012:10122
bgw21  bgw21-vr     10130  2    2.4.0.1:16 65012:10130  65012:10130
bgw21  bgw21-vr     10131  2    2.4.0.1:17 65012:10131  65012:10131
bgw21  bgw21-vr     10140  2    2.4.0.1:20 65012:10140  65012:10140
bgw21  bgw21-vr     10145  2    2.4.0.1:33 65012:10145  65012:10145
bgw21  bgw21-vr     10146  2    2.4.0.1:13 65012:10146  65012:10146
bgw21  bgw21-vr     10147  2    2.4.0.1:19 65012:10147  65012:10147
bgw21  bgw21-vr     10148  2    2.4.0.1:32 65012:10148  65012:10148
bgw21  bgw21-vr     10149  2    2.4.0.1:12 65012:10149  65012:10149
bgw21  bgw21-vr     10150  2    2.4.0.1:28 65012:10150  65012:10150
```

Similarly, it is possible to check the Layer 3 table's contents (MAC/IP address pairs) populated by EVPN and then also verify that the low-level entries match. For that you can use the following two commands (in this example, for the sake of brevity, only the entries associated with VLAN 211 and VNI 10211 are shown):

```
CLI (network-admin@bgw21*) > l3-table-show vlan 211

mac               ip                vlan vxlan state
----------------- ----------------- ---- ----- -----------
00:11:01:00:00:06 100.0.211.6        211  10211 active,evpn
00:11:01:00:00:02 2000::100:0:211:2 211   10211 active,evpn
00:11:01:00:00:0a 2000::100:0:211:a 211   10211 active,evpn
00:11:01:00:00:04 100.0.211.4        211  10211 active,evpn
00:11:01:00:00:03 2000::100:0:211:3 211   10211 active,evpn
00:11:01:00:00:07 100.0.211.7        211  10211 active,evpn
00:11:01:00:00:05 2000::100:0:211:5 211   10211 active,evpn
00:11:01:00:00:08 2000::100:0:211:8 211   10211 active,evpn
00:11:01:00:00:07 2000::100:0:211:7 211   10211 active,evpn
```

```
00:11:01:00:00:09 100.0.211.9        211   10211 active,evpn
00:11:01:00:00:05 100.0.211.5        211   10211 active,evpn
00:11:01:00:00:04 2000::100:0:211:4  211   10211 active,evpn
00:11:01:00:00:08 100.0.211.8        211   10211 active,evpn
00:11:01:00:00:02 100.0.211.2        211   10211 active,evpn
00:11:01:00:00:03 100.0.211.3        211   10211 active,evpn
00:11:01:00:00:0a 100.0.211.10       211   10211 active,evpn
00:11:01:00:00:06 2000::100:0:211:6  211   10211 active,evpn
00:11:01:00:00:09 2000::100:0:211:9  211   10211 active,evpn
```

The corresponding low-level command is:

```
CLI (network-admin@bgw21*) > vrouter-evpn-arp-cache-show vni 10211

name     vni   ip                mac               remote-vtep type   state  remote-seq local-
seq
-------- ----- ----------------- ----------------- ----------- ------ ------ ----------
---------
bgw21-vr 10211 2000::100:0:211:8 00:11:01:00:00:08 7.1.1.1     remote active 0          0
bgw21-vr 10211 100.0.211.7       00:11:01:00:00:07 7.1.1.1     remote active 0          0
bgw21-vr 10211 100.0.211.6       00:11:01:00:00:06 7.1.1.1     remote active 0          0
bgw21-vr 10211 2000::100:0:211:9 00:11:01:00:00:09 7.1.1.1     remote active 0          0
bgw21-vr 10211 2000::100:0:211:a 00:11:01:00:00:0a 7.1.1.1     remote active 0          0
bgw21-vr 10211 100.0.211.3       00:11:01:00:00:03 7.1.1.1     remote active 0          0
bgw21-vr 10211 2000::100:0:211:6 00:11:01:00:00:06 7.1.1.1     remote active 0          0
bgw21-vr 10211 100.0.211.2       00:11:01:00:00:02 7.1.1.1     remote active 0          0
bgw21-vr 10211 2000::100:0:211:7 00:11:01:00:00:07 7.1.1.1     remote active 0          0
bgw21-vr 10211 2000::100:0:211:4 00:11:01:00:00:04 7.1.1.1     remote active 0          0
bgw21-vr 10211 100.0.211.10      00:11:01:00:00:0a 7.1.1.1     remote active 0          0
bgw21-vr 10211 100.0.211.8       00:11:01:00:00:08 7.1.1.1     remote active 0          0
bgw21-vr 10211 2000::100:0:211:2 00:11:01:00:00:02 7.1.1.1     remote active 0          0
bgw21-vr 10211 100.0.211.4       00:11:01:00:00:04 7.1.1.1     remote active 0          0
bgw21-vr 10211 100.0.211.5       00:11:01:00:00:05 7.1.1.1     remote active 0          0
bgw21-vr 10211 100.0.211.1       00:11:01:00:00:01 7.1.1.1     remote active 0          0
bgw21-vr 10211 100.0.211.9       00:11:01:00:00:09 7.1.1.1     remote active 0          0
bgw21-vr 10211 2000::100:0:211:3 00:11:01:00:00:03 7.1.1.1     remote active 0          0
bgw21-vr 10211 2000::100:0:211:1 00:11:01:00:00:01 7.1.1.1     remote active 0          0
bgw21-vr 10211 2000::100:0:211:5 00:11:01:00:00:05 7.1.1.1     remote active 0          0
```

Furthermore, to check the Layer 2 table's contents (e.g., the MAC addresses learned locally and remotely via EVPN) and then also to verify that the low-level entries match, you can use the following two commands (in this example, for the sake of brevity, only the entries associated with VNI 10211 are displayed):

```
CLI (network-admin@bgw21*) > l2-table-show vlan 211 format
mac,vlan,vxlan,ip,state,peer-state,peer-owner-state,status

mac               vlan vxlan ip                  state       peer-state  peer-owner-state status
----------------- ---- ----- ------------------- ----------- ----------- ---------------- ------
00:11:01:00:00:01 211  10211 2000::100:0:211:1   tunnel,evpn evpn-wo-bgp
00:11:01:00:00:04 211  10211 2000::100:0:211:4   tunnel,evpn
00:12:01:00:00:08 211  10211 2000::100:0:211:107 tunnel      tunnel      active            host
00:11:01:00:00:08 211  10211 2000::100:0:211:8   tunnel,evpn
00:11:01:00:00:07 211  10211 2000::100:0:211:7   tunnel,evpn
00:12:01:00:00:01 211  10211 2000::100:0:211:100 tunnel      tunnel      active            host
00:11:01:00:00:05 211  10211 2000::100:0:211:5   tunnel,evpn
00:12:01:00:00:06 211  10211 2000::100:0:211:105 tunnel      tunnel      active            host
00:11:01:00:00:03 211  10211 2000::100:0:211:3   tunnel,evpn
00:11:01:00:00:02 211  10211 2000::100:0:211:2   tunnel,evpn
00:12:01:00:00:02 211  10211 2000::100:0:211:101 tunnel      tunnel      active            host
00:11:01:00:00:06 211  10211 2000::100:0:211:6   tunnel,evpn
```

```
00:12:01:00:00:07 211  10211 2000::100:0:211:106 tunnel       tunnel       active            host
00:12:01:00:00:05 211  10211 2000::100:0:211:104 tunnel       tunnel       active            host
00:12:01:00:00:0a 211  10211 2000::100:0:211:109 tunnel       tunnel       active            host
00:11:01:00:00:0a 211  10211 100.0.211.10        tunnel,evpn
00:12:01:00:00:03 211  10211 2000::100:0:211:102 tunnel       tunnel       active            host
00:12:01:00:00:04 211  10211 2000::100:0:211:103 tunnel       tunnel       active            host
00:11:01:00:00:09 211  10211 2000::100:0:211:9   tunnel,evpn
00:12:01:00:00:09 211  10211 2000::100:0:211:108 tunnel       tunnel       active            host
```

In the above command output it's possible to look at the state of a MAC address entry to check if it's learned from a local `tunnel` and/or from the `evpn` control plane. When the `peer-state is evpn-wo-bgp`, it means that the entry was learned (initially) directly from the data plane.

The corresponding low-level command to check the Layer 2 entries is:

```
CLI (network-admin@bgw21*) > vrouter-evpn-vni-mac-show vni 10211

vrouter-name vni    mac                type    intf/remote-vtep vlan local-seq remote-seq
------------ -----  -----------------  ------  ---------------  ---- ---------  ----------
bgw21-vr     10211 00:12:01:00:00:02 local  9000becvni10211  0    0          0
bgw21-vr     10211 00:12:01:00:00:0a local  9000becvni10211  0    0          0
bgw21-vr     10211 00:11:01:00:00:05 remote 7.1.1.1           0    0          0
bgw21-vr     10211 00:11:01:00:00:0a remote 7.1.1.1           0    0          0
bgw21-vr     10211 00:12:01:00:00:07 local  9000becvni10211  0    0          0
bgw21-vr     10211 00:11:01:00:00:02 remote 7.1.1.1           0    0          0
bgw21-vr     10211 00:12:01:00:00:09 local  9000becvni10211  0    0          0
bgw21-vr     10211 00:11:01:00:00:07 remote 7.1.1.1           0    0          0
bgw21-vr     10211 00:11:01:00:00:04 remote 7.1.1.1           0    0          0
bgw21-vr     10211 00:12:01:00:00:03 local  9000becvni10211  0    0          0
bgw21-vr     10211 00:11:01:00:00:08 remote 7.1.1.1           0    0          0
bgw21-vr     10211 00:12:01:00:00:08 local  9000becvni10211  0    0          0
bgw21-vr     10211 00:11:01:00:00:01 remote 7.1.1.1           0    0          0
bgw21-vr     10211 00:11:01:00:00:03 remote 7.1.1.1           0    0          0
bgw21-vr     10211 00:12:01:00:00:06 local  9000becvni10211  0    0          0
bgw21-vr     10211 00:12:01:00:00:01 local  9000becvni10211  0    0          0
bgw21-vr     10211 00:11:01:00:00:09 remote 7.1.1.1           0    0          0
bgw21-vr     10211 00:12:01:00:00:05 local  9000becvni10211  0    0          0
bgw21-vr     10211 00:12:01:00:00:04 local  9000becvni10211  0    0          0
bgw21-vr     10211 00:11:01:00:00:06 remote 7.1.1.1           0    0          0
```

Displaying EVPN Route History Information

Displaying EVPN route history information is useful to verify the inter-pod information exchanged over type 2, 3 and 5 messages and for forensic tracing.

To display the type 2 route history, you can use the `vport-history-show` command. For example, to display the injected (i.e., created/added) vPorts, you can use:

```
CLI (network-admin@bgw21) > vport-history-show caller inject, within-last 2d

time           log-type  reason owner mac               vxlan state vtep-ip
-------------- --------- ------ ----- ----------------- ----- ----- ---------
01-18,06:33:40 l2-modify create bgw21 66:0e:94:3f:64:4a 40400 evpn  20.0.12.1
```

Or you can use this command to show all the latest actions, for example limited to a certain VNI:

```
CLI (network-admin@bgw21*) > vport-history-show within-last 2d vxlan 100200
```

state **evpn-wo-bgp**,

```
time           log-type  reason owner mac               vlan vxlan  state              hostname status
-------------- --------- ------ ----- ----------------- ---- ------ ------------------ -------- -----------
01-18,06:23:37 l2-modify modify bgw21 20:04:0f:12:95:62 200   100200 tunnel,evpn-wo-bgp bgw21     PN-internal
01-18,06:23:52 l2-modify modify bgw21 50:9a:4c:d3:fd:f0 200   100200 tunnel,evpn-wo-bgp bgw21     PN-internal
01-18,06:29:34 l2-modify flush  bgw21 20:04:0f:12:95:62 200   100200 evpn-wo-bgp        bgw21    PN-internal
01-18,06:29:34 l2-modify flush  bgw21 50:9a:4c:d3:fd:f0 200   100200 evpn-wo-bgp        bgw21    PN-internal
```

You can display the type 5 route history with the following command, for example filtered for a specific address:

```
CLI (network-admin@bgw21) > vrouter-rib-history-show within-last 2d flags evpn, ip 200.1.1.0

time           caller reason vrid ip        prelen nexthop    flags      vlan intf-ip
-------------- ------ ------ ---- --------- ------ ---------- ---------- ---- --------
01-17,17:12:12 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,20:17:32 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,20:26:39 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,20:56:09 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,21:09:06 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,21:23:45 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,21:32:43 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,21:36:37 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,21:42:27 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,22:50:22 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,23:32:06 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,23:32:07 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
01-17,23:35:48 vrf    add    1    200.1.1.0 24     190.12.1.1 in-hw,evpn 3434 33.3.3.3
```

Furthermore, to see the type 3 message-triggered VTEP changes you can filter the audit log with grep, for example like so:

```
CLI (network-admin@bgw21*) > log-audit-show name user_command format message, | grep evpn

message

------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------
---------------------
------------------------------------------------------------------------
Command "vtep-create scope local vtep-source evpn name __evpn__190.12.1.1 location 16777214 ip 190.12.1.1"
result success

Command "vtep-vxlan-add name __evpn__190.12.1.1 vxlan 10400" result success

<snip>
```

Supported Releases

Command/Parameter	NetVisor OS Version
`evpn-border/no-evpn-border, l2vpn-evpn, l3-vni, state evpn`	Parameters added in version 6.1.0
`vrouter-evpn-bgp-routes-show, vrouter-evpn-bgp-neighbor-summary-show, vrouter-evpn-bgp-vni-show, vrouter-evpn-arp-cache-show, vrouter-evpn-vni-mac-show`	Commands added in version 6.1.0
`evpn-dup-addr-max-moves, evpn-dup-addr-moves-duration, evpn-dup-addr-freeze`	Parameters added in version 6.1.1
`vrouter-evpn-duplicate-mac-show`	Command added in version 6.1.1
`system-settings-modify no-l3-use-host-table | l3-use-host-table system-settings-show format l3-use-host-table`	Command and parameter added in version 7.0.0
`export-leaked-networks evpn, evpn-export-leaked-networks deny`	Parameters added in version 7.0.0
`match-evpn-default-route, matchevpn-route-type, match-evpn-vni,match-evpn-rd`	Parameters added in version 7.0.0

Please also refer to the *Arista NetVisor OS Command Reference* document.

Related Documentation

For further information on concepts mentioned in this section, refer to these sections of the Configuration Guide:

- *Configuring Layer 3 Features*
- *Configuring and Administering the Unified Cloud Fabric*
- *Configuring High Availability*
- *Configuring VXLAN*
- *Configuring Advanced Layer 2 Transport Services*

for further information on concepts mentioned in this section (such as Layer 2, Layer 3, etc.)

Configuring and Using Network Management and Monitoring

This chapter provides required information about the Network Management and Monitoring features on a NetVisor OS switch using the NetVisor OS Command Line Interface.

Overview

Network Management and Monitoring involves an array of methods and tools that can help to operate and maintain an error-free network. Network management tools primarily cater to the collection and organization of information regarding a network, enabling you to make informed decisions while configuring, troubleshooting, or overseeing the operation of the network. Network management is essential to ensure optimal performance and fault-free operation of network elements. NetVisor OS provides a handful of methods to manage and monitor a network including different logging schemes, Simple Network Management Protocol (SNMP), Network Packet Broker (NPB), mirroring of network traffic, and system statistics.

Supported Network Management Components

NetVisor OS supports systems statistics, logging, Simple Network Management Protocol (SNMP), and traffic mirroring, as the primary methods of network management and monitoring.

System Statistics: This network monitoring method provides information about the CPU and memory usage of a switch. This gives you an insight into the performance and efficiency of the network device.

Logging: Logs serve as a record of all important events that occur in a network. NetVisor OS broadly has four types of logging: event logging, audit logging, system logging, and perror logging. Logging tracks activities such as connection state changes, administrative changes, user login or logout, protocol events, and so on. Logging information can be viewed on the switch and can also be sent to dedicated servers from where it can be accessed later through the CLI.

SNMP: NetVisor OS allows monitoring of network nodes through this protocol. The SNMP manager installed on a server called Network Management Station (NMS) polls the SNMP agents installed on the network devices periodically for information regarding network events. This collected information can be accessed through the CLI. In addition, SNMP traps can be configured to receive instantaneous alerts for desired events, instead of relying solely on periodic SNMP polls.

Traffic Mirroring for Ports and vFlows: This functionality is used to copy packets from one port to another for increased visibility into the network traffic. This feature supports a basic configuration analogous to local Switched Port Analyzer (SPAN) as well as advanced configurations similar to Remote Switched Port Analyzer (RSPAN) and Encapsulated Remote Switched Port Analyzer (ERSPAN).

Understanding System Statistics

Measuring the hardware utilization of each device is the key to evaluate the efficiency of networking infrastructure. The performance of a device is determined primarily by two hardware resources: CPU and main memory. System statistics in NetVisor OS comprise the CPU and memory usage of the switch.

System statistics on a switch also include memory swapping and paging parameters. Swapping is when an entire process' address space is mapped from the main memory (RAM) to a secondary memory (disk memory) that is used to "swap" pages to or from in order to free up space for the relevant processes in main memory. Paging allows a granular method of copying where, instead of an entire process, frames or pages of a process are copied to and from the main memory. The paging and memory swap parameters are important indicators of performance as consistently high values in these fields are predictors of slowness of the switch.

Configuring and Displaying System Statistics on a Switch

You can display system statistics on a switch using the `system-stats-show` command. This command displays the memory and CPU usage statistics. The parameters under `system-stats-show` command are:

`switch` - The current switch of which the statistics are being displayed.
`uptime` - The amount of time for which the switch has been online.
`used-mem` - Percentage of memory used by the switch.
`used-mem-val` - The amount of memory used by the switch.
`used-swap` - The percentage of swap memory used by the switch.
`used-swap-val` - The amount of swap memory used by the switch.
`paging` - The swap scan rate.
`cpu-user` - The percentage of CPU usage associated with user processes.
`cpu-sys` - The percentage of CPU usage associated with system processes.
`cpu-total` - The total usage of the CPU in percentage.
`cpu-idle` - The percentage of CPU idle time.
`cpu-avg` - The average CPU usage (`cpu-sys` + `cpu-user`) over last 30 seconds.

For example:

```
CLI (network-admin@switch1) > system-stats-show layout vertical

switch:       switch1
uptime:       7h7m10s
used-mem:     54%
used-swap:    0%
paging:       0
cpu-user:     2%
cpu-sys:      5%
cpu-idle:     92%
cpu-avg:      3%
```

To view all the details, add the parameter `format all`:

```
CLI (network-admin@switch1) > system-stats-show layout vertical format all

switch:         switch1
uptime:         7h5m41s
used-mem:       54%
used-mem-val:   3.84G
used-swap:      0%
used-swap-val:  0
paging:         0
cpu-user:       4%
cpu-sys:        4%
cpu-total:      9%
cpu-idle:       90%
cpu-avg:        3%
```

To view the system statistics within last 5 minutes, use the command:

```
CLI (network-admin@switch1) > system-stats-show within-last 5m layout
vertical
switch:       switch1
time:         08:06:26
uptime:        3426684193d8h52m32s
used-mem:     63%
used-swap:    0%
paging:       0
cpu-user:     0%
cpu-sys:      99%
cpu-idle:     0%
switch:       switch1
time:         08:07:26
uptime:        3427378633d4h10m24s
used-mem:     63%
used-swap:    0%
paging:       0
cpu-user:     2%
cpu-sys:      2%
cpu-idle:     95%
switch:       switch1
time:         08:08:26
uptime:        3428073072d10h3m41s
used-mem:     63%
used-swap:    0%
paging:       0
cpu-user:     2%
cpu-sys:      1%
cpu-idle:     95%
switch:       switch1
time:         08:09:26
uptime:        3428767511d20h1m42s
used-mem:     63%
used-swap:    0%
paging:       0
cpu-user:     2%
cpu-sys:      2%
cpu-idle:     95%
```

The paging field of the `system-stats-show` command displays the swap scan rate. A non-zero value in this field shows that memory is being paged from the physical memory (RAM) to virtual memory (disk or swap). A consistently high value in this field indicates that all memory, both physical and virtual, is exhausted and the system may stop responding.

Configuring and Displaying System Statistics History

In releases prior to NetVisor OS version 6.10, the `system-stats-show` command provided instantaneous values of CPU usage, which was not representative of the average CPU usage over a duration. NetVisor OS release 6.1.0 improves the infrastructure for collecting and displaying system statistics to provide average usage information. NetVisor OS currently supports viewing of historical CPU and memory usage over the last 24 hours with a granularity of up to 10s.

You can enable system statistics history by using the command:

```
CLI (network-admin@switch1) > system-stats-history-settings-modify enable
```

`system-stats-history-settings-modify`	Modify system statistics history settings.
`enable|disable`	Enable or disable system statistics history. The default status is `enable`.

When you enable the collection of historical system usage data, a new thread is created in nvOSd to collect statistics periodically. You can turn off this thread by using the `disable` option the CLI. If you disable statistics history collection and re-enable the feature, the statistics collection starts from scratch. Also, if you restart nvOSd or reboot the switch while system statistics history is enabled, statistics history is cleared and statistics collection starts afresh.

To view the settings for collection of system statistics history, use the command:

```
CLI (network-admin@switch1) > system-stats-history-settings-show
switch:         switch1
enable:         yes
read-interval:  10s
```

Note that the read interval is set to 10s in the software and is not configurable.

To display system statistics history over the last one minute (with a granularity of 10s), last 1 hour (with a granularity of 1 minute), and last 24 hours (with a granularity of 1 hour), use the command:

```
CLI (network-admin@switch1*) > system-stats-history-show
### Usage over last 1 minute, 60 minutes and last 24 hours ###

time             mem-used(%) cpu-used(%) cpu-peak(%) over-last
-------------- ----------- ----------- ----------- ---------
01:48:24       54          4                       10s
01:48:14       54          3                       10s
01:48:04       54          3                       10s
01:47:54       54          3                       10s
01:47:44       54          3                       10s
01:47:34       54          3                       10s
01:47:34       54          3           3           1m
01:46:34       53          3           5           1m
01:45:34       53          3           4           1m
01:44:34       53          2           3           1m
```

```
01:43:34        53          3           4           1m
01:42:34        53          3           3           1m
01:41:34        53          3           4           1m
01:40:34        53          3           4           1m
01:39:34        53          2           3           1m
.
.
.
00:56:34        53          2           3           1m
00:55:34        53          3           4           1m
00:54:34        53          3           4           1m
00:53:34        53          3           4           1m
00:52:34        53          3           3           1m
00:51:34        53          3           4           1m
00:50:34        53          3           4           1m
00:49:34        53          3           4           1m
00:48:34        53          3           4           1m
01:33:34        53          2           3           1h
00:33:34        53          2           3           1h
02-24,23:33:34 53           2           3           1h
02-24,22:33:34 53           2           3           1h
02-24,21:33:34 53           2           3           1h
```

Note: The system statistics history values are populated with time. For example, if you want to view the statistics history for the last 24 hours, nvOSd must have run for the last 24 hours with system statistics history enabled.

If you disable system statistics history, running the commands `system-stats-history-show` and `sytem-stats-history-average-show` returns an error. For example:

```
CLI (network-admin@switch1) > system-stats-history-settings-modify disable

CLI (network-admin@switch1) > system-stats-history-show
system-stats-history-show: system stats history collection is disabled

CLI (network-admin@switch1) > system-stats-history-average-show over-last
10s
system-stats-history-average-show: system stats history collection is
disabled
```

To view the average system usage statistics over the last 10 to 60 seconds, use the command:

```
CLI (network-admin@switch) > system-stats-history-average-show
```

`system-stats-history-average-show`	Display average system statistics over a given duration.
`over-last` *`duration:#s`*	Specify a duration for average usage statistics (in multiples of 10s from 10s up to 60s). **Note:** You cannot specify the duration in days, hours, or minutes.

For example, view the average memory and CPU usage for the last 40s by using the command:

```
CLI (network-admin@switch1*) > system-stats-average-show over-last 40s
time      mem-used(%) cpu-used(%) over-last
--------  ----------- ----------- ---------
02:00:34 54          3           40s
```

For system statistics older than one day, use the `system-stats-show within-last` <#d#h#m#s> command.

Understanding Logging

NetVisor OS logs all important activities that occur on the switches and fabrics created on them. Logging is enabled by default and can be viewed using the CLI. You can also configure system logging to send syslog-formatted messages to servers configured to receive them, as part of centralized logging and monitoring.

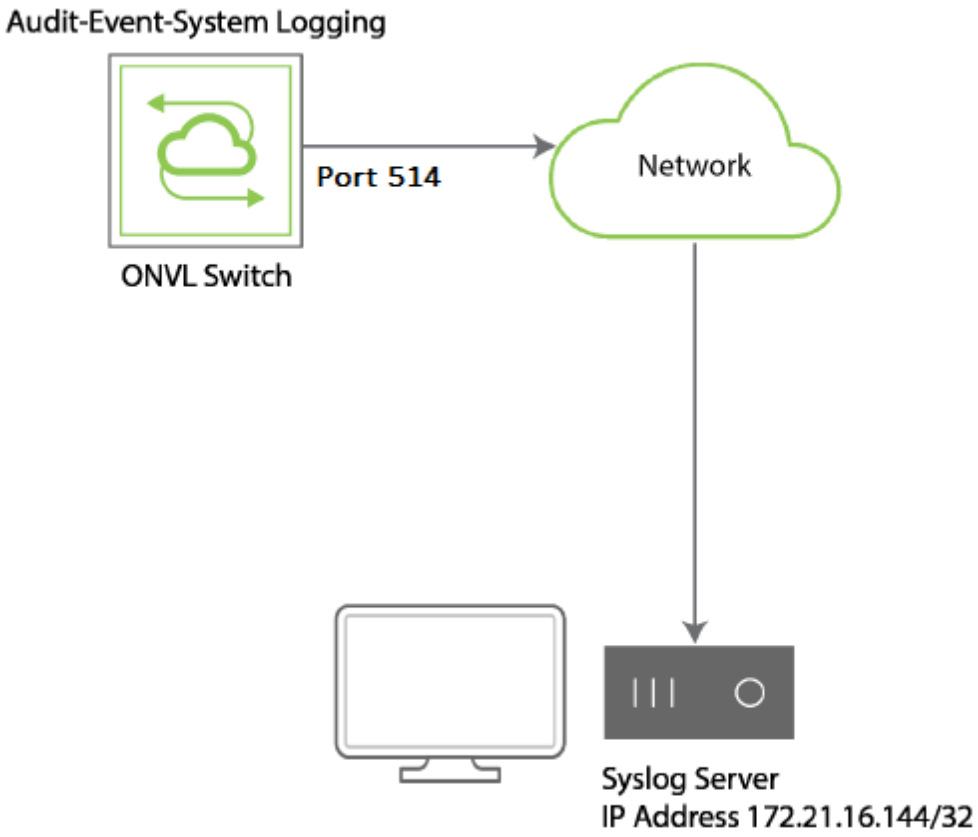


Figure 11-1 - NetVisor OS Switch with Syslog Server

The following types of activities are logged:

Log Type	Description
Event	Records action observed or performed by switches. Each Event type can be enabled or disabled. Events are collected on a best effort basis. If events occur too rapidly to be recorded, the event log is annotated with the number of events lost. The following are examples of event types: • Port state changes • TCP connections • STP port changes
Audit	When an administrative change to the configuration is made, an audit log is

	recorded. An audit log consists of the command and parameters along with the success or failure indication. When a command fails, an error message is also recorded.
System	The system log records error conditions and conditions of interest.
Perror	The Perror log records messages on standard error output, describing the last error encountered.

Each log message includes the following information:

- Category - event, audit, or system
- Timestamp within a microsecond
- Process name and process ID of the process producing the message
- Unique message name
- Unique five digit numerical message code
- Message: additional message-specific parameters and explanation

A log message consists of common parameters separated by spaces and a colon (:), optional parameters, another colon, and then the log-specific message. The optional parameters, which may include the associated VLAN, VXLAN, or switch ports, appear as key/value pairs. An audit log message includes additional information:

- User
- Process ID
- Client IP of the remote computer issuing the command

An event log also includes the event type.

For information about specific log events and their meaning, see the *NetVisor OS Log Messages Guide*.

The maximum number of repeated messages detected by NetVisor OS is ten (10). After five seconds, if NetVisor OS detects repeated messages, then the log prints "Last X messages(s) repeated Y time(s)". If the log message detects "X" and "Y" as both 1, then NetVisor OS prints the message rather than "Last 1 message(s) repeated 1 time(s)". Log events are printed after a five (5) second delay.

Currently, accessing system log information may require assistance from TAC to retrieve the `logs` from NetVisor OS. To enable log auditing in NetVisor OS, use the following command:

```
CLI (network-admin@Leaf1) > log-admin-audit-modify enable|disable
```

Log auditing is disabled by default. To display auditing status, use the following command:

```
CLI (network-admin@Leaf1) > log-admin-audit-show
switch: Leaf1
enable: yes
```

Using Facility Codes with Log Messages

NetVisor OS labels log messages with a facility code indicating the area of the software that generated the log message.

The following are the default facility codes:

- `Log_Daemon` for events and system messages
- `Log_AUDIT` for audit messages

The following severity levels are used by default:

- `Log_INFO` = informational
- `Log_Critical` = critical
- `Log_ERROR` = error
- `Log_WARNING` = warn
- `Log_NOTICE` = note

Configuring Event Logging

Event log messages in NetVisor OS covers events related to the system, protocols (STP, TCP, LLDP, LACP, IGMP), and TACACS+, among others. By default, system, port, LLDP, and TACACS+ events are logged. You can add or remove log events by using the `log-event-settings-modify` command.

`log-event-settings-modify`	Modify log event settings.
Specify one or more of the following options:	
`system|no-system`	Enable or disable system event logging. Enabled by default
`port|no-port`	Enable or disable port event logging. Enabled by default.
`tcp|no-tcp`	Enable or disable TCP event logging.
`stp|no-stp`	Enable or disable STP event logging.
`igmp|no-igmp`	Enable or disable IGMP event logging.
`lldp|no-lldp`	Enable or disable LLDP event logging.
`lacp|no-lacp`	Enable or disable LACP event logging.
`tacacs|no-tacacs`	Enable or disable TACACS+ event logging.Enabled by default.
`mld|no-mld`	Enable or disable MLD event logging.
`mroute|no-mroute`	Enable or disable mroute event logging.
`vport|no-vport`	Enable or disable vPort event logging.
`lacp-port-event|no-lacp-port-event`	Enable or disable LACP port event logging.

For instance, to enable logging of STP events, use the following command:

```
CLI (network-admin@Leaf1) > log-event-settings-modify stp
```

To display log event settings, use the `log-event-settings-show` command:

```
CLI (network-admin@Leaf1) > log-event-settings-show
switch:          Leaf1
system:          on
port:            on
tcp:             off
stp:             on
igmp:            off
lldp:            on
lacp:            off
tacacs:          on
```

```
mld:              off
mroute:           off
vport:            off
lacp-port-event:  off
```

Displaying Event Log Information

To view event log information, run the command `log-event-show.`

`log-event-show`	Display event log information.
Specify between zero to two of the following options:	
`start-time date/time:` *`yyyy-mm-ddThh:mm:ss`*	Specify the start time for the log file.
`end-time date/time:` *`yyyy-mm-ddThh:mm:ss`*	Specify the end time for the log file.
`duration duration: #d#h#m#s`	Specify the duration of the log file.
Specify any of the following parameters to view the information related to those parameters:	
`program` *`program-string`*	Specify the program that generates log messages.
`pid` *`pid-number`*	Specify the product ID generating the log messages.
`name` *`name-string`*	Specify the message name.
`code` *`code-number`*	Specify the message code.
`level critical|error|warn|note|info`	Specify the severity level of event log messages.
`event-type system|port|tcp|stp|igmp|lldp|lacp|vdp|ecp|evb|ptp|storage|tacacs|mld|mroute|vport|lacp-port-event`	Specify one among the options as the log event type.
`vnet` *`vnet-name`*	Specify the associated VNET.
`remote_switch node` *`name`*	Specify the name of the remote switch.
`client-pid` *`client-pid-number`*	Specify the client product ID.
`client-addr` *`ip-address`*	Specify the client IP address.
`port` *`port-number`*	Specify the port number.
`vlan` *`vlan-id`*	Specify the VLAN ID as a value between 2 and 4092.
`bd` *`bridge-domain name`*	Specify the bridge domain.

`vxlan` *`vxlan-id`*	Specify the VXLAN ID.
`count` *`number 1..50000`*	Specify the number of events to be displayed in a range from 1 to 50000.
`starting-point` *`starting-point-number`*	Specify the starting point of the log audit.
`length` *`length-number`*	Specify the length of the log audit.
`reverse|no-reverse`	Use this option to enable or disable displaying the messages in reverse order.

For example:

```
CLI (network-admin@Leaf1) >log-event-show count 3
category time                        name    code level event-type port message
-------- --------------------------- ------- ---- ----- ---------- ---- -------
event    2013-06-04,13:12:18.304740 port_up 62   info  port       62   up
event    2013-06-04,13:12:18.304740 port_up 62   info  port       50   up
event    2013-06-04,13:12:18.304740 port_up 62   info  port       10   up
```

Configuring Audit Logging

Audit logging includes messages for user login or logout, authorization and denial of sessions or commands by TACACS+ server, audit drops, and commands run internally or by the user, among others. To view audit log Information, enter the following command:

```
CLI (network-admin@Leaf1) > log-audit-show
```

`log-audit-show`	Display audit log information.
Specify up to two options from the following:	
`start-time date/time: yyyy-mm-ddThh:mm:ss`	Specify the start time for the log file.
`end-time date/time: yyyy-mm-ddThh:mm:ss`	Specify the end time for the log file.
`duration duration: #d#h#m#s`	Specify the duration of the log file.
Specify any of the following parameters to view the information related to those parameters:	
`program program-string`	Specify the program that generates log messages.
`pid pid-number`	Specify the product ID generating the log messages.
`name name-string`	Specify the message name.
`code code-number`	Specify the message code.
`level critical|error|warn|note|info`	Specify the severity level of audit log messages.
`vnet vnet-name`	Specify the associated VNET.
`remote_switch node name`	Specify the name of the remote switch.
`client-pid client-pid-number`	Specify the client product ID.
`client-addr ip-address`	Specify the client IP address.
`port port-number`	Specify the port number.
`vlan vlan-id`	Specify the VLAN ID as a value between 2 and 4092.
`bd bridge-domain name`	Specify the bridge domain.
`vxlan vxlan-id`	Specify the VXLAN ID.
`count number 1..50000`	Specify the number of events to be displayed in a range from 1 to 50000.
`starting-point starting-point-number`	Specify the starting point of the log audit.
`length length-number`	Specify the length of the log audit.

`reverse|no-reverse`	Use this option to enable or disable displaying the messages in reverse order.

For example:

```
CLI (network-admin@leaf1) > log-audit-show count 2 layout vertical
category:          audit
time:               2020-07-29,07:21:09.297988-07:00
name:              login
code:              11099
level:             info
user:              network-admin
client-addr:       10.140.0.158
message:           login
category:          audit
time:               2020-07-29,07:21:25.593283-07:00
name:              user_command
code:              11001
level:             info
user:              network-admin
client-addr:       10.140.0.158
message:           Command "vflow-delete name span1" result success
```

Exceptions for Audit Logging

When NetVisor OS supports a command for auditing, the command is added to the audit log and sent to the TACACS+ server as authorization and accounting messages.The commands `log-audit-exception-create`, `log-audit-exception-delete`, and `log-audit-exception-show` are used to control which CLI, shell, and vtysh commands are audited.

```
CLI (network-admin@Spine1) > log-audit-exception-create
```

`log-audit-exception-create`	Create an audit logging exception.
`cli|shell|vtysh`	Specify the type of audit exception.
`pattern` *`pattern-string`*	Specify a regular expression to match exceptions.
`any|read-only|read-write`	Specify the access type to match exceptions.
`scope local|fabric`	Specify the scope of exceptions.

```
CLI (network-admin@Spine1) > log-audit-exception-delete
```

`log-audit-exception-delete`	Delete an audit logging exception.
`cli|shell|vtysh`	Specify the type of audit exception.
`pattern` *`pattern-string`*	Specify a regular expression to match exceptions.
`any|read-only|read-write`	Specify the access type to match exceptions.

```
CLI (network-admin@Spine1) > log-audit-exception-show
```

`log-audit-exception-show`	Display audit logging exceptions.
`cli|shell|vtysh`	Display the type of audit exception.
`pattern` *`pattern-string`*	Display a regular expression to match exceptions.
`any|read-only|read-write`	Display the access type to match exceptions.
`scope local|fabric`	Display the scope of exceptions.

By default, NetVisor OS audits every command except for read-only CLI commands and shell commands with `^/usr/bin/nvmore` pattern which is the pager for NetVisor OS CLI:

```
CLI (network-admin@switch) > log-audit-exception-show

switch type  pattern              access     scope
------ ----- ---------------- --------- -----
switch cli                              read-only local
switch shell ^/usr/bin/nvmore any         local
```

To enable auditing of all CLI commands, you can delete the `cli/read-only` exception:

```
CLI (network-admin@switch) > log-audit-exception-delete cli read-only
```

Modifying User Roles

You can impart privileges to a user through the `role-create` command. To add shell access to a user's role, use the following syntax:

```
CLI (network-admin@switch) > role-create name role1 scope local shell
```

`role-create`	Create a user role.
`name` *`name-string`*	Specify a name for the user role.
`scope local|fabric`	Specify a scope for the user role.
Specify one or more of the following options:	
`access read-only|read-write`	Specify the type of access for the user role. The default is read-write.
`running-config|no-running-config`	Specify if the user role allows access to the switch running configuration.
`shell|no-shell`	Specify if the user role allows access to the shell.
`sudo|no-sudo`	Specify if the user role allows the sudo command.

The `role-modify` command can be used to modify a user role configuration.

Configuring System Logging

NetVisor OS allows you to send log messages to syslog servers over a UDP session or a TCP session with TLS encryption. By default, the messages are sent to syslog servers through UDP.

`CLI (network-admin@Leaf1) > admin-syslog-create`

`admin-syslog-create`	Use this command to configure syslog parameters.
Specify the following options:	
`name name-string`	Specify the name for the syslog configuration.
`scope local|fabric`	Specify one among the options the scope of the syslog.
`host` *`host-string`*	Specify the host name.
`port` *`port-number`*	Specify the host port number.
`transport tcp-tls|udp`	Specify TCP with TLS or UDP as the transport protocol for log events. UDP is used by default.
`message-format structured|legacy`	Specify one among the options as the message format for log events.
`export-container-logs|no-export-container-logs`	Enable or disable the export of container logs.
`export-os-logs|no-export-os-log`	Enable or disable the export of OS logs.

For example, to configure the switch to send all log messages to a syslog server with an IP address of `172.16.21.67` through port 10514, use the following command:

```
CLI (network-admin@Leaf1) > admin-syslog-create name <name> scope fabric
host 172.16.21.76 port 10514 message-format structured
```

To display the configuration, use the `admin-syslog-show` command:

`CLI (network-admin@Leaf1) > admin-syslog-show`

`admin-syslog-show`	Use this command to view the syslog configuration.
Specify the following options:	
`name name-string`	Specify the name of the syslog.
`scope local|fabric`	Specify the scope of the syslog.
`host` *`host-string`*	Specify the host name.
`port` *`port-number`*	Specify the host port number.
`transport tcp-tls|udp`	Specify TCP with TLS or UDP as the transport protocol for log events.

`message-format structured|legacy`	Specify one among the options as the message format for log events.
`status status-string`	Specify the syslog export status.
`export-container-logs|no-export-container-logs`	Enable or disable the export of container logs.
`export-os-logs|no-export-os-log`	Enable or disable the export of OS logs.

```
CLI (network-admin@Leaf1) > admin-syslog-show layout vertical
switch:                 leaf1
name:                   log-all
scope:                  fabric
host:                   172.16.21.76
port:                   10514
transport:              udp
message-format:          structured
export-container-logs:  off
export-os-logs:         off
```

You can send messages to syslog servers using either RFC5424 (structured) or RFC3164 (legacy) formats. To send syslog messages in a legacy format, add the `message-format` parameter to `admin-syslog-modify` command:

```
CLI (network-admin@Leaf1) > admin-syslog-modify name <name> message-format
legacy
```

You can also modify the port that the service listens on by using the `port` option in the CLI.

```
CLI (network-admin@Leaf1) > admin-syslog-modify name <name> port 22
```

By default, all log messages are sent to syslog servers. NetVisor OS offers the flexibility of configuring more than one syslog server and sending selective syslog messages to each of them. Filters for forwarding log messages can be created by using the command:

```
CLI (network-admin@Leaf1) > admin-syslog-match-add
```

`admin-syslog-match-add`	Add a syslog event match filter.
`syslog-name` *`name-string`*	Specify the name of the syslog file.
Specify the following match arguments:	
`name` *`name-string`*	Specify a name for the matching scheme.
Specify any of the following options:	
`msg-category event|audit|system|perror|container|os-logs|xact-truncate`	Specify one among the options as the category of the message to match.
`msg-program program-string`	Specify the name of the program used to generate the log messages.

`msg-name` *`name-string`*	Specify the name of the message to match.
`msg-code` *`code-number`*	Specify the message code to match.
`msg-level critical|error|warn|note|info`	Specify one among the options as the severity level to match. By default, all messages are forwarded to the syslog server.
`msg-event-type system|port|tcp|stp|igmp|lldp|lacp|vdp|ecp|evb|ptp|storage|tacacs|mld|mroute|vport|lacp-port-event`	Specify the type of event to match.
`msg-vnet` *`vnet-name`*	Specify the vNET name to match.
`msg-remote_switch` *`node-name`*	Specify the remote switch to match.
`msg-user` *`user-name`*	Specify user name to match.
`msg-client-addr ip-address`	Specify the client IP address.
`msg-port` *`port-number`*	Specify the port number to match.
`msg-vlan` *`0..4095`*	Specify the VLAN to match.
`msg-bd` *`bridge-domain-name`*	Specify the bridge domain to match.
`msg-vxlan` *`vxlan-ID`*	Specify the VXLAN to match.

In the absence of a match condition, NetVisor OS forwards all messages to the syslog server. However, if you configure a match condition, only the messages that match the specified parameters are forwarded. For example:

```
CLI (network-admin@Leaf1) > admin-syslog-create name to-10.10.10.10 scope fabric host 10.10.10.10 port 10514 message-format structured export-container-logs export-os-logs

CLI (network-admin@Leaf1) > admin-syslog-match-add syslog-name to-10.10.10.10 msg-category event name match-1
```

The above configuration forwards only event log messages to the syslog server, and all other categories are denied.

To display the configuration, use the `admin-syslog-match-show` command:

```
CLI (network-admin@Leaf1) > admin-syslog-match-show
syslog-name     msg-category msg-level name
--------------  ------------ --------- -------
to-10.10.10.10 event                   match-1
```

Displaying Syslog Information

To view system log information, use the command `log-system-show`:

`log-system-show`	Display system log information.
Specify between zero to two of the following options:	
`start-time date/time: yyyy-mm-ddThh:mm:ss`	Specify the start time from which to display the logs.
`end-time date/time: yyyy-mm-ddThh:mm:ss`	Specify the end time for the log file.
`duration duration: #d#h#m#s`	Specify the duration of the log file.
Specify any of the following parameters to view the information related to those parameters:	
`program program-string`	Specify the program that generates log messages.
`pid pid-number`	Specify the product ID generating the log messages.
`name name-string`	Specify the message name.
`code code-number`	Specify the message code.
`level critical|error|warn|note|info`	Specify one among the options as the level of severity.
`vnet vnet-name`	Specify the associated VNET.
`remote_switch node name`	Specify the name of the remote switch.
`client-pid client-pid-number`	Specify the client product ID.
`client-addr ip-address`	Specify the client IP address.
`port port-number`	Specify the port number.
`vlan vlan-id`	Specify the VLAN ID as a value between 2 and 4092.
`bd bridge-domain name`	Specify the bridge domain.
`vxlan vxlan-id`	Specify the VXLAN ID.
`count number 1..50000`	Specify the number of events to be displayed in a range from 1 to 50000.
`starting-point starting-point-number`	Specify the starting point of the log audit.
`length length-number`	Specify the length of the log audit.
`reverse|no-reverse`	Use this option to enable or disable displaying the messages in reverse order.

For example:

```
CLI (network-admin@Leaf1) > log-system-show count 3 layout vertical
category:          system
time:               2020-04-13,12:36:00.966426-07:00
name:              fabric_node_status
code:              11403
level:             note
message:            pavo-colo-5: Node status changed to mgmt-only-online
category:          system
time:               2020-04-13,12:36:46.462439-07:00
name:              congestion_relieved_on_port
code:              11402
level:             critical
message:            Congestion relieved on port=0
category:          system
time:               2020-04-13,12:36:46.509220-07:00
name:              congestion_relieved_on_port
code:              11402
level:             critical
message:            Congestion relieved on port=126
```

Note: Prior to NetVisor OS 6.0.0 release, during MAC and IP move, the nvOSd.log file gets flooded with MAC move notifications. Starting from NetVisor OS version 6.0.0, the MAC and IP move messages are logged in to the system.log file and can be accessed using the `log-system-show` command. The output of the command is in a summarized format that displays the repetition count instead of printing each message.

Note: Starting from NetVisor OS version 6.0.1, all system log messages are logged into system.log file. This prevents selective system log messages from being logged into of nvOSd.log or perror.log files.

Displaying Syslog Counters

You can view the number of events that have occurred in the network belonging to the default severity levels by using the `log-system-counters-show` command:

```
CLI (network-admin@Leaf1) > log-system-counters-show layout vertical
switch:       Leaf1
critical:     0
error:        0
warn:         1061
note:         9
```

To reset the log counters, issue the `log-system-counters-reset` command.

Forwarding Log Files to an External Server

Log messages can be sent to an external Linux server and encrypted using TLS over TCP. NetVisor OS supports only one external server for TCP-TLS export while the UDP syslog export can be done to more than one server.

Follow the steps below to configure exporting of logs to an external server:

- Enable SFTP import/export using command below:

```
CLI (network-admin@Leaf1) > admin-sftp-modify enable
```

- Create the private key and the Certificate Signing Request (CSR) for the switch using the command `syslog-tls-cert-request-create.`

`syslog-tls-cert-request-create`	This command creates a certificate request for the TLS connection.
`country country-string`	Specify the contact address starting with the country code.
`state state-string`	Specify the state or province.
`city` *`city-string`*	Specify the city.
`organization` *`organization-string`*	Specify the organization.
`organizational-unit` *`organizational-unit-string`*	Specify the organizational unit.
`common-name` *`common-name-string`*	Specify the common name. This name must match the switch hostname.

For example:

```
CLI (network-admin@Leaf1) > syslog-tls-cert-request-create country US state
CA city Palo Alto organization QA organizational-unit engineering common-
name Leaf1
```

This command creates a Certificate Signing Request (CSR) and places it in the directory /sftp/export used by NetVisor OS. You must get the CSR signed by the Certificate Authority (CA) and import the ca.pem and server-cert.pem files to NetVisor OS.

- To import the signed certificate and CA root certificate files, you must upload the my-cert.pem and the ca.pem files to `/sftp/import` directory in NetVisor OS and run the following command:

```
CLI (network-admin@Leaf1) > syslog-tls-cert-import file-ca ca.pem file-cert
my-cert.pem
```

`syslog-tls-cert-import`	Import certificates from /sftp/import directory.
Specify the following options:	

`file-ca` *`file-ca-string`*	Name of the CA certificate file.
`file-cert` *`file-cert-string`*	Name of switch certificate file (signed by CA).

- To enable TLS-TCP logging export, use the following syntax:

```
CLI (network-admin@Leaf1)>admin-syslog-create name audit-logs scope local
host 172.16.21.33 transport tcp-tls port 10514
```

This command can be executed anywhere in the sequence.

- To display and verify the syslog export configuration, use the `admin-syslog-show` command:

```
CLI (network-admin@leo-ext-23) > admin-syslog-show layout vertical
switch:                 leo-ext-23
name:                   audit-logs
scope:                  local
host:                   172.16.21.33
port:                   10514
transport:              tcp-tls
message-format:         legacy
export-container-logs:  off
export-os-logs:         off
```

To display alert messages related to syslog export, use the command `log-alert-show`. This command displays events such as a disruption in connection to the syslog TLS server and the restoration of the connection. For example:

```
CLI (network-admin@switch1) > log-alert-show
time:         07:31:32
switch:       switch1
code:         20006
name:         syslog_tls_server_down
count:        1
last-message: tcp-tls connection to syslog server=MYTLS down. Logs are not getting exported
time:         07:32:50
switch:       switch1
code:         20007
name:         syslog_tls_server_down
count:        1
last-message: tcp-tls connection to syslog server=MYTLS restored. Log export is operational
```

Related Commands

- `syslog-tls-cert-clear`

Use this command to delete imported certificates.

For example:

```
CLI (network-admin@switch1) > syslog-tls-cert-clear
Successfully deleted all certificate files.
```

- `syslog-tls-cert-info-show`

Use this command to display certificate information.

`syslog-tls-cert-info-show`	Display the certificate information.
Specify any of the following options:	
`cert-type ca|intermediate|server`	Specify the one among the options as the certificate type.
`subject` *`subject-string`*	Specify the the subject of the certificate.
`issuer` *`issuer-string`*	Specify the issuer of the certificate.
`serial-number` *`serial-number`*	Specify the serial number of the certificate.
`valid-from` *`valid-from-string`*	Specify the time from which the certificate is valid.
`valid-to` *`valid-to-string`*	Specify the time at which the certificate expires and is no longer valid.

For example:

```
CLI (network-admin@switch1) > syslog-tls-cert-info-show
switch:         switch1
cert-type:      server
subject:          /C=US/ST=CA/L=PA/O=Eng/OU=TT/CN=switch1.pluribusnetworks.com
issuer:           /C=US/ST=CA/L=PA/O=Eng/OU=TT/CN=switch1.pluribusnetworks.com
serial-number:  1
valid-from:     Oct 20 09:06:02 2016 GMT
valid-to:       Oct 20 09:06:02 2017 GMT
```

- The `syslog-tls-cert-show` displays the syslog TLS import certificate configuration.

`syslog-tls-cert-show`	Displays the certificate information.
Specify any of the following options:	
`file-ca` *`file-ca-string`*	Specify the name of CA certificate file.
`file-cert` *`file-cert-string`*	Specify the name of switch certificate file (signed by CA).
`cert-ca` *`cert-ca-string`*	Specify the CA certificate.
`cert-switch` *`cert-switch-string`*	Specify the switch certificate.

For example:

```
CLI (network-admin@switch1) > syslog-tls-cert-show
file-ca  file-cert
-------  -----------
ca.pem   my-cert.pem
```

Understanding Network Packet Broker

The Arista Network Packet Broker (NPB) solution enables users to deploy a dedicated modular, scalable monitoring fabric with a distributed architecture that connects to shared analytics and security tools located anywhere in the network. A monitoring fabric deployed as part of Arista's NPB solution features centralized management capabilities and logically functions as a distributed virtual chassis.

A network packet broker topology is built on top of the familiar NetVisor OS's Unified Cloud Fabric: it does not require specialized software and a custom fabric, and consequently provides a high degree of flexibility, resiliency, and operational simplicity.

Using network taps or port mirrors, the NPB service copies traffic from a production network to the ingress ports of the monitoring fabric which, in turn, redirects the copied packets to the monitoring tools (which can be located geographically apart).

Arista's implementation employs a VXLAN overlay to transport packets from ingress ports to a “tool farm” (that is, a group of monitoring tools) and supports Equal Cost Multi-Path (ECMP) forwarding in the underlay to address link failures.

For more details on platform support, an example of NPB topology, and the configuration steps, refer to the *Configuring Network Packet Broker* section.

Configuring Network Packet Broker

Note: The Arista Network Packet Broker (NPB) solution is available on all Dell, Edgecore, and Freedom series platforms supported by Arista Networks.

An example of Arista Network Packet Broker topology is depicted in the figure below and will be used to illustrate the configuration steps:

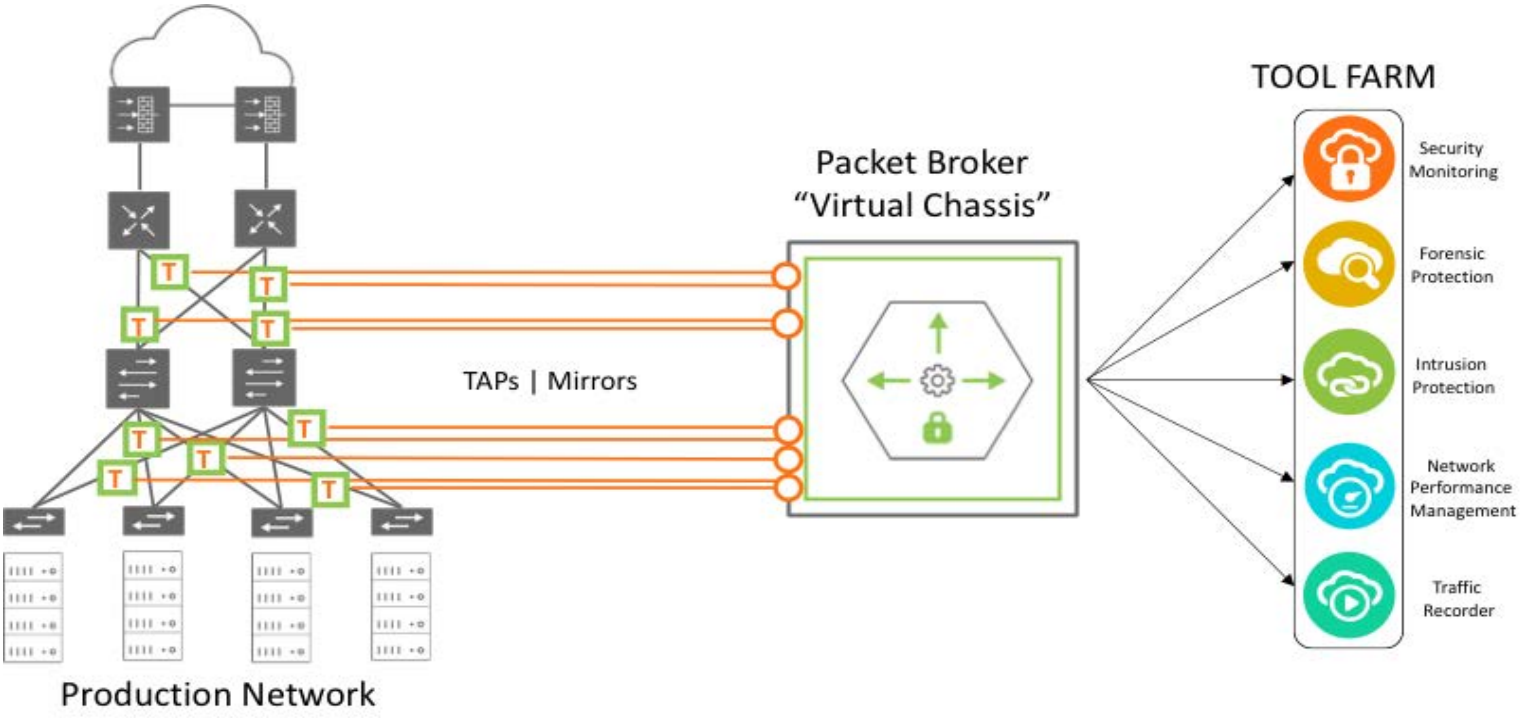


Figure 11-2: Network Packet Broker Architecture

The monitoring fabric can have any physical topology, including leaf/spine, ring, hub and spoke, mesh, and tree. NetVisor OS supports the virtual Port Group (vPG) CLI construct to define a set of ingress or source ports and of destination ports so that the hardware will flood the traffic that arrives at the source ports to (multiple) desired destination ports where the monitoring tools are connected. This type of vPG configuration is referred to as a unidirectional vPG topology (as the monitored traffic flows in one direction only).

Refer to the Configuring Virtual Port Group section of the Configuring Advanced Layer 2 Transport Services chapter.

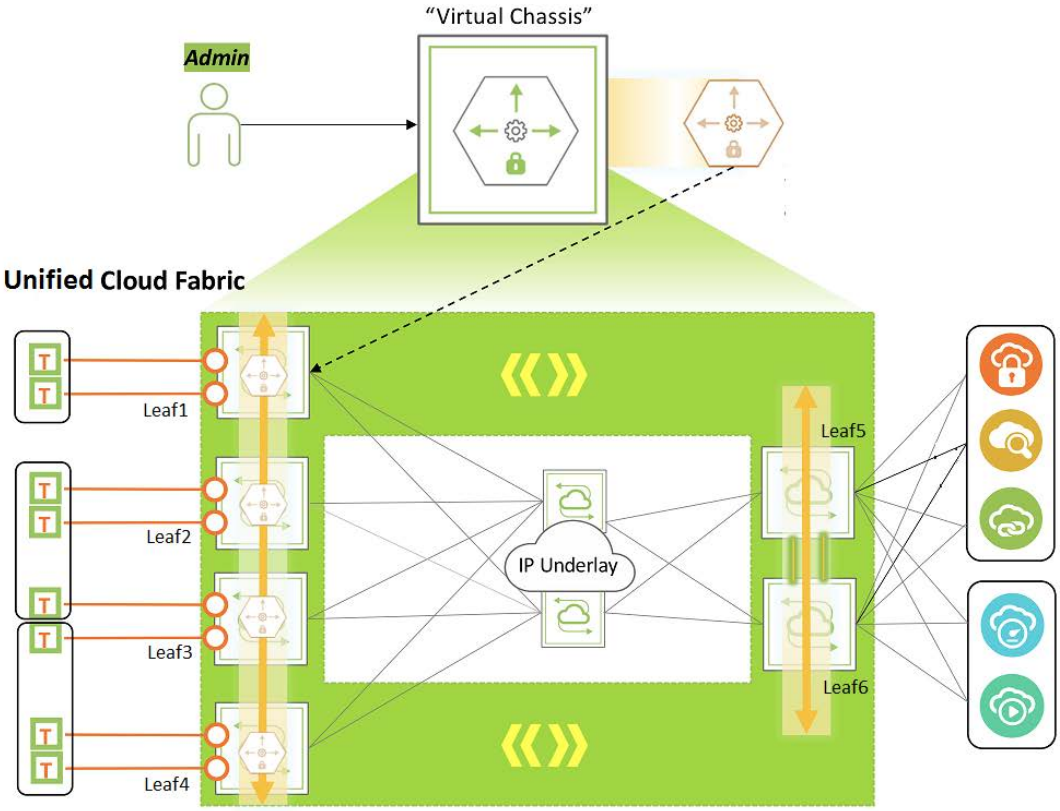


Figure 11-3: Monitoring Fabric Topology

Consider, for example, a monitoring fabric with a leaf-spine topology as shown above in Figure 11-3. Network taps copy traffic from the production network to the source ports or trunks on Leaf1, Leaf2, Leaf3, and Leaf4. These ports constitute the source vPGs: TAP-GROUP-1, TAP-GROUP-2, and TAP-GROUP3. The switches Leaf5 and Leaf6 form a cluster. The monitoring tools are connected to ports on Leaf5 and Leaf6 which constitute the destination vPGs: TOOL-GROUP-1 and TOOL-GROUP-2.
This section describes the steps to configure source and destination vPGs and to create a forwarding policy.

Before creating the vPGs, you must configure a VXLAN underlay network and VTEPs for the overlay. For details, refer to the sections, *Configuring the VXLAN Underlay Network* and *Configuring VTEP Objects with Automatic Fabric Connections*

Also, to deploy the Packet Broker fabric that spreads across geographical locations, you must create a Fabric over Layer 3 configuration. For details, refer to the section, *Configuring a Fabric over a Layer 3 Network*

Follow the steps below to configure the port groups and to send traffic from a source port group to a destination port group:

1. Configuring Source vPGs

Create vPGs for the source ports of the monitoring fabric, by using the command `vpg-create`.

```
CLI (network-admin@switch1) > vpg-create
```

`vpg-create`	Creates a virtual Port Group (vPG)
`vnet` *`vnet-name`*	Specify the name of the vNET that you want to associate with this vPG.
`name` *`name-string`*	Specify the name of the vPG.
`type source|destination| bidirectional`	Specify the type of the vPG as `source`, `destination`, or `bidirectional`.
`ports` *`port-list`*	Specify the ports assigned to this vPG.
`port-desc port-desc-string`	Specify a previously configured port description to add the ports with that description to the vPG.
`monitor-ports port-list`	Specify the monitoring ports for destination or bidirectional vPGs.

Below is an example configuration as per ***Figure 11-3***.

```
CLI (network-admin@Leaf1) > vpg-create name TAP-GROUP-1 type source

CLI (network-admin@Leaf1) > vpg-create name TAP-GROUP-2 type source

CLI (network-admin@Leaf1) > vpg-create name TAP-GROUP-3 type source
```

Add the source ports to the vPGs by using the `vpg-port-add` command:

```
CLI (network-admin@Leaf1) > vpg-port-add
```

`vpg-port-add`	Adds ports to a vPG.
`vpg-name` *`name-string`*	Specify a name for this vPG.
`ports` *`port-list`*	Specify the source ports for this vPG.
`port-desc` *`port-desc-string`*	Specify a previously configured port description to add the ports with that description to the vPG.

The switches Leaf1, Leaf2, and Leaf3 form trunks (Link Aggregation Groups) with network taps, with trunk IDs 273, 274, and 275 as shown in ***Figure 11-3***.

```
CLI (network-admin@Leaf1) > vpg-port-add vpg-name TAP-GROUP-1 ports 273

CLI (network-admin@Leaf1) > switch Leaf2 vpg-port-add vpg-name TAP-GROUP-2
ports 274

CLI (network-admin@Leaf1) > switch Leaf3 vpg-port-add vpg-name TAP-GROUP-2
ports 22
```

```
CLI (network-admin@Leaf1) > switch Leaf3 vpg-port-add vpg-name TAP-GROUP-3
ports 25

CLI (network-admin@Leaf1) > switch Leaf4 vpg-port-add vpg-name TAP-GROUP-3
ports 275
```

Note: The source vPG can include ports from all the nodes in the fabric. However, this port group can only include at most one port or trunk from each node.

To view the vPG configuration, use the command `vpg-show`.

```
CLI (network-admin@Leaf1) > vpg-show
```

`vpg-show`	Displays vPG configuration details.
`scope`	The scope of the vPG.
`vnet` *`vnet-name`*	vNET name associated with this vPG.
`name` *`name-string`*	The name of the vPG.
`type source|destination| bidirectional`	The vPG type that was configured.
`ports` *`port-list`*	The ports assigned to this vPG.
`port-desc` *`port-desc-string`*	The port description.
`vni` *`0..16777215`*	The VNI or VXLAN identifier for auto BD configuration.
`vlan` *`0..4095`*	The VLAN identifier for auto BD configuration.
`monitor-ports` *`monitor-ports-list`*	The list of monitoring ports.

```
CLI (network-admin@leaf1) > vpg-show
switch scope  name        type   ports vni vlan
------ ------ ---------- ------ ----- --- ----
Leaf1  fabric TAP-GROUP-3 source none  0   0
Leaf2  fabric TAP-GROUP-3 source 25    0   0
Leaf3  fabric TAP-GROUP-3 source 275   0   0
Leaf4  fabric TAP-GROUP-3 source none  0   0
Leaf5  fabric TAP-GROUP-3 source none  0   0
Leaf6  fabric TAP-GROUP-3 source none  0   0
Leaf1  fabric TAP-GROUP-2 source none  0   0
Leaf2  fabric TAP-GROUP-2 source 274   0   0
Leaf3  fabric TAP-GROUP-2 source 22    0   0
Leaf4  fabric TAP-GROUP-2 source none  0   0
Leaf5  fabric TAP-GROUP-2 source none  0   0
```

```
Leaf6  fabric TAP-GROUP-2 source none  0   0
Leaf1  fabric TAP-GROUP-1 source 273   0   0
Leaf2  fabric TAP-GROUP-1 source none  0   0
Leaf3  fabric TAP-GROUP-1 source none  0   0
Leaf4  fabric TAP-GROUP-1 source none  0   0
Leaf5  fabric TAP-GROUP-1 source none  0   0
Leaf6  fabric TAP-GROUP-1 source none  0   0
```

2. Configuring Destination vPGs

Create destination tool groups by using the following commands:

```
CLI (network-admin@Leaf1) > switch Leaf5 vpg-create name TOOL-GROUP-1 type destination

CLI (network-admin@Leaf1) > switch Leaf5 vpg-create name TOOL-GROUP-2 type destination
```

Note: For destination vPG creation to be successful, you must add at least one port to the `vxlan-loopback-trunk` on all the nodes of the NPB fabric, including spine switches. For more details, refer to '*Configuring the VXLAN Loopback Trunk*' topic of the '*Configuring VXLAN*' chapter.

Add the ports connected to the monitoring tools into the tool groups by using the command `vpg-port-add`:

For example:

```
CLI (network-admin@Leaf1) > switch Leaf5 vpg-port-add vpg-name TOOL-GROUP-1 ports 22,34,45

CLI (network-admin@Leaf1) > switch Leaf6 vpg-port-add vpg-name TOOL-GROUP-1 ports 47,50

CLI (network-admin@Leaf1) > switch Leaf5 vpg-port-add vpg-name TOOL-GROUP-2 ports 17,32

CLI (network-admin@Leaf1) > switch Leaf6 vpg-port-add vpg-name TOOL-GROUP-2 ports 48,55
```

Note: Starting from NetVisor OS version 7.0.0, you can add the same port to multiple destination vPGs. This enhancement allows tool ports to receive partially overlapping traffic. For more information, see *'Use cases for Network Packet Broker*' topic.

The `vpg-port-add` commands automatically create bridge domains for TOOL-GROUP1 and TOOL-GROUP-2. These bridge domains flood the arriving traffic to the ports in the respective tool groups.

Use the `vpg-show` command to view the configuration:

```
CLI (network-admin@leaf1) > vpg-show type destination
switch  scope  name         type        ports    vni      vlan
------  ------ ------------ ----------- -----    -------- ----
Leaf1   fabric TOOL-GROUP-2 destination none     12666666 2833
```

```
Leaf2   fabric TOOL-GROUP-2 destination none     12666666 2833
Leaf3   fabric TOOL-GROUP-2 destination none     12666666 2833
Leaf4   fabric TOOL-GROUP-2 destination none     12666666 2833
Leaf5   fabric TOOL-GROUP-2 destination 17,32    12666666 2833
Leaf6   fabric TOOL-GROUP-2 destination 48,55    12666666 2833
Leaf1   fabric TOOL-GROUP-1 destination none     12000000 2500
Leaf2   fabric TOOL-GROUP-1 destination none     12000000 2500
Leaf3   fabric TOOL-GROUP-1 destination none     12000000 2500
Leaf4   fabric TOOL-GROUP-1 destination none     12000000 2500
Leaf5   fabric TOOL-GROUP-1 destination 22,34,45 12000000 2500
Leaf6   fabric TOOL-GROUP-1 destination 47,70    12000000 2500
```

Note: NetVisor OS auto-generates a VNI and reserved VLAN ID for the destination vPGs.

3. Configuring the vFlow Policy

As the final step of the configuration, create a vFlow to forward traffic from the desired source port group to a destination port group. For example, use the following command to send traffic from TAP-GROUP-1 to TOOL-GROUP-2 with protocol transparency enabled:

```
CLI (network-admin@Leaf1) > vflow-create name TAP1-TOOL2 scope fabric src-vpg TAP-GROUP-1 dst-vpg TOOL-GROUP-2 transparency enable
```

This configuration floods the ingress traffic matching the vFlow to all the ports in TOOL-GROUP-2 over the VXLAN tunnel.

The above configuration also sends control plane traffic from source to destination as protocol transparency is enabled. In the absence of protocol transparency, the control plane traffic that originates from a switch is generally processed or dropped by the next hop switch. With protocol transparency, the traffic belonging to various Layer 2, Layer 3, and Layer 4 protocols can be sent to switches located anywhere within the fabric. These control plane protocols include:

- Layer 2: LLDP, STP, LACP, and VLAN
- Layer 3: IPv4 and IPv6
- Layer 4: TCP, UDP, and ICMP
- Layer 4 ports: HTTP, DNS, and DHCP

When you enable protocol transparency, NetVisor OS automatically configures additional vFlows to ensure that the control traffic reaches the destination vPG. This auto-configuration is necessary as system vFlows generally consume the protocol traffic and therefore, there must be pathways for the control traffic to bypass system vFlows. To display the vFlow configuration, use the `vflow-show` command:

```
CLI (network-admin@Leaf1) > vflow-show format all layout vertical
name:                       TAP1-TOOL2
id:                         b0022cd:1a9
scope:                      fabric
type:                       vflow
hidden:                     false
in-port:                    275
burst-size:                 auto
precedence:                 3
action-set-svp-value:       0x80000006
log-stats:                  no
stats-interval:             60
```

```
hw-stats:                    enabled
src-vpg:                     TAP-GROUP-1
dst-vpg:                     TOOL-GROUP-2
transparency:                enable
enable:                      enable
table-name:                  System-VCAP-table-1-0
name:                        npb-system-bypass-proto
id:                          b0022cd:1a8
scope:                       fabric
type:                        vflow
precedence:                  15
action:                      none
metadata:                    1000
hw-stats:                    enabled
enable:                      enable
table-name:                  System-L1-L4-Tun-1-0
```

Note: The auto-configured vFlow, `npb-system-bypass-proto` having a high precedence of 15 bypasses system vFlow treatment for packets at the ingress side.

```
CLI (network-admin@Leaf6) > vflow-show format all layout vertical
name:                        npb-tunnel-decap-b0022cd:1a9-1
id:                          b0022cf:16e
scope:                       local
type:                        vflow
precedence:                  15
action:                      none
vxlan:                       12666668
from-tunnel-decap:           yes
hw-stats:                    enabled
enable:                      enable
table-name:                  System-L1-L4-Tun-1-0
```

The auto-configured vFlow on the destination switch bypasses system vFlow treatment for the decapsulated packets at the egress end of tunnels.

Note: The running configuration for the `vflow-create` command has parameters such as `in-port`. However, while configuring a vFlow, you should not configure these parameters as these fields are auto-populated.

Additional CLI Commands

1. `vpg-delete`

Use this command to delete a vPG. You cannot delete a vPG that is associated with a vFlow. In such cases, you must delete the vFlow before you delete the vPG.

```
CLI (network-admin@Leaf1) > vpg-delete
```

`vpg-delete`	Deletes a vPG.
`name` *`name-string`*	Enter the name of the vPG you want to delete.

For example:

```
CLI (network-admin@Leaf1) > vpg-delete name <vpg-name>
```

2. `vpg-port-remove`

Use this command to remove ports from a vPG.

```
CLI (network-admin@Leaf1) > vpg-port-remove
```

`vpg-port-remove`	Removes ports from a vPG.
`vpg-name` *`name-string`*	Enter the name of the vPG from where you want to remove the ports.
`ports` *`port-list`*	Specify the ports to be removed from the vPG.
`port-desc` *`port-desc-string`*	Specify the port description to remove the ports with that description from the vPG.

For example:

```
CLI (network-admin@Leaf1) > vpg-port-remove vpg-name <vpg-name> ports
<port-list>
```

Note: For a given switch, if you remove a port or trunk from a source vPG, the associated vFlow on that switch is disabled in hardware but it is retained by the software.

Guidelines and Limitations

- If you configure a protocol transparent vFlow, you cannot use the `set-metadata` keyword while modifying the vFlow thereafter. Conversely, if `set-metadata` is configured for a vFlow, you cannot enable protocol transparency for that vFlow.
- The metadata value of 1000 is reserved for protocol transparency and you cannot use this metadata value while configuring other vFlows.
- All control plane protocols are tunnel transparent if no additional filters are enabled. However, if you specify a vFlow filter, only the control plane packets pertaining to that filter is copied between the vPGs. For example, if you specify a VLAN filter, only the control plane packets belonging to that VLAN are copied. This limitation applies to the global vNET and any user-defined vNET in which vFlows are created with a VLAN filter. As a consequence of this limitation, adjacency over LLDP or LACP fails to be established.

Configuring IPv6 Filters for Network Packet Broker

NetVisor OS version 6.1.0 expands IPv6 filtering options for Network Packet Broker. To enable IPv6 filtering for NPB, you must first configure an IPv6 VLAN Content Aware Processing (VCAP) table. This IPv6 VCAP table is created by dividing the 512-entry VCAP table into 256 IPv4 entries and 256 IPv6 entries.

Create an IPv6 VCAP table by using the command:

```
CLI (network-admin@switch) > vflow-table-profile-modify profile ipv6-vcap
hw-tbl switch-main enable
Note: Please reboot the system for this new profile setting to take effect
```

As seen from the CLI, the switch must be rebooted before the vFlow table profile modification can take effect.

After restarting the switch you can view the vFlow table profile configuration by using the command:

```
CLI (network-admin@switch*) > vflow-table-profile-show layout vertical
switch:               switch
profile:              ipv6-vcap
hw-tbl:               switch-main
enable:               enable
flow-capacity:        0
flow-slices-needed:   1
flow-slices-used:     0
comment:              VCAP-IPv6
```

View the vFlow table configuration by using the command:

```
CLI (network-admin@switch*) > vflow-table-show
name                  flow-max-per-group flow-used flow-tbl-slices capability     flow-profile
--------------------- ------------------ --------- --------------- -------------- ------------
Egress-Table-1-0      512                1         2               match-metadata system
System-L1-L4-Tun-1-0  2048               55        2               set-metadata   system
System-VCAP-table-1-0 256                0         2               none           system
VCAP-IPv6-table-1-0   256                0         1               none           vcap-v6
```

You can now create a vFlow to direct traffic from a source vPG to a destination vPG with IPv6 filters. For example, direct all traffic with IPv6 address 1000::1/64 from vPG-1 to vPG-10 by using the command:

```
CLI (network-admin@switch) > vflow-create name vpg_v6 scope local src-ip
1000::1/64 src-vpg vPG-1 dst-vpg vPG-10 table-name VCAP-IPv6-table-1-0
```

Note: You must specify the table name as `VCAP-IPv6-table-1-0` in the `vflow-create` command to enable IPv6 filtering for NPB.

Use the `vflow-show` command to view the configuration.

```
CLI (network-admin@leo-norma2) > vflow-show name vpg_v6 layout vertical
switch:                 switch
name:                   vpg_v6
type:                   vflow
src-ip:                  1000::1/ffff:ffff:ffff:ffff::
burst-sizeprecedence:   autodefault
src-vpg:                vPG-1
dst-vpg:                vPG-10
enable:                 enable
table-name:              VCAP-IPv6-table-1-0
```

To disable the IPv6 VCAP table, use the command:

```
CLI (network-admin@switch) > vflow-table-profile-modify profile ipv6-vcap
hw-tbl switch-main no-enable
Note: Please reboot the system for this new profile setting to take effect
```

Configuring Regular Traffic and vPG Traffic Over the Same Fabric Topology

Starting with NetVisor OS version 7.0.1, you can configure the Network Packet Broker functionality on the same network topology used for regular traffic (the so-called production network).

With NetVisor OS, only one VTEP configuration with auto-tunnel is allowed on a per switch (or switch pair) basis. The Network Packet Broker (NPB) functionality uses VTEPs with MAC learning disabled, whereas a production network uses VTEPs with MAC learning enabled. Therefore, because of this divergence, before NetVisor OS version 7.0.1, the fabric could be used either for production traffic or for NPB functionality. NetVisor OS version 7.0.1 enables both production and NPB traffic to coexist in the same fabric by using the same underlay and the same VTEPs. For a single VTEP, two tunnels are created, one for production traffic and one for NPB functionality. For production traffic, VXLAN tunnels is configured in standard mode. Additionally, for NPB traffic VXLAN tunnels are configured in transparent mode. This combination is referred to as hybrid mode.

Hybrid Mode

NetVisor OS can interconnect VTEPs automatically upon creation: when the switch is in hybrid mode (i.e., is configured for both standard and transparent mode), it creates two tunnels, one for regular traffic and one for NPB traffic. When hybrid mode is enabled for the same source IP, source port, destination IP and destination port, two tunnels are created: one standard tunnel with MAC learning enabled and another transparent tunnel with MAC learning disabled. Per-mode VNIs are associated to the VTEPs and are added to the respective tunnels in the hardware. (For more details on VTEPs, automatic tunnels and the VXLAN feature, refer to the Configuring VXLAN chapter.)

> **Note:** Once the feature is enabled, and when a regular VTEP is created, for this special VTEP an automatic tunnel is created for standard mode only, as an additional tunnel is not supported nor required for NPB.

> **Note:** While upgrading from earlier versions to NetVisor OS version 7.0.1, ensure that the hybrid-mode is enabled before proceeding with upgrade if you want to configure both vLE and vPG on the same fabric.

The feature is disabled by default and can be enabled using the following command:

```
CLI (network-admin@switch) > system-setting-modify  hybrid-mode
```

To disable the hybrid mode, use the command:

```
CLI (network-admin@switch) > system-setting-modify  no-hybrid-mode
```

The hybrid mode value is visible only after enabling the hybrid mode feature. To view details, use the command:

```
CLI(network-admin@switch) > bridge-domain-show
switch   name         scope  vxlan    auto-vxlan vxlan-hybrid-mode rsvd-vlan ports         vxlan-inner-
packet mac-learning l2-tunneling
-------- ------------ ------ -------- ---------- ----------------- --------- -------------
----------------- ------------ ------------
switch-1 BD-1         fabric 10800    no         standard          1090                    auto
  on           none
switch-2 BD-1         fabric 10800    no         standard          1090                    auto
  on           none
```

```
switch-3 BD-1         fabric 10800    no         standard          1090                    auto
  on           none
switch-1 auto-dst_vpg local  12333333 no         transparent       2666      57-59,272,397 auto
  off
switch-2 auto-dst_vpg local  12333333 no         transparent       2666      25-27,272,397 auto
  off
switch-3 auto-dst_vpg local  12333333 no         transparent                 397           auto
  off
```

Note: You must reboot the switch after enabling or disabling feature.

Note: Bridge domains created as part of the NPB functionality will have the `transparent` value in the `vxlan-hybrid-mode` output column.

Note: When hybrid mode is disabled, the entire column of `vxlan-hybrid-mode` field is not visible in the show command output.

Note: When hybrid mode is enabled, more hardware resources will be utilized: the number of vFlow entries in the system table is increased with one additional vFlow entry allocated per BD. If the system table gets full, new vPG creation will fail. So, if only NPB/production traffic is to be used in the fabric, hybrid mode can be disabled.

Note: Enabling this feature results in creating a vFlow to block the CPU's incoming traffic at tunnel decapsulation. The tupple for this ICAP vFlow is VXLAN of vPG bridge-domain. This will hinder any additional vFlow the user has to create from vPG traffic on the egress side.

VTEP Tunnel Creation in Hybrid Mode

In NetVisor OS version 7.0.1 or later, when hybrid mode feature is enabled two tunnels are created, one for standard and one for transparent mode as shown in the figure below.

- In standard mode, auto-tunnel naming follows the standard rules. In transparent mode, the keyword _transparent_mode is appended.
- A transparent mode tunnel uses the same tunnel source and tunnel destination as a standard tunnel, but with MAC learning disabled and enabled for transparent as well as standard tunnel respectively.

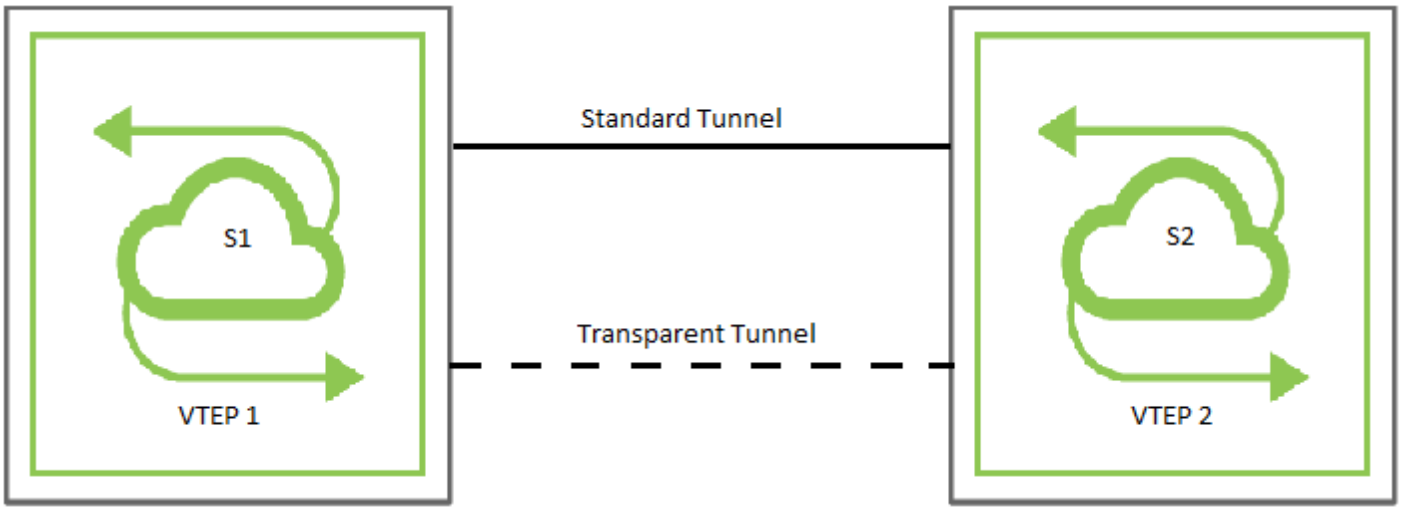


Figure 11-4 - Switches with two auto tunnels in Hybrid mode

For more details, visit the Configuring the Overlay: VTEP Interconnections and VNIs topic.

Use Cases for Network Packet Broker

- From NetVisor OS version 7.0.0, you can add the same port to multiple destination vPGs.

For example, you can add a common port to two distinct destination vPGs dvpg1 and dvpg2 to obtain partially overlapping traffic on the destination ports, as demonstrated by the configuration steps below:

Configure the source vPG:

```
CLI (network-admin@Leaf1) > vpg-create name svpg1 type source ports 27
```

Configure destination vPGs dvpg1 and dvpg2 with a common port 30:

```
CLI (network-admin@Leaf2) > vpg-create name dvpg1 type destination ports
30,40

CLI (network-admin@Leaf2) > vpg-create name dvpg2 type destination ports 30
```

Configure vFlows that forward ICMP traffic to dvpg1 and HTTP traffic to dvpg2.

```
CLI (network-admin@Leaf1) > vflow-create name flow1 scope fabric src-vpg
svpg1 dst-vpg dvpg1 proto icmp

CLI (network-admin@Leaf1) > vflow-create name flow2 scope fabric src-vpg
svpg1 dst-vpg dvpg2 proto http
```

With this configuration, port 30 receives both HTTP and ICMP traffic as it is a part of both the destination vPGs, while port 40 receives ICMP traffic alone.

- From NetVisor OS version 7.0.0, to configure a vFlow to send traffic from a source vPG to destination vPG after stripping the outer VLAN tag of single-tagged or double-tagged packets, use the vFlow action `strip-outer-vlan`. For example:

```
CLI (network-admin@switch) > vflow-create name flow1 scope fabric src-vpg vpg1 dst-vpg vpg2 action strip-
outer-vlan

CLI (network-admin@switch) > vflow-show format name,scope,src-vpg,dst-vpg,action,table-name
name  scope  src-vpg dst-vpg action            table-name
----- ------ ------- ------- --------------- --------------------
flow1 fabric vpg1    vpg2    strip-outer-vlan System-VCAP-table-1-0
```

- From NetVisor OS 6.1.0 onward, the vFlow parameters `src-mac` and `dst-mac` are supported in the System-VCAP table in addition to the System-L1-L4 table. This enhancement allows you to use these parameters while configuring the Network Packet Broker (NPB) solution. For example:

```
CLI (network-admin@switch) > vflow-create name flow1 scope fabric src-mac
00::0a dst-mac 00::aa src-vpg vpg1 dst-vpg vpg2

CLI (network-admin@switch) > vflow-show format,name,id,scope,in-port,src-mac,dst-mac,src-vpg,dst-vpg,table-
name

name  id           scope  type  in-port src-mac dst-mac src-vpg dst-vpg  table-name
----- ---------- ------ ----- ------- ------- ------- ------- -------  --------------------
flow1 9000a92:41 fabric vflow 13       00::aa  00::aa  vpg1    vpg2     System-VCAP-table-1-0
```

- NetVisor OS version 6.1.0 introduces the `inner-vlan` vFlow parameter to support filtering of traffic based on the inner VLAN of a QinQ frame. This parameter is supported by the System-L1-L4 hardware table and can be configured as part of NPB deployments. You can set a metadata value for the NPB vFlow in System-VCAP table, and this value can be supplied along with `inner-vlan` parameter in another vFlow for filtration of NPB traffic based on inner VLAN ID.

For example, create a NPB vFlow and assign a metadata value to it:

```
CLI (network-admin@switch) > vflow-create name vflow2 scope fabric src-vpg
vPG1 dst-vpg vPG2 set-metadata action-value 201
```

Use the metadata value to configure vFlows that permit and drop NPB traffic with respect to the `inner-vlan` parameter.

Create a vFlow to permit NPB traffic with inner VLAN ID 10:

```
CLI (network-admin@switch) > vflow-create name vflow3 scope fabric metadata
201 inner-vlan 10 precedence 14 action none
```

```
CLI (network-admin@switch) > vflow-show
switch name  scope type  inner-vlan in-port burst-size precedence enable table-name
------ ----- ----- ----- ---------- ------- ---------- ---------- ------ -------------------
switch flow2 local vflow 10         10      auto       default    enable System-L1-L4-Tun-1-0
```

Create a vFlow to drop all other NPB traffic:

```
CLI (network-admin@switch) > vflow-create name vflow4 scope fabric metadata
201 precedence 13 action drop
```

- NetVisor OS version 6.1.0 supports the configuration of a vFlow action along with traffic redirection from a source vPG to a destination vPG, using a single command. In earlier versions of NetVisor OS, this required two separate vFlow commands: one for configuring source and destination vPGs and another for configuring the action.

For example, you can configure an action of `setvlan` to assign a VLAN to all the packets that are copied between the source and destination vPGs:

```
CLI (network-admin@switch1) > vflow-create name TAP1-TOOL2 scope fabric
proto tcp src-vpg TAP-GROUP-1 dst-vpg TOOL-GROUP-2 action setvlan action-
value 20
```

The command above assigns VLAN 20 to the copied packets.

Understanding Port Mirroring

The port mirroring feature is used to gain visibility into the traffic flowing through the network, which assists in troubleshooting connectivity issues and monitoring the performance of network devices. Port mirroring copies traffic from specific ports on a switch to other ports on the same switch or a different switch. To analyze the traffic, you must configure a destination port for the mirrored traffic where network analyzer tools can be connected. Consider the use case of an application that needs access to data flowing through the switch, but does not want to impede the flow. The port mirroring feature can provide the configuration required by this application.

NetVisor OS supports three distinct port mirroring functionalities:

Local Switched Port Analyzer (SPAN) mode: The simplest port mirroring configuration which copies traffic from one or more source ports to one or more destination ports on the same switch.

Remote Switched Port Analyzer (RSPAN) mode: An extension of SPAN which copies packets between different switches by using a special user-specified VLAN to carry the traffic. This feature enables remote monitoring of multiple switches.

Encapsulated Remote Switched Port Analyzer (ERSPAN) mode: This configuration transfers mirrored traffic over a routed network (over IP) with GRE encapsulation.

Configuring Port Mirroring

You can create a port mirror and configure the parameters using the `mirror-create` command.

```
CLI (network-admin@switch) > mirror-create
```

`mirror-create`	Create mirrored ports.
`name` *`name-string`*	Specify a mirror name.
Specify the following options:	
`direction [ingress|egress| bidirectional]`	Specify the direction of the traffic on the source port to be mirrored. Use this option to mirror the traffic that is received on source ports or traffic that leaves source ports, or both. The default direction is ingress.
`out-port` *`port-list`*	Specify one or more outgoing traffic ports.
`out-trunk` *`trunk name`*	Specifying outgoing traffic trunk (link aggregation). `out-trunk` option load balances the outgoing traffic among trunk ports. You can either configure an `out-port` or an `out-trunk.`
`in-port` *`port-list`*	Specify one or more incoming traffic ports. The in-ports can overlap among other mirror instances.
`filtering [port|vflow-and-port| vflow-or-port]`	Specify the traffic filter policy. With `vflow-and-port` policy, only if a packet matches both the vFlow and the `in-port` for the mirror will it get mirrored. With `vflow-or-port` policy, the packet gets mirrored if it matches either the vFlow or the `in-port` of the mirror.
`enable|disable`	Enable or disable the mirror. A mirror, once created, is enabled by default.
`other-egress-out [allow|prevent]`	Specify to allow or prevent switching of other traffic to `out-port`. The default status is `prevent`.
`span-encap [none|over-ip|over-vlan]`	Specify the mirror encapsulation type. Specify `over-ip` to enable ERSPAN and `over-vlan` to enable RSPAN. The default is `none`.
`span-local-ip` *`ip-address`*	Specify the local IPv4 address.
`span-remote-ip` *`ip-address`*	Specify the remote IPv4 address.
`span-src-mac` *`mac-address`*	Specify the source MAC address for the mirror.
`span-dst-mac` *`mac-address`*	Specify the destination MAC address for the mirror.
`span-tagging-vlan` *`vlan-id`*	Specify the mirror SPAN tagging VLAN ID. This VLAN carries the traffic in RSPAN configuration.
`span-tos 0..255`	Specify the mirror SPAN Type of Service (ToS) as a value between 0 and 255.

`nvie-mirror|no-nvie-mirror`	Specify to mark/unmark this mirror as an NVIE mirror used to mirror traffic to NVIE virtual machines.

Configuring Local SPAN

For Local SPAN, the `in-port` and `out-port` are on the same switch. For example:

```
CLI (network-admin@switch) > mirror-create name mirror1 direction ingress
in-port 10 out-port 15
```

NetVisor OS defines a mirror configuration, but does not add any traffic into that mirror. A sniffer tool like Wireshark can capture and analyze the mirrored traffic at the destination port. You can modify a mirror configuration by using the `mirror-modify` command. To view the details of a mirror configuration, use the `mirror-show` command.

For example:

```
CLI (network-admin@switch) > mirror-create name mirror2 direction
bidirection out-port 10 in-port 15
```

The details of the mirror configured above can be viewed using the command:

```
CLI (network-admin@switch) > mirror-show layout vertical
name:              mirror2
direction:          bidirection
out-port:          10
in-port:           15
filtering:          port
enable:            yes
other-egress-out:   prevent
nvie-mirror:        false
```

To modify the above configuration, use the command:

```
CLI (network-admin@switch) > mirror-modify name mirror2 out-port 20
```

To view the modified configuration, use the command:

```
CLI (network-admin@switch) > mirror-show layout vertical
name:              mirror2
direction:          bidirection
out-port:          20
in-port:           15
filtering:          port
enable:            yes
other-egress-out:   prevent
nvie-mirror:        false
```

Use the following command to modify a configuration and setup mirroring to send traffic from a range of data ports to a destination SPAN port.

```
CLI (network-admin@switch) > mirror-modify mirror25 in-port 1-5 out-port 50
```

To disable the configuration, use the following command:

```
CLI (network-admin@switch) > mirror-modify mirror25 in-port 1-5 out-port 50
disable
```

By default, a port configured as `out-port` of a mirror only functions as egress port for mirrored traffic. The `out-port` does not allow transit traffic to flow through which, in certain cases, can lead to traffic black holing. To overcome this problem, the `out-port` may be configured to allow other egress traffic. For example:

```
CLI (network-admin@switch1) > mirror-create name mirror20 direction ingress
in-port 81 out-port 86 other-egress-out allow
```

Configuring Multiple Port Mirrors

NetVisor OS supports the creation of multiple mirrors. At a time, up to four unidirectional mirrors can be configured on any platform.

For example:

```
CLI (network-admin@switch) > mirror-create name rule1 in-port 1,2 out-port
50 span-encap over-vlan span-tagging-vlan 50
CLI (network-admin@switch) > mirror-create name rule2 in-port 3,4 out-port
51 span-encap over-vlan span-tagging-vlan 50
CLI (network-admin@switch) > mirror-create name rule3 in-port 5,6 out-port
52 span-encap over-vlan span-tagging-vlan 50
CLI (network-admin@switch) > mirror-create name rule4 in-port 7,8 out-port
53 span-encap over-vlan span-tagging-vlan 50
```

Note: All platforms support two bidirectional or four unidirectional mirrors. They also support either an `out-port` or an `out-trunk` per mirror.

Configuring Remote Port Mirroring

Configuring VLAN-based Remote Port Mirroring

Remote port mirroring copies traffic between different switches and the mirrored traffic is carried over a specified VLAN. This functionality is also known as RSPAN.

Take for example, the topology below:

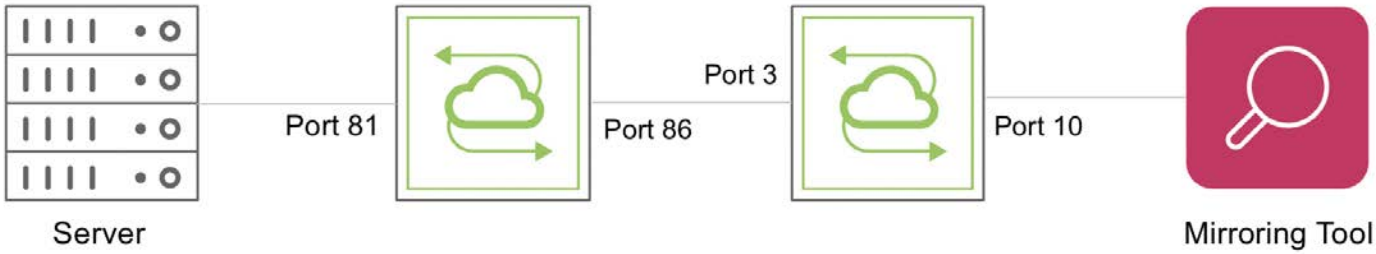


Figure 11-5: VLAN-based Remote Port Mirroring Topology

To configure remote port mirroring over a VLAN, you must first create a mirror instance on the source switch (switch1). You must also specify `over-vlan` as encapsulation scheme. For example:

```
CLI (network-admin@switch1) > mirror-create name mirror33 direction ingress
in-port 81 out-port 86 other-egress-out allow span-encap over-vlan span-
tagging-vlan 200
```

Use the `mirror-show` command to view the configuration:

```
CLI (network-admin@switch1) > mirror-show
name:               mirror33
direction:          ingress
out-port:           86
in-port:            81
filtering:          port
enable:             yes
other-egress-out:   allow
span-encap:         over-vlan
span-tagging-vlan:  200
nvie-mirror:        false
```

This command tags all ingress packets on port 81 with VLAN 200 and these packets are sent out on port 86. If VLAN 200 is not configured on switch2, the packets are dropped at port 3. Thus, VLAN 200 must be configured on port 3 and port 10 of switch2.

```
CLI (network-admin@switch2) > vlan-create id 200 scope local ports 3,10
```

The alternate method is to configure a mirror on switch2 with a VLAN 200 tagging.

For example:

```
CLI (network-admin@switch2) > mirror-create name mirror35 in-port 3 out-
```

```
port 10 span-encap over-vlan span-tagging-vlan 200
```

Use the `mirror-show` command to view the configuration:

```
CLI (network-admin@switch2) > mirror-show
name:               mirror35
direction:          ingress
out-port:           10
in-port:            3
filtering:          port
enable:             yes
other-egress-out:   prevent
span-encap:         over-vlan
span-tagging-vlan:  200
nvie-mirror:        false
```

With this configuration, the mirrored packets can be analyzed at port 10 of switch2 by using a packet analyzer tool.

> **Note**: If the path for the mirrored packets to reach the destination include multiple switches, the SPAN tagging VLAN must be configured on all the intermediate switches.

Configuring IP-based Remote Port Mirroring

Remote port mirroring can be configured to send packets to a destination reachable over Layer 3. This is achieved by encapsulating mirrored packets with an L3 header so that the packets can be routed. This functionality is also known as ERSPAN.

Consider for example, the topology below:

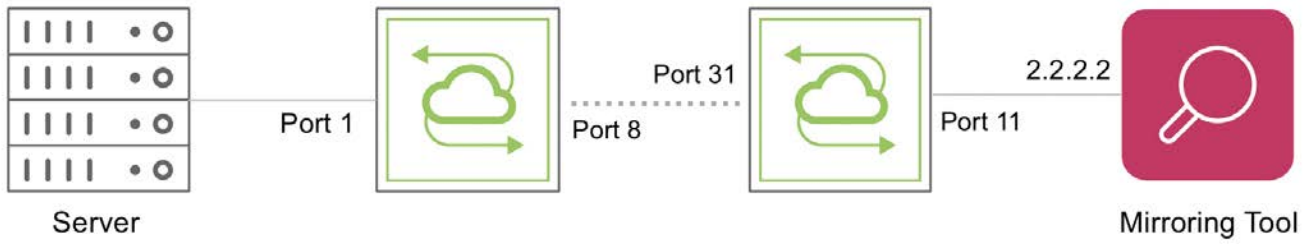


Figure 11-6 - IP-based Remote Port Mirroring Topology

To create a mirror over IP configuration, you must specify `over-ip` as encapsulation scheme.

For example:

```
CLI (network-admin@switch1) > mirror-create name mirror37 in-port 1 out-
port 8 span-encap over-ip span-local-ip 1.1.1.1 span-remote-ip 2.2.2.2
span-src-mac 33:44:55:66:77:88 span-dst-mac 99:aa:bb:cc:dd:ee other-egress-
out allow
```

To view the configuration, use the `mirror-show` command:

```
CLI (network-admin@switch1) > mirror-show layout vertical
```

```
switch:             switch1
name:               mirror37
direction:          ingress
out-port:           8
in-port:            1
filtering:          port
enable:             yes
other-egress-out:   allow
span-encap:         over-ip
span-local-ip:      1.1.1.1
span-remote-ip:     2.2.2.2
span-src-mac:        33:44:55:66:77:88
span-dst-mac:        99:aa:bb:cc:dd:ee
span-tos:           0
nvie-mirror:        false
```

Note that the IP-based configuration has local and remote IP addresses and MAC addresses explicitly specified. VLAN tagging and ToS setting are optional parameters.

- `span-local-ip` and `span-src-mac` can be specified as the router IP and MAC addresses of the local switch on which the mirror is created.
- `span-remote-ip` must be the IP address of the packet analyzer tool.
- `span-dst-mac` must be the next-hop MAC address.
- `other-egress-out  allow` must be configured as the L3 protocol fails to come up otherwise.
- `out-port` must be the egress port to reach the next-hop IP address.

Note: The local router should have a route to reach the packet analyzer's IP address.

Configuring vFlow Filters with Port Mirrors for Logging Packets

A vFlow filter, in conjunction with a port mirror, gives granular control over the traffic that is mirrored through SPAN, RSPAN, or ERSPAN configurations. By configuring a vFlow with a mirror, you can select the traffic you need for analysis with precision.

To create a vFlow-mirror, you should first configure a port mirror and you must identify the SPAN port before configuring the port mirror. For example,

To create a port mirror *mirror11*, use:

```
CLI (network-admin@switch) > mirror-create name mirror11 out-port 130 in-port 1-128 filtering vflow-and-port
```

To view the details:

```
CLI (network-admin@switch) > mirror-show

name    direction out-port in-port filtering      enable other-egress-out nvie-mirror
------- --------- -------- ------- -------------- ------ ---------------- -----------
mirror11 ingress   130      1-128   vflow-and-port yes    prevent          false
```

Note: The SPAN Port is different for each platform. For NRU02 platform, the SPAN port is *port 130*.

To create the corresponding vFlow - *to_mirror*, use the `vflow-create` command:

```
CLI (network-admin@switch) > vflow-create name to_mirror scope local mirror mirror11 src-ip 100.1.1.200 dst-ip 10.0.40.2 precedence default action none
```

The `vflow-create` command allows numerous filtering policies. Refer to the Configuring and Using vFlows chapter for more information.

To view the details, use the command:

```
CLI (network-admin@switch) > vflow-show name to_mirror

name      scope type  dst-ip    precedence action mirror from-tunnel-decap transparency enable table-name
--------- ----- ----- --------- ---------- ------ ------ ----------------- ------------ ------ --------------------
to_mirror local vflow 10.0.40.2  default    none  mirror11 none              disable      enable System-L1-L4-Tun-1-0
```

A logical combination of a port mirror and a vFlow-based one can be configured using the `filtering` parameter in the `vflow-create` command.

- Use the `port` option to consider only the parameters configured in the `mirror-create` command for filtering the traffic.
- Use the `vflow-or-port` option to mirror traffic that meets either the vFlow or the mirror constraints. With this option, packets that match either the vFlow policy or the `in-port` parameter of the mirror get mirrored.

- Use the `vflow-and-port` option to mirror traffic that meets both the vFlow and the mirror constraints. With this option, only packets that match both the vFlow policy and the `in-port` parameter of the mirror get mirrored.

> **Note:** IP-based remote port mirroring supports only sources (`in-port`) in the `ingress` direction with `vflow-and-port` filtering. This limitation applies to all platforms.

For Remote SPAN (RSPAN):

```
CLI (network-admin@switch) > mirror-create name mirror5 out-port 80 in-port
40 filtering vflow-and-port span-encap over-vlan span-tagging-vlan 300
```

> **Note**: Port 80 is associated with VLAN 300.

```
CLI (network-admin@switch) > vflow-create name flow1 scope local dst-ip
10.10.10.10 action none mirror mirror5
```

With the above configuration, only the packets that ingress on port 40 of *switch* with a destination IP address of 10.10.10.10 are mirrored.

For packet logging to local SPAN port on platforms that have rear-facing NICs:

```
CLI (network-admin@switch) > mirror-create name mirror5 out-port 80 in-port
40 filtering vflow-and-port
```

where out-port parameter is the rear-facing NIC SPAN port.

> **Note**: Use `tcpdump` command on Linux shell with rear-facing NIC SPAN port interface to create a PCAP file or to view the traffic live.

Guidelines to remember while configuring SPAN port:

- To view the SPAN port, use the command:

```
CLI (network-admin@switch*) > port-cos-rate-setting-show port span-
ports format port,ports,
port           ports
----------  -----
span-ports  130
```

here, `130` is the *SPAN port* for the switch.

- To determine the physical SPAN port interface used for *tcpdump* in Linux shell, use the command:

```
root@switch:~#  cat  /var/nvos/hw_pid.xml
<?xml  version="1.0"?>
<hw_pids>
    <hw_pid  code="NRU02-
```

```
ONVL" .. mgmt0="em0" mgmt1="em1" data0="em3" data1="em2" data2="" data3="
" ports="none" ..
</hw_pids>
```

where, "**data1**" determines the span port physical interface (which is "**em2**" in this case). This "em2" interface must be used along with *tcpdump* to capture the packets.

- You can set the aggregate egress rate limit for traffic to the local SPAN port depending on the CPU utilization and traffic profile. For example, to set the limit to 100 MB, use the command:

```
CLI (network-admin@switch) > port-config-modify port 130 egress-rate-
limit 100m

CLI (network-admin@switch) > port-config-show port
130 format intf,switch,port,speed,egress-rate-limit,

intf  switch  port  speed  egress-rate-limit
----  ------  ----  -----  -----------------
130   switch 130   10g    100000000
```

For a more granular rate setting, use the `port-cos-rate-setting-modify` command.

Understanding and Configuring SNMP

This chapter provides information for understanding and configuring SNMP services on NetVisor OS switches using the Arista Networks NetVisor OS Command Line Interface (CLI).

Overview - Understanding and Configuring SNMP

Simple Network Management Protocol (SNMP) is an application-layer protocol used to manage and monitor network devices and functions. SNMP provides a common language for network devices to relay management information between network elements and monitor the health of network devices such as routers, switches, access points, and even devices such as UPS, printers etc. SNMP is part of the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite and comes bundled with an SNMP agent. The SNMP agents are configured and enabled to communicate with the network management system (NMS).

Configuring SNMP

NetVisor OS supports SNMP v1, v2, and v3 and the implementation is based on Net-SNMP, where the SNMP daemon runs as a service. When an external browser or SNMP collector contacts the SNMP daemon on NetVisor, the SNMP daemon queries the nvOSd for details.

NetVisor OS uses SNMP only for monitoring the network: that is, you can use the SNMP service for read-only MIB support and does not provide support for managing the network. The following sections explain how to configure access controls and enable SNMP on a switch.

> **Note:** To configure SNMP v1 and v2, you must create community strings and for SNMP v3, you should create users.

Activating the SNMP Service

By default, the SNMP service is disabled on NetVisor OS. You can enable the SNMP service by using the `admin-service-modify` command. NetVisor OS retains the status of the SNMP service (enabled or disabled) when you upgrade from one release version to another version.
To enable SNMP service, use the command:

```
CLI (network-admin@switch) admin-service-modify if if-string snmp|no-snmp
```

`if` *`if-string`*	Specify the administrative service interface. The options are `mgmt` or `data`.
`snmp|no-snmp`	Specify to enable or disable SNMP.

To enable SNMP service through management interface, use the command:

```
CLI (network-admin@switch)admin-service-modify if mgmt snmp
```

To enable SNMP service on data interface, use the command:

```
CLI (network-admin@switch)admin-service-modify if data snmp
```

The above commands launch the daemon, sub-agents, and opens the firewall so that remote queries can reach the daemon. Use the no-snmp parameter for disabling SNMP service on the administrative interface:

```
CLI (network-admin@switch)admin-service-modify if mgmt no-snmp
```

For using SNMP services, you must configure access controls either by creating SNMP communities and/or by creating SNMP users.

Creating SNMP Communities on SNMP V1 and V2

SNMPv1 and v2 protocol uses communities as a method of controlling access to information. A community consists of the community string and community type. You can create a community using the following command:

```
CLI (network-admin@switch) > snmp-community-create community-string
community-string-string community-type read-only|read-write
```

`community-string community-string-string`	Specify a community name
`community-type read-only|read-write`	Specify the community type having read-only or read-write privileges.

For example, to create a SNMP community string named, **community1**, with read-only privileges, use the following command:

```
CLI (network-admin@switch) > snmp-community-create community-string
community1 community-type read-only
```

Note: Although NetVisor OS allows you to configure community strings "*public*" or "*private*", due to security reasons, we recommend not to use *public*/*private* community strings as SNMP community strings. This is because the *public*/*private* community strings are the industry default and commonly used community strings. Some vulnerability scanning tools may report the use of *public/private* community strings as a vulnerable configuration.

In NetVisor OS, the `snmp-show` command enables an SNMP walk internally on specific MIB tables or MIB elements. You can run an SNMP walk from any host where SNMP software is enabled.

To display the details of a Pluribus custom MIB using SNMP walk, pnFabricTable, use the command:

```
CLI (network-admin@switch) > snmp-show community-string community1 name
pnFabricTable show-type walk
switch name                  value
------  --------------------  -------------------
switch FbIndex.100663455     Gauge32: 100663455
switch NodeName.100663455    STRING: switch.
switch FabricName.100663455  STRING: switch.
switch NodeState.100663455   Gauge32: 1
```

To modify the SNMP community, **community1**, to read-write, use the following command:

```
CLI (network-admin@switch) > snmp-community-modify community-string
community1 community-type read-write
```

To display information about the SNMP community, **community1**, use the following command:

```
CLI (network-admin@switch) > snmp-community-show community-string
```

```
community1
```

```
switch  community-string  community-type
------  ----------------  --------------
switch community1         read-write
```

To delete the SNMP community, **community1**, use the following command:

```
CLI (network-admin@switch) > snmp-community-delete community-string
community1
```

Enabling SNMP Walk

To run an SNMP walk on the supported switches, you must ensure:

- The SNMP service is enabled on the switch by using the `admin-service-show` command:

```
CLI (network-admin@switch) > admin-service-show
switch     if   ssh nfs web web-ssl web-ssl-port web-port vrrp snmp net-api icmp
---------- ---- --- --- --- ------- ------------ -------- ---- ---- ------- ----
leo-ext-23 mgmt on  off on  off     443          80       off  on   on      on
```

- The community-string is configured correctly on the switch using the command:

```
CLI (network-admin@switch) > snmp-community-show
switch  community-string  community-type
------  ----------------  --------------
switch community1         read-write
```

For more details, see the *Creating SNMP Communities* section.

- To get traps for particular SNMP trap receiver or NMS software, configure the SNMP trap sink host using the command:

```
CLI (network-admin@switch) > snmp-trap-sink-create community <community-
string-name> type TRAP_TYPE_V2C_TRAP dest-host <destination-host>
```

For more details, see the *Enabling SNMP Traps* section.

Creating SNMP Users on SNMPv3

The SNMPv3 protocol supports the creation of users and optionally allows the usage of authentication and encryption. NetVisor OS supports SHA or MD5 as authentication protocols and DES as the encryption algorithm. The default authentication protocol is SHA, however, NetVisor allows you to change the authentication protocol to MD5 by using the CLI.

- You can also create a user without providing the authentication and privilege password options. For example:

```
CLI (network-admin@switch) > snmp-user-create user-name name-string auth
priv
```

- To create a user by providing the authentication and privilege passwords for encryption, use the following command. You must provide a password for authentication (`auth-password`) and encryption (`priv-password`):

```
CLI (network-admin@switch) > snmp-user-create user-name user-name-string
auth-password auth-password-string [auth|no-auth] [auth-hash md5|sha] priv-
password priv-password-string [priv|no-priv]
```

- To create the user, `pluribus`, with an authentication password and authentication hash as SHA1, use the following command:

```
CLI (network-admin@switch) > snmp-user-create user-name pluribus auth auth-
hash sha

auth password: ********
confirm password: ********
```

The password should have at least eight (8) characters and can be a combination of letters, numbers, and special characters.

- To modify the SNMP user and add privilege with a password, use the following command:

```
CLI (network-admin@switch) > snmp-user-modify user-name pluribus auth-
password auth priv-password priv
priv-password priv
auth password: ********
confirm password: ********
priv password: ********
confirm password: ********
```

To display information about the SNMP user created earlier, use the following command:

```
CLI (network-admin@switch) > snmp-user-show user-name pluribus
user-name auth auth-hash priv
--------- ---- --------- ----
pluribus  yes   sha       yes
```

- Create another user with user name, `pluribus2` and authentication hash as `MD5`:

```
CLI (network-admin@switch) > snmp-user-create user-name pluribus2 auth
auth-password priv priv-password auth-hash md5

auth password:********
confirm auth password:********
priv password:********
confirm priv password:********
```

To display the details, use the following command:

```
CLI (network-admin@switch) > snmp-user-show

switch  user-name   auth auth-hash priv
------- ----------- ---- --------- ----
switch  pluribus1   yes  sha       yes
switch  pluribus2   yes  md5       yes
```

- To delete the SNMP user, use the `snmp-user-delete` command:

```
CLI (network-admin@switch) > snmp-user-delete user-name
```

- After you create the SNMP user, you must grant permission to view the SNMP objects by using the View Access Control Model (VACM). To grant permission, use the command:

```
CLI (network-admin@switch) > snmp-vacm-create user-name snmp-user user-type
[rouser|rwuser] oid-restrict oid-restrict-string [auth|no-auth] [priv|no-
priv]
```

The parameter, `oid-restrict`, is an optional argument and specifies a MIB sub-tree with a restricted view. In other words, if you specify an OID, you can only see that OID and the descendants in the tree.

- Using the `snmp-vacm-create` command can restrict a particular user, `snmp-user` from accessing a specified OID. For example, to restrict access to sysContact OID, use the command:

```
CLI (network-admin@switch) > snmp-vacm-create user-name snmp-user user-type
rouser oid-restrict sysContact no-auth no-priv
```

Note: Ensure to create the SNMP users before enabling the SNMP traps. In cases where the SNMP traps were already enabled before user creation, then disable the SNMP traps, create users, and re-enable the traps.

- To modify the VACM configuration of the user and to change from no authentication to authentication, use the following command:

```
CLI (network-admin@switch) > snmp-vacm-modify user-name snmp-user user-type
rouser auth
```

To display information about the VACM configuration, use the `snmp-vacm-show` command:

```
CLI (network-admin@switch) > snmp-vacm-show

user-type user-name oid-restrict view auth priv
```

```
--------- --------- ------------ ---- ---- ----
rouser    snmp-user sysContact         no   no
```

- To delete the VACM of the user from the SNMP configuration, use the `snmp-vacm-delete` command:

```
CLI (network-admin@switch) > snmp-vacm-delete user-name snmp-user
```

Modifying the SNMP Engine ID

NetVisor OS allows you to modify the SNMP Engine ID and retrieve previous SNMP agent information for a switch that is no longer in use. The SNMP Engine ID is used only by SNMPv3 entities for uniquely identifying the V3 entries.

If you remove a switch due to an RMA or other reasons from the network, you can modify the SNMP Engine ID to use the old SNMP Engine ID. This enables NetVisor OS to query and maintain the same history records for the new switch. The SNMP engine ID is a unique string of characters specific to a switch and identifies the device for administrative purposes.

After modifying the SNMP engine ID, you must recreate the SNMP user. To modify the SNMP engine ID, use the command:

```
CLI (network-admin@switch) > snmp-engineid-modify engineid <string>
```

`engineid string`	Specify the 28 character unique ID for the SNMP engine.

For example, to modify the SNMP engine ID, use the steps below:

- View the current SNMP engine ID by using the command:

```
CLI (network-admin@switch) > running-config-show | grep snmp

snmp-engineid-modify  engineid  0x80001f8880077f7820da49395a00000000
```

You can also view the SNMP engine ID using the command:

```
CLI (network-admin@switch*) > snmp-engineid-show
engineid:  0x80001f8880077f7820da49395a00000000
```

The asterisk (*) next to the switch name in the above command indicates a specific local switch.

- To modify the above SNMP engine ID to 0x80001f8880077f7820da49395a00000001, use the command:

```
CLI (network-admin@switch) > snmp-engineid-modify engineid
0x80001f8880077f7820da49395a00000001
Warning: All SNMP users will be erased.

Please confirm y/n (Default: n):y
Modified snmp engineID, Deleted all SNMP users.Please re-create SNMP
users.
```

- View the modified SNMP engine ID using the command:

```
CLI (network-admin@switch) > running-config-show | grep snmp
snmp-engineid-modify  engineid  0x80001f8880077f7820da49395a00000001
```

To display the SNMP engine IDs configured on the switches in a fabric, use the command:

```
CLI  (network-admin@aries-vxlan-01)  >  snmp-engineid-show
switch:    aries-vxlan-01
engineid:   0x80001f8880f3ac9a7e4b3fb15c00000000
switch:    ursa-vxlan-01
engineid:   0x80001f888030aae147f240b15c00000000
switch:    ursa-vxlan-02
engineid:   0x80001f8880cae2b0662f40b15c00000000
```

Supported SNMP MIBs

NetVisor OS supports several MIBs which are run in a sub-agent to the SNMP daemon (master). NetVisor OS uses the AgentX protocol for communicating between the SNMP daemon and the sub-agent. When you issue an `snmp-show` command, the SNMP daemon receives the SNMP walk request and transfers the call to a sub-agent based on the SNMP OID of the request. The sub-agent populates the SNMP container to NetVisor OS and displays the data. NetVisor OS supports the following MIBs:

- IfTable (IF-MIB)
- IfXTable (IF-MIB)
- EntPhySensorTable (ENTITY-SENSOR-MIB)
- BGP MIB. The following BGPv4 tables are supported:
  - bgpPeerTable
  - bgp4PathAttrTable
- OSPF MIB. The following OSPF tables are supported:
  - ospfGeneralGroupTable
  - ospfAreaTable
  - ospfLsdbTable
  - ospfIfTable
  - ospfExtLsdbTable
  - ospfNbrTable
  - ospfIfMetricTable
- OSPFv3 MIB Tables. The following OSPFv3 tables are supported:
  - ospfrv3GenealGroupTable
  - ospfv3AreaTable
  - ospfv3NbrTable
  - ospfv3IfTable
  - ospfv3AreaLsdbTable
  - ospfv3AsLsdbTable
  - ospfv3LinkLsdbTable
- IEEE8021-Q-BRIDGE-MIB (ieee8021QBridgePortVlanStatisticsTable)
- IEEE8021-Q-BRIDGE-MIB (ieee8021QBridgeVlanStaticTable)
- IEEE8021-SPANNING-TREE-MIB (ieee8021SpanningTreePortTable)
- IEEE8021-SPANNING-TREE-MIB (ieee8021SpanningTreeTable)
- IP-MIB (ipv4InterfaceTable, ipv6InterfaceTable)
- DISMAN-EVENT-MIB
- UCD-SNMP-MIB
- Pluribus Enterprise PN-VRRP-MIB (supports IPv6 VRRP state change)
- Pluribus Enterprise PN-HR-MIB (pnHrCPUTable, pnHRMemTable)
- Pluribus Enterprise PN-FABRIC-MIB (pnFabricTable)
- Pluribus Enterprise PN-COS-MIB (pnCosStatsTable, pnCosBezelStatsTable)
- Pluribus Enterprise PN-LOG-MIB (pnLogMatchName, pnLogFileName, pnLogMatchData, pnLogMatchCount)

Note: Pluribus Enterprise PN-LOG-MIB is used for the generation of message-based traps only.

NetVisor OS version 6.0.0 adds a new OID (pnVrrpPreemptMode.0) to the Pluribus proprietary PN-VRRP-MIB, which is mapped to the standard VRRP-MIB v3. The value of this OID is 'true(1)' when VRRP preempt

mode is enabled (default status) and 'false(2)' when VRRP preempt mode is disabled.

The above MIBs consolidate into a single sub-agent, called **nvOS_snmpd** agent, which communicates with the master SNMP daemon and serves requests to the MIBs. All the MIBs except the routing MIBs are part of the **nvOS_snmpd** agent.

Routing MIBs

For the OSPF and BGP routing MIBs, NetVisor OS hosts the routing protocols inside a vRouter. Use the commands `vrouter-create` and `vrouter-modify` to enable the routing MIBs. You can also enable the SNMP notifications for OSPF and BGP protocols while enabling the routing MIBs. Enabling Routing MIBs is supported only on individual *vRouters* in a *fabric*. You can enable routing MIBs only on one node at a time.

To enable or disable the BGP and OSPF routing MIBs on a vRouter, use the command:

```
CLI (network-admin@switch) > vrouter-create name name-string
```

Specify one or more options:	
`bgp-snmp|no-bgp-snm`	Enable or disable BGP SNMP MIB.
`bgp-snmp-notification|no-bgp-snmp-notification`	Enable or disable BGP SNMP notifications.
`ospf-snmp|no-ospf-snmp`	Enable or disable OSPF SNMP MIB.
`ospf-snmp-notification|no-ospf-snmp-notification`	Enable or disable OSPF SNMP notifications.
`ospf6-snmp|no-ospf6-snmp`	Enable or disable OSPFv3 SNMP MIB.
`ospf6-snmp-notification|no-ospf6-snmp-notification`	Enable or disable OSPFv3 SNMP notifications
`ip-snmp|no-ip-snmp`	Enable or disable ipForward SNMP MIB.

For example, to create a vrouter, **vr1** and to enable the routing MIBs and notifications on **vr1**, use the command:

```
CLI (network-admin@switch) > vrouter-create vr1 bgp-snmp bgp-snmp-
notification ospf-snmp ospf-snmp-notification
```

Use the `vrouter-modify` command to modify or disable the SNMP notifications:

```
CLI (network-admin@switch) > vrouter-modify vr1 bgp-snmp no-bgp-snmp-
notification ospf-snmp no-ospf-snmp-notification
```

Configuring SNMP Traps

To enable SNMP traps, you must:

1. Create an SNMP trap sink (set up a destination host). Use separate commands for SNMPv1 or SNMPv2c and SNMPv3 to setup the destination host.
2. Enable the SNMP traps to be sent.

Configuring SNMP Trap Sink

To set up the destination host for SNMPv1 or SNMPv2c, use the command:

```
CLI (network-admin@switch) > snmp-trap-sink-create community community-string [type TRAP_TYPE_V1_TRAP|TRAP_TYPE_V2C_TRAP|TRAP_TYPE_V2_INFORM] dest-host dest-host-string [dest-port dest-port-number]
```

`community` *`community-string`*	Specify the community type.
`[type TRAP_TYPE_V1_TRAP| TRAP_TYPE_V2C_TRAP| TRAP_TYPE_V2_INFORM]`	Specify the trap type. The default trap is: TRAP_TYPE_V2C_TRAP.
`dest-host` *`dest-host-string`*	Specify the destination host. This can be an IP address or a hostname.
`[dest-port` *`dest-port-number`*`]`	Specify the destination port. The default port number is 162.

For example, to create a community *test1*, with destination host as *lyra10.pluribusnetworks.com* and trap type as *TRAP_TYPE_V1_TRAP*, use the command:

```
CLI (network-admin@switch) > snmp-trap-sink-create community test1 dest-host lyra10.pluribusnetworks.com type TRAP_TYPE_V1_TRAP
```

This command sends the SNMPv1 traps to a receiver on the host *lyra10* with the community string *test1*.

To setup the destination host for SNMPv3 trap receiver, use the command:

```
CLI (network-admin@switch) > snmp-v3-trap-sink-create user-name user-name-string [type TRAP_TYPE_V3_TRAP|TRAP_TYPE_V3_INFORM] dest-host dest-host-string [dest-port dest-port-number]
```

`user-name` *`user-name-string`*	Specify the user name for the SNMPv3 trap.
`[type TRAP_TYPE_V3_TRAP| TRAP_TYPE_V3_INFORM]`	Specify the SNMPv3 trap type.
`dest-host` *`dest-host-string`*	Specify the destination host.
`[dest-port` *`dest-port-number`*`]`	Specify the destination port number. The default port number is 162

For example, to configure SNMPv3 traps for user *test2*, with destination host as *lyra10.pluribusnetworks.com* and with authentication enabled, use the command:

```
CLI (network-admin@switch) > snmp-v3-trap-sink-create user-name test2 dest-host lyra10.pluribusnetworks.com auth
```

This command configures the traps to be sent to *lyra10* with the username *test2*, engine id *0xABCDEF* and with authentication enabled. You will be prompted for an authentication password with this command.

Enabling SNMP Traps

After configuring the destination, you must enable the type of traps to be sent using the command:

```
CLI (network-admin@switch) > snmp-trap-enable-modify
```

Specify one or more of the following options:	
`link-up-down|no-link-up-down`	Enable or disable traps for the port link state changes.
`interface-up-down|no-interface-up-down`	Enable or disable traps for vRouter interface status.
`default-monitors|no-default-monitors`	Enable or disable traps for default monitoring.
`physical-sensors|no-physical-sensors`	Enable or disable physical sensor traps for temperature, fan speed, etc.
`low-disk-space|no-low-disk-space`	Enable or disable traps for low disk space.
`low-disk-space-threshold` *`low-disk-space-threshold-string`*	Specify the Threshold value of low-disk-space in percentage.
`system-usage|no-system-usage`	Enable or disable traps for memory and CPU usage.
`high-system-usage-threshold` *`high-system-usage-threshold-string`*	Specify the Threshold value of system-usage in percentage.
`login-failure|no-login-failure`	Enable or disable traps for incorrect passwords during login.
`cluster-tr-diverge|no-cluster-tr-diverge`	Enable or disable traps when cluster transaction lists diverge.
`lacp-status|no-lacp-status`	Enable or disable traps for LACP status.
`vport-modified|no-vport-modified`	Enable or disable vPort modification traps.
`stp-port-modified|no-stp-port-modified`	Enable or disable STP port modification traps.
`mirror-to-cpu|no-mirror-to-cpu`	Enable or disable traps when mirror to CPU is configured.
`stp-port-state-failed|no-stp-port-`	Enable or disable STP port state failed traps.

`state-failed`	
`link-congestion-detected|no-link-congestion-detected`	Enable or disable traps for link congestion detection.
`fabric-node-state-changed|no-fabric-node-state-changed`	Enable or disable traps for fabric node state changes.
`stp-new-root|no-stp-new-root`	Enable or disable traps when new STP root is detected.
`stp-topology-changed|no-stp-topology-changed`	Enable or disable traps for STP topology changes.
`vrrp-new-master|no-vrrp-new-master`	Enable or disable traps for VRRP master changes.
`disable-start-stop|no-disable-start-stop`	Enable or disable traps for disable start/stop.
`cert-expiry|no-cert-expiry`	Enable or disable traps for the expiry of the NetVisor OS certificates (shipped with the switches) .
`sysup-alert|no-sysup-alert`	Enable or disable traps for system up alert.
`port-oir-error-state|no-port-oir-error-state`	Enable or disable traps for port OIR error.
`port-bw-threshold-exceed-event|no-port-bw-threshold-exceed-event`	Enable or disable traps for port bandwidth usage exceeding the threshold or falling below the threshold.

The above command enables the notifications to be sent out when a link changes the state to up or down.

Further, You can enable SNMP trap notifications for the following events:

- Physical interface is up/down
- Data/vRouter interface is up/down
- CPU usage is higher than the configured threshold
- Memory utilization is higher than the configured threshold
- Disk utilization is higher than the configured threshold
- STP port is modified
- vPort is modified
- LAG/vLAG status changes
- Mirror-to-CPU vFlow is configured
- Link congestion is detected
- Fabric node state changes
- New STP root is elected
- VRRP master changes
- Disable start/stop
- NetVisor OS certificate expires
- System up alert
- Port OIR error

- Port bandwidth usage exceeds a threshold

Using Additional SNMP Commands

You can use the following additional SNMP commands to check the details of the SNMP configuration and notifications on the Arista switches:

- `snmp-trap-enable-show` — Display enabled SNMP traps.

```
CLI (network-admin@switch1) > snmp-trap-enable-show
link-up-down:                   no
interface-up-down:              no
default-monitors:               no
physical-sensors:               no
low-disk-space:                 no
low-disk-space-threshold:       no
system-usage:                   no
high-system-usage-threshold:    no
login-failure:                  no
cluster-tr-diverge:             no
lacp-status:                    no
vport-modified:                 no
stp-port-modified:              no
mirror-to-cpu:                  no
stp-port-state-failed:          no
link-congestion-detected:       no
fabric-node-state-changed:      no
stp-new-root:                   no
stp-topology-changed:           no
vrrp-new-master:                no
disable-start-stop:             no
cert-expiry:                    no
sysup-alert:                    no
port-oir-error-state:           no
port-bw-threshold-exceed-event: no
```

- `snmp-trap-sink-show` — Display SNMP trap sinks.

```
CLI (network-admin@switch) > snmp-trap-sink-show community community-string
[type TRAP_TYPE_V1_TRAP|TRAP_TYPE_V2C_TRAP|TRAP_TYPE_V2_INFORM] dest-host
dest-host-string [dest-port dest-port-number]
```

`community` *`community-string`*	Specify the community type.
`[type TRAP_TYPE_V1_TRAP| TRAP_TYPE_V2C_TRAP| TRAP_TYPE_V2_INFORM]`	Specify the trap type. The default trap is: TRAP_TYPE_V2C_TRAP.
`dest-host` *`dest-host-string`*	Specify the destination host. This can be an IP address or a hostname.
`[dest-port` *`dest-port-number`*`]`	Specify the destination port. The default port number is 162.

- `snmp-trap-sink-delete` — Delete SNMP trap sinks.

```
CLI (network-admin@switch) > snmp-trap-sink-delete community community-string dest-host dest-host-string
```

The above command deletes the SNMP trap sink for the specified community. For details about the command parameters, see the CLI table for the `snmp-trap-sink-show` command.

- `snmp-v3-trap-sink-create` — Used to specify an SNMPv3 trap receiver.

```
CLI (network-admin@switch) > snmp-v3-trap-sink-create user-name user-name-string [type TRAP_TYPE_V3_TRAP|TRAP_TYPE_V3_INFORM] dest-host dest-host-string [dest-port dest-port-number]
```

`user-name` *`user-name-string`*	Specify the user name for the SNMPv3 trap.
`[type TRAP_TYPE_V3_TRAP| TRAP_TYPE_V3_INFORM]`	Specify the SNMPv3 trap type.
`dest-host` *`dest-host-string`*	Specify the destination host. This can be an IP address or a hostname.
`[dest-port` *`dest-port-number`*`]`	Specify the destination port number. The default port number is 162.

The above command enables you to create an SNMPv3 trap receiver.

- `snmp-v3-trap-sink-delete` — Used to delete an SNMPv3 trap receiver.

```
CLI (network-admin@switch) > snmp-v3-trap-sink-delete user-name user-name-string dest-host dest-host-string [dest-port dest-port-number]
```

The above command deletes the SNMPv3 trap sink for the specified user. For details about the command parameters, see the CLI table for the `snmp-v3-trap-sink-create` command.

- `snmp-v3-trap-sink-show` — Used to display an SNMPv3 trap receiver information.

```
CLI (network-admin@switch) > snmp-v3-trap-sink-show user-name user-name-string dest-host dest-host-string [dest-port dest-port-number]
```

The above command displays the SNMPv3 trap receivers for the specified options (see table below). Specify one or more of the formatting options to display the output in the desired format.

Specify one or more options:	
`user-name` *`user-name-string`*	Displays the specified user name.
`engine-id` *`engine-id-string`*	Displays the engine ID.
`[type TRAP_TYPE_V3_TRAP| TRAP_TYPE_V3_INFORM]`	Displays the trap type.

`dest-host` *`dest-host-string`*	Displays the destination host.
`dest-port` *`dest-port-number`*	Displays the destination port.
`auth|no-auth`	Displays the authentication status.
`priv|no-priv`	Displays the privacy status.

- `port-stats-snmp-show` — Used to display the accumulated port statistics for SNMP.

```
CLI (network-admin@switch) > port-stats-snmp-show port port-list port-
clear-count port-clear-count-number port-clear-time #d#h#m#s
port_clear_time high resolution time: #ns all-ports
```

Specify any of the following options:	
`port` *`port-list`*	Displays one or more switch network data port numbers. The port number must be in the range of 1 to 64. Multiple ports can be specified as a comma-separated list of numbers or a range (-).
`port-clear-count` *`port-clear-count-number`*	Displays the number of times `port-stats-clear` or `trunk-stats-clear` commands were issued on a port or trunk.
`port_clear_time` *`high resolution time: #ns`*	Displays the high resolution time (ns) since the last `port-stats-clear` command.
`port-clear-time` *`duration: #d#h#m#s`*	Displays the time since last `port-stats-clear`. It is initialized with time atnvOS_init by default.
`all-ports`	A flag to specify if you want to view the stats of all ports on the switch. The default view contains only enabled ports.
`counter` *`counter-number`*	Displays the counter number.
`ibytes ibytes-number`	Displays the incoming number of bytes.
`ibits` *`ibits-number`*	Displays the number of bits.
`iUpkts` *`iUpkts-number`*	Displays the number of incoming unicast packets.
`iBpkts` *`iBpkts-number`*	Displays the number of incoming broadcast packets.
`iMpkts` *`iMpkts-number`*	Displays the number of incoming multicast packets.
`iPauseFs` *`iPauseFs-number`*	Displays the number of incoming pause frames.
`iCongDrops` *`iCongDrops-number`*	Displays the number of incoming packets dropped due to congestion.
`idiscards` *`idiscards-number`*	Displays the number of incoming packets discarded.

ierrs *ierrs-number*	Displays the number of incoming errors.
obytes *obytes-number*	Displays the number of outgoing bytes.
oUpkts *oUpkts-number*	Displays the number of outgoing unicast packets.
oBpkts *oBpkts-number*	Displays the number of outgoing broadcast packets.
oMpkts *oMpkts-number*	Displays the number of outgoing multicast packets.
oPauseFs *oPauseFs-number*	Displays the number of outgoing pause frames.
oCongDrops *oCongDrops*-number	DisplaysDisplays the number of outgoing packets dropped due to congestion.
odiscards *odiscards-number*	Displays the number of outgoing discarded packets.
oerrs *oerrs-number*	Displays the number of outgoing errors.
mtu-errs *mtu-errs-number*	Displays the number of MTU errors.
HER-packets *HER-pts-number*	Displays the number of Head End Replicated (HER) packets.
HER-bytes *HER-bytes-number*	Displays the number of Head End Replicated (HER) bytes.

- snmp-system-modify — Modify the system information for the SNMP configuration.

```
CLI (network-admin@switch) > snmp-system-modify syscontact syscontact-string syslocation syslocation-string
```

Specify one or more options:	
syscontact *syscontact-string*	Specify or change the SNMP system contact.
syslocation *syslocation-string*	Specify or change the SNMP system location.

The above command enables you to modify the system details such as the system contact or location for the SNMP configuration.

- snmp-system-show — Displays the system details for the SNMP configurations.

```
CLI (network-admin@switch) > snmp-system-show
```

The above command displays the changed or modified system information for the SNMP configuration.

Supported SNMP Notifications (Traps)

NetVisor OS supports two types of SNMP notifications (a.k.a. traps): event-based and message-based traps.

Event-based traps are the traps generated by the SNMP agent based on specific events when an OID value changes. For example, when there is a change in *link-up-down* or *low-disk-space*. Message-based traps are triggered based on messages logged in the local logging mechanism. For instance, a *login-failure trap* is triggered when a login failure message is saved in the */var/log/auth.log* log file. Message-based traps are of the type `pnLogMatchNotification`. For example, a `link-congestion` relieved trap message is as below:

```
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (21245) 0:03:32.45
SNMPv2-MIB::snmpTrapOID.0 = OID: PN-LOG-MIB::pnLogMatchNotification
PN-LOG-MIB::pnLogMatchName.4 = STRING: linkCongestionRelieved
PN-LOG-MIB::pnLogFileName.4 = STRING: /nvOS/log/system.log
PN-LOG-MIB::pnLogMatchCount.4 = Gauge32: 15
PN-LOG-MIB::pnLogMatchData.4 = STRING: 2020-02-24,19:28:46.985665-08:00
nru03-proto-1 nvOSd(24190) system congestion_relieved_on_port(11402) :
level=critical : port=126 : Congestion relieved on port=126
```

Table 1 explains the supported SNMP notifications.

Table 1: Details of Supported SNMP Notifications

Trap Name	Description	Trap Type	Trigger
link-up-down	Port link is up or down	Event-based	If enabled, SNMP generates a trap when a port is up or down.
default-monitors	Use default SNMP monitoring	Event-based	If enabled, SNMP generates a trap for various error conditions.
physical-sensors	Physical sensors are enabled	Event-based	If enabled, SNMP generates a trap for physical sensors such as power supplies and fans.
low-disk-space	Monitors for low-disk-space	Event-based	If enabled, SNMP generates a trap if disk space is lower than threshold. The threshold can be set using the `low-disk-space-threshold` parameter in `snmp-trap-enable-modify` command. SNMP checks the output of the command, `storage-pool-show`.
system-usage	Monitors memory & CPU usage	Event-based	If enabled, SNMP generates a trap if memory and CPU usage [Total CPU = sys + user] is greater than the threshold. The threshold can be set using the `system-usage-threshold` parameter in `snmp-trap-`

			`enable-modify` command. SNMP checks the output of the command, `system-stats-show`.
login-failure	Monitors login failures	Message-based	If enabled, SNMP generates a trap when user login with wrong password.
lacp-status	Monitors LACP enable or disable	Message-based	If enabled, SNMP generates a trap when the LACP state changes from enable to disabled or vice versa.
vport-modified	Monitors vPort modifications	Message-based	If enabled, SNMP generates a trap when vPort modifications occur on the switch.
stp-port-modified	Monitors STP port status	Message-based	If enabled, SNMP generates a trap when STP port state is modified using the command, `switch-local stp-port-modify port 1 <block|edge|bpdu|root-guard>`
stp-port-state-failed	Monitors STP port state failures	Message-based	If enabled, SNMP generates a trap when STP port state is modified using the command, `switch-local stp-port-modify port 128 edge bpdu-guard`.
mirror-to-cpu	Monitors mirror-to-cpu configuration	Message-based	If enabled, SNMP generates a trap when created a vflow using the command, `vflow-create name mirror scope local action copy-to-cpu` and also generates a trap for perror.log.
link-congestion-detected	Monitors congestion drop at port	Message-based	If enabled, SNMP generates a trap indicating a link is congested.
fabric-node-state-changed	Monitors fabric node states	Message-based	If enabled, SNMP generates a trap when the a fabric node changes state.
ospfIfStateChange	Monitors OSPF interface states	Event-based	If enabled, this notification is triggered when interface state changes from DR to Down or vice versa. Originator for this trap is by changing router-id.
ospfNbrStateChange	Monitors OSPF NBR states	Event-based	If enabled, this notification is triggered when neighbor state changes from DR to Down or vice versa. Originator for this trap is designated router on broadcast networks.

bgpEstablished	Monitors BGP NBR state	Event-based	If enabled, this notification is triggered when the BGP FSM enters the ESTABLISHED State. Originator for this trap is to bring up BGP session between two BGP Peers. The objects present in the trap includes: bgpPeerRemoteAddr, bgpPeerLastError and bgpPeerState.
bgpBackwardTransition	Monitors BGP NBR state transition	Event-based	If enabled, this notification is triggered when the BGP FSM moves from higher number to lower numbered state. Originator for this trap when BGP state changes from active to idle (higher state to lower state). The objects present in the trap includes: bgpPeerRemoteAddr, bgpPeerLastError and bgpPeerState.
stp-new-root	Monitors new STP root	Event-based	If enabled, SNMP generates a trap to monitor a new root for STP.
stp-topology-changed	Monitors STP topology change	Event-based	If enabled, SNMP generates a trap to monitor topology changes for STP.
interface-up-down	Monitors vRouter interfaces	Event-based	If enabled, SNMP generates a trap for an interface with the state up or down.
disable-start-stop	Monitors disable traps for start and stop	Event-based	If Enabled, This notification gets triggered to disable cold-start notifications. By default cold-start notifications are enabled.
fabric-node-state-changed	Monitors fabric node states	Event-based	If enabled, SNMP generates a trap to monitor fabric node state changes.
vrrp-new-master	Monitors VRRP master changes	Event-based	If enabled, SNMP generates a trap to monitor VRRP master state changes.
ospfv3IfStateChange	Monitors OSPF Interface state changes	Event-based	If enabled, this notification is triggered when interface state changes from DR to Down or vice versa.
ospfv3NbrStateChange	Monitors OSPF neighbor state changes	Event-based	If enabled, this notification is triggered when neighbor state changes from DR to Down or vice versa. Originator for this trap is designated router on broadcast networks.

cluster-tr-diverge	Monitors Cluster Transaction list for divergence	Message-based	If enabled, this notification is triggered when Transaction Diverge message is generated in perror.log.
cert-expiry	Monitors expiry of Switch Certificate	Message-based	If enabled, this notification gets triggered when switch-certificate expires in xx number of days in /nvOS/log/event.log. You can control value of the number of days by using the `cert-expiration-alert-modify <days-before-expiration>` command.
sysup-alert	Monitor whether all the admin up ports are made up. (L3, vLAG, orphan, cluster)	Message-based	If enabled, this notification gets triggered when you reboot the switch or restart nvOS.
port-bw-threshold-exceed-event	Monitors whether port bandwidth usage on a port exceeds or falls below a threshold.	Message-based	If enabled, this notification gets triggered when bandwidth usage on a port exceeds a configured threshold or when the usage falls below the threshold.

Expected Behavior of Link Up/Down Traps

The timing and generation of `link-up-down` traps for data ports are dependent on the sequence in which the ports are brought up and down.

The port bringup sequence when a switch boots up is:
1) Cluster ports, vxlan-loopback-trunk ports, and loopback ports
2) L3 ports
3) vLAG ports
4) Rest of the ports

The port bringdown sequence when a switch goes offline is:

1. Orphan ports (i.e., non-vLAG, non-L3, non-cluster, or any form of loopback ports)
2. vLAG ports
3. L3 ports
4. Rest of the ports (e.g. cluster ports and internal ports)

As can be seen from the bringup and bringdown sequences, cluster ports are the first ports to be brought up when a switch boots up and are among the last ports to go down when a switch goes offline. As the last ports are being brought down, link scan is disabled, implying that NetVisor OS provides link-up notifications for cluster ports but there are no link-down notifications. Therefore, a mismatch can always be expected in the number of link-up and link-down SNMP traps, with the number of link-up traps being higher than the number of link-down traps.

Currently, `defer-bringup` option in `port-config-modify` command delays the time at which a port is brought up but does not affect the number of SNMP traps in any capacity. `defer-bringup` is enabled for all orphan ports by default. This prevents traffic loss by ensuring that other ports (e.g. cluster, vLAG, and L3 ports) are up and the network is ready before the orphan ports come up.

> **Note:** Management ports, loopback ports, and internal ports do not have SNMP traps for link-up or link-down events.

Additional MIBs Supported in NetVisor OS

MIB	Description
Entity-Sensor	This module defines Entity MIB extensions for physical sensors. The Entity Sensor MIB contains a single group called, **entitySensorValueGroup**, which allows objects to convey the current value and status of a physical sensor. Click here for the RFC details. **entPhySensorOperStatus** — Object identifies the current operational status of the sensor (as it is known to the agent). For example, entPhySensorUnitsDisplay.1 STRING: Temp Outlet. >60 entPhySensorUnitsDisplay.2 STRING: Temp Inlet. ... entPhySensorUnitsDisplay.22 STRING: PS1 Status.
Host-Resources (For CPU Utilization)	Use this MIB in managing host systems. Click here for the RFC details. • To monitor the CPU utilization of the switch, use the custom MIB: **PN-HR-MIB: PnHrCpuTable** • To monitor the fabric node status change, use the custom MIB: **PN-FABRIC-MIB: PnFabricTable**
IF-MIB, Entity-Sensor-MIB	The MIB module to describe generic objects for network interface sub-layers. This MIB is an updated version of the ifTable for MIB-II, and incorporates the extensions defined in RFC 1229.
Pluribus Enterprise PN-VRRP-MIB	This MIB addresses the information collected with the VRRP master change trap and supports IPv6 VRRP state change.

Sample CLI outputs

To display the SNMP details for the OID Ieee8021SpanningTreeEntry, with the community-string `community1`, use the command:

```
CLI (network-admin@cetus-colo-01) > switch-local snmp-show community-string community1 name
Ieee8021SpanningTreeEntry show-type walk show-interval 1
switch        name                                                value
------------    ------------------------------------------------    ---------------------------------------
cetus-colo-01 ieee8021SpanningTreeProtocolSpecification.1          INTEGER: 0
cetus-colo-01 ieee8021SpanningTreeProtocolSpecification.144        INTEGER: 0
cetus-colo-01 ieee8021SpanningTreeProtocolSpecification.4093       INTEGER: 0
cetus-colo-01 ieee8021SpanningTreePriority.1                       INTEGER: 8193
cetus-colo-01 ieee8021SpanningTreePriority.144                     INTEGER: 8336
cetus-colo-01 ieee8021SpanningTreePriority.4093                    INTEGER: 12285
cetus-colo-01 ieee8021SpanningTreeTimeSinceTopologyChange.1        Timeticks: (0) 0:00:00.00 centi-seconds
cetus-colo-01 ieee8021SpanningTreeTimeSinceTopologyChange.144      Timeticks: (0) 0:00:00.00 centi-seconds
cetus-colo-01 ieee8021SpanningTreeTimeSinceTopologyChange.4093     Timeticks: (0) 0:00:00.00 centi-seconds
cetus-colo-01 ieee8021SpanningTreeTopChanges.1                     Counter64: 0 topology changes
cetus-colo-01 ieee8021SpanningTreeTopChanges.144                   Counter64: 0 topology changes
cetus-colo-01 ieee8021SpanningTreeTopChanges.4093                  Counter64: 0 topology changes
```

```
cetus-colo-01 ieee8021SpanningTreeDesignatedRoot.1                Hex-STRING: 66 0E 94 D2 AF 08
cetus-colo-01 ieee8021SpanningTreeDesignatedRoot.144              Hex-STRING: 66 0E 94 D2 AF 08
cetus-colo-01 ieee8021SpanningTreeDesignatedRoot.4093             Hex-STRING: 66 0E 94 D2 AF 08
cetus-colo-01 ieee8021SpanningTreeRootCost.1                      INTEGER: 0
cetus-colo-01 ieee8021SpanningTreeRootCost.144                    INTEGER: 0
cetus-colo-01 ieee8021SpanningTreeRootCost.4093                   INTEGER: 0
cetus-colo-01 ieee8021SpanningTreeRootPort.1                      Gauge32: 0
cetus-colo-01 ieee8021SpanningTreeRootPort.144                    Gauge32: 0
cetus-colo-01 ieee8021SpanningTreeRootPort.4093                   Gauge32: 0
cetus-colo-01 ieee8021SpanningTreeMaxAge.1                        INTEGER: 20 centi-seconds
```

To display the SNMP details for the OID, pnHRCpuTable, with the community-string `community1`, use the command:

```
CLI (network-admin@aquila-m) > snmp-show community-string community1 name
pnHrCpuTable show-type walk
switch   name             value
--------  ---------------  ------------
aquila-m pnHrCpuIdx.0     INTEGER: 0
aquila-m pnHrCpuUsr.0     INTEGER: 0
aquila-m pnHrCpuSys.0     INTEGER: 0
aquila-m pnHrCpuIdle.0    INTEGER: 99
aquila-m pnHrCpuTotal.0   INTEGER: 0
```

To display the system details including the `cpu-total` output, use the command:

```
CLI (network-admin@Leaf1) > system-stats-show format all layout vertical
switch:         Leaf1
uptime:         1d1h42m29s
used-mem:       62%
used-mem-val:   9.19G
used-swap:      0%
used-swap-val:  0
paging:         0
cpu-user:       8%
cpu-sys:        12%
cpu-total:      21%
cpu-idle:       78%
```

To display the accumulated port statistics for SNMP, use the `port-stats-snmp-show` command:

```
CLI (network-admin@switch1) > port-stats-snmp-show
switch:      switch1
port:        12
ibits:       0
iUpkts:      0
iBpkts:      0
iMpkts:      0
iCongDrops:  0
ierrs:       0
obits:       174M
oUpkts:      1.32K
oBpkts:      801
oMpkts:      73.6K
oCongDrops:  0
```

```
oerrs:       0
mtu-errs:    0
switch:      switch1
port:        21
ibits:       374M
iUpkts:      37.3K
iBpkts:      579
iMpkts:      199K
iCongDrops:  0
ierrs:       0
obits:       360M
oUpkts:      37.3K
oBpkts:      0
oMpkts:      174K
oCongDrops:  0
oerrs:       0
mtu-errs:    0
....
```

To display the modified system information, for example, if you modified the system contact to `bob@xyz.com` with location as `brisbane`, use the `snmp-system-show` command:

```
CLI (network-admin@switch-04) > snmp-system-show
switch:         switch-04
syscontact:     bob@xyz.com
syslocation:    brisbane
```

You can also view the details using an SNMP walk.

Supported Releases

Command/Parameter	NetVisor OS Version
snmp-community-create	Command with parameters first supported in version 2.0
snmp-community-delete	Command introduced in version 2.0
snmp-community-modify	Command introduced in version 2.0
snmp-community-show	Command introduced in version 2.0
snmp-engineid-modify	Command introduced in version 3.0.0
snmp-engineid-show	Command introduced in version 2.0
snmp-show	Command introduced in version 1.2.1
snmp-system-clear	Command introduced in version 3.1.0
snmp-system-modify	Command introduced in version 3.1.0
snmp-system-show	Command introduced in version 3.1.0
snmp-trap-enable-modify	Command introduced in version 2.0
snmp-trap-enable-show	Command introduced in version 2.0
snmp-trap-sink-create	Command introduced in version 2.0
snmp-trap-sink-delete	Command introduced in version 2.0
snmp-trap-sink-show	Command introduced in version 2.0
snmp-user-create	Command introduced in version 2.0
snmp-user-delete	Command introduced in version 2.0
snmp-user-modify	Command introduced in version 2.0
snmp-user-show	Command introduced in version 2.0
snmp-v3-trap-sink-create	Command introduced in version 2.0
snmp-v3-trap-sink-delete	Command introduced in version 2.0
snmp-v3-trap-sink-show	Command introduced in version 2.0

Refer to the Arista NetVisor OS *Command Reference* document for more details on the commands.

Related Documentation

For more information on concepts mentioned in the SNMP chapter, refer to the documents below:

- Net-SNMP
- RFC2790
- RFC1229
- RFC3433

Managing NetVisor switch via NETCONF

NETCONF

NetVisor OS 7.0.2 introduces support for the Network Configuration Protocol (NETCONF) (originally defined in RFC 6241). NETCONF is a network management protocol that provides a mechanism to install, view, manage, and delete the configuration of network devices. Operations are implemented over a Remote Procedure Call (RPC) layer using an Extensible Markup Language (XML)-based data encoding for the configuration data as well as the protocol messages.

> **Note:** The NETCONF protocol is supported over Secure Shell (SSH) using (TCP) port 830 as defined in RFC 6242.

YANG

Yet Another Next Generation (YANG) is a data modeling language used for NETCONF-based operations as defined in RFC 6020. It provides network descriptions for the network nodes and resources.
NetVisor OS uses YANG to specify NETCONF data models and protocol operations.

NETCONF support in NetVisor OS

To support NETCONF services on NetVisor OS, you must first enable NETCONF services by using the `admin-service-modify` command. When you enable the NETCONF services, all configuration services or service requests made through NETCONF services are routed to NetVisor OS, where the service requests are processed.

In NetVisor version 7.0.2, the following operations are supported:

- get-schema --model nvOS
- get --filter-xpath /nvOS:vlans
- get --filter-xpath /nvOS:port-configs
- user-rpc
    - vlan-create
    - vlan-create-by-range
    - vlan-modify
    - vlan-delete
    - vlan-delete-by-range
    - port-config-modify
    - port-config-by-description

Configuring NETCONF:

Use the following command to enable NETCONF:

```
CLI (network-admin@switch) > admin-service-modify if mgmt netconf
```

Note: By default, NETCONF service is disabled. You can enable NETCONF admin service by using the `admin-service-modify` command.

Use the following command to display NETCONF status:

```
CLI (network-admin@switch) > admin-service-show
switch if    ssh nfs web web-ssl web-ssl-port web-port vrrp snmp netconf
icmp
------ ---- --- --- --- ------- ------------ -------- ---- ---- -------
----
switch mgmt on  off on  off     443          80       off  off  on      on

switch data on  off off off     443          80       off  off  off     on
```

Examples of NETCONF supported operations

In NetVisor, the NETCONF services can be leveraged by using any of the supported clients. Below is a sample output by using one of the supported clients (*ncclient*):

get-schema `--model nvOS:`

```
root@VNV-7000119597:~# nvOS_ncclient.py --host switch --username network-
admin --get-schema --model nvOS
Password:
<rpc-reply  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"  message-
id="urn:uuid:7ac4c700-899e-4335-97f1-d79b5929ad52"><data
xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">module  nvOS  {
   namespace "urn:sysrepo:nvOS";
  prefix nvOS;

  revision 2022-09-28 {
    description
      "nvOS version: 7000219751";
  }

  typedef boolean_t {
    type boolean;
  }

  typedef int8_t {
    type int8;
  }

  typedef int16_t {
    type int16;
  }

  typedef int32_t {
    type int32;
  }
```

```
  typedef int64_t {
    type int64;
  }
….
    output {
      uses nvOS_result_t;
    }
  }
}
</data></rpc-reply>
```

get:

```
root@VNV-7000119597:~# nvOS_ncclient.py --host switch --username network-
admin --get /vlans/vlan[id=1098]
Password:
<?xml version="1.0" ?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-
id="urn:uuid:ec78121f-33a2-4af3-9d42-6ebaab79b8f5">
root@VNV-7000119597:~# nvOS_ncclient.py --host switch --username network-
admin --rpc /root/vlan33.xml
Password:
<?xml version="1.0" ?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-
id="urn:uuid:2858d646-f04c-4ef9-b8c3-0c7ee07c3b5a">
     <result_status   xmlns="urn:sysrepo:nvOS">SUCCESS</result_status>
     <result_code   xmlns="urn:sysrepo:nvOS">0</result_code>
     <result_msg xmlns="urn:sysrepo:nvOS">Vlans 33 created</result_msg>
</rpc-reply>

root@VNV-7000119597:~#
```

User-rpc: vlan modify

```
root@VNV-7000119597:~# cat test.xml
<vlan-modify  xmlns="urn:sysrepo:nvOS">
     <id>33</id>
       <description>vlan-modified-description-33_via_user-
rpc_ncclient</description>
</vlan-modify>
root@VNV-7000119597:~#
root@VNV-7000119597:~# nvOS_ncclient.py --host switch --username network-
admin --rpc  test.xml
Password:
<?xml version="1.0" ?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-
id="urn:uuid:96d18e6f-63f3-495d-a3a3-e89fd0a0f5d5">
      <result_status xmlns="urn:sysrepo:nvOS">SUCCESS</result_status>
      <result_code xmlns="urn:sysrepo:nvOS">0</result_code>
     <result_msg xmlns="urn:sysrepo:nvOS"/>
</rpc-reply>
```

```
root@VNV-7000119597:~#
```

Configuring and Using vFlows

This chapter provides information for understanding, configuring, and using the vFlow feature in NetVisor OS using the NetVisor OS CLI.

Understanding vFlows and vFlow Objects

The vFlow functionality in NetVisor OS is a unique Arista feature, which defines fabric-wide policies (using match conditions) to facilitate the manipulation and redirection of traffic flows using physical or logical filtering methods (using action parameters) at line rate. NetVisor OS implements vFlow objects in hardware that have no impact on the forwarding performance of the switch.

The vFlows can be applied to traffic flows regardless of the forwarding method or provisioning construct employed. As such, vFlow objects can be implemented for bridging, routing and extended bridging operations and also for transparent forwarding services such as Virtual Wire and Virtual Link extension (vLE).

The vFlows can also be viewed as Access Control Lists (ACL) with advanced capabilities.

The vFlow functionality offers a versatile, programmable, and distributed method for implementing security access control policies, security service insertion, flow monitoring and telemetry, quality of service, and optimized flow-based forwarding.

In NetVisor OS the vFlow filters operate at wirespeed without any performance degradation because the vflow actions and filtering are applied in the ASIC pipeline at line rate, which ensures no latency or performance degradation.

The vFlow object enables you to:

- Configure traffic filtering based on L2, L3, and L4 layer parameters
- Configure traffic filtering based on action parameters such as blocking and forwarding traffic
- Configure vFlows to copy packets to CPU, packet mirroring, packet classification, traffic metering and bandwidth guarantee
- Gathering statistics for evaluation and analytics

At a high level, vFlow feature supports the following actions, which can be configured using the CLI commands:

- Creating a vFlow Object

```
CLI (network-admin@switch-1) > vflow-create name <vflow-name> scope [local|
fabric] {specify one or more parameters} {specify any action}
```

- Modifying an existing vFlow Object

```
CLI (network-admin@switch-1) > vflow-modify name <vflow-name> {specify one
or more parameters}
```

- Deleting an existing vFlow Object

```
CLI (network-admin@switch-1) > vflow-delete name <vflow-name>
```

- Displaying the applicable actions and use cases for a selected vFlow Object

```
CLI (network-admin@switch-1) > vflow-show {specify one or more parameters}
```

These actions are explained in detail in the subsequent sections for configuring vFlow objects with specific parameters.

Elements of a vFlow

NetVisor OS identifies a vFlow object by a unique name and is composed of the following elements:

- Administrative scope and state
- Implementation stage
- Traffic flow filter
- Forwarding action

Configuring the Administrative Scope and State

The administrative *state* of a vFlow object determines if you enable or disable the corresponding flow policy in the switch hardware, as defined by mutually exclusive keywords `enable` and `no-enable`. By default, NetVisor OS enables newly created vFlow objects.

The administrative scope defines the set of switches in the fabric where you create the vFlow object, which is controlled by the keyword scope, and can be either `fabric` or `local` scope. The administrative parameters for a `vflow-create` command are:

```
CLI (network-admin@switch-1) > vflow-create name <vflow-name> scope
[fabric|local]  [enable|no-enable]
```

`name`	The vFlow object's unique identifier
`scope [fabric|local]`	Defines the scope of the vFlow object. Once a Vflow object is created using either the local or the fabric scope, you cannot modify the scope of the vFlow object later. To modify, you must delete the vFlow object and create a new one.
`enable|no-enable`	Enables or disables the flow policy in hardware. By default, NetVisor enables the vFlow objects. You can disable the vflow policy using the `no-enable` parameter.
`{parameters}`	Specify one or more of the parameters • Table name • Filtering parameters • Action parameters • Flow class For details, see the *Filtering of Traffic Flows*, *Forwarding Action in vFlow Filters*, and *Commands and Parameters Applicable to vFlow Traffic* sections. Also, see the *Command Reference* Guides

Note: You can specify the hardware table name while creating a vFlow object, however, if not specified, NetVisor OS uses the default table, **System-L1-L4-Tun-1-0**.

Fabric Scope

A fabric-scoped vFlow is a single managed object distributed across all switches that are part of the Unified Cloud Fabric in NetVisor OS. To create a fabric scoped vFlow object, for example, use the command:

```
CLI (network-admin@switch-1) > vflow-create name example_fabric_scope scope
fabric enable {parameters}
```

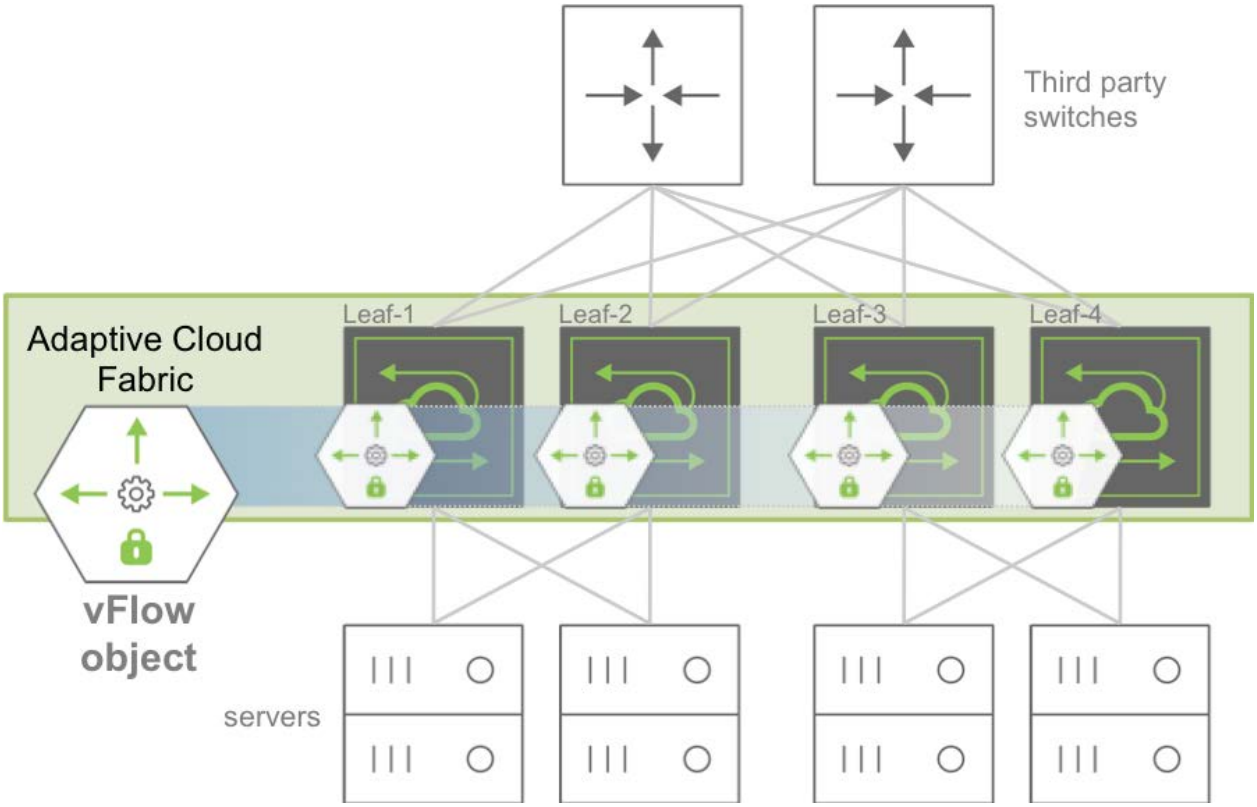


Figure 12-1: Fabric Scoped vFlow Object Example

Figure 12-1 illustrates a fabric scoped vFlow object topology, where a single vFlow object is created on all four switches: Leaf-1, Leaf-2, Leaf-3, and Leaf-4 that are part of the Unified Cloud Fabric. The switches in the Unified Cloud Fabric are also connected to multiple servers and other third party switches. In this scenario, the fabric-scoped vFlow can be modified concurrently on all switches of the fabric with a single CLI or API command, by referencing the unique name. For example, the below command disables the previously created vFlow object, **example_fabric_scope** for the entire fabric, where NetVisor OS does not delete the object, but uninstalls the object from the hardware tables.

For example:

```
CLI (network-admin@switch-1) > vflow-modify name example_fabric_scope scope
fabric no-enable
```

Local Scope

A local-scoped vFlow is an object defined and instantiated on one single switch. To create a locally scoped vFlow, for example, use the following command:

```
CLI (network-admin@switch-1) > vflow-create name example_local_scope scope
local enable {parameters}
```

NetVisor OS allows you to apply or modify the same vFlow policy on multiple switches concurrently using a single CLI or API command by including the `switch` keyword followed by the list of individual switches or switch groups. Below is an example on creating a vFlow object on four switches, leaf-1, leaf-2, leaf-3, and leaf-4:

```
CLI (network-admin@leaf-1) > switch leaf-1,leaf-2 \ vflow-create name
example_local_scope scope local

CLI (network-admin@leaf-1) > switch leaf-3,leaf-4 \ vflow-create name
example_local_scope scope local
```

The above commands create the same vFlow object, *example_local_scope* on the four switches, leaf-1, leaf-2, leaf-3, and leaf-4 (see ***Figure 12-2***).

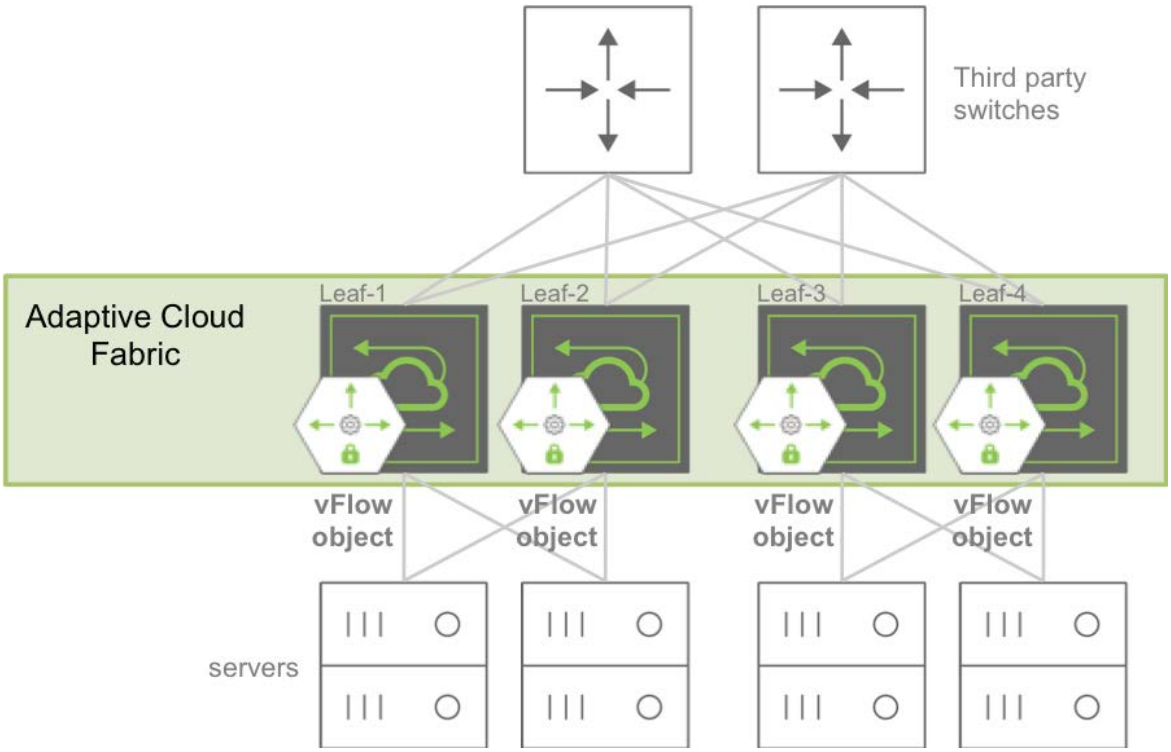


Figure 12-2: Local Scoped vFlow Object Example

You can now modify or delete the vFlow objects on individual switches as explained in the example below:

To disable the vFlow object, example_local_scope on the switch, leaf-1, use the command:

```
CLI (network-admin@leaf-1) > switch leaf-1 vflow-modify name
example_local_scope no-enable
```

To delete the vFlow object, example_local_scope on the switch, leaf-2, use the command:

```
CLI (network-admin@leaf-1) > switch leaf-2 vflow-delete name
example_local_scope
```

Implementing the vFlow Policies

NetVisor OS allows you to apply multiple policies in parallel or in series to a particular traffic flow by providing the vFlow construct with two main attributes to control the sequential order of execution relative to other vFlows such as the hardware table and precedence.

The following command keywords enable this functionality:

- `table-name` – hardware vFlow table name
- `precedence` – processing priority value

Hardware Table

NetVisor OS provides multiple filter tables along the internal flow hardware data path. However, by default, the vFlow is installed in the ingress filter table, but allows you to optionally implement the vFlow in any other available table, although flow filtering, manipulation, and redirection capabilities may become limited. ***Figure 12-3*** describes the available hardware tables with the corresponding vFlow table names and how the tables are concatenated, allowing both cascading and parallel execution policies.

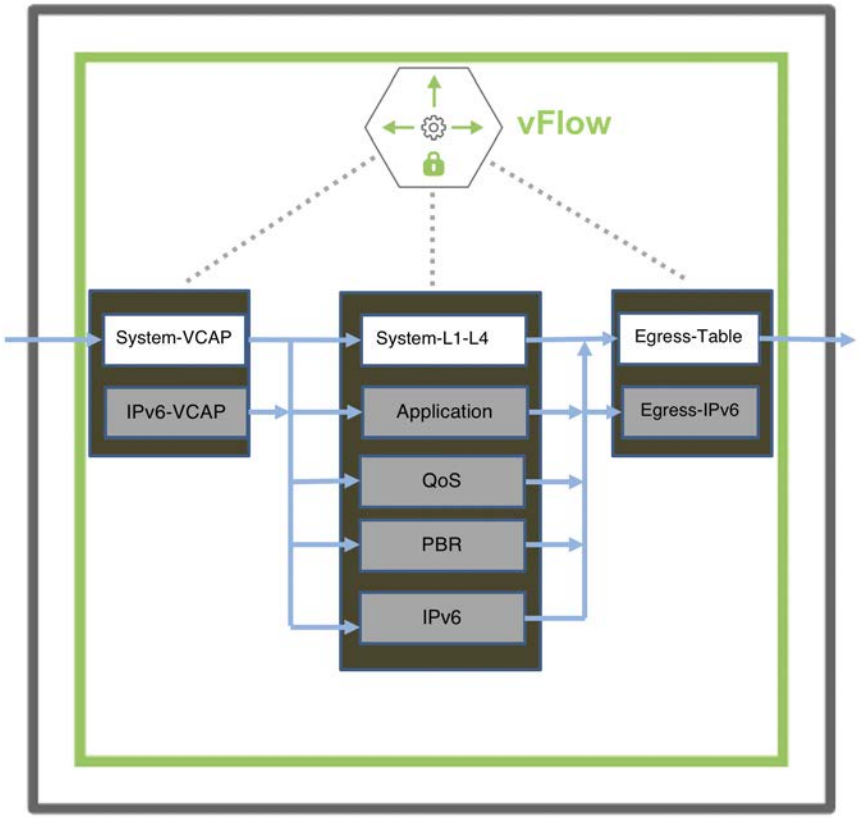


Figure 12-3: Concatenation of vFlow Hardware Tables

Additionally, ***Figure 12-3*** highlights the data-path forwarding stage for each filter table, where some tables are always enabled (displayed in white), while some tables require manual enabling (displayed in grey) such as the Application, QoS, PBR, and IPv6 tables. Use the `table-name` keyword to install or program the vFlow in the specified hardware table.

Table 12-1: Hardware Filter Tables with Descriptions

Hardware Filter Tables	Description
System-VCAP	Where the system VCAP policies are defined at the pre-ingress stage
System-L1-L4	Where the system ingress traffic filtering policies are defined for L2, L3, and L4 packet parameters at the ingress or ICAP table. All system rules are defined in ICAP
Egress-Table	Where the system egress policies are defined at the egress or ECAP table. Supports `drop` and `forward` actions.
Application Table	Where the user application level policies are defined.
QoS Table	Where the ACL policies are defined
PBR Table	Where the policy based routing policies are defined. For details, see the Configuring Policy-Based Routing section.
IPv6 Table	Where IPv6 policies are defined.
IPv6 VCAP Table	Where IPv6 VCAP policies are defined.

You can view the configurable hardware tables by using the command:

```
CLI (network-admin@leaf-1) > vflow-table-profile-show layout vertical
profile:             system
hw-tbl:              switch-main
enable:              enable
flow-capacity:       768
flow-slices-needed:  4
flow-slices-used:    7
comment:              System-L1-L4-flows
profile:             npu-app
hw-tbl:              npu-main
enable:              disable
flow-capacity:       0
flow-slices-needed:  0
flow-slices-used:    0
comment:              L1-L4-flows
profile:              application
hw-tbl:              switch-main
enable:              disable
flow-capacity:       0
flow-slices-needed:  1
flow-slices-used:    0
comment:              User-Application
profile:             qos
hw-tbl:              switch-main
enable:              disable
```

```
flow-capacity:       0
flow-slices-needed:  1
flow-slices-used:    0
comment:             QoS
profile:             ipv6
hw-tbl:              switch-main
enable:              disable
flow-capacity:       0
flow-slices-needed:  2
flow-slices-used:    0
comment:             IPv6
profile:             pbr
hw-tbl:              switch-main
enable:              disable
flow-capacity:       0
flow-slices-needed:  0
flow-slices-used:    0
comment:             PBR
profile:             egress-v6
hw-tbl:              switch-main
enable:              disable
flow-capacity:       0
flow-slices-needed:  1
flow-slices-used:    0
comment:             Egress-IPv6
profile:             ipv6-vcap
hw-tbl:              switch-main
enable:              enable
flow-capacity:       256
flow-slices-needed:  1
flow-slices-used:    1
comment:             VCAP-IPv6
```

Note: The capacity and availability of the hardware tables vary between switch models.

The optional tables (in grey in ***Figure 12-3***) are disabled by default. You can enable optional tables with the `vflow-table-profile-modify` command.

`CLI (network-admin@leaf-1) > vflow-table-profile-modify`

`vflow-table-profile-modify`	Modify vFlow table profiles.
`profile application|ipv6|qos|ipv6-vcap`	Specify the type of vFlow profile.
`hw-tbl switch-main|switch-hash|npu-main|npu-hash`	Specify the hardware used by vFlow.
`enable|no-enable`	Enable or disable vFlow table profile.

For example, enable the qos table using the command:

```
CLI (network-admin@leaf-1) > vflow-table-profile-modify profile qos enable
hw-tbl switch-main
```

NetVisor OS version 6.1.0 introduces ipv6-vcap table profile to enable IPv6 filtering for features like Network Packet Broker. When you enable ipv6-vcap table profile, NetVisor OS constructs an IPv6 VCAP table by allocating half the space in the VCAP table for IPv6 entries.

Enable IPv6 filtering in VCAP table by using the command:

```
CLI (network-admin@leaf-1) > vflow-table-profile-modify profile ipv6-vcap
hw-tbl switch-main enable
```

You must reboot the switch or restart the nvOSd service for the settings to take effect. When you enable optional hardware tables, NetVisor OS allocates a minimum number of entries in the order of 256 vFlow objects (the number of vFlow objects varies based on the platform and the type of the table). For maximum vFlow scalability, enable hardware tables only when necessary. You can monitor the resource consumption of active hardware tables with the following command:

```
CLI (network-admin@leaf-1) > vflow-table-show layout vertical
name:                Egress-Table-1-0
flow-max-per-group:  512
flow-used:           0
flow-tbl-slices:     3
capability:           match-metadata
flow-profile:        system
name:                 System-L1-L4-Tun-1-0
flow-max-per-group:  4096
flow-used:           62
flow-tbl-slices:     4
capability:           set-metadata
flow-profile:        system
name:                 System-VCAP-table-1-0
flow-max-per-group:  256
flow-used:           3
flow-tbl-slices:     3
capability:          none
flow-profile:        system
name:                 VCAP-IPv6-table-1-0
flow-max-per-group:  256
flow-used:           0
flow-tbl-slices:     1
capability:          none
flow-profile:         ipv6-vcap
```

Precedence

When you implement two or more vFlow objects within the same hardware table, it may be necessary to enforce a particular evaluation order. Use the keyword `precedence` to enforce the evaluation order as NetVisor OS executes vFlows with higher precedence value first. See a sample configuration below:

Figure 12-4 displays the precedence or evaluation order for different vFlow objects. When a flow matches two or more vFlows with the same precedence, the corresponding vFlow actions are merged and executed together. When you create the vFlow, NetVisor OS validates that the new object is consistent and can be merged with objects with the same precedence.

The precedence value is within a numerical range of 2 and 15, with 2 as the default value. You cannot configure the evaluation order or `precedence` value beyond 15.

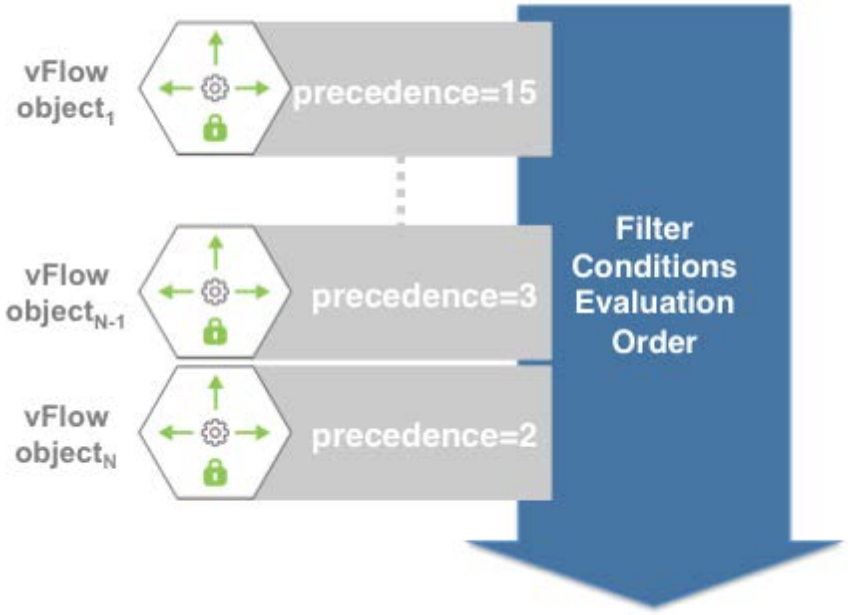


Figure 12-4: Evaluation Order for vFlow Objects with Different Precedence

When you create multiple vflow objects within the same hardware table without specifying the precedence value (default value being 2), NetVisor displays an error message about the vFlow conflicts. For example,

- Create a vFlow:

```
CLI (network-admin@Leaf1) >  vflow-create name example_vflow1 scope fabric
 bw flow-class meter bw-max 2g
```

- Create a second vFlow:

```
CLI (network-admin@Leaf1) >  vflow-create name example_vflow2 scope fabric
 bw flow-class meter bw-max 5g src-ip 192.168.20.1

vflow-create: Flow conflicts with Flow example_vflow1, ID68: specify fields
to make flows mutually exclusive or change the flow precedence
```

The error message is generated because the vFlow configurations conflict with each other. To differentiate

between the two vflows, assign a different precedence to example_vflow2:

```
CLI (network-admin@Leaf1) >  vflow-create name example_vflow2 scope fabric
 bw flow-class meter bw-max 5g src-ip 192.168.20.1 precedence 5
```

Managing Traffic Classes with vFlow

The vFlow classes indicate the priority assigned to a packet within a switch for internal processing and prioritization and specifies a service type: traffic metering or traffic shaping, bandwidth guarantee. NetVisor OS supports two types of vFlow classes:

- System Flow Classes
    - Metered flow class, where the traffic is not allowed to exceed a set rate.
    - Guaranteed bandwidth flow class, where the vFlow object guarantees a certain bandwidth and the switch priority is 9.
    - Lossless flow class, where `drop` action is unavailable and the switch priority is 10.
- User Defined Flow Classes
    - Flow classes created by users with priorities between 1 and 8
    - Used for traffic metering
    - Used for traffic shaping and bandwidth guarantee

Filtering of Traffic Flows

In the context of vFlows, filtering describes the traffic characteristics on which the network administrator intends to undertake a certain action. You can define the filter through one of the arbitrary set of matching qualifiers as illustrated in ***Figure 12-5*** and ***Figure 12-6***:

- Operating at an OSI or Internet Protocol layer, which is based on the packet header content

OR

- Based on NetVisor OS logical constructs, such as vNET, vFlow metadata, or vRouter.

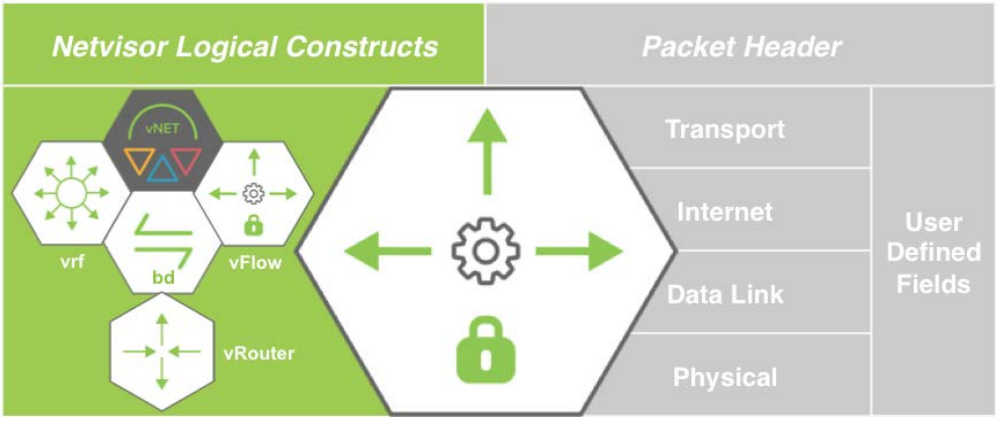


Figure 12-5: vFlow Filtering Using Packet Header Fields and NetVisor Logical Constructs

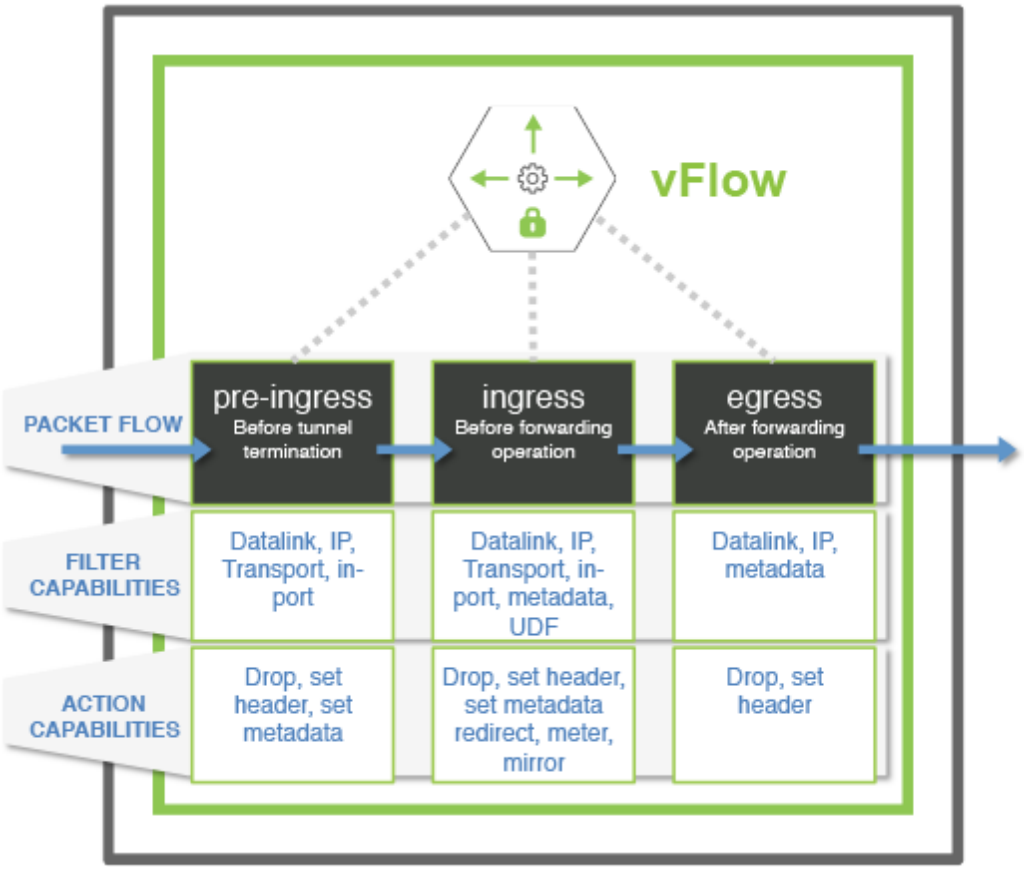


Figure 12-6: vFlow Tables - Packet Forwarding Stage and Filter or Action Capabilities

To define the vFlow filter using the components of OSI layer, use the following keywords or parameters in the `vflow-create` command:

Filtering Qualifiers at Physical Layer

- `in-port` *`port-list`* — ingress physical interface or LAG interface identifiers. NetVisor OS accepts values as a single value, a dash-separated value range, or comma-separated list of values and ranges.
- `out-port` *`port-list`* — egress physical interface or LAG interface identifier. NetVisor OS accepts values as a single value, a dash-separated value range, or comma-separated list of values and ranges. This out-port is applicable for Egress_table.

Filtering Qualifiers at Data Link Layer

- `packet-res` — packet resolution in the ASIC. The resolution types can be L2-unicast, L2-unknown-unicast, L2 multicast, L2-unknown-multicast, L2-broadcast. For more details, see the Enhancing the vFlow Capability to Match Forwarding Type and Packet Resolution in ASIC section.
- `fwding-type` — ASIC forwarding type: VLAN, VXLAN, or vLE
- `vlan` — VLAN (Virtual LAN) identifier (IEEE 802.1q). Range is from 0 through 4095.
- `vxlan` —VxLAN Network Identifier or VNI

- `vxlan-ether-type` — Ethernet type such IPv4, ARP, WAKE, RARP, VLAN, IPv6, LACP, MPLS-uni, MPLS-multi, Jumbo, dot1X, AOE, Q-in-Q, LLDP, MACSEC, ECP, PTP, FCOE, FCOE-init, or Q-in-Q-old
- `vxlan-proto` — Protocol type for the VXLAN. Includes TCP, UDP, ICMP, IGMP, IP, ICMPv6
- `src-mac` — Source MAC address
- `src-mac-mask` — Mask for source MAC address
- `dst-mac` — Destination MAC address for the vFlow
- `dst-mac-mask` — Mask for destination MAC address
- `vlan-pri` — Class of Service (CoS) or VLAN priority (IEEE 802.1p), ranges from 0 through 7.

Filtering Qualifiers at Internet Layer

- `src-ip` — Source IP address for the vFlow
- src-ip-mask — Mask for source IP address
- `dst-ip` — Destination IP address
- `dst-ip-mask` — Mask destination IP address
- `ttl` — Packet time-to-live
- `proto` — Layer 3 protocol for the vFlow
- `dscp` — 6-bit Differentiated Services Code Point (DSCP) for Quality of Service (QoS), in the range of 0 to 63
- `dscp-start` — Start value for DSCP
- `dscp-end` — End value for DSCP
- `tos` — Type of Service (ToS) value for Quality of Service (QoS)
- `tos-start` — Start value for Type of Service (ToS) range
- `tos-end` — End value for Type of Service (ToS) range

Filtering Qualifiers at Transport Layer

- `src-port` — Source transport port
- `src-port-mask` — Mask for source transport port
- `src-port-end` — Ending port for a range of source ports. Use this qualifier along with `src-port` to specify a range of ports. This option is mutually exclusive with `src-port-mask`
- `dst-port` — Destination transport port
- `dst-port-mask` — Mask for destination transport port
- `dst-port-end` — Ending port for a range of destination ports. Use this qualifier along with `dst-port`

to specify a range of ports. This option is mutually exclusive with `dst-port-mask`

- `tcp-flags` — Comma-separated list of TCP control flag values

Filtering Qualifiers at User Defined Fields (UDFs)

- `udf-name[1-3]` — Reference to a User Defined Field (UDF) object, and defines advanced multi-layer filtering. Up to 3 objects are supported.
- `udf-data[1-3]` — Data value applied to the corresponding UDF object
- `udf-data[1-3]-mask` — Mask value applied to the corresponding UDF object

For details on configuring vFlows with UDFs, see the Configuring vFlows with user Defined Fields section.

NetVisor OS Logical Filtering

To define the vFlow filter using the NetVisor OS logical constructs, use the following keywords or parameters in the `vflow-create` command:

- `metadata` — Metadata tag value assigned to packets along the internal hardware forwarding path, which can be used to correlate different vFlow objects operating at different ingress and egress stages
- `bridge-domain` — Logical abstraction of data-link learning and forwarding domain, implemented using a combination of physical interfaces, VLANs and VNI
- `vrouter-name` — Reference to an Internet layer VRF context defined on a local vRouter
- `vrf` - Parameter used to enable vFlows to operate at scale in a distributed VRF environment. For details, see the *Configuring VRF-aware vFlow* section of the *Configuring VXLAN* chapter.
- `vnet` — Virtual Network (vNET) value, used for identifying traffic belonging to a logical network partition for multi-tenancy and network segmentation purposes.
- src-vpg, dst-vpg, bidir-vpg-1, bidir-vpg-2 — Source, destination, and bidirectional Virtual Port Groups(vPGs) are logical constructs that allows you to club ports together in order to support the Network Packet Broker solution. For details, see the *Configuring Network Packet Broker* section of the *Configuring Network Management and Monitoring* chapter.

Configuring vFlows with User Defined Fields (UDFs)

NetVisor allows you to define policy filters through one of the arbitrary set of matching qualifiers as explained in the Filtering of Traffic Flows section. One of the qualifier is the User Defined Field (UDF).

A UDF can match up to 128 bytes of a packet starting from the first byte of the packet. The length of the match can be from 1 to 4 bytes. Hardware with a Trident chip supports the creation of 8 UDF IDs. Each id can match a 2 byte portion of a packet. Creating a UDF with a length of 3 or 4 bytes requires 2 UDF IDs whereas a UDF with length of 1 or 2 bytes required 1 UDF id. The length specified for each UDF determines the total number of UDFs supported by NetVisor OS. If you specify a length of 3 or 4 bytes, a maximum of 4 UDFs can be created. If you specify a length of 1 or 2 bytes, a maximum of 8 UDFs can be created.

> **Limitation**: UDF offset range supported for UDF header packet-start type ranges from 0-63. This limitation is applicable for all NRU03 platforms.

A UDF adds a qualifier to the vFlow group, and you should create all UDFs before creating any vFlows. This feature is disabled by default, and you can enable it by using the following command:

```
CLI(network-admin@Spine1) > vflow-settings-modify enable-user-defined-flow|no-enable-user-defined-flow
```

`vflow-settings-modify`	Use this command to update a user vflow setting
`Specify one of more of the following options`	
`enable-user-defined-flow|no-enable-user-defined-flow`	Specify to enable or disable the user defined flows.
`vxlan-analytics|no-vxlan-analytics`	Specify to enable or disable VXLAN analytics. **Note**: You must disable VXLAN analytics before enabling the longlived tcp connection
`inflight-vxlan-analytics|no-inflight-vxlan-analytics`	Specify to enable or disable the inflight VXLAN analytics. **Note**: You must disable inflight VXLAN analytics before enabling the longlived tcp connection.
`longlived-tcp-conn-stats|no-longlived-tcp-conn-stats`	Specify to enable or disable the long-lived TCP connection statistics. **Note**: You must enable the user-defined-flow before enabling the longlived tcp connection statistics.

To enable the user defined vflow, use the command:

```
CLI(network-admin@Spine1) > vflow-settings-modify enable-user-defined-flow
```

To disable the feature, use the command:

```
CLI(network-admin@Spine1) > vflow-settings-modify no-enable-user-defined-flow
```

Note: Reboot NetVisor OS for the changes (`enable` or `disable` commands) to take effect on the platform.

The command, `udf-create`, adds the qualifier to the UDF group in the hardware. This allocates UDF IDs based on the length. The command, `vflow-create`, has parameter fields to provide the data and mask to be matched by the vFlow. You can create vFlows with either one or two UDFs.

You cannot modify a UDF after adding it to a vFlow. You must delete the vFlow, modify the UDF, and re-create the vFlow with the modified UDF.

New Commands for UDF

To create a new UDF, use the following command:

```
CLI(network-admin@Spine1) > udf-create name udf1 scope local offset 10 length 2 header packet-start
```

`udf-create`	Create the UDF qualifier list
`name` *`name-string`*	Create the UDF name
`scope local|fabric`	Scope for the UDF
`offset` *`number-bytes`*	The offset in bytes. This is a value between 1 and 128.
`length` *`number-bytes`*	The length in bytes. This is a value between 1 and 4 bytes.
`header [packet-start|l3-outer|l3-inner|l4-outer|l4-inner]`	The header from where offset is calculated.

To delete an UDF command:

```
CLI(network-admin@Spine1) > udf-delete name udf1
```

`udf-delete`	Delete UDF qualifier list
`name` *`name-string`*	The name of the UDF to delete.

To modify an existing UDF command:

```
CLI(network-admin@Spine1) > udf-modify name udf1 scope local offset 20 length 4 header packet-start
```

`udf-modify`	Modify UDF qualifier list
`name` *`name-string`*	The name of the UDF to modify.

One or more of the following options:	
`offset` *`number-bytes`*	The offset in bytes. This is a value between 1 and 128.
`length` *`number-bytes`*	The length in bytes. This is a value between 1 and 4 bytes.
`header packet-start|l3-outer|l3-inner|l4-outer|l4-inner`	The header from where offset is calculated.

```
CLI(network-admin@Spine1) >  udf-show

switch      name  scope   offset  length   header
------      ----  -----  ------  ------   ------------
spine1    u1    local   20      4        packet-start
spine1    u2    local   24      4        packet-start
```

`switch`	Displays the name of the switch
`udf-show`	Displays the UDF qualifier list
`name` *`name-string`*	Displays the UDF name
`scope local|fabric`	Displays the scope for the UDF
`offset` *`number-bytes`*	Displays the offset in bytes. This is a value between 1 and 128.
`length` *`number-bytes`*	Displays the length in bytes. This is a value between 1 and 4 bytes.
`header packet-start|l3-outer|l3-inner|l4-outer|l4-inner`	Displays the header from where the offset is calculated.

The command, `vflow-create`, has the following additional parameters:

`udf-name1` *`udf-name`*	Specify the name of the UDF.
`udf-data1` *`udf-data1-number`*	Specify UDF data1q with the format 0xa0a0a01
`udf-data1-mask` *`udf-data1-mask-number`*	Specify he mask for udf-data with the format 0xffffffff.
`udf-name2` *`udf-name`*	Specify the name of the UDF.
`udf-data2` *`udf-data2-number`*	Specify UDF data2 with the format 0xa0a0a01
`udf-data2-mask` *`udf-data2-mask-number`*	Specify the mask for udf-data with the format 0xffffffff.

For example, to create a vflow with UDF parameters, use the command:

```
CLI(network-admin@Spine1) >  vflow-create name udf1 scope local udf-name1
udf1 udf-data 0x0a0a0a01 udf-data-mask1 0xffffffff udf-name2 udf2 udf-data2
0x0a0a1400 udf-data-mask2 0xffffff00
```

```
CLI(network-admin@Spine1) >   vflow-show

name   scope  type   precedence  udf-name1  udf-data1   udf-data-mask1  udf-name2   udf-data2   udf-data-mask2
---- ------ ------ ---------- --------- ---------   --------------   --------- ---------    --------------
udf1   local  vflow   default       udf1      0xa0a0a01   0xffffffff     udf2         0xa0a1400   0xffffff00
```

Configuring a UDF vFlow for Filtering ARP Requests

Consider a scenario where you need to allow only the ARP requests destined for Anycast gateway IP (`anycast-gw-ip`) to proceed with ARP processing, while blocking other transit ARP requests. You can create a UDF with a higher precedence value than system vFlows to achieve this. Follow the steps below to create such a configuration:

- First enable UDF by using the command:

```
CLI  (network-admin@switch)  >  vflow-settings-modify  enable-user-defined-flow
```

`udf-name1` *`udf-name`*	Specify the name of the UDF.
`udf-data1` *`udf-data1-number`*	Specify UDF data1q with the format 0xa0a0a01
`udf-data1-mask` *`udf-data1-mask-number`*	Specify he mask for udf-data with the format 0xffffffff.

- Restart the nvOSd.
- Create a UDF, *target-ip*, for the ARP request using the command:

```
CLI (network-admin@switch) > udf-create name target-ip scope local offset
42 length 4 header packet-start
```

- Create UDF to punt ARP packets to CPU using the command:

```
CLI (network-admin@switch) > vflow-create name UDF-Allow-AnyGW-243 scope
fabric precedence 15 action copy-to-cpu udf-name1 target-ip udf-data1
0xa65f302 udf-data1-mask 0xffffffff table-name System-L1-L4-UDF-1-0 flow-cb
arp-cb
```

`vflow-create`	Creates a virtual flow definition for L2 or L3 IP.
`name  name-string`	Enter a name for the vFlow.
`scope  local|fabric`	Specify the scope for vFlow.
`precedence  default|0..15`	Specify the traffic priority value between 2 and 15.
`action  copy-to-cpu`	Specify the forwarding action to apply to the vFlow, in this case, `copy-to-cpu`.
`udf-name1`	Specify the UDF name created in step c.
`udf-data1  udf-data1-number`	Enter the UDF data (hexa-decimal value equivalent to the IP address of anycast gateway.
`udf-data1-mask  udf-data1-mask-number`	Enter the mask for UDF data.
`table-name  vflow-table  name`	Enter the vFlow table name.

`flow-cb default-cb|arp-cb|bcast-cb| igmp-cb|pim-cb|dhcp-cb|dhcpv6-cb| dmac-miss-cb|l2-miss-cb|no-cb`	Specify the call-back option (here `arp-cb`).

To verify the configuration, use the `vflow-show` command:

```
CLI (network-admin@switch) > vflow-show name UDF-Allow-AnyGW-243

name                scope  type  burst-size precedence action      udf-name1 udf-data1 udf-data1-mask enable table-name           flow-cb
------------------- ------ ----- ---------- ---------- ----------- --------- --------- -------------- ------ -------------------- -------
UDF-Allow-AnyGW-243 fabric vflow auto       15         copy-to-cpu target-ip 0xa65f302 0xffffffff     enable System-L1-L4-UDF-1-0 arp-cb
```

Forwarding Action in vFlow Filters

The forwarding action defines the switch behavior for traffic that matches with the vFlow filter. Depending on the use case, the possible actions for the traffic flow include:

- Dropping of filtered traffic
- Regular or customized forwarding of filtered traffic
- Redirection or replication of filtered traffic to a switch processor, a physical port, or to an IP address destination.

Access Control

To provide security access control policies that operate on high-speed distributed networks, the switch, which employs low-cost high-capacity technology like ASIC Ternary Content Addressable Memory (TCAM), is the ideal point of insertion. With Arista Unified Cloud Fabric, high-performance security access control using a **blacklist** or **whitelist** approach can be implemented consistently across the entire network using fabric-wide vFlow policies.

NetVisor OS provides several vFlow actions for securing the network traffic with Access Control Lists (ACL) as described in the table :

Table 12-2: vFlow Actions Supported in NetVisor OS

Action	Description
`action none`	Traffic is forwarded (permit filtered traffic) This is the default action.
`action drop`	Traffic matching the filter is removed from traffic path (block filtered traffic)
`action to-port`	Traffic matching the filter is forwarded to specified ports
`action to-cpu`	Traffic matching the filter is sent to CPU only
`action trap`	Traffic matching the filter is trapped to CPU
`action copy-to-cpu`	Traffic matching the filter is forwarded normally and copied to CPU, this action does not affect the traffic policy, but creates a copy of the policy to the CPU
`action copy-to-port`	Traffic matching the filter is forwarded normally and copied to port
`action setvlan`	VLAN ID is set for the traffic matching the filter
`action add-outer-vlan`	New outer VLAN tag is added to the traffic matching the filter
`action set tpid`	Traffic matching the filter is tagged with the provided TPID
`action to-port-set-vlan`	Traffic matching the filter is sent to provided port on specified VLAN
`action to-span`	Traffic matching the filter is forwarded to the span ports
`action set-metadata`	Metadata is set for the traffic matching the filter
`action set-dscp`	DSCP value is set for the traffic matching the filter
`action set-dmac`	Destination MAC is set for traffic matching the filter

`action to-next-hop-ip`	The next hop IP address for traffic redirection
`action set-dmac-to-port`	Destination MAC is set and it is forwarded to port specified
`action to-ports-and-cpu`	Traffic matching the filter is forwarded to ports and CPU
`action set-vlan-pri`	Set VLAN priority for traffic matching the filter
`action set-smac`	Source MAC is set for the traffic matching the filter
`action drop-cancel-trap`	Traffic matching the flow is dropped and not trapped
`action to-ecmp-group`	ECMP group for traffic redirection
`action redirect-to-vrouter`	Redirect packets to vrouter
`action strip-outer-vlan`	Strip outer-vlan of single/double tagged traffic
`action cancel-switch-to-cpu`	Cancel copy/switch to CPU

Typically the action resolution process gathers the actions from each slice match and decides on a collective action. However, when there is an action conflict, the collective action is not possible and the `drop` action takes higher priority. With the implementation of virtual slide mapping feature, action resolution is handled by the precedence or priority value.

Understanding Blacklist Policy

You can implement access control policies using a blacklist approach, that is, approving all traffic by default and explicitly restricting certain traffic categories.

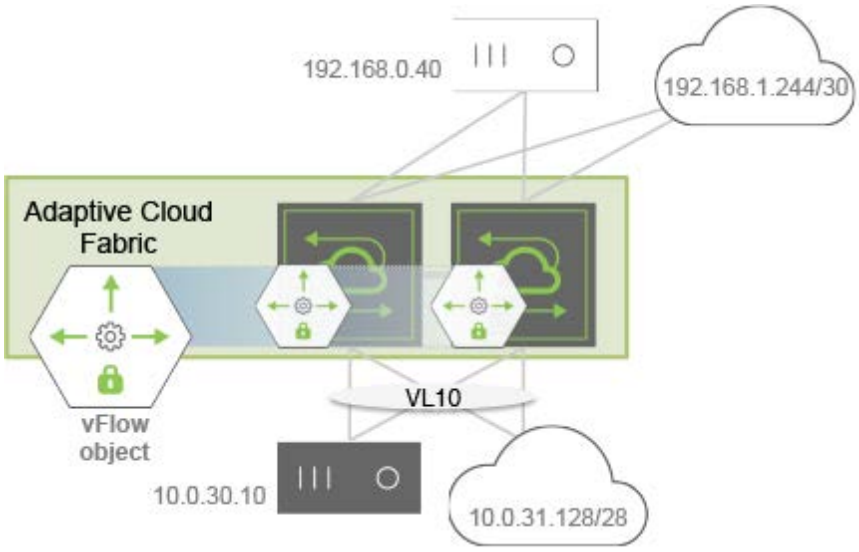


Figure 12-7 - Blacklist Topology Example

For example, as shown in ***Figure 12-7***, you want to block HTTP traffic ingressing the switches in VLAN 10 going from the black host to the white host, while permitting all other traffic. Use the following parameters to configure a vFlow for this purpose:

- vFlow name: `ACL01`
- Filter: `VLAN 10`
- TCP port: `80`
- Source IP address: `10.0.30.10`

- Destination IP address: `192.168.0.40`
- Forwarding Action: `drop`

```
CLI (network-admin@switch) > vflow-create name ACL01 scope fabric vlan 10
src-ip 10.0.30.10 dst-ip 192.168.0.40 proto tcp dst-port 80 action drop
```

For maximum ACL scalability, the blacklist policy can be also implemented using the System-VCAP table (see **Figure 12-8**) and provide additional capacity based on the switch models. In this approach, NetVisor OS drops the blacklisted traffic at the pre-ingress stage without utilizing the resources in the default ingress table.

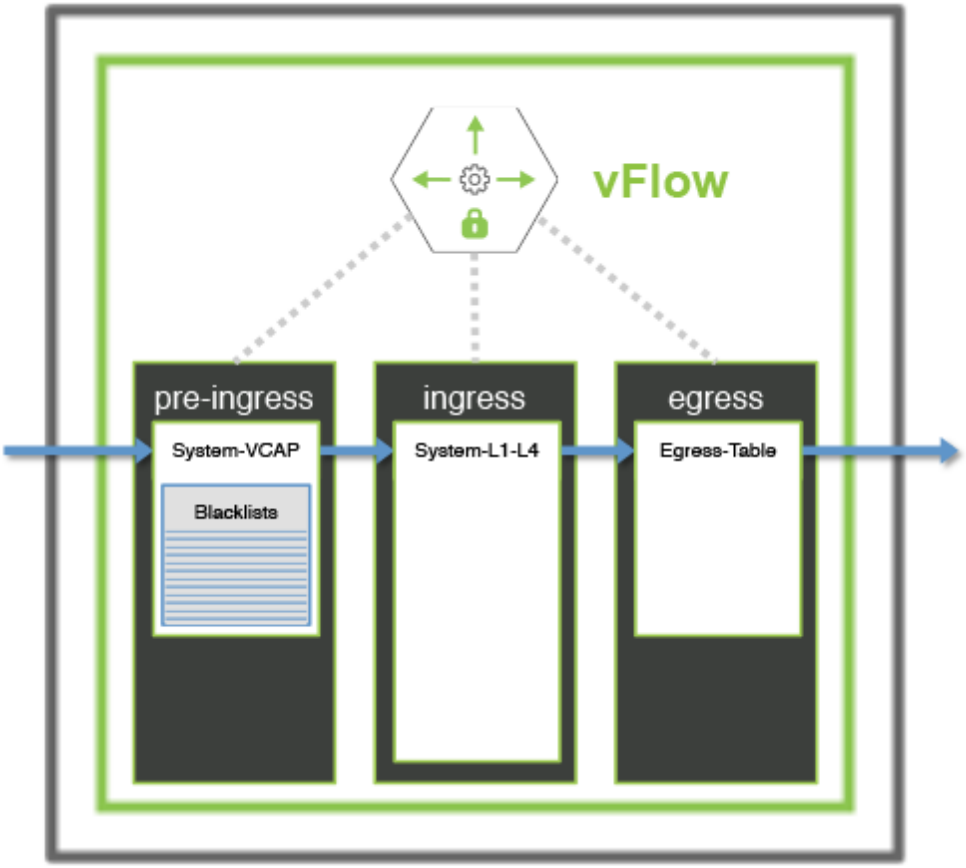


Figure 12-8: Scaling the Blacklist ACLs using a Pre-Ingress Filter

Understanding Whitelist Policy

Contrary to the blacklist policy, the Whitelist model explicitly defines network traffic that is allowed and uses a default permit policy. ***Figure12- 9*** illustrates an example where you want to deny default traffic and approve traffic ingressing the switches on VLAN 10 for the following conditions:

- Traffic between the black hosts
- Traffic between the two subnets (clouds)
- Secure Shell (SSH) traffic between the white hosts where the host below (10.0.31.4) acts as a responder.

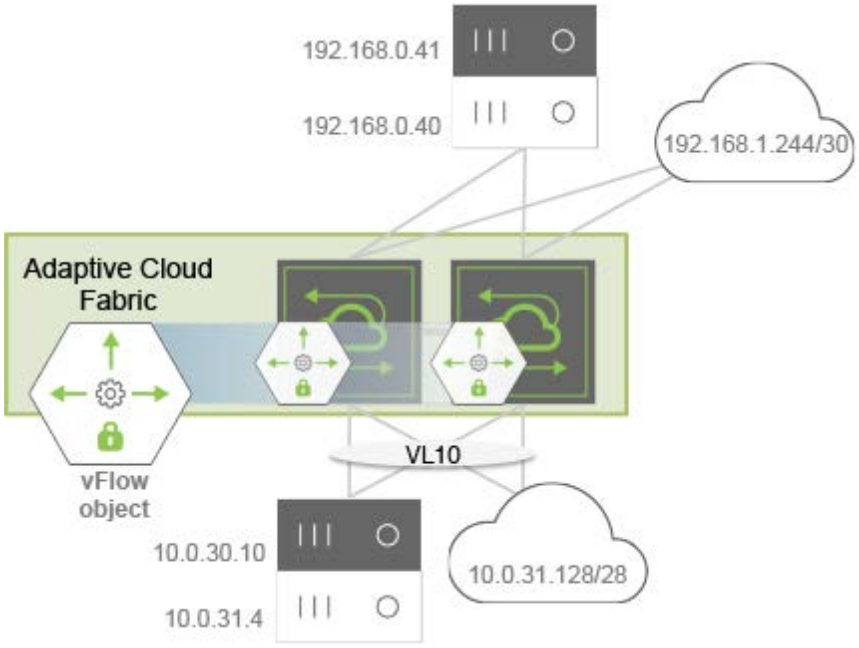


Figure 12-9: Whitelist Topology Example

To implement this policy, create four `fabric`-scoped vFlow objects, with the last one (**vFlow ACL04**) having the least (default) precedence and by using the following parameters:

1. **vFlow ACL01**

   - Filter: VLAN `10`
   - Source IP address: `10.0.30.10`
   - Destination IP address: `192.168.0.41`
   - Forwarding Action: `none`
   - Precedence: `4`

```
CLI (network-admin@switch) > vflow-create name ACL01 scope fabric vlan 10
precedence 4 src-ip 10.0.30.10 dst-ip 192.168.0.41
```

2. **vFlow ACL02**

   - Filter: VLAN `10`
   - Source IP address: `10.0.31.128/28`
   - Destination IP address: `192.168.1.144/30`
   - Forwarding Action: `none`
   - Precedence: `4`

```
CLI (network-admin@switch) > vflow-create name ACL02 scope fabric vlan 10
precedence 4 src-ip 10.0.31.128 src-ip-mask 255.255.255.250 dst-ip
192.168.1.144 dst-ip-mask 255.255.255.252
```

3. **vFlow ACL03**

- Filter: `VLAN 10`
- TCP port: `22`
- TCP flags: `ACK`
- Source IP address: `10.0.31.4`
- Destination IP address: `192.168.0.40`
- Forwarding Action: `none`
- Precedence: 4

```
CLI (network-admin@switch) > vflow-create name ACL03 scope fabric vlan 10
precedence 4 src-ip 10.0.31.4 dst-ip 192.168.0.40 proto tcp dst-port 22
tcp-flags ack
```

4. **vFlow ACL04**

- Filter: `VLAN 10`
- Forwarding Action: `drop`
- Precedence: 2 (default)

```
CLI (network-admin@switch) > vflow-create name ACL04 scope fabric vlan 10
action drop
```

For maximum ACL scalability, you can implement the whitelist policy using the System-VCAP table and provide additional capacity for vFlow objects based on the switch models.

In this approach, NetVisor OS examines Whitelisted traffic and labels the traffic with an arbitrary metadata value at the pre-ingress stage, without consuming resources in the default ingress table. During next ingress stage, NetVisor OS uses a single vFlow object with highest precedence to permit traffic already allowed at pre-ingress stage. You can also build the same Whitelist policy using both System-VCAP and the default System-L1-L4 tables as shown in ***Figure 12-10***.

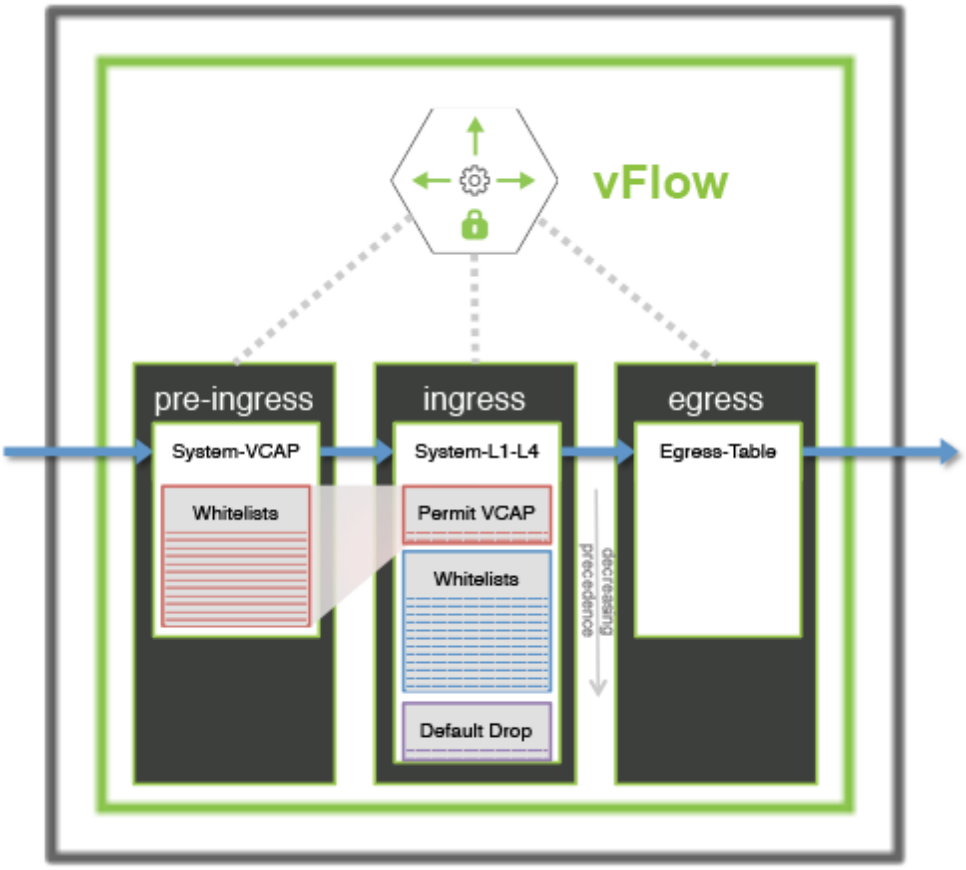


Figure 12-10: Scaling Whitelist ACLs Using a Pre-ingress Filter

Commands and Parameters Applicable to vFlow Traffic

The vFlow feature includes several commands and keyword parameters to configure and monitor various vFlow actions. Some of the important commands and parameters are explained in this section.

- To create a new vFlow object, use the command:

```
CLI (network-admin@switch-1) > vflow-create name <vflow-name> scope [local|
fabric] {parameters}
```

- To modify an existing vFlow object,

```
CLI (network-admin@switch-1) > vflow-modify name <vflow-name> {parameters}
```

- To delete an existing vFlow object,

```
CLI (network-admin@switch-1) > vflow-delete name <vflow-name>
```

- To display existing vFlow objects,

```
CLI (network-admin@switch-1) > vflow-show {parameters}
```

The key parameters required to create a vFlow object are categorized as:

- Scope (2)
  - local OR
  - fabric

- Tables (4)
  - table-name Egress-Table-1-0
  - table-name System-L1-L4-Tun-1-0
  - table-name System-VCAP-table-1-0
  - table-name VCAP-IPv6-table-1-0

- Match Conditions: Specify one or more match conditions:
  - vlan
  - inner-vlan
  - vnet
  - bd < bridge-domain name>
  - in-port <port>
  - out-port <port>
  - ether-type <type>
  - src-mac <mac> src-mac-mask <mask>
  - dst-mac <mac> dst-mac-mask<mask>
  - src-ip <ip> src-ip-mask <mask>
  - dst-ip <ip> dst-ip-mask <mask>
  - src-port <port> proto <tcp|udp>
  - dst-port <port> proto <tcp|udp>
  - src-port <port> src-port-mask <mask>
  - dst-port <port> dst-port-mask <mask>
  - src-port <port> src-port-end <port>

- dst-port <port> dst-port-end <port>
- dscp-start <start value> dscp-end <end value>
- tos-start <start value> tos-end <end value>
- tos <tos>
- src-vpg <name> dst-vpg <name>
- bidir-vpg-1 <name> bidir-vpg-2 <name>
- vlan-prio <802.1p priority>
- inner-vlan-pri <priority>
- vrf <name>
- ttl <ttl>
- proto <IP proto>
- tcp-flags <tcp control flags>
- ingress-tunnel <tunnel>
- egress-tunnel <tunnel>
- vrouter-name <vrouter>
- mirror <mirror-name>
- packet-log-max <1..50G>
- metadata <number>
- vxlan <vxlan-id>
- vxlan-ether-type <ether-type for VXLAN>
- stp-state <state>
- packet-res <packet resolution in ASIC>
- fwding-type <vlan|vxlan>
- udf-name1 <name> udf-data1 <data> udf-data1-mask <mask>
- udf-name2 <name> udf-data2 <data> udf-data2-mask <mask>
- udf-name3 <name> udf-data3 <data> udf-data3-mask <mask>
- if <mgmt|data>
- description <description-string>

Note:

- From NetVisor OS 6.1.0 onward, the parameters `src-mac` and `dst-mac` are supported in the System-VCAP table in addition to the System-L1-L4 table. This enhancement allows you to use these parameters while configuring the Network Packet Broker (NPB) solution.

- NetVisor OS version 6.1.0 introduces the `inner-vlan` parameter to support filtering of traffic based on the inner VLAN of a QinQ frame. This parameter is supported by the System-L1-L4 hardware table and can be configured as part of NPB deployments. You can set a metadata value for the NPB vFlow in System-VCAP table, and this value can be supplied along with `inner-vlan` parameter in another vFlow for filtration of NPB traffic based on inner VLAN ID.

- Special Actions
  - burst-size <size>
  - precedence <IP precedence>
  - process-mirror|no-process-mirror
  - log-stats|no-log-stats
  - from-tunnel-decap|no-from-tunnel-decap
  - log-stats stats-interval <sec> dur <sec>
  - set-src <ip> set-src-port <port>
  - set-dst <ip> set-dst-port <port>
  - dropped|no-dropped

    - log-stats|no-log-stats
    - transient|no-transient
    - enable|no-enable
    - transparency <enable|disable>
    - flow-cb default-cb|arp-cb|bcast-cb|igmp-cb|pim-cb|dhcp-cb|dhcpv6-cb|dmac-miss-cb|l2-miss-cb|no-cb

- Specific Actions with **Action** Keyword
    - action none
    - action drop
    - action to-port action-to-ports-value <port>
    - action to-cpu
    - action trap
    - action copy-to-cpu
    - action copy-to-port
    - action setvlan action-value <vlan>
    - action add-outer-vlan
    - action set-tpid
    - action to-port-set-vlan
    - action to-span
    - action set-metadata
    - action set-dscp
    - action set-dmac
    - action to-next-hop-ip action-to-next-hop-ip-value <next-hop-ip>
    - action set-dmac-to-port
    - action to-ports-and-cpu
    - action set-vlan-pri
    - action set-smac
    - action drop-cancel-trap
    - action to-ecmp-group action-to-ecmp-group-value <ecmp-group>
    - action redirect-to-vrouter
    - action strip-outer-vlan
    - action cancel-switch-to-cpu

- Flow Class
    - flow-class meter
    - flow-class guaranteed_bw
    - flow-class lossless
    - flow-class class0
    - flow-class class1
    - flow-class class2
    - flow-class class3
    - flow-class class4
    - flow-class class5
    - flow-class class6
    - flow-class class7
    - flow-class class8
    - flow-class control
    - flow-class control2
    - flow-class control3

Note: The flow classes control2 and control3 are available only on NRU01, NRU02, NRU03, and NRU03-

SFF platforms.

- Bandwidth parameters
    - bw-min <min>
    - bw-max <max>

Refreshing vFlow Level Statistics for Long-lived Connections

Prior to NetVisor OS 5.1.1 release, the connection analytics displayed the connection statistics (incoming bytes, outgoing bytes, total bytes, and the age of the connection) only after the connection is completed, which worked well for short-timed connections. However, for long-lived TCP connections, the connection statistics was unreliable. This was due to the fact that the parameters involved were calculated with reference to the TCP sequence numbers, which, for a long-lived connection, always wrapped around.

To eliminate inaccuracies in long-lived TCP connection statistics, the TCP data packets with sequence numbers that are about to wrap around are sent to the CPU. This is implemented by defining a new vflow rule (policy). NetVisor OS provides an option to enable this functionality through the `vflow-settings-modify` command. When the feature is active, the `connection-stats-show` and `connection-show` commands show accurate outputs for long-lived TCP connections.

Note:

- To enable *long-lived TCP connection statistics*, you must first enable the *user-defined-flow* knob.
- Disable *vxlan-analytics* before enabling the *long-lived TCP connection statistics* knob.
- You cannot enable *long-lived TCP connection statistics* knob if the *inflight-vxlan-analytics* is enabled or vice-versa.

Note: You must restart nvOSd when you enable or disable the long-lived TCP connection statistics knob.

Use the `vflow-settings-modify` command to enable long-lived TCP connection statistics:

```
CLI (network-admin@switch) > vflow-settings-modify
```

`vflow-settings-modify`	Use this command to update a user vflow setting.
`Specify one or more of the following options:`	
`enable-user-defined-flow|no-enable-user-defined-flow`	Specify to enable or disable the user defined flows. **Note**: You must enable the user-defined-flow before enabling the longlived tcp connection statistics.
`vxlan-analytics|no-vxlan-analytics`	Specify to enable or disable VXLAN analytics. **Note**: You must disable VXLAN analytics before enabling the longlived tcp connection
`inflight-vxlan-analytics|no-inflight-vxlan-analytics`	Specify to enable or disable the inflight VXLAN analytics. **Note**: You must disable inflight VXLAN analytics before enabling the longlived tcp connection.
`longlived-tcp-conn-stats|no-`	Specify to enable or disable the long-lived TCP

`longlived-tcp-conn-stats`	connection statistics.

For example, to enable the long-lived TCP connection statistics, use the commands below:

```
CLI (network-admin@Leaf1) > vflow-settings-modify enable-user-defined-flow

CLI (network-admin@Leaf1) > vflow-settings-modify no-vxlan-analytics no-
inflight-vxlan-analytics

CLI (network-admin@Leaf1) > vflow-settings-modify longlived-tcp-conn-stats
```

To view the user vflow settings, use the command `vflow-settings-show`. For example, after enabling long-lived TCP connection statistics, the typical output would be:

```
CLI (network-admin@Leaf1) > vflow-settings-show

enable-user-defined-flow:  on
vxlan-analytics:           off
inflight-vxlan-analytics:  off
longlived-tcp-conn-stats:  on
```

To view the connection statistics, use the show commands:

```
CLI (network-admin@Leaf1) > connection-stats-show

vlan     ip       port iconns    oconns ibytes obytes total-bytes
---- ------------ ---- --------- ------ ------ ------ -----------
100  132.10.3.152 32   402617119        813G   809G    1.58T
100  132.10.3.113 32   402439803        822G   818G    1.60T
100  132.10.3.191 32   402379008        828G   822G    1.61T
100  132.10.3.160 32   402531295        828G   824G    1.61T
100  132.10.3.147 32   402620992        833G   829G    1.62T
100  132.10.3.131 32   402466573        840G   836G    1.64T

CLI (network-admin@Leaf1) > connection-show

vlan   src-ip     dst-ip     dst-port cur-state latency obytes ibytes total-bytes age
---- ---------- ------------ -------- --------- ------- ------ ------ ----------- ---
100  132.10.3.2 132.10.3.127 http     fin       67.9us  198    188    386         1s
100  132.10.3.2 132.10.3.186 http     fin       62.3us  198    188    386         1s
100  132.10.3.2 132.10.3.153 http     fin       511us   198    188    386         1s
100  132.10.3.2 132.10.3.205 http     fin       66.4us  198    188    386         1s
100  132.10.3.1 132.10.3.160 http     fin       305us   198    188    386         1s
```

Configuring Ingress QoS Policing based on Internal Priority

Internal priority is an intermediary priority value that maps DSCP values to CoS values. NetVisor OS version 6.1.0 allows you to configure vFlows with internal priority as a filtering parameter. In effect, you can now perform traffic policing for any received traffic based on DSCP values. In earlier versions of NetVisor OS, you needed to configure at least one vFlow per port per DSCP value or DSCP value range. However, with NetVisor OS release 6.1.0, you need to configure only one vFlow per internal priority value.

The allowed range of internal priority values is between 0 and 7. For example, you can assign a maximum bandwidth limit of 100Mbps for packets with an internal priority of 1 by using the command:

```
CLI (network-admin@switch) > vflow-create name flow1 scope local internal-
pri 1 bw-max 100M
```

Display the configuration by using the command:

```
CLI (network-admin@switch) > vflow-show
name:          flow1
scope:         local
type:          vflow
in-port:
internal-pri:  1
bw-max:        100M
burst-size:    auto
precedence:    default
action:
packet-res:
fwding-type:
enable:        enable
table-name:     System-L1-L4-Tun-1-0
```

To clear the internal priority field for a vFlow, use the command:

```
CLI (network-admin@switch) > vflow-create name flow1 internal-pri none
```

Note:

- To achieve filtering of traffic based on the desired DSCP values, you must configure the internal priority value on the basis of the configured DSCP to CoS mapping. For more information, see the *Configuring DSCP to CoS Mapping* section of the *Configuring and Using vFlows* chapter.
- This feature does not support CoS values of 8 and 9 as these queues are used for internal control plane traffic.
- You can configure the `internal-pri` parameter in different hardware filter tables except `System-VCAP-table-1-0` and `VCAP-IPv6-table-1-0`.

Configuring Bridge Domain Aware vFlow

Starting from NetVisor OS release 6.1.0, you can create vFlows that accept bridge domains as a filtering parameter. This release also allows you to configure vFlows that use the direction of VXLAN traffic as a qualifier.

You can use the `bd` parameter in the `vflow-create` command to filter network traffic based on the bridge domain. For example, to configure a vFlow that drops packets that hits the ingress port 10 of bridge domain `bd1`, use the command:

```
CLI (network-admin@switch) > vflow-create name bd-vflow scope local bd bd1
in-port 10 action drop

CLI (network-admin@switch) > vflow-show name bd-vflow

switch   name      scope type   bd   in-port burst-size precedence action enable
-------  --------  ----- -----  ---  ------- ----------  ----------  ------ ------
switch   bd-vflow local vflow bd1 10       auto        default     drop   enable
```

Use the `in-port` and `from-tunnel-decap` parameters in the `vflow-create` command to filter traffic based on whether VXLAN traffic is entering a port or is decapsulated from a tunnel. For example, to configure a vFlow that copies VXLAN ingress traffic to the CPU, use the parameters `vxlan` and `in-port` in conjunction.

```
CLI (network-admin@switch) > vflow-create name vxlan-ingress-vflow scope
fabric vxlan 10100 in-port 25 action copy-to-cpu

CLI (network-admin@switch) > vflow-show name vxlan-decap-vflow layout
vertical
switch:             switch
name:               vxlan-ingress-vflow
scope:              fabric
type:               vflow
in-port:            25
burst-size:         auto
precedence:         8
action:             copy-to-cpu
vxlan:              10100
from-tunnel-decap:
enable:             enable
table-name:         System-L1-L4-Tun-1-0
```

Similarly, to configure a vFlow that filters decapsulated VXLAN tunnel traffic, use the `vxlan` and `from-tunnel-decap` parameters together:

```
CLI (network-admin@switch) > vflow-create name vxlan-decap-vflow scope
fabric vxlan 10101 from-tunnel-decap action drop

CLI (network-admin@switch) > vflow-show name vxlan-decap-vflow layout
vertical
switch:             switch
name:               vxlan-decap-vflow
```

```
scope:              fabric
type:               vflow
burst-size:         auto
precedence:         8
action:             drop
vxlan:              10101
from-tunnel-decap:  yes
enable:             enable
table-name:          System-L1-L4-Tun-1-0
```

Note:

- You can configure the `from-tunnel-decap` parameter only if the `vxlan` parameter is configured.
- The `from-tunnel-decap` and `in-port` parameters are mutually exclusive when used along with the `vxlan` parameter as these parameters pertain to opposite directions of traffic flow.

Guidelines and Limitations

When you configure the vFlow policies, it is recommended to follow the guidelines and limitation mentioned in this section.

Guideline: While specifying the `burst-size` in the `vflow-create` and `vflow-modify` commands, ensure that:

- The burst-size is large enough to handle the maximum transmission unit (MTU) size of the packets.
- The `burst-size` limit should not be set lower than 10 times the MTU size of the traffic on the interface to be policed.
  If the configured `burst-size` value is less than ten times the MTU size, then NetVisor OS takes the default burst-size value for configuration. If you configure the burst-value higher than ten times the MTU size, then the configured value is accepted.

Examples and Use Cases in vFlow

- Supporting TCP Parameters using vFlows
- Configuring Burst Size in vFlow for Maximum Bandwidth
- Configuring Bandwidth Sharing for a Single VLAN
- Enhancing the vFlow Capability to Match Forwarding Type and Packet Resolution in ASIC

Supporting TCP Parameters using vFlows

Packet Broker requires the ability to create flows based on TCP control bits in a packet. The commands, `vflow-create` and `vflow-modify` have a new option `tcp-flags`. The supported TCP control bits include FIN, SYN, RST, PUSH, ACK, and URG.

Setting the ACK bit is supported only if it is combined with other TCP bits such as SYN and FIN and not as a single parameter.

Only `to-port` and `mirror` actions are supported by vFlow with `tcp-flags` filter. The actions added for vFlows with `tcp-flags` configured are mirror-to-port.

If analytics is enabled, then `copy-to-cpu` are also applied on the same vFlow. Also, these flows are created with a precedence of 3 or above.

System vFlows are created with precedence 2 so that analytics can also work even with these vFlows.

To create a vFlow for the default system table, use the following syntax:

```
CLI (network-admin@Spine1) > vflow-create name Redirect-TCP-Reset tcp-flags RST action to-port

CLI (network-admin@Spine1) > vflow-create name Redirect-TCP-ECN-Capable tcp-flags ECN,RST action to-port

CLI (network-admin@Spine1) > vflow-create name Mirror-TCP-Finished tcp-flags FIN action mirror
```

You can use the `vflow-table-show` command to display vFlow tables:

```
CLI (network-admin@Spine1) > vflow-table-show format all layout vertical

switch: Spine1
name: Egress-Table-1-0
id: a0000d7:1
flow-max: 1024
flow-used: 0
flow-tbl-slices: 1
capability: match-metadata
flow-tbl-bank: Egress
flow-profile: system
switch: Spine1
name: Decap-Table-1-0
id: a0000d7:2
flow-max: 1024
flow-used: 0
flow-tbl-slices: 2
capability: none
flow-tbl-bank: Match-Metadata
flow-profile: vxlan
switch: tac-f64-sw5
name: OpenFlow-L2-L3-1-0
```

```
id:  a0000d7:3
flow-max:  1024
flow-used:  0
flow-tbl-slices:  7
capability:  none
flow-tbl-bank:  Match-Metadata
flow-profile:  openflow
```

Configuring Burst Size in vFlow for Maximum Bandwidth

The `vflow-create` and `vflow-modify` commands support a configurable burst-size parameter. This feature enables you to specify different burst-sizes for different types of metered traffic. For example, you can configure higher burst levels for a metered application that may produce bursty traffic patterns when you click on it, such as a media-rich Web page link.

This feature defaults to burst-size auto, which auto-calculates the burst size based on the maximum bandwidth settings for the vFlow. You can configure a burst-size number between 0 through 134MB.

The command syntax is:

```
CLI (network-admin@switch) > vflow-create name name-string scope local|
fabric in-port port-list bw-max bw-max-number burst-size number
```

For example, to create a vFlow with a burst size of 12 MB, use the following syntax:

```
CLI (network-admin@switch) > vflow-create name flow1 scope local in-port 12
bw-max 5G burst-size 12M
```

Configuring Bandwidth Sharing for a Single VLAN

In some instances, you may want to configure bandwidth sharing for a single VLAN with different IP addresses or subnets.

To do this, you must create a VRG with the required bandwidth:

```
CLI (network-admin@Leaf1) >  vrg-create name admin-vrg vlans 100 data-bw-
min 1g data-bw-max 2g scope fabric
```

You have now created a VRG with the guaranteed bandwidth of 1 Gbps and limited to a maximum of 2 Gbps. Now, create a vFLow for each IP address:

```
CLI (network-admin@Leaf1) >  vflow-create name vfl-1 scope fabric vlan 100
src-ip 1.1.1.1

CLI (network-admin@Leaf1) >  vflow-create name vfl-2 scope fabric vlan 100
src-ip 2.2.2.2

CLI (network-admin@Leaf1) >  vflow-create name vfl-3 scope fabric vlan 100
src-ip 3.3.3.3

CLI (network-admin@Leaf1) >  vflow-create name vfl-4 scope fabric vlan 100
src-ip 4.4.4.4
```

In this example, the specified IP addresses each have a guaranteed bandwidth between 1 Gbps and 2 Gbps.

If you want to specify a subnet, 100.100.100.0/28, and VLAN 53 with maximum bandwidth of 50 Mbps, use the following syntax:

```
CLI (network-admin@Leaf1) >  vrg-create name vrg-custom scope fabric data-
bw-min 50M data-bw-max 50M vlan 53

CLI (network-admin@Leaf1) >  vflow-create name vfl-cust scope fabric src-ip
100.100.100.0 src-ip-mask 255.255.255.240 vlan 53
```

However, later on, you found that sixteen IP addresses were not enough and you needed an additional 8 with the subnet, 101.101.101.8/29 that require the same bandwidth as the previous subnet. Use the following syntax:

```
CLI (network-admin@Leaf1) >  vflow-create name vfl-cust-2 scope fabric src-
ip 101.101.101.8 src-ip-mask 255.255.255.248 vlan 53
```

You now have two vFlows on VLAN 53.

Then, you discover that 50 Mbps is not sufficient to support the network traffic affected by the vFlow, and you want to upgrade to 80 Mbps:

```
CLI (network-admin@Leaf1) >  vrg-modify name vrg-custom data-bw-min 80M
data-bw-max 80M
```

Enhancing the vFlow Capability to Match Forwarding Type and Packet Resolution in ASIC

On certain platforms, where the VXLAN routing is supported using recirculation of packets by leveraging the vxlan-loopback-trunk parameter, the Layer 2 entries for route RMAC address, VRRP MAC address on VXLAN VLAN, or the Virtual Forwarder Interface (VFI) are programmed to point to vxlan-loopback-trunk ports in the hardware. As a result, any Layer 2 unicast packets destined for route RMAC address or the programmed VFIs do not reach the vRouter. NetVisor OS allows you to mitigate this problem by enabling you to create vFlow objects and specify the desired policy.

To create the vFlow object and to enable the match forwarding and packet resolution capability, use the command:

```
CLI (network-admin@switch) > vflow-create name <name-string> scope [local|
fabric] in-port <port-list> fwding-type [vlan|vxlan|vle] packet-res [l2-
unicast|l2-unknown-unicast|l2-multicast|l2-unknown-multicast|l2-broadcast]
action [none|drop|to-port|to-cpu|trap|copy-to-cpu|copy-to-port|setvlan|add-
outer-vlan|set-tpid|to-port-set-vlan|to-span|set-metadata|set-dscp|set-
dmac|to-next-hop-ip|set-dmac-to-port|to-ports-and-cpu|set-vlan-pri|set-
smac|drop-cancel-trap|to-ecmp-group|redirect-to-vrouter]
```

`name <name-string>`	Specify the name of the vFlow object.
`scope [local|fabric]`	Specify if the scope of the vFlow object is local or fabric
`in-port <port-list>`	Specify the incoming port for the vFlow object
`fwding-type [vlan|vxlan|vle]`	Specify the ASIC forwarding type
`packet-res [l2-unicast|l2-unknown-unicast|l2-multicast|l2-unknown-multicast|l2-broadcast]`	Specify the packet resolution in the ASIC
`action [none|drop|to-port|to-cpu|trap|copy-to-cpu|copy-to-port|setvlan|add-outer-vlan|set-tpid|to-port-set-vlan|to-span|set-metadata|set-dscp|set-dmac|to-next-hop-ip|set-dmac-to-port|to-ports-and-cpu|set-vlan-pri|set-smac|drop-cancel-trap|to-ecmp-group|redirect-to-vrouter]`	Specify the forwarding action to apply to the vFlow object

For example, to create a vFlow object: *vflow1*, scope: *local*, in-port number (port number of vxlan-loopback-trunk): *397*, with forwarding type: *vxlan*, packet resolution in ASIC as *l2-unicast* and forwarding action to be applied to the vFlow object as *redirect-to-vrouter*, use the command:

```
CLI (network-admin@switch) > vflow-create name vflow1 scope local in-port
397 fwding-type vxlan packet-res l2-unicast action redirect-to-vrouter
```

In the above example, port **397** is the port number of the vxlan-loopback-trunk and the action **redirect-to-vrouter** redirects the packets unmodified to data port.

To view the details, use the command:

```
CLI (network-admin@switch) > vflow-show

name   scope type  in-port burst-size precedence action             packet-res fwding-type enable table-name
----   ----- ----- ------- ---------- ---------- ----------------- ---------- ---------  ------ ----------------
vflow1 local vflow  397     auto       13       redirect-to-vrouter l2-unicast  vxlan      enable System-L1-L4-Tun-1-0
```

Using Application Flows and Statistics

Displaying Statistics for vFlows for all Switches in the Fabric

To create vFlows across the entire fabric, configure the vFlow with the scope `fabric` and stats `enable` option. By using these parameters, you can enable statistics for the flow on all switches that are members of the fabric and you can display the statistics for any switch in the fabric.

For example, to create a vFlow for VLAN1 with the scope fabric, use the following command:

```
CLI (network-admin@Leaf1) >  vflow-create name example_fabric_flow1 scope
fabric log-stats enable vlan 1
```

To display the statistics for the new vFlow for a switch in the fabric, use the following syntax:

```
CLI (network-admin@Leaf1) >  switch switch-name vflow-stats-show name
example_fabric_flow1
```

```
name                   packets bytes cpu-packets cpu-bytes
---------------------- ------- ----- ----------- ---------
example_fabric_flow1   51.4K     13.8M  50.1K               13.1M
```

If you omit the switch name, all vFlow statistics for the fabric are displayed.

```
name                 packets  bytes  cpu-packets cpu-bytes
-----------------    -------  -----  ----------- ---------
example_fabric_flow1  1.32K    305K   1.29K       291K
example_fabric_flow1  910      256K   884         243K
```

Understanding vFlow Statistics

The virtual network-based flows- vflows, display statistics for packet traffic flows on a switch and across the fabric. The vFlows are very powerful and provide many features such as quality of service (QoS), traffic shaping, packet redirect, drop actions, mirror, and capture.

A vFlow can be configured to store log statistics to a file accessible to clients using NFS and SFTP. If statistics logging is enabled, NetVisor OS periodically polls the switch for the most recent statistics for each flow and saves the statistics to an exported file. NetVisor OS also saves individual statistics received from other switches in the fabric and combines the statistics from all switches to record aggregate statistics for the entire fabric.

The switch consists of two components, the switch and the server. vFlows with operations such as drop executes within the switch component. Some vFlows operations for QoS take place in the switch component, while others operate within the co-processor by directing pertinent traffic to the co-processor.

There, the traffic is managed and then sent back to the switch component.Other actions such as copy-to-cpu sends the match traffic to the server component where the traffic is managed and then forwards packets for delivery. In general, the details are managed by NetVisor OS including fabric scope commands that cause all switches within a fabric to participate in an operation and then sends the compiled results to the CLI or to log files.

Before you can access the files, you must enable NFS or SFTP access to the log files by using the `admin-service-modify` command.

```
CLI (network-admin@switch-1) > vflow-share-show

vnet                            enable                          share-path
----                            ------                          ----------
fab1-global                     no                              switch-1://fab1-global
fab1-global                     no                              switch-1://fab1-global
fab1-global                     no                              switch-1:///fab1-global
fab1-global                     no                              switch-1://fab1-global
fab1-global                     no                              switch-1://fab1-global

CLI (network-admin@switch) > vflow-share-modify fab1-global enable

CLI (network-admin@switch) > vflow-share-show

vnet                            enable                          share-path
-----                           ------                          ----------
fab1-global                     yes                             switch-1://fab1-global
fab1-global                     yes                             switch-1://fab1-global
fab1-global                     no                              switch-1://fab1-global
fab1-global                     no                              switch-1://fab1-global
```

```
fab1-global                    no                              switch-1://fab1-global
```

You can then access the statistics log files using NFS in the following locations:

For the switch scope, the files are located in: `/net/switch-name//-name/flow/flow-name/switch/switch-name/stats`

For the fabric scope, the files are located in: `/net/switch-name//-name/flow/flow-name/fabric/stats`

To create a vFlow for example, Host-Agent-Discover, and measure statistics, enter the following command:

```
CLI (network-admin@switch) > vflow-create name Host-Agent-Discover scope local system
```

To view all vFlows currently tracked by the switch or fabric, use the vflow-show command:

```
CLI (network-admin@switch) > vflow-show

switch: pleiades24
name: Host-Agent-Discover
scope: local
type: system
dst-ip: 224.4.9.6
precedence: 2
action: copy-to-cpu
switch: pleiades24
name: DHCP-client
scope: local
type: system
in-port: 1-68
src-port: 68
proto: udp
precedence: 2
action: copy-to-cpu
switch: pleiades24
name: Host-Agent-Discover
scope: local
type: system
dst-ip: 224.4.9.6
precedence: 2
action: copy-to-cpu
switch: pleiades24
name: DHCP-client
scope: local
type: system
in-port: 1-68
src-port: 68
proto: udp
precedence: 2
action: copy-to-cpu
```

From the information displayed in the output, you can review the switch, the name of the vFlow, scope, type

of vFlow, destination IP address, precedence, and action for the vFlow.

To display statistics for all vFlows, use the `vflow-stats-show` command:

```
CLI (network-admin@switch) > vflow-stats-show

name                    packets      bytes   cpu-packets   cpu-bytes
---------- -------- ----- ----------- ---------
IGMP-Flow      368K              23.0M   392K                      23.0M
LLDP-Flow  82.9K    26.3M 82.9K       26.0M
Host-Agent 17.8K    1.11M 0           0
ECP                    0                    0             0
```

To monitor statistics of a vFlow and update every 10 seconds, use the following syntax:

```
CLI (network-admin@switch) > vflow-stats-show name flow1 show-diff-interval
10
```

To log persistent records of flow statistics, use the logging parameter and collect statistics every 10 seconds:

```
CLI (network-admin@switch) > vflow-create name monitor-flow scope local
ether-type arp stats log stats-interval 5
```

You can display the statistics logs for the new flow using the `vflow-stats-show` command.

> **Note:** Conflicting vFlows - Multiple vFlows can be active at once, but cannot apply them at the same time. You can use the precedence parameter to set the order of the vFlows. If you set the precedence to a higher value (0 - 10 with 0 as the lowest precedence), the vFlow has a higher precedence than those with lower values. If you are seeing error messages about vFlow conflicts, try adding a precedence value to new or existing vFlows.

Examples and Use Cases for Network Monitoring and Security

Using vFlows to Disable Communication for Security Monitoring

You can use vFlows to control the traffic by specifying the communications that are not allowed in a switch or a fabric. Use the following steps to create a vFlow as a firewall:

Define a VLAN and destination IP-based flow and specify that the flow is dropped by the switch, with statistics monitoring enabled:

```
CLI (network-admin@Leaf1) > vflow-create name vflow10 scope local vlan 99
dst-ip 172.168.24.1 action drop stats enable
```

Display the statistics for the new flow above as the traffic is dropped:

```
CLI (network-admin@Leaf1) > vflow-stats-show name vflow10 show-diff-
interval 5

switch    name     packets     bytes  cpu-packets  cpu-bytes
-------  ----    --------   -----   -----------   ---------
Leaf1     vflow10   864        116K         0           0
Leaf1     vflow10   5          936K         0           0
```

There are many options available for creating vFlows, and vFlows can be used to shape traffic, capture statistics, capture flow metadata, capture packets, or manage communications. The options include:

- vlan
- in-port
- out-port
- ether-type
- src-mac
- src-mac-mask
- dst-mac
- dst-mac-mask
- src-ip
- src-ip-mask
- dst-ip
- dst-ip-mask
- src-port
- dst-port
- dscp
- tos
- proto
- flow-class
- uplink-ports
- bw-min
- bw-max
- precedence
- action
- action-value
- no-mirror

- mirror
- no-process-mirror
- process-mirror
- packet-log-max
- stats
- stats-interval
- duration
- no-transient
- transient
- vxlan
- vxlan-ether-type

Configuring vFlows to Filter Packets on Management Interfaces

The Arista Networks switches support administrative services and protocols such as SSH, HTTP, SSL, ICMP, etc. (for all supported protocols, see the show command output below). Management vflow feature enables the use of IPTables to support filtering based on filter parameters on management interfaces traffic.
The management traffic on Arista switches are handled in two ways:

- Out-of band management interface traffic: Uses IPTables to perform kernel based filtering
- In-band management interface traffic: Uses vflow based programming approach

In dual stack networks, both IPv4 and IPv6 filters can be used on the management port (in-band/out-of-band). By default, the management traffic allows all SSH, NFS, SNMP traffic and denies all web traffic as displayed in the command output below:

```
CLI  (network-admin@spine1)  >  admin-service-show

if        ssh       nfs       web       web-ssl   web-ssl-port  web-port  snmp      net-api   icmp
----      ----      ----      -----     ------    ------------  --------  -----     -------   -----
mgmt      on        on        off       off       443           80        on        on        on
data      on        on        off       off       443           80        on        on        on
```

This feature uses the existing vflow commands to add filters on the out-of-band and in-band management interfaces that are specific for these administrative services. The vflow rules uses precedence numbering to maintain the order of filters and helps in enforcing rules at specific locations in the IPTables. However, when you configure vflow rules, make sure that the vflow rules do not have a conflict with the system rules because the system rules may take precedence over the user configured vflow rules.

While configuring the vflow rules, be aware of the following configuration considerations:

- The parameter if is used to configure management vflows.
- The vflow rules support only permit and drop actions.
- The order of the configuration aligns with the order in which the rules are programmed. However, the user can re-arrange the rules using precedence.
- The vflow rules take precedence in both IPTables and TCAM are:
- By default, the vflow rules have a precedence value of four (4).
- Implicit drop priority is always lower than the user configured management vflows
- IPTables filter is added such that it precedes the existing system rule.
- The following are the applicable scaling numbers:
- For in-band traffic: the egress TCAM table limitation of 256 entries or as per hardware limits.

- For out-of-band traffic: The IPTables scale limitation is applied.

For example, create a vflow with the following parameters, use the command:

```
CLI (network-admin@Spine1) > vflow-create name <mgmt_flow> if <mgmt|data>
scope <local|fabric> src-ip <IP> src-mask <MASK> dstip <IP> dst-mask <MASK>
proto <num_or_name> src-port <src-port-number> dst-port <dst-port-number>
action <permit|drop> precedence <num>
```

`name`	Name of the vFlow that you are creating
`if`	Specify the vflow administrative service as management or data
`scope`	Specify the scope as local or fabric
`src-ip`	Specify the source IP Address
`src-mask`	Specify the source IP address mask
`dstip`	Specify the destination IP address
`dst-mask`	Specify the destination IP mask
`proto`	Specify the name or number of the protocol
`src-port`	Specify the Layer 3 protocol source port for the vFlow
`dst-port`	Specify the Layer 3 protocol destination port for the vFlow
`action`	Specify the action, whether to drop the packet or allow/permit the flow of packet
`precedence`	Specify the traffic priority value. The default values range between 2 and 15.

To delete a vflow, use the command:

```
CLI (network-admin@spine1) > vflow-delete name <mgmt_flow>
```

To modify the vflow rule, use the command:

```
CLI (network-admin@spine1) > vflow-modify name <mgmt_flow>if <mgmt|data>
src-ip <IP> src-mask <MASK> dstip <IP> dst-mask <MASK>
proto <num_or_name> src-port <num> dst-port <num> action <permit|drop>
precedence <num>
```

To display the configured vflow rules from the IPTables, use the command:

```
CLI (network-admin@spine1) > vflow-mgmt-show name <string>
```

The following example displays an In-band filter configured in Egress Content Aware Processing (ECAP) TCAM on two IPV4 addresses, where the vflow filters are applied to block the ssh connection from the source IP address, 10.10.10.19 whereas the ssh connection is allowed from the IP address, 10.10.10.20:

```
CLI (network-admin@spine1) > switch-local vflow-show
```

```
name     scope   type    in-port  src-ip       dst-port  precedence  action   enable
fdata    local   vflow   73       10.10.10.20  22        4           none     enable
fdata1   local   vflow   73       10.10.10.19  22        4           drop     enable
```

```
tcp_22   local   vflow   73                    22         default      drop     enable
```

To display the examples for out-of-band management filters.

```
CLI  (network-admin@spine1)  >  vflow-mgmt-show
```

```
name          scope        type          src-ip       dst-port     precedence    action       enable
data1         local        iptable       153.1.1.120
                                         /255.255.25
                                         5.255        22           15                         enable
implicitv4_
drop_tcp_22   local        iptable                    22           15            drop         enable
_vmgmt0
mgmt_ipv4     local        iptable       2.1.1.1                   default       none         enable
implicitv4_
drop_icmp_v   local        iptable                    0            15            drop         enable
mgmt0
                                         2000::2/fff
mgmt1_ipv6    local        iptable       f:ffff:ffff                default       none         enable
                                         :ffff::
                                         2000::1/fff
mgmt_ipv6     local        iptable       f:ffff:ffff                default       none         enable
                                         :ffff::
implicitv6_
drop_ipv6-    local        iptable                    0            15            drop         enable
icmp_vmgmt0
```

To display the packets and byte count from the IPTables, use the command:

```
CLI  (network-admin@spine1)  >  vflow-mgmt-stats-show  name  <string>
```

```
CLI  (network-admin@spine1)  >  vflow-mgmt-stats-show
```

```
switch                 name                  pkts                  bytes
---------              -----------           ------                ---------
spine1                 data1                 0                     0
spine1                 implicitv4_drop_tcp_2
                       2_vmgmt0              16                    976
spine1                 mgmt_ipv4             0                     0
spine1                 implicitv4_drop_icmp_
                       vmgmt0                29                    2.38K
spine1                 mgmt1_ipv6            0                     0
spine13                mgmt_ipv6             0                     0
spine1                 implicitv6_drop_ipv6-
                       icmp_vmgmt0           0                     0
```

To clear all the IPTable rules, use the command:

```
CLI  (network-admin@spine1)  >  vflow-mgmt-stats-clear  name  <string>
```

The following example displays an In-band filter configured in Egress Content Aware Processing (ECAP) TCAM on two IPV4 addresses, where the vflow filters are applied to block the ssh connection from the source IP address, 10.10.10.19 whereas the ssh connection is allowed from the IP address, 10.10.10.20:

```
CLI (network-admin@spine1) > switch-local vflow-show
```

name	scope	type	in-port	src-ip	dst-port	precedence	action	enable	if
fdata	local	vflow	73	10.10.10.20	22	4	none	enable	data
fdata1	local	vflow	73	10.10.10.19	22	4	drop	enable	data
tcp_22	local	vflow	73		22	default	drop	enable	data

Configuring vFlow for Analytics

A vFlow can be used to capture packets for analysis, and you can determine if the vFlow captures packets across the fabric or on a single switch. Packets are captured by forwarding them from the data plane of the switch to the control plane.

Snooping only works if you use the parameters, `copy-to-cpu` or `to-cpu`.

The `copy-to-cpu` parameter ensures that the data plane forwards the packets and sends a copy to the CPU. Use this parameter if you want traffic to flow through the switch.

The `to-cpu` parameter doesn't forward packets and interrupts traffic on the switch. To snoop all application flow packets of protocol type TCP, enter the following CLI commands at the prompt:

```
CLI (network-admin@Leaf1) > vflow-create name snoop_all scope local proto
tcp action copy-to-cpu
```

Then use the following command to display the output:

```
CLI (network-admin@Leaf1) > vflow-snoop

switch: pleiades24, flow: snoop_all, port: 65, size: 66, time:
20:07:15.03867188
smac: 64:0e:94:28:00:fa, dmac: 64:0e:94:2c:00:7a, etype: ip
sip: 192.168.2.51, dip: 192.168.2.31, proto: tcp
sport: 42120, dport: 33399

switch: pleiades24, flow: snoop_all, port: 65, size: 184, time:
20:07:15.03882961
smac: 64:0e:94:28:00:fa, dmac: 64:0e:94:2c:00:7a, etype: ip
sip: 192.168.2.51, dip: 192.168.2.31, proto: tcp
sport: 42120, dport: 33399

switch: pleiades24, flow: snoop_all, port: 43, size: 66, time:
20:07:15.03893740
smac: 64:0e:94:2c:00:7a, dmac: 64:0e:94:28:00:fa, etype: ip
sip: 192.168.2.31, dip: 192.168.2.51, proto: tcp
sport: 33399, dport: 42120
```

Note: Use the `vflow-snoop` command only on platforms that do not have *rear-facing NICs*.

To restrict the flows captured to TCP port 22, SSH traffic, create the following vFlow:

```
CLI (network-admin@Leaf1) > vflow-create name snoop_ssh scope local action
copy-to-cpu src-port 22 proto tcp vflow-add-filter name snoop_ssh
```

Then use the `vflow-snoop` command to display the results:

```
switch: pleiades24, flow: snoop_ssh, port: 41, size: 230, time:
10:56:57.05785917 src-mac: 00:15:17:ea:f8:70, dst-mac: f4:6d:04:0e:77:60,
```

```
etype: ip src-ip: 10.9.11.18, dst-ip: 10.9.10.65, proto: tcp src-port: 22,
dst-port: 62356
switch: pleiades24, flow: snoop_ssh, port: 41, size: 118, time:
10:56:57.05922560 src-mac: 00:15:17:ea:f8:70, dst-mac: f4:6d:04:0e:77:60,
etype: ip src-ip: 10.9.11.18, dst-ip: 10.9.10.65, proto: tcp src-port: 22,
dst-port: 62356
```

The optional parameter `vflow-add-filter` restricts the output of the `vflow-snoop` command to the packets matching the snoop_ssh flow definition.

To capture traffic packets for a flow across the entire fabric, you create a flow with the scope of fabric:

```
CLI (network-admin@Leaf1) > vflow-create name fab_snoop_all scope fabric
action copy-to-cpu port 22
```

Support for IPv6 Addresses and vFlow Configurations

You must modify the vFlow table profile using the new command, `vflow-table-profile-modify`:

```
CLI (network-admin@Leaf1) > vflow-table-profile-modify profile ipv6 hw-tbl
switch-main
```

You must reboot the switch in order for the settings to take effect. To ensure that the profile is available after rebooting, use the `vflow-table-show` command:

```
CLI (network-admin@Leaf1) > vflow-table-show
```

name	flow-max-per-group	flow-used	flow-tbl-slices	capability	flow-profile
Egress-Table-1-0	256	0	2	match-metadata	system
Egress-Table-v6-1-0	256	0	1	none	egress-v6
IPv6-Table-1-0	1536	0	1	none	ipv6
System-L1-L4-Tun-1-0	1536	57	2	set-metadata	system
System-VCAP-table-1-0	512	1	1	none	system

vFlow Capability Changes in the Z9432F-ON Platforms

Starting from NetVisor OS release 7.0.1 the Z9432F-ON platforms are supported. They introduce a new generation of programmable ASICs, which has some differences in vFlow support.

The new ASIC architecture changes the forwarding pipelines introducing various processing stages that are mostly backward compatible with previous generations. Some enhancements and changes are introduced.

Regarding QoS, a new vFlow table is introduced called `Ingress-Policer-Table-1-0`. The `vflow-table-show` and `vflow-table-profile-show` commands also display the new table.

The `max-bw` action (for traffic policing) is supported in the new `Ingress-Policer-Table-1-0`, as well as in the `Egress-Table-1-0`. For example:

```
CLI (network-admin@switch) > vflow-create name band scope local in-port 129
bw-max 1M table-name  Ingress-Policer-Table-1-0 precedence 12

CLI (network-admin@switch) > vflow-show

name scope type  in-port bw-max precedence transparency tracking tracking-status enable table-name
---- ----- ----- ------- ------ ---------- ------------ -------- --------------- ------ --------------------
band local vflow 129     1M     12         disable      disable  disabled        enable Ingress-Policer-Table-1-0
```

Furthermore, the `set-dscp` action is not supported in the `Egress-Table-1-0` and `Egress-Table-v6-1-0` (to perform this action you can use the ingress tables). The action `set-vlan-pri` is not supported in `System-VCAP-Table-1-0`. Only the `Ingress-Policer-Table-1-0` and `System-L1-L4-Tun-1-0` tables support it.

Switch ports support 4 *dedicated queues for multicast, broadcast and unknown unicast* (BUM, in short) traffic. The default weight of each queue is:

```
CLI (network-admin@switch) > port-cos-weight-show
cos8-weight:   32
cos9-weight:   64
cos10-weight:  64
cos11-weight:  127
```

For BUM traffic CoS priorities from 0 to 4 are mapped to queue 8, CoS 5 is mapped to queue 9, CoS 6 is mapped to queue 10, and CoS 7 is mapped to queue 11.

Queues 8-11 can be configured like other queues: weights and bandwidth can be set and verified with the `port-cos-weight-modify/show` and `port-cos-bw-modify/show` commands, while the queue stats can be displayed with the `port-cos-stats-show` command.

The functionality of the port and vFlow commands has remained unchanged. The only difference is the addition of support for queues 8-11. For example, queue 8's bandwidth limit can be configured on port 1 like so:

```
CLI (network-admin@switch*) > port-cos-bw-modify port 1 cos 8 max-bw-limit
90
```

```
CLI (network-admin@switch*) > port-cos-bw-show cos 8,9,10,11

cos port                                 min-bw-guarantee max-bw-limit weight
---  ----------------------------------  ----------------  ------------  ------
8    3,5,7,9,13,15,17,19,21,25,29,33-37 0%                100%          32
8    1                                   0%                90%           32
9    1,3,5,7,9,13,15,17,19,21,25         0%                100%          64
10   1,3,5,7,9,13,15,17,19,21,25         0%                100%          64
11   1,3,5,7,9,13,15,17,19,21,25         0%                100%          127
```

Queue 8-11's stats can be displayed like so:

```
CLI (network-admin@switch) > port-cos-stats-show format port,cos8-obits,cos8-dropbits,cos9-obits,cos9-dropbits,cos10-obits,cos10-dropbits,cos11-
obits,cos11-dropbits,

port cos8-obits cos8-dropbits cos9-obits cos9-dropbits cos10-obits cos10-dropbits cos11-obits cos11-dropbits
---- ---------- ------------ ---------- ------------ ----------- ------------- ----------- -------------
1    11.3K      0            1.61K      0            0           0             1.22K       0
13   0          0            53.0K      0            0           0             0           0
97   2.20K      0            132K       0            0           0             11.0K       0
101  2.30K      0            140K       0            0           0             11.0K       0
113  11.7K      0            79.1K      0            0           0             0           0
125  0          0            3.80K      0            0           0             0           0
129  2.30K      0            137K       0            0           0             5.52K       0
```

Other vFlow functionality differences are

For details on the vFlow differences on Z9432F-ON Platforms, see the *GSG for Dell Z9432F Platforms*.

Configuring Quality of Service (QoS)

This chapter provides information about the configuration of the Quality of Service (QoS) features on a NetVisor OS switch using the NetVisor OS command line interface (CLI).

Understanding Quality of Service (QoS)

In computer networking, the term Quality of Service (QoS) refers to the use of techniques and technologies to ensure that different applications and/or devices can benefit from appropriate network guarantees to operate optimally, even in the presence of constrained bandwidth and network congestion.

QoS technologies can be used to optimize network performance, prevent service disruption, and provide protection to devices in the network. For example, they can be used to ensure that the traffic of mission-critical applications is prioritized and not disrupted during times of high network congestion.

QoS enables organizations to optimize their overall network utilization by applying different prioritization levels to the traffic of different applications, that is, to different traffic classes.

On the other hand, when QoS is not enabled, a network operates by delivering traffic on a best-effort basis, in other words all traffic is forwarded with the same priority and therefore has an equal chance of being delivered or of being dropped in case of congestion.

When QoS is enabled, traffic can be classified based on a number of criteria (such as certain packet fields or granular hardware-based policies). Once service classes are associated to the packets, they can be prioritized according to policies implemented by the network administrator that take into account the requirements of each traffic class. Some applications, for example, may be latency sensitive, other applications may not be very tolerant to packet drops, etc. Therefore, the network administrator can apply different prioritization policies to different types of traffic to optimize each application or protocol's behavior. For instance, certain protocols may be very sensitive to the effects of congestion, therefore the network administrator may elect to use QoS techniques for traffic congestion avoidance.

In a nutshell, implementing QoS in a network improves the predictability of the network performance (in terms of speed and traffic latency) and optimizes bandwidth utilization.

In general, the broad term 'QoS' refers to a set of techniques and mechanisms that include:

- **Port buffering**: consists of techniques for the allocation of temporary storage to packets during network congestion (i.e., when packets cannot be immediately forwarded) to avoid drops in the network choke points.
- **Port queuing**: refers to segregating traffic into different transmission queues based on one or more classification criteria.
- **Traffic classification and marking**: is the process of selecting traffic and associating it with a class value. Packets can carry a class or marking directly in their headers: 802.1Q-tagged packets carry an 802.1p Class of Service (CoS) priority field (see figure below), while IP packet headers include a differentiated services code point (DSCP) field. Traffic classification can be derived directly from the packet fields or, when those fields are missing or cannot be trusted, the traffic is (re)marked by the hardware based on some preconfigured policies. A per-port ingress trust attribute determines which classification behavior to apply to incoming packets.
- **Traffic scheduling**: packet buffers can be divided into multiple queues to which packets get scheduled based on their classification (for example, based on their CoS value).
- **Congestion avoidance**: consists of various techniques that aim to prevent or minimize downstream traffic congestion.
- **Traffic policing (or rate limiting**): is a technique to limit the speed of a flow by dropping the packets that exceed a configured threshold (in addition, re-marking can also be used to conditionally drop traffic in case of downstream congestion).
- **Traffic shaping**: is a technique to limit the speed of a flow by queuing up the packets that exceed a

configured threshold. This technology is more drop-averse and hence TCP friendly (compared to policing) but, when queuing up traffic, the latency of the buffered packets may be significantly increased (which is not always desirable).

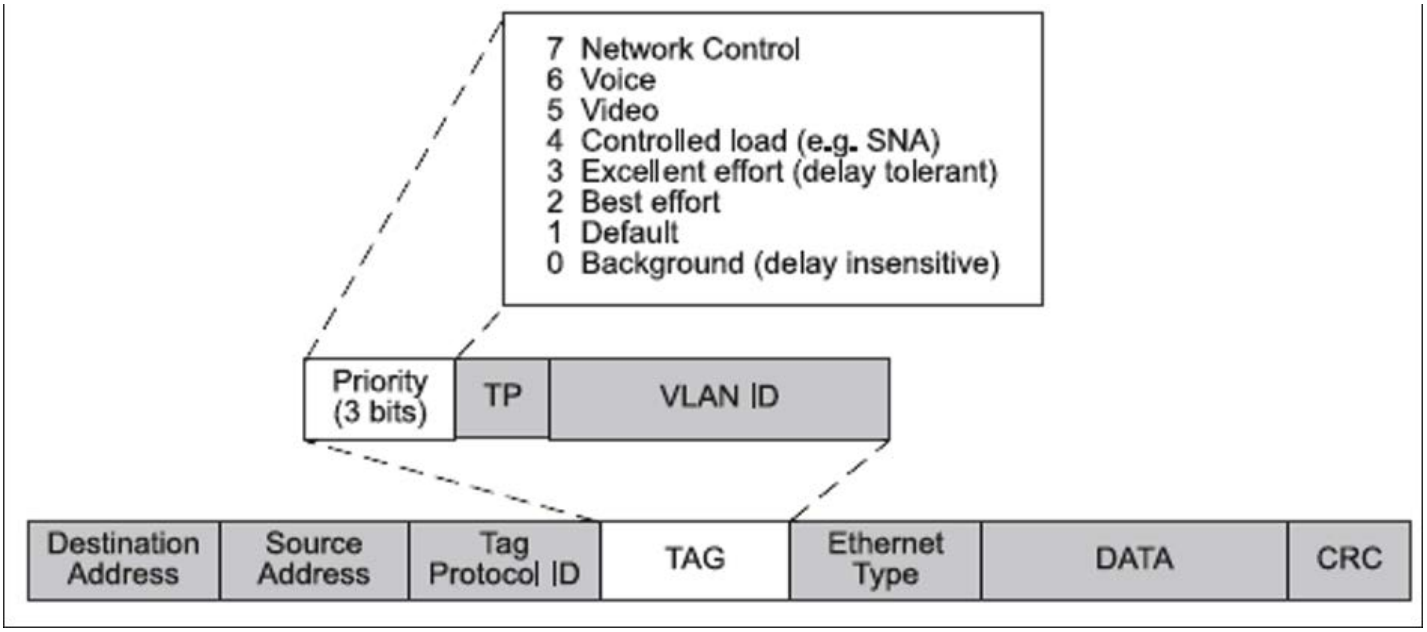


Figure 13-1: IEEE 802.1Q Header with Class of Service/Priority Field

QoS can logically be applied in ingress or in egress to a switch. The switch forwarding engine can also apply hardware policies to the traffic, for example to remark or police the traffic.

Layer 2 QoS revolves around 802.3 and 802.1Q technologies such as 802.1p class of service (CoS) and packet flow control. Hence, it can be applied to both IP and non-IP traffic.
Layer 3 QoS applies to IP traffic and can therefore leverage the differentiated services code point (DSCP) field (defined in RFC 2474), which for IPv4 replaced the outdated Type of Service (TOS) field (defined in RFC 791).

QoS can be applied to the data plane as well as to the control plane traffic. In the former case, it is used to optimize the traffic flow, the resource allocation, and the application behavior. It can also be used to protect certain classes of traffic, devices, or users.

When the device to be protected (for example, from denial-of-service attacks or network malfunctions) is the control plane CPU of a switch, then QoS can be very useful to ensure network stability and security (because QoS-based optimizations can be applied to benefit the CPU's computing and memory resources and to save CPU cycles).

Regarding QoS applied to control plane protection, Arista Network has implemented the Control Plane Traffic Protection (CPTP) feature. For more information about it, refer to the Configuring Control Plane Traffic Protection (CPTP) section of the Configuring Network Security chapter.

Understanding NetVisor Quality of Service (QoS) Technology

NetVisor switches follow Ethernet standards and hence can parse and extract the 802.1p Class of Service/Priority bits in the received frames. The CoS field represents the least common denominator form of packet marking available to Ethernet devices as it can be used for any type of traffic (whether IP or non-IP). Historically, Arista Networks has used the CoS field across different and heterogeneous generations of hardware. The packet CoS can be trusted or untrusted. When it's trusted, for simplicity's sake, CoS values have been mapped 1:1 to the queue numbers in the ASIC, as shown in the table below:

CoS Value	Output Queue Number
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7

For this reason, in the NetVisor OS CLI a queue number is sometimes referred to as 'CoS'.

For IP packets, NetVisor OS supports also classification based on the 6-bit DSCP field as defined in the IETF standards:

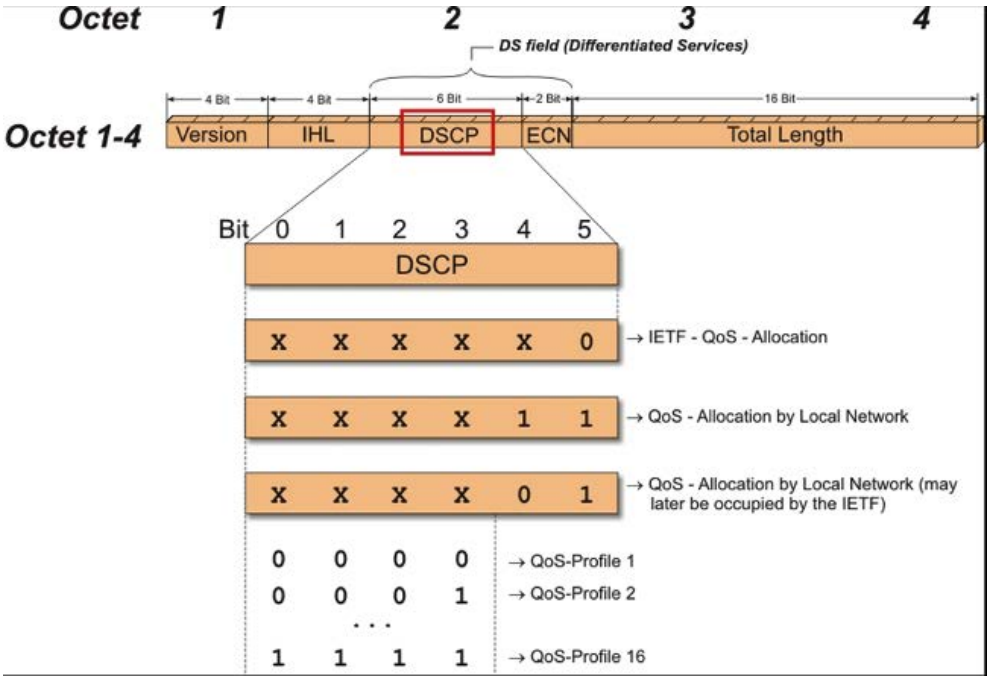


Figure 13-2: DSCP Field with QoS Profiles

Understanding QoS-based Packet Processing

A switch's packet processing logic is divided into three main stages:

- The ingress logic (corresponding to the port-related ingress functions)
- The centralized logic (corresponding to switch-wide processing and forwarding functions)
- The egress logic (corresponding to the port-related egress functions)

QoS is implemented in all of the above processing stages:

- In the logic associated to the ingress port, the input traffic's classification can be trusted, or it can be remarked per the network administrator's configuration.
- In the centralized logic, traffic can be filtered and QoS policies can be applied to it: traffic can be remarked and/or policed.
- In the logic associated to the egress port, traffic is scheduled for transmission in an egress queue, in which it can be shaped per the network administrator's configuration.

Figure 20-3 below shows the logical sequence of the QoS operations applied by the forwarding hardware to a packet:

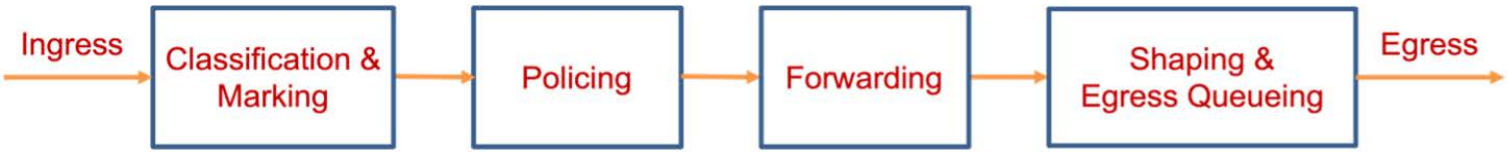


Figure 13-3: QoS Packet Flow

Understanding QoS Features

Let's look at the various QoS features available to the network administrator to optimize the traffic flows.

Port Trust Attribute and DSCP Map

A port's configuration includes an implicit trust attribute. Its default setting is to trust the priority marking (CoS) in the 802.1Q header to use internally to the switch for subsequent QoS functions (see the sequence above in Figure 20-3).

In addition to trusting the CoS field, it is possible to trust the DSCP field, when present in a packet. This functionality can be enabled by applying a DSCP map to a port.

A DSCP map specifies the associations between CoS values (called priorities) and DSCP values. These associations are used for queuing and for 802.1Q marking of the IP packets egressing the switch.

NetVisor OS supports interoperable DSCP-based markings:

- A DSCP value range from 0 to 63.
- Standard differentiated services definition: these values are used by different vendors to facilitate common QoS levels.
- Class selector code points (CS1 through CS7) are backwards compatible with IP Precedence values in the TOS field.
- Assured Forwarding (AF) code points have 4 priority classes, each class has three code points indicating the drop precedence:
    - Class1: AF11/12/13 (DSCP 10, 12, 14)
    - Class2: AF21/22/23 (DSCP 18, 20, 22)
    - Class3: AF31/32/33 (DSCP 26, 28, 30)
    - Class4: AF41/42/43 (DSCP 34, 36, 38)
- DSCP (and IP Precedence) 0 corresponds to the best effort behavior
- DSCP 46 is the Expedited Forwarding (EF) code point, indicating critical traffic.

Policing (Rate Limiting)

Policing is the function of limiting the speed of the forwarded traffic.

NetVisor OS leverages vFlow policies to select the subset of the traffic to apply a maximum forwarding rate to. It supports the token bucket algorithm to implement the rate metering and limiting function. The maximum rate is called `bw-max` and is expressed in bps. It can be in a range between 0 and 40 Gbps.

Policing can be configured with a vFlow policy on a per-port basis or on a per-VLAN basis.
Per-port policers can be applied to traffic ingressing (in-port) or egressing (out-port) a switch port. VLAN-based policers instead apply to the aggregate of the traffic from all the ports in a certain VLAN.
Other filtering parameters that can be used in a vFlow policy to apply even more granular policing include: DSCP or TOS field, source and/or destination MAC address, source and/or destination IP address, source and/or destination L4 port, etc.

For more information on vFlow policies, refer to the Configuring and Using vFlows chapter.

With a plain rate limiter, traffic that exceeds a specified threshold is dropped. However, this action is not very TCP-friendly, and in general it could be undesirable in certain circumstances. For example, it may be desirable for certain protocols to allow traffic bursting for optimal performance, even if a burst may temporarily exceed the configured rate.
Therefore, NetVisor OS supports policing by using the popular token bucket algorithm, which includes both a maximum rate and a token bucket size (burst size). This allows for TCP traffic bursts up to the maximum configured size, and therefore it can make a host TCP stack's behavior less "jittery" in case of traffic speeds being capped by the switch hardware.

Limiting traffic speeds in hardware (i.e., without any software overhead) can be used as a security measure too, typically to protect devices from Denial of Service (DoS) attacks (for specific destinations) or from potentially misbehaving sources. This security strategy can be critical to guarantee a deterministic behavior for devices that are particularly vulnerable to overloads or for mission critical ones that need to optimize their CPU utilization.

Of all the mission-critical entities in a network, the switch control planes play a very important role as they guarantee the stability and assist with the redundancy of the entire network. QoS is of great importance for the protection of the control plane and therefore historically network device vendors have created special QoS-based features to ensure network robustness and stability. As a case in point, NetVisor OS supports the Control Plane Traffic Protection (CPTP) feature.

For more information on the CPTP feature, refer to the Configuring Control Plane Traffic Protection (CPTP) section of the Configuring Network Security chapter.

Traffic Remarking

Traffic remarking is a useful hardware forwarding capability to change the prioritization of the traffic downstream. By default, forwarded traffic is not remarked (in other words, the QoS fields of the packets are not modified) but a policy can be configured to implement that.

In vFlow policies, along with forwarding actions (such as `copy-to-port` or `setvlan`), NetVisor OS also supports the `set-dscp` action that can be used to remark the traffic while not modifying its egress queuing. This is useful to signal to a downstream device that traffic should be treated differently based on the remarked DSCP value (for example, with a lower priority compared to the original one, according to the network administrator's requirements). As a matter of fact, it is not uncommon that host devices and network devices may not agree on the QoS markings to use. Or even different sections of the network, managed by different entities, may not have a complete agreement on the QoS markings to use. So remarking the traffic is used to apply the appropriate re-classification, when needed.

In addition, for example for non-IP traffic, it is possible to remark the frames by changing the priority/CoS field with the `set-vlan-pri` action.

In addition to `set-dscp`, NetVisor OS supports the dscp-map action to modify the egress port queuing based on the chosen remarked DSCP value. This type of action enables the administrator to select a different egress priority for the remarked traffic (for example, a lower priority compared to the original one). To apply the CoS/DSCP association on an egress port basis, NetVisor OS supports the `dscp-map` parameter in the `port-config-modify` command. The DSCP map associates each CoS value to one or more DSCP values: hence, for each remarked DSCP value in a forwarded IP packet the egress queue can be selected by the hardware based on the associated CoS value.

Queue Scheduling and Shaping

NetVisor OS supports the Weighted Round Robin (WRR) algorithm for traffic scheduling. On egress ports eight queues are available to schedule traffic to, based on the packets' CoS values. Network administrators can specify a minimum guaranteed bandwidth parameter, called `min-bw-guarantee`, as a percentage for each queue. They can also specify a per-queue scheduling weight (a value between 1 and 127) that is used by the hardware after the bandwidth guarantee is met. The queue number is referred to as "cos".

In addition to specifying a minimum bandwidth guarantee, it is also possible to configure a maximum bandwidth limit parameter, called `max-bw-limit`, on a per queue basis. This function is called shaping: it buffers the traffic that exceeds the configured limit up to the capacity of the queue.

Strict Priority Queue

As discussed above, NetVisor OS supports the WRR algorithm to schedule traffic out of port queues. However, traffic that is highly latency-sensitive and jitter-sensitive may not always work well with this type of scheduling, even when high weights are allocated to it. Therefore, for this kind of traffic, a special case (equivalent to an infinite weight) is supported and is called strict priority: NetVisor OS supports the weight priority queue configuration to enable special traffic to be scheduled for forwarding as soon as it's queued. This configuration minimizes forwarding latency and jitter, but it's generally reserved to low bandwidth protocols (for example, VoIP) that cannot monopolize the entire bandwidth of the port.

QoS References

For more details on QoS, you can also refer to the following IEEE and IETF standards:

- IEEE 802.1Q-2018 - IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks
- IEEE 802.1D-2004 - IEEE Standard for Local and metropolitan area networks: Media Access Control (MAC) Bridges
- RFC 791 “INTERNET PROTOCOL”
- RFC 795 “SERVICE MAPPINGS”
- RFC 1349 “Type of Service in the Internet Protocol Suite”
- RFC 2474 “Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers”
- RFC 2475 “An Architecture for Differentiated Services”
- RFC 2597 “Assured Forwarding PHB Group”

Guidelines and Limitations

These are some limitations to consider when configuring QoS:

- User-defined classification mappings for 802.1p CoS values are not supported. Currently NetVisor OS supports fixed 1-to-1 CoS-to-queue mappings.
- You can configure the internal-pri parameter in different hardware filter tables except for System-VCAP-table-1-0 and VCAP-IPv6-table-1-0.
- Once a DSCP map is used in a vFlow policy, it cannot be deleted until it remains in use.

Configuring QoS

The switch hardware's architecture and the corresponding QoS packet flow (described in Figure 20-3 above) determines the logical sequence and the types of configurations that can be utilized to achieve the desired results.

The network administrator should determine what kind of functionality is required and how that functionality should affect the traffic (TCP, UDP, non-IP, etc.). Therefore, advance knowledge of the user applications and their traffic patterns aids in the process of configuring QoS. For example, ingress traffic can already be marked appropriately by the upstream devices, so QoS policies can directly rely on such marking. On the other hand, some traffic may lack marking or may be marked differently from what is desired. In such case the network administrator needs to implement appropriate policies. Granular and accurate traffic classification is of paramount importance to optimizing traffic flows: for that purpose, vFlow policies can represent a powerful tool for selecting and (re)marking traffic according to the network design's requirements.

Below are the QoS functionalities described in the sequence from ingress to egress:

- Ingress Port Trust
- DSCP Map
- Policing (Rate Limiting)
- Traffic Remarking
- Port Buffering
- Queue Scheduling and Shaping
- Strict Priority Queuing

About Ingress Port Trust

By default, there is no configuration required for trusting the traffic's marking in ingress: the hardware uses the 802.1Q packets' CoS field for classification.

However, when the network administrator needs to trust the IP marking in the DSCP field, then trust can be changed to DSCP-based for IP traffic by configuring a DSCP to CoS mapping, as described in the next section. In such case, the CoS values associated to the trusted DCSP values also reflect in egress in the packets' CoS field.

Configuring DSCP to CoS Mapping

NetVisor OS supports creating hardware maps that associate Differentiated Services Code Point (DSCP) values in ingress in the received IP headers to Class of Service (CoS) priorities. This functionality also helps in prioritizing traffic based on DSCP markings by using the appropriate egress CoS-based queues to forward traffic out.

A DSCP map can be created, deleted and modified with the following commands:

```
CLI (network-admin@switch) > dscp-map-create
```

`dscp-map-create`	Create a DSCP priority mapping table with default DSCP to priority mappings.
`name name-string`	Create a name for the DSCP map.
`scope local|fabric`	Specifies the scope for DSCP map.

```
CLI (network-admin@Spine1) > dscp-map-delete
```

`dscp-map-delete`	Delete a DSCP priority mapping table.
`name name-string`	Name for the DSCP map.

```
CLI (network-admin@switch) > dscp-map-pri-map-modify
```

`dscp-map-pri-map-modify`	Update priority mappings in tables.
dscp-map selector:	
`name name-string`	Specify the name for the DSCP map to modify.
the following pri-map arguments:	
`pri number`	Specify a CoS priority from 0 to 7.
`dsmap number-list`	Specify a DSCP value(s) as a single value, comma separated list, or a number range.

To show the list of DSCP maps, you can use the command:

```
CLI (network-admin@switch) > dscp-map-show
```

`dscp-map-show`	Display a DSCP priority mapping table.
`name name-string`	Display the name of the DSCP map.
`scope local|fabric`	The scope for DSCP map.

For example:

```
CLI (network-admin@switch) > dscp-map-create name dscp-map1 scope fabric

CLI (network-admin@switch) > dscp-map-show name dscp-map1
name       scope
---------  ------
dscp-map1  fabric
```

The output of the command is empty if no maps have been created.

To show the associations in a DSCP map, you can use the command:

```
CLI (network-admin@switch) > dscp-map-pri-map-show
```

`dscp-map-pri-map-show`	Display priority mappings in tables.
dscp-map selector:	
`name name-string`	Display the name of the DSCP map
the following pri-map arguments:	
`pri number`	Display a CoS priority from 0 to 7.
`dsmap number-list`	Display a DSCP value(s) as a single value, comma separated list, or a number range.

The default values are listed in the following dscp-map-pri-map-show output:

```
CLI (network-admin@switch) > dscp-map-create name dscp-map1 scope fabric

CLI (network-admin@switch) >  dscp-map-pri-map-show name dscp-map1

switch name pri dsmap
------ ---- --- -----------
switch  ds2 0   none
switch  ds2 1   8,10,12,14
switch  ds2 2   16,18,20,22
switch  ds2 3   24,26,28,30
switch  ds2 4   32,34,36,38
switch  ds2 5   40
switch  ds2 6   48
switch  ds2 7   56
```

Furthermore, the `port-config-modify` command has a parameter, `dscp-map map-name|none`, to apply a DSCP map to a port. (Using the option none removes the DSCP map previously configured on the port.)

For example:

```
CLI (network-admin@switch) > port-config-modify port 17,21 dscp-map dscp-
map1
```

```
CLI (network-admin@switch) > port-config-show port 17,21 format
switch,intf,dscp-map,

switch   intf dscp-map
-------  ---- ---------
switch   17   dscp-map1
switch   21   dscp-map1
```

To modify a DSCP map that is in use, you can change its associations like so:

```
CLI (network-admin@switch) > dscp-map-pri-map-modify name dscp-map1 pri 5
dsmap 32

CLI (network-admin@switch) > dscp-map-pri-map-show
name       pri dsmap
---------  --- -----------
dscp-map1 0   40
dscp-map1 1    8,10,12,14
dscp-map1 2    16,18,20,22
dscp-map1 3    24,26,28,30
dscp-map1 4    34,36,38
dscp-map1 5   32
dscp-map1 6   48
dscp-map1 7   56
```

You can delete a DSCP map from a port like so:

```
CLI (network-admin@switch) > port-config-modify port 17,21 dscp-map none
```

Configuring Policing

Policing can be applied per port or globally (for example, on a per VLAN basis). Starting from NetVisor OS release 6.1.0, it can also be applied based on a 3-bit internal priority value used to categorize the traffic based on 8 possible classes.

Port-based Policing

You can configure a vFlow policy that applies to a port and specifies a maximum bandwidth (and optionally a token bucket size), for example like so:

```
CLI (network-admin@switch) > vflow-create name policer1 scope fabric in-
port 21 bw-max 0.4g

CLI (network-admin@switch) > port-stats-show port 21 show-diff-interval 1
format
port,ibytes,ibits,iUpkts,iBpkts,iMpkts,obytes,obits,oUpkts,oBpkts,oMpkts

port ibytes ibits iUpkts iBpkts iMpkts obytes obits oUpkts oBpkts oMpkts
---- ------ ----- ------ ------ ------ ------ ----- ------ ------ ------
21   47.4M  397M  5.38K  0      0      82.2K  673K  1.11K  0      0
21   85.2K  698K  1.15K  0      0      49.2M  412M  5.59K  0      0
...
```

VLAN-based Policing

You can configure a vFlow policy that applies to all the traffic in a VLAN and specifies a maximum bandwidth (and optionally a token bucket size), for example like so:

```
CLI (network-admin@switch) > vflow-create name policer1 scope fabric in-
port 21 bw-max 400m

CLI (network-admin@switch) > vflow-create name policer2 scope fabric vlan
10 bw-max 2g

CLI (network-admin@switch) > port-stats-show port 21 show-diff-interval 1
format    port,ibytes,ibits,iUpkts,iBpkts,iMpkts,obytes,obits,oUpkts

port ibytes ibits iUpkts iBpkts iMpkts obytes obits oUpkts
---- ------ ----- ------ ------ ------ ------ ----- ------
21   0      62.4M 970    0      0      0      51.0M 1.20K
...
```

Internal Priority-based Policing

Internal priority is an intermediary priority value that maps DSCP values to CoS values. NetVisor OS version 6.1.0 (or later release) allows you to configure vFlow policies using the internal priority as a filtering parameter. This means that you can now perform traffic policing for any received traffic based on DSCP values. In earlier versions of NetVisor OS you needed to configure at least one vFlow policy per port per DSCP value or DSCP value range. However, starting from NetVisor OS release 6.1.0, you need to configure only one vFlow per internal priority value.

The allowed range of internal priority values is 0-7. For example, you can assign a maximum bandwidth limit of 100 Mbps for packets with an internal priority value of 1 by using the command:

```
CLI (network-admin@switch) > vflow-create name flow1 scope local internal-
pri 1 bw-max 100M
```

You can display the configuration by using the command:

```
CLI (network-admin@switch) > vflow-show
name:          flow1
scope:         local
type:          vflow
in-port:
internal-pri:  1
bw-max:        100M
burst-size:    auto
precedence:    default
action:
packet-res:
fwding-type:
enable:        enable
table-name:     System-L1-L4-Tun-1-0
```

To clear the internal priority field for a vFlow entry, use the command:

```
CLI (network-admin@switch) > vflow-create name flow1 internal-pri none
```

Note: To achieve filtering of traffic based on the desired DSCP values, you must configure the internal priority value on the basis of the configured DSCP to CoS mapping. For more information, see the Configuring DSCP to CoS Mapping section of the Configuring and Using vFlows chapter.

Note: This feature does not support queue numbers/classes 8 and 9 as these queues are available exclusively for internal control plane traffic.

Note: You can configure the internal-pri parameter in different hardware filter tables except for System-VCAP-table-1-0 and VCAP-IPv6-table-1-0.

Configuring Burst Size in vFlow

The `vflow-create` and `vflow-modify` commands support a configurable burst-size parameter (i.e., the token bucket size). This feature enables you to specify different burst sizes for different types of metered traffic.
For example, you can configure higher burst levels for a metered application that may produce bursty traffic patterns when you click on it, such as a media-rich Web page link.

This feature defaults to burst-size auto, which auto-calculates the burst size based on the maximum bandwidth settings for the vFlow. You can configure a burst-size value between 256 bytes and 134 MB (128 MiB).

For example, to create a vFlow with a maximum bandwidth of 5 Gbps and a burst size of 12 MB, you can use the following command:

```
CLI (network-admin@switch) > vflow-create name flow1 scope local in-port 12
bw-max 5G burst-size 12M
```

Applying CoS Queue Mapping based on Re-Marked DSCP in vFlow

NetVisor OS allows a vFlow policy to mark or re-mark matched outgoing packets with a DSCP value. For this purpose, it supports the set-dscp action-value parameter.
For example, a vFlow policy called myflow can be modified to remark to DSCP 46 any matching traffic like so:

```
CLI (network-admin@switch) > vflow-modify name myflow action set-dscp
action-value 46
```

In this case, NetVisor OS does not modify the egress port queue selection for outgoing traffic.

As discussed above, NetVisor OS also allows you to create named DSCP maps as independent objects, and to apply the maps to ingress ports for classification of packets based on the DSCP markings.

In addition to that, you can apply the same DSCP maps in vFlow policies by using the dscp-map map-name parameter.

> **Note:** Before you can apply a DSCP map, you should verify that it's already created with the appropriate name. Once the map is used in a vFlow policy, it cannot be deleted until it remains in use.

By using the DSCP maps in vFlow policies, NetVisor OS can modify the egress port queue selection for outgoing traffic based on the DSCP-to-CoS associations configured by the network administrator.

You can specify the name of a DSCP map in the `vflow-create` command:

`dscp-map dscp-map name | none`	Specify the DSCP map to apply on the flow.

Please reapply if map priorities are updated.

For example, you can modify the aforementioned vFlow policy called myflow to affect egress queue selection by applying a (default) DSCP map like so:

```
CLI (network-admin@switch) > dscp-map-create name dscp-map1 scope fabric

CLI (network-admin@switch) > dscp-map-pri-map-show name dscp-map1
name       pri dsmap
---------  --- -----------
dscp-map1 0   none
dscp-map1 1    8,10,12,14
dscp-map1 2    16,18,20,22
dscp-map1 3    24,26,28,30
dscp-map1 4    32,34,36,38
dscp-map1 5   40
dscp-map1 6   48
dscp-map1 7   56
```

```
CLI (network-admin@switch) > vflow-modify name myflow dscp-map dscp-map1

CLI (network-admin@switch) > vflow-show name myflow format name,dscp-
map,precedence,action,action-value,enable,table-name

name    dscp-map   precedence action    action-value enable table-name

------  ---------  ----------  --------  ------------  ------
--------------------
myflow dscp-map1 default     set-dscp 32            enable System-L1-L4-Tun-
1-0
```

Configuring Port Buffering

Port buffering allocates storage for packets that cannot be immediately forwarded or processed on a port.

To display port buffer information (in percentages and bytes), use the port-buffer-show command. This command displays ingress and egress buffer utilization for each port:

```
CLI (network-admin@switch) > port-buffer-show
```

`port-buffer-show`	Display port buffer information.
`time date/time: yyyy-mm-ddThh:mm:ss`	Time to start statistics collection.
`start-time date/time: yyyy-mm-ddThh:mm:ss`	Start time of statistics collection
`end-time date/time: yyyy-mm-ddThh:mm:ss`	End time of statistics collection
`duration duration: #d#h#m#s`	Duration of statistics collection
`interval duration: #d#h#m#s`	Interval between statistics collection.
`since-start`	Specify to display port buffering since start.
`older-than duration: #d#h#m#s`	Specify an older-than time to display port buffering.
`within-last duration: #d#h#m#s`	Specify a within-last time to display port buffering.
`port port-list`	The list of ports.

```
CLI (network-admin@switch*) > port-buffer-show

port ingress-used-buf ingress-used-buf-val egress-used-buf egress-used-buf-val
---- --------------- ------------------- -------------- ------------------
5    2%              150K                0%             0
9    0%              0                   0%             0
13   0%              0                   0%             0
17   0%              22.5K               0%             0
21   0%              0                   0%             0
25   0%              0                   0%             0
29   0%              0                   0%             0
33   0%              0                   0%             0
34   0%              0                   0%             0
35   0%              0                   18%            1.12M
37   0%              0                   0%             0
41   0%              0                   0%             0
45   6%              405K                0%             0
49   0%              0                   0%             0
61   0%              0                   0%             0
69   0%              0                   0%             0
121  8%              567K                0%             0
129  0%              0                   0%             0
130  0%              0                   0%             0
```

Use the port-buffer-settings-modify command to enable the collection of port buffer information and to modify the logging interval and the internal storage space (called disk-space) allocated for the data collected for the command output above:

```
CLI (network-admin@switch) > port-buffer-settings-modify
```

`port-buffer-settings-modify`	Modify port buffer settings.
`enable|disable`	Specify if you want to enable or disable the collection of port buffer information.
`interval duration: #d#h#m#s`	Specify the interval for the logging of port buffer information.
`disk-space disk-space-number`	Specify the amount of memory to be allotted for the collection of port buffering information. The default value is 50MB.

For example, to set an interval of 2 minutes for logging port buffer information, use the command:

```
CLI (network-admin@switch) > port-buffer-settings-modify interval 2m
```

To display the port buffering settings, use the port-buffer-settings-show command:

```
CLI (network-admin@switch) > port-buffer-settings-show
switch: switch
enable: yes
interval: 2m
disk-space: 50M
```

Configuring Minimum and Maximum Port Bandwidths

NetVisor OS supports the configuration of a minimum bandwidth guarantee and of a maximum bandwidth limit at the egress queue level to manage prioritized traffic. You can use this feature to set up Service Level Agreements (SLAs).

You can configure bandwidth guarantees as a percentage of the port speed, and NetVisor OS determines the scheduling ratio internally upon command execution. Additionally, when you update a port speed, the port configuration internally re-adjusts the minimum or maximum bandwidths on the applicable ports. However, when you logically divide a breakout cable into multiple ports of lower bandwidth, you need to adjust the port queue bandwidths manually.

Use the port-cos-bw-modify command to configure minimum and maximum bandwidth on a particular port and on a queue number (referred to as 'cos'):

```
CLI (network-admin@switch) > port-cos-bw-modify
```

`port-cos-bw-modify`	Modify port CoS bandwidth settings.
`port port-list`	Specify the physical port(s).
`min-bw-guarantee min-bw-guarantee-string`	Specify the minimum bandwidth as a percentage.
`max-bw-limit max-bw-limit-string`	Specify the maximum bandwidth as a percentage.
`weight priority|no-priority`	Specify to enable or disable weight scheduling after the bandwidth guarantee is met.

For example, to configure a minimum bandwidth guarantee of 10 percent on ports 2-5 for queue 5, use the command:

```
CLI (network-admin@switch) > port-cos-bw-modify port 2-5 cos 5 min-bw-guarantee 10
```

Use the `port-cos-bw-show` command to check the configuration:

```
CLI (network-admin@switch) > port-cos-bw-show
```

`cos integer`	Specify the CoS priority between 0 and 11.
`port port-list`	Specify the physical port(s).

```
CLI (network-admin@switch) > port-cos-bw-show
switch  cos  port    min-bw-guarantee  max-bw-limit  weight
------  ---  ------  ----------------  ------------  ------
switch 0   1-72   0%                100%          16
switch 1   1-72   0%                100%          32
switch 2   1-72   0%                100%          32
switch 3   1-72   0%                100%          32
switch 4   1-72   0%                100%          32
```

```
switch 5   1,6-72 0%                100%          32
switch 5   2-5    10%               100%          32
switch 6   1-128  0%                100%          64
switch 7   1-128  0%                100%          127
```

In the weight column the above command displays the default weights for the queues (expressed by values in the range 1-127).

> **Note:** The `port-cos-bw-show` command displays only modified port configurations. Ports not displayed in the show command output default to the settings of 100% link capacity and no minimum bandwidth guarantee for each port queue.

Configuring a maximum bandwidth limit helps in shaping the traffic on the egress queue(s). For example, to configure a minimum bandwidth of 20% and a maximum bandwidth of 80% on ports 11-13 for queue 4, use the command:

```
CLI (network-admin@switch) > port-cos-bw-modify port 11-13 cos 4 min-bw-
guarantee 20 max-bw-limit 80

CLI (network-admin@switch) > port-cos-bw-show
switch    cos port    min-bw-guarantee max-bw-limit weight
--------- --- ------ ---------------- ------------ ------
switch    0   1-72   0%               100%         16
switch    1   1-72   0%               100%         32
switch    2   1-72   0%               100%         32
switch    3   1-72   0%               100%         32
switch    4   1-72   0%               100%         32
switch    4   11-13  20%              80%          32
switch    5   1,6-72 0%               100%         32
switch    5   2-5    10%              100%         32
switch    6   1-72   0%               100%         64
switch    7   1-72   0%               100%         127
```

Changing the port settings to new values overrides the previous settings.

Configuring Queue Weights

After a minimum bandwidth guarantee is met by the traffic on a queue, NetVisor OS supports per-queue scheduling weights for the remaining bandwidth. No user intervention is needed by default, as NetVisor OS automatically associates scheduling weights to port queues in accordance with the minimum bandwidth guarantees.

When you configure a minimum bandwidth guarantee without specifying a weight value, the weight for the port queue is set on a scale between 1 and 100 when auto-configuration is enabled. (In other words, instead of a binary value, a decimal one is used in the queue weights, which represents a non-zero percentage.)

If auto-configuration of weights is disabled, you can manually assign the desired weight values to the queues. When you configure queue weights manually, the bandwidth that remains after meeting the minimum bandwidth guarantee is proportionally divided among the queues in accordance with the assigned weights.

To enable or disable automatic weight assignment for port queues, use the command:

```
CLI (network-admin@switch) > system-settings-modify cosq-weight-auto|no-cosq-weight-auto

CLI (network-admin@switch) > system-settings-show format cosq-weight-auto,
cosq-weight-auto: off

CLI (network-admin@switch) > system-settings-modify cosq-weight-auto

CLI (network-admin@switch) > port-cos-bw-show
switch cos port   min-bw-guarantee max-bw-limit weight
------ --- -----  ---------------- ------------ ------
switch 0   1-128 0%                100%         1
switch 1   1-128 0%                100%         1
switch 2   1-128 20%               100%         20
switch 3   1-128 0%                100%         1
switch 4   1-128 40%               100%         40
switch 5   1-128 0%                100%         1
switch 6   1-128 0%                100%         1
switch 7   1-128 0%                100%         1
```

Note: If you enable auto-configuration of queue weights, you cannot change the weights using the port-cos-bw-modify command thereafter. You can still use the command to enable the strict priority queue (see below).

Note: When you enable auto-configuration, the weights for all the queues that do not have the minimum bandwidth configured are set to the default value of 1 (which is the minimum possible value, since zero is not allowed).

You can disable the auto-configuration function and then use the `port-cos-bw-modify` command to change the port queue weights:

```
CLI (network-admin@switch) > port-cos-bw-modify
```

`port-cos-bw-modify`	Modify port CoS bandwidth settings.
`cos integer`	Specify the CoS priority between 0 and 9. Note: queue 8 and 9 are platform specific.
`port port-list`	Specify the physical port(s).
`min-bw-guarantee min-bw-guarantee-string`	Specify the minimum bandwidth as a percentage.
`max-bw-limit max-bw-limit-string`	Specify the maximum bandwidth as a percentage.
`weight priority|no-priority`	Specify to enable or disable weight scheduling after the bandwidth guarantee is met.

For example:

```
CLI (network-admin@switch) > port-cos-bw-modify cos 5 port 10 weight 64
```

Configuring the Strict Priority Queue

It may be critical for certain types of traffic (e.g., voice traffic that is latency sensitive) to be scheduled for transmission as soon as they are enqueued. For this purpose, a queue can be designated as strict priority.

You can configure strict priority scheduling for any of the queues (for example, for queue 5 mapped to CoS 5-marked voice traffic). If strict priority is configured on a queue, NetVisor OS gives this queue the highest possible priority over the other queues, therefore the strict priority queue is typically reserved for relatively low bandwidth traffic (such as VoIP).

Strict priority scheduling is not enabled by default, it has to be configured explicitly by the user.

For example, to configure strict priority for queue 5 on port 2, use the following command:

```
CLI (network-admin@switch) > port-cos-bw-modify port 2 cos 5 weight
priority

CLI (network-admin@switch) > port-cos-bw-show port 2
switch       cos port min-bw-guarantee max-bw-limit weight
-----------  --- ---- ---------------- ------------ --------
<snip>
switch       5   2    0%               100%         priority
```

You can also configure a second queue as strict priority, for example, queue 6. In that case, those two queues are scheduled using equal-weight round robin. Using multiple strict priority queues is generally not recommended (as they take precedence over all the other queues), but may be useful in certain cases.

Displaying and Configuring Port Queue Statistics

You can view the port queue statistics by using the `port-cos-stats-show` command.

This command displays the number of bits of transmitted and dropped packets on a per queue basis.

```
CLI (network-admin@switch) > port-cos-stats-show
```

`port-cos-stats-show`	Display per port CoS queue statistics.
`time date/time: yyyy-mm-ddThh:mm:ss`	Time to start statistics collection.
`start-time date/time: yyyy-mm-ddThh:mm:ss`	Start time for statistics collection.
`end-time date/time: yyyy-mm-ddThh:mm:ss`	End time for statistics collection.
`duration: #d#h#m#s`	Duration of statistics collection.
`interval duration: #d#h#m#s`	Interval between statistics collection.
`since-start`	Displays statistics from the start of collection.
`older-than duration: #d#h#m#s`	Displays statistics older than a specified duration.
`within-last duration: #d#h#m#s`	Displays statistics within the last specified duration.

For example:

```
CLI (network-admin@switch) > port-cos-stats-show layout vertical
switch:          switch
time:            21:36:59
port:            21
cos0-obits:      0
cos0-dropbits:   0
cos1-obits:      0
cos1-dropbits:   0
cos2-obits:      0
cos2-dropbits:   0
cos3-obits:      0
cos3-dropbits:   0
cos4-obits:      0
cos4-dropbits:   0
cos5-obits:      0
cos5-dropbits:   0
cos6-obits:      0
cos6-dropbits:   0
cos7-obits:      0
cos7-dropbits:   0

CLI (network-admin@switch) > port-cos-stats-show format all layout
vertical
switch:          switch
```

```
time:          04:24:19
port:          73
cos0-obits:    0
cos0-dropbits: 0
cos0-opkts:    0
cos0-droppkts: 0
cos1-obits:    0
cos1-dropbits: 0
cos1-opkts:    0
cos1-droppkts: 0
cos2-obits:    0
cos2-dropbits: 0
cos2-opkts:    0
cos2-droppkts: 0
cos3-obits:    0
cos3-dropbits: 0
cos3-opkts:    0
cos3-droppkts: 0
cos4-obits:    0
cos4-dropbits: 0
cos4-opkts:    0
cos4-droppkts: 0
cos5-obits:    40.4K
cos5-dropbits: 0
cos5-opkts:    80
cos5-droppkts: 0
cos6-obits:    0
cos6-dropbits: 0
cos6-opkts:    0
cos6-droppkts: 0
cos7-obits:    624K
cos7-dropbits: 0
cos7-opkts:    434
cos7-droppkts: 0
```

To modify the port queue statistics' collection settings, use the command:

```
CLI  (network-admin@switch)  >  port-cos-stats-settings-modify
```

`port-cos-stats-settings-modify`	Modify port CoS statistics collection settings.
`enable|disable`	Specify if you want to enable or disable statistics collection.
`interval duration: #d#h#m#s`	Specify the interval to collect statistics.
`disk-space disk-space-number`	Specify the amount of memory allocated for statistics (including rotated log files). The default value is 50MB.

For example:

```
CLI (network-admin@switch) > port-cos-stats-settings-modify enable
```

To view the port queue statistics' collection settings, use the command:

```
CLI (network-admin@switch) > port-cos-stats-settings-show
switch:      switch
enable:      yes
interval:    30m
disk-space:  50M
```

You can clear the port queue statistics by using the port-cos-stats-clear command.

Supported Releases

Command/Parameter	NetVisor OS Version

`dscp-start`
`dscp-end`
`tos-start`
`tos-end`
`vlan-pri`
	vFlow parameters added in version 2.2.3

`set-dscp`	vFlow parameter added in version 2.2.5
`port-cos-rate-setting-modify` `port-cos-rate-setting-show`	Commands added in version 2.3.2

`port-cos-weight-modify`
`port-cos-weight-show`
`port-cos-stats-clear`
	Commands added in version 2.4.0

`port-cos-stats-settings-modify`	Command added in version 2.4.1

`dscp-map-create`
`dscp-map-delete`
`dscp-map-show`
`dscp-map-pri-map-modify`
`dscp-map-pri-map-show`
	Commands added in version 2.5.3

`port-cos-bw-modify` `port-cos-bw-show`	Parameters added in version 2.6.0
`dscp-map`	vFlow parameter added in version 2.6.2
`port-cos-stats-show` (output in bits)	Command added in version 3.0.0
`inner-vlan-pri` `internal-pri`	vFlow parameters added in version 6.1.0

Please also refer to the Arista NetVisor OS Command Reference document.

Related Documentation

For further information on concepts mentioned in this section, refer to these sections of the Configuration Guide:

- Configuring Switch Ports
- Configuring and Using vFlows
- Configuring Network Security

For further information on concepts mentioned in this section (such as switch ports, vFlow policies, CPTP, etc.)

Configuring Kubernetes Visibility

This chapter provides information about configuring the Kubernetes visibility feature on a NetVisor OS switch using the NetVisor OS Command Line Interface (CLI).

Understanding Kubernetes Visibility

Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications. A Kubernetes deployment creates a Kubernetes cluster which consists of at least one control plane node and a set of worker nodes that run containers. The worker nodes host the Pods that handle application workloads. The Kubernetes control plane administers the worker nodes and the Pods in a cluster. The core of Kubernetes' control plane is the API server through which users, different parts of the Kubernetes cluster, and external components interact.The Kubernetes API lets you query the state of objects in Kubernetes. For more information, see Kubernetes Documentation.

An increasing number of applications today use Kubernetes for container orchestration. The cloud-native application environments built using Kubernetes are highly distributed, and the majority of traffic consists of East-West flows between containers or pods. From a network perspective, there is very less visibility into these flows, which can make troubleshooting application performance issues difficult.

NetVisor OS version 7.0.0 introduces the Kubernetes visibility feature which gives you full visibility into East-West traffic flows between containers inside a Kubernetes cluster, without external network TAP infrastructure. This feature allows you to correlate containers with applications, to know on which hosts they reside and how they are connected to the fabric.

You can track and monitor containers (for information including IP, name, image, and location in the fabric) and Kubernetes cluster components (kube-controller and kube-scheduler). This granular information helps to perform root cause analysis of application availability and performance issues in a container environment. NetVisor OS implements a Kubernetes client that subscribes to the Kubernetes API server for real time notifications of events and changes.

By configuring the Kubernetes visibility feature, you can track historical data regarding a Kubernetes cluster which helps you to identify and fix availability issues even for containers that existed in the past. The feature lets you trace all available shortest paths between two Pods or nodes. For fine-grained Pod to Pod traffic analytics, you can also enable vPort table creation and connection statistics.

Configuring Kubernetes Visibility

The Kubernetes client in NetVisor OS connects to the Kubernetes API server for real time notifications of events and changes. To create a connection between the Kubernetes client in NetVisor OS and the Kubernetes API server, use the command:

```
CLI (network-admin@switch1) k8s-connection-create
```

`k8s-connection-create`	Create a new connection to the Kubernetes API server.
`cluster-name` *`cluster-name-string`*	Specify a name for the Kubernetes cluster.
`rest-user` *`user-name`*	Specify the username for vREST access.
`kube-config` *`kube-config-string`*	Specify the absolute path to the kubeconfig file. The default location is `/root/.kube/config`.
`password`	Specify the vREST password.
`enable|disable`	Enable or disable the connection to Kubernetes API server.
`track-history|no-track-history`	Enable or disable Kubernetes cluster history tracking.
`disk-space` *`disk-space-number`*	Specify the disk-space allocated for history (including rotated log files).

For example:

```
CLI (network-admin@switch) > k8s-connection-create cluster-name k8s01 kube-config /root/.kube/config
rest-user network-admin enable
rest-user password :
confirm rest-user password:
```

While configuring Kubernetes, please remember the following guidelines:

- NetVisor OS supports Kubernetes versions 1.19, 1.21, and 1.22.
- The Kubernetes client uses REST API to interact with NetVisor OS. Thus, you must enable web services on the data interface or the management interface of the switch by using the `admin-service-modify` command.
- You must enable LLDP on all the fabric nodes before configuring Kubernetes visibility. All Kubernetes servers must be visible in the `lldp-show` outputs of the fabric switches.
- To establish a connection with the Kubernetes API server succesfully, you must first copy the Kubernetes configuration file called kubeconfig from the Kubernetes control plane node to the switch. You can then specify the path to the file in the `k8s-connection-create` command.
- The Kubernetes API server must be reachable from linux global routing table and the API server's fully qualified domain name (FQDN) must be resolvable via nameserver.

Issue the `k8s-connection-show` command to display the Kubernetes connection information.

```
CLI (network-admin@switch1) k8s-connection-show
```

`k8s-connection-show`	Display the connection to the Kubernetes API server.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
`kube-config` *`kube-config-string`*	The absolute path to the kube config file.
`rest-user` *`user-name`*	The username for vRest access on switch.
`enable|disable`	Status of the connection to Kubernetes API server.
`track-history|no-track-history`	Enable or disable Kubernetes cluster history tracking.
`disk-space` *`disk-space-number`*	The disk-space allocated for history (including rotated log files).
`create-vport|no-create-vport`	Enable or disable vPort creation for Pod traffic.
`overlay-vxlan-analytics|no-overlay-vxlan-analytics`	Enable or disable Pod traffic analytics.
`location location-number`	Location where the connection is running.
`connection-error` *`connection-error-string`*	Kubernetes connection error.
`k8s-vlan` *`vlan-id`*	The VLAN ID used by Kubernetes for Pod vPort creation

```
CLI (network-admin@switch1) > k8s-connection-show format all layout
vertical
cluster-name:             k8s01
kube-config:              /root/.kube/config
rest-user:                network-admin
enable:                   yes
track-history:            yes
disk-space:               50M
create-vport:             yes
overlay-vxlan-analytics:  no
location:                 1262261009
```

Use the `k8s-connection-delete` and the `k8s-connection-modify` commands to delete or modify the configuration.

Displaying Kubernetes Cluster Information

A Kubernetes cluster consists of a control plane node and a set of worker nodes that run containerized applications. Every cluster runs at least one worker node. You can view the Kubernetes cluster information by using the `k8s-cluster-show` command.

```
CLI (network-admin@switch1) k8s-cluster-show
```

`k8s-cluster-show`	Display Kubernetes cluster information.

`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
`control-plane-node` *`control-plane-node-string`*	The name of the Kubernetes control plane node.
`control-plane-node-ip` *`control-plane-node-ip-string`*	The IP address of the control plane node.
`controller-status` *`controller-status-string`*	The status of the Kubernetes controller.
`scheduler-status` *`scheduler-status-string`*	The status of the Kubernetes scheduler.
`action none|add|remove|update`	The last action.

For example:

```
CLI (network-admin@switch1) > k8s-cluster-show

cluster-name  control-plane-node  control-plane-node-ip  controller-stauts  scheduler-status  last-changed
------------  -----------------  ---------------------  -----------------  ----------------  --------------
k8s01         server-test-93     192.168.10.10          healthy            healthy           07-12,23:25:21
```

Displaying Kubernetes Node Information

The control plane machine and the worker machines that are part of a Kubernetes cluster are called nodes. Nodes host the Pods that are the components of the application workload. You can display information on Kubernetes nodes by using the command:

`CLI (network-admin@switch1) k8s-node-show`

`k8s-node-show`	Display information on Kubernetes master and worker nodes.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
`node` *`node-string`*	The name of the Kubernetes node.
`role` *`role-string`*	The role assigned to the node.
`status` *`status-string`*	The status of the node.
`ip` *`ip-string`*	The IP address of the node.
`pod-CIDR` *`pod-CIDR-string`*	The Pod CIDR for the node.
`labels` *`labels-string`*	The Pod labels.
`action none|add|remove|update`	The last action performed.

```
CLI (network-admin@switch) k8s-node-show
cluster-name node            role          status       ip            pod-CIDR      last-changed
------------ --------------  ------------  -----------  ------------  ------------  --------------
k8s01         server-test-103 worker         KubeletReady 192.168.99.22 10.244.2.0/24 11-11,05:59:13
k8s01          server-test-93   control-plane KubeletReady 192.168.99.20 10.244.0.0/24 11-11,06:02:21
k8s01         server-test-94   worker         KubeletReady 192.168.99.21 10.244.1.0/24 11-11,05:58:36
```

Displaying Information on Kubernetes Pods

Pods are the smallest units that you can deploy using Kubernetes. Pods consist of one or more containers that are always scheduled together. Each Kubernetes pod has a unique IP address. To display information on Pods, use the command:

```
CLI (network-admin@switch1) k8s-pod-show
```

`k8s-pod-show`	Display Kubernetes Pod information.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
`name` *`name-string`*	The name of the Daemonset.
`namespace` *`namespace-string`*	The namespace.
`ip` *`ip-string`*	The IP address of the Pod.
`state` *`state-string`*	The state of the Pod.
`container-state` *`container-state-string`*	The state of the Pod's container.
`node` *`node-string`*	The name of the Pod.
`containers containers-string`	The container images running on the Pod.
`labels` *`labels-string`*	The Pod labels.
`action none|add|remove|update`	The last action performed.
`connections` *`connections-string`*	The connections to fabric switches.

For example:

```
CLI (network-admin@switch1) > k8s-pod-show format name,namespace,ip,state,node,containers,labels

name                                        namespace             ip             state   node
containers              labels        ----------------------------------------- -------------------- -------------
-------  --------------  -----------------  ---------
kube-apiserver-server-test-93                kube-system            192.168.99.20 Running server-test-93
k8s.gcr.io/kube-...    component:kube-apiserver,tier...
kube-proxy-d28w9                              kube-system           192.168.99.22 Running server-test-103
k8s.gcr.io/kube-...    controller-revision-hash:84f9...
kube-flannel-ds-ltctg                         kube-system           192.168.99.22 Running server-test-103
quay.io/coreos/f...    app:flannel,controller-revisi...
dashboard-metrics-scraper-79c5968bdc-hwpcg  kubernetes-dashboard  10.244.1.35     Running  server-test-94
kubernetesui/met...    k8s-app:dashboard-metrics-scr...
kube-scheduler-server-test-93                kube-system            192.168.99.20 Running server-test-93
k8s.gcr.io/kube-...    component:kube-scheduler,tier...
kube-controller-manager-server-test-93      kube-system            192.168.99.20 Running server-test-93
k8s.gcr.io/kube-...    component:kube-controller-man...
gloo-7dc68dd65b-clz98                         gloo-system           10.244.2.36    Running server-test-103
quay.io/solo-io/...    pod-template-hash:7dc68dd65b,...
```

Displaying Kubernetes Deployment Information

Deployments represent a set of multiple, identical Pods with no unique identities. A deployment runs multiple replicas of your application and automatically replaces any instances that fail or become unresponsive. In this way, deployments help ensure that one or more instances of your application are

available to serve user requests.

To display Kubernetes deployment information, use the command:

`CLI (network-admin@switch1) k8s-deployment-show`

`k8s-deployment-show`	Display Kubernetes deployment information.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes node.
`name` *`name-string`*	The name of the deployment.
`namespace` *`namespace-string`*	The name of the namespace.
`replicas` *`replicas-number`*	The total replicas of Pod.
`ready-replicas` *`ready-replicas-number`*	The ready replicas of Pod.
`selector` *`selector-string`*	The Pod label in deployment.
`action none|add|remove|update`	The last action performed.

For example:

```
CLI (network-admin@switch1) > k8s-deployment-show

cluster-name name                      namespace            replicas ready-replicas selector
          last-changed
------------ ------------------------- -------------------- -------- -------------
--------------------------------- ------------
pluribus     my-app                    default              1        1              app:my-app
          01:07:38
pluribus     pingtest                  default              2        2              app:pingtest
          01:07:38
pluribus     kubernetes-dashboard      kubernetes-dashboard 1        1              k8s-app:kubernetes-
dashboard      01:07:38
pluribus     dashboard-metrics-scraper kubernetes-dashboard 1        1              k8s-app:dashboard-
metrics-scraper 01:07:39
pluribus     hello-world               default              5        5              run:load-balancer-
example         01:07:39
pluribus     petstore                  default              1        1              app:petstore
          01:07:39
pluribus     discovery                 gloo-system          1        1              gloo:discovery
          01:07:39
pluribus     gloo                      gloo-system          1        1              gloo:gloo
          01:07:39
pluribus     ingress                   gloo-system          1        1              gloo:ingress
          01:07:39
pluribus     ingress-proxy             gloo-system          1        1              gloo:ingress-proxy
          01:07:39
pluribus     coredns                   kube-system          2        2              k8s-app:kube-dns
          01:07:39
```

Displaying Information on Daemonsets

A DaemonSet manages groups of replicated Pods and ensures that all (or some) nodes run a copy of a Pod. DaemonSets adds Pods to nodes that are added to a cluster and removes the Pods from nodes which are removed from the cluster. Deleting a DaemonSet also deletes the Pods that it created.To display information on DaemonSets, use the command:

`CLI (network-admin@switch1) k8s-daemonset-show`

`k8s-daemonset-show`	Display Kubernetes Daemonset information.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
`name` *`name-string`*	The name of the Daemonset.
`namespace` *`namespace-string`*	The name of the namespace.
`desired-cnt` *`desired-cnt-number`*	The desired number of Pods.
`ready-cnt` *`ready-cnt-number`*	The number of Pods ready.
`selector` *`selector-string`*	The Pod label selector for Daemonset.
`last-changed` *`date/time: yyyy-mm-ddTHH:mm:ss`*	The time when Daemonset was last changed.
`action none|add|remove|update`	The last action performed.

For example:

```
CLI (network-admin@switch) > k8s-daemonset-show
cluster-name name                   namespace      desired-cnt ready-cnt last-changed
------------ ---------------------- -------------- ----------- --------- ------------
k8s01        speaker                metallb-system 6           6         23:24:49
k8s01        csi-cephfsplugin       rook-ceph      6           6         23:24:50
k8s01        csi-rbdplugin          rook-ceph      6           6         23:24:50
k8s01        calico-node            kube-system    6           6         23:24:50
k8s01        fluentd-elasticsearch  kube-system    6           6         23:24:50
k8s01        kube-proxy             kube-system    6           6         23:24:50
k8s01        nodelocaldns           kube-system    6           6         23:24:50
k8s02        fluentd-elasticsearch  kube-system    6           6         23:25:22
k8s02        kube-flannel-ds        kube-system    6           6         23:25:22
k8s02        kube-proxy             kube-system    6           6         23:25:22
```

Displaying Kubernetes Services

Kubernetes services provide a stable virtual IP addresses for a group of Pods. Unlike Pods, which are non-permanent resources that can be created and destroyed dynamically, services remain consistent. Services act as loadbalancers and are based on Linux IPTable or IPVS table. Use the `k8s-service-show` command to view the Kubernetes services information.

`CLI (network-admin@switch1) k8s-service-show`

`k8s-service-show`	Display Kubernetes service information.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
`name` *`name-string`*	The name of the Daemonset.

`namespace` *`namespace-string`*	The name of the namespace.
`type` *`type-string`*	The service type.
`service-ip` *`service-ip-string`*	The IP address of the service.
`external-ip` *`external-ip-string`*	The external IP for services of type load balancer.
`selector` *`selector-string`*	The label selector for the service.
`ports` *`ports-string`*	The ports for the service.
`target-ports` *`target-ports-string`*	The ports targeted on Pods.
`action none|add|remove|update`	The last action performed.

For example:

```
CLI (network-admin@switch) > k8s-service-show layout vertical
cluster-name:  k8s02
name:          kubernetes
namespace:     default
type:          ClusterIP
service-ip:    10.96.0.1
ports:         443/TCP
last-changed:  01-30,23:25:21
cluster-name:  k8s02
name:          my-service
namespace:     default
type:          ClusterIP
service-ip:    10.98.20.142
ports:         71/TCP
last-changed:  01-30,23:25:21
cluster-name:  k8s02
name:          kube-dns
namespace:     kube-system
type:          ClusterIP
service-ip:    10.96.0.10
ports:         53/UDP,53/TCP,9153/TCP
last-changed:  01-30,23:25:23
...
```

Displaying Kubernetes Endpoints

To view information about Pods targeted by a service (Pod IP addresses) in a Kubernetes cluster, use the command:

`CLI (network-admin@switch) k8s-endpoint-show`

`k8s-endpoint-show`	Display Kubernetes endpoint information.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
`name` *`name-string`*	The name of the endpoint.
`namespace` *`namespace-string`*	Specify the namespace.
`selector` *`selector-string`*	Specify the label selector.

`target-ports` *`target-ports-string`*	Ports targeted on ports.
`endpoints` *`endpoints-string`*	The list of Pod IP addresses.
`action none|add|remove|update`	The last changed action.

```
CLI (network-admin@switch) > k8s-endpoint-show layout vertical
cluster-name:  k8s02
name:          kubernetes
namespace:     default
target-ports:  6443
last-changed:  01-30,23:25:21
cluster-name:  k8s02
name:          my-service
namespace:     default
target-ports:  9378
endpoints:     pending
last-changed:  01-30,23:25:21
cluster-name:  k8s02
name:          kube-dns
namespace:     kube-system
target-ports:  53,53,9153
endpoints:     172.22.0.39,172.22.1.8
last-changed:  01-30,23:25:23
...
```

Displaying Kubernetes Ingress Information

Kubernetes Ingress manages the external access to services within a cluster. Ingress exposes HTTP and HTTPS routes from outside the cluster to services within the cluster. Traffic routing is controlled by rules defined on the Ingress resource. An Ingress can give Services externally-reachable URLs, load balance traffic, terminate SSL/TLS, and offer name-based virtual hosting. To display Ingress information, use the command:

`CLI (network-admin@switch) k8s-ingress-show`

`k8s-ingress-show`	Display Kubernetes Ingress information.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
`name` *`name-string`*	The name of the endpoint.
`namespace` *`namespace-string`*	The namespace.
`host` *`host-string`*	Specify the hostname of the ingress.
`path` *`path-string`*	Path for the ingress rule.
`service-name` *`service-name-string`*	The backend service name.
`service-port` *`service-port-string`*	The backend service port.
`resource-api-group` *`resource-api-group-string`*	The backend resource API group
`resource-kind` *`resource-kind-string`*	The backend resource kind.
`resource-name` *`resource-name-string`*	The backend resource name.

`action`	`none|add|remove|update`	The last performed action.

For example:

```
CLI  (network-admin@switch)  >  k8s-ingress-show
cluster-name name                          namespace host              path service-name service-port last-
changed
------------  ------------------------  ---------  ---------------  ----  ------------  ------------
--------------
pluribus      petstore-ingress            default    gloo.example.com /.*   petstore      8080          07-
12,23:25:22
pluribus      name-virtual-host-ingress default    foo.bar.com       /     service1      80            07-
12,23:25:22
pluribus      name-virtual-host-ingress default    bar.foo.com       /     service2      80            07-
12,23:25:22
```

Displaying Kubernetes Connectivity information.

the Kubernetes visibility feature enables you to locate where Kubernetes Pods for specific applications are connected to the fabric to resolve any connectivity or performance issues. Use the `k8s-connectivity-show` command to view the connectivity information for a deployment, Daemonset, services or a Pod. Without any filter arguments, `k8s-connectivity-show` gives connectivity information for each Kubernetes node to the fabric.

`CLI (network-admin@switch) > k8s-connectivity-show`

`k8s-connectivity-show`	Display Kubernetes endpoint information.
`namespace` *`namespace-string`*	The namespace.
`deployment` *`deployment-string`*	The Kubernetes deployment.
`daemonset` *`daemonset-string`*	The Daemonset name.
`service` *`service-string`*	Kubernetes service name.
`pod` *`pod-string`*	The name of the Pod.
`node` *`node-string`*	The cluster node name.

```
CLI  (network-admin@switch)  >  k8s-connectivity-show  deployment  petstore
cluster-name namespace deployment pod                        connections
------------  ---------  ----------  ------------------------  ---------------------------------------
pluribus       default    petstore    petstore-9d499b76f-hhmg4  server-test-94:enp1s0f0  --  43:switch-1
pluribus       default    petstore    petstore-9d499b76f-psv8w  server-test-103:enp1s0f0  --  16:switch-1
                                                                server-test-94:enp1s0f1 -- 44:switch-1
pluribus       default    petstore    petstore-9d499b76f-qj58s  server-test-94:enp1s0f0  --  43:switch-1
                                                                 server-test-94:enp1s0f1 -- 44:switch-1
pluribus       default    petstore    petstore-9d499b76f-649g6  server-test-103:enp1s0f0  --  16:switch-1
```

The output displays the connectivity of a deployment named petstore which has 4 replicas. The output shows how each replica (Pod) is connected to Arista fabric. For example, the first replica (`petstore-9d499b76f-hhmg4`) running on `server-test-94` is connected through enp1s0f0 to port 43 of Arista switch `switch-1`.

Tracing Paths between Kubernetes Pods or Nodes

You can inspect all the paths that Kubernetes Pods or Kubernetes nodes use to communicate with each

other across the fabric. Use the command `k8s-trace-path` and specify the source and destination to view all available shortest paths between them.

`k8s-trace-paths`	Display the available paths between two Kubernetes Pods or nodes.
`src-pod`	The name of the Kubernetes source Pod.
`dst-pod`	The name of the Kubernetes destination Pod.
`src-node`	The name of the Kubernetes source node.
`dst-node`	The name of the Kubernetes destination node.

The example below shows all the shortest paths from the Pod `petstore` running on node `server-test-94` to Pod `ingress-proxy` running on node `server-test-103`. You can locate where Kubernetes Pods for specific applications are connected to the fabric to resolve any connectivity issues.

```
CLI (network-admin@switch) > k8s-trace-paths src-pod petstore dst-pod ingress-proxy
server-test-94:enp1s0f0 -- swp1:Leaf1:swp2 -- swp3:spine1:swp4 -- swp5:Leaf3:swp6 -- enp1s0f0:server-test-
103
server-test-94:enp1s0f0 -- swp1:Leaf1:swp2 -- swp3:spine2:swp4 -- swp5:Leaf3:swp6 -- enp1s0f0:server-test-
103
server-test-94:enp1s0f0 -- swp1:Leaf1:swp2 -- swp3:spine1:swp4 -- swp5:Leaf4:swp6 -- enp1s0f0:server-test-
103
server-test-94:enp1s0f0 -- swp1:Leaf1:swp2 -- swp3:spine1:swp4 -- swp5:Leaf3:swp7 -- swp7:Leaf4:swp6 --
enp1s0f0:server-test-103
```

Analyzing the Impact on Applications during Host or Leaf Replacement

The Kubernetes visibility feature allows you to proactively determine the impact on containerized applications during switch outages or server replacements. You can view the impact on a Kubernetes deployment, Daemonset, service, or Pods in the occasion of a switch or host replacement by using the `k8s-impact-show` command. The impact can be `none` if there is no impact, `partial` if alternate paths are present, or `full` if the Kubernetes Pods lose all connectivity.

`CLI (network-admin@switch) > k8s-impact-show`

`k8s-impact-show`	Display the available paths between two Kubernetes Pods or nodes.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
`host` *`host-string`*	The host or node for which the impact needs to be checked.
`switch` *`fabric-node-name`*	The switch for which the impact needs to be checked.
`namespace` *`namespace-string`*	The name of the namespace.
`deployment` *`deployment-string`*	The Kubernetes deployment name.
`daemonset` *`daemonset-string`*	The Kubernetes Daemonset name.
`service` *`service-string`*	The Kubernetes service name.
`pod` *`pod-string`*	The Kubernetes Pod.

`impact unknown|none|partial|full`	The impact on the Pod when node or switch is replaced.

For example, you can view the impact on a Kubernetes deployment named `petstore` upon replacing a server named `server-test-94` by using the command below:

```
CLI (network-admin@switch) > k8s-impact-show host server-test-94 deployment petstore
cluster-name namespace deployment pod                       impact  connections
------------ --------- ---------- ------------------------  ------  -------------------------------------
pluribus     default   petstore   petstore-9d499b76f-psv8w  none    server-test-103:enp1s0f0 -- 16:switch-1
pluribus     default   petstore   petstore-9d499b76f-hhmg4  full    server-test-94:enp1s0f0 -- 43:switch-1
                                                                   server-test-94:enp1s0f1 -- 44:switch-1
pluribus     default   petstore   petstore-9d499b76f-qj58s  full    server-test-94:enp1s0f0 -- 43:switch-1
                                                                   server-test-94:enp1s0f0 -- 43:switch-1
pluribus     default   petstore   petstore-9d499b76f-649g6  none    server-test-103:enp1s0f0 -- 16:switch-1
```

Displaying Kubernetes Network Information

Use the `k8s-network-show` command to show the relation between Kubernetes networking components like ingress, services, and Pods.

`k8s-network-show`	Display Kubernetes Pod information.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
Specify any of the following options:	
`namespace` *`namespace-string`*	The name of the namespace.
`ingress-name` *`ingress-name-string`*	The IP address of the Pod.
`host` *`host-string`*	The state of the Pod.
`type` *`type-string`*	The name of the Pod.
`external-ip` *`external-ip-string`*	The external IP address for services of type load balancer.
`node-port` *`node-port-string`*	The service type.
`service-name` *`service-name-string`*	The name of the service.
`selector` *`selector-string`*	The service selector to identify Pods.
`endpoints` *`endpoints-string`*	The list of Pod IP addresses.

For example, this command displays the ingress to Pods to services mapping:

```
CLI (network-admin@switch) > k8s-network-show layout vertical format all
cluster-name:  k8s01
namespace:     default
ingress-name:  demo
host:          www.demo.io
type:          LoadBalancer
external-ip:   10.13.22.154
node-port:     31437
service-name:  demo
selector:      app:demo
```

```
endpoints:     172.19.126.9,172.19.127.74
cluster-name:  k8s01
namespace:     default
ingress-name:  example-ingress
host:          example.com
type:          ExternalName
external-ip:   my.database.example.com
service-name:  my-service
...
```

Viewing Historical Information on Kubernetes Cluster

NetVisor OS implements a time machine feature that eliminates the need to reproduce past events that impacts Kubernetes applications in order to troubleshoot availability issues. This feature enables you to track historical data in a Kubernetes cluster. You can view various historical attributes of a cluster since the creation of a Kubernetes connection and inspect the changes to the cluster in a specific time interval.

Use the `k8s-connection-modify` command to enable history tracking and to specify the disk space for logging of historical data. For example:

```
CLI (network-admin@switch) > k8s-connection-modify cluster-name pluribus
track-history disk-space 60M
```

The Kubernetes commands that allow history tracking have a set of statistics collection parameters that allows you to specify the start time, end time, duration of the data collection and so on. Listed below are the commands that you can view historical data for:

```
k8s-cluster-show
k8s-node-show
k8s-deployment-show
k8s-service-show
k8s-ingress-show
k8s-pod-show
k8s-connectivity-show
k8s-impact-show
k8s-cni-show
```

The data collection parameters of the commands above which enable you to view the history in different formats are:

`start-time date/time: yyyy-mm-ddTHH:mm:ss`	The start time of statistics collection.
`end-time date/time: yyyy-mm-ddTHH:mm:ss`	The end time of statistics collection.
`duration duration: #d#h#m#s`	The duration of statistic collection.
`interval duration: #d#h#m#s`	The interval between statistic collection.
`since-start`	Statistics collected since start time.
`older-than duration: #d#h#m#s`	Display statistics older than the specified duration.

`within-last duration: #d#h#m#s`	Display statistics within the last specified duration.

For example, to display the changes to the deployment named `petstore` in the last 30 minutes, use the command:

```
CLI (network-admin@switch) > k8s-deployment-show name petstore within-last 30m
cluster-name name      namespace replicas ready-replicas selector      last-changed action
------------ -------- --------- -------- -------------- ------------ ------------ ------
pluribus     petstore default   4        4              app:petstore 09:11:51     add
pluribus     petstore default   6        4              app:petstore 09:24:35     update
pluribus     petstore default   6        5              app:petstore 09:24:45     update
pluribus     petstore default   6        6              app:petstore 09:24:45     update
```

Configuring Kubernetes Visibility through vPort Table and Connection Table

The Container Network Interface (CNI) is a proposed standard for configuring network interfaces for Linux application containers. A CNI plugin is responsible for inserting a network interface into the container network namespace. Kubernetes uses CNI as an interface between network providers and Kubernetes Pod networking. CNI plugins can implement their network either using encapsulated network model (overlay) such as VXLAN or an unencapsulated network model (underlay) such as BGP. Also, each plugin can operate in different modes to implement either overlay or underlay network types. For more information on Kubernetes networking and CNI plugins, see Kubernetes cluster networking documentation.

NetVisor OS version 7.0.0 uses the information from the Kubernetes client to detect the CNI configuration on the Kubernetes cluster to build vPort table and connection table for Pod to Pod traffic analytics. This feature currently supports only Flannel and Calico as CNI plugins. For more information on vPorts, see *Understanding Fabric Status Updates, vPorts and Keepalives.*

Use the `k8s-connection-modify` command to configure vPort table creation and overlay VXLAN analytics.

If you want to enable vPort table creation for an existing Kubernetes connection, use the command:

```
CLI (network-admin@switch) > k8s-connection-modify cluster-name k8s01
create-vport
```

To enable overlay VXLAN analytics, issue the command:

```
CLI (network-admin@switch) > k8s-connection-modify cluster-name k8s01
overlay-vxlan-analytics
```

Note:

- When you enable `overlay-vxlan-analytics`, you may need to reboot all the switches in the fabric or restart NetVisor OS on several nodes in the fabric.
- You cannot enable `overlay-vxlan-analytics` on more than one Kubernetes connection in a fabric.
- When you enable `overlay-vxlan-analytics`, NetVisor OS internally enables `inflight-vxlan-analytics`. For more information on `inflight-vxlan-analytics`, see *Configuring vFlows with User Defined Fields (UDFs).*

As an example, consider a Kubernetes environment illustrated in the figure below having Flannel as the CNI plugin and `host-gw` as the backend. The worker nodes server1 and server 2 host Pod A and Pod B respectively.

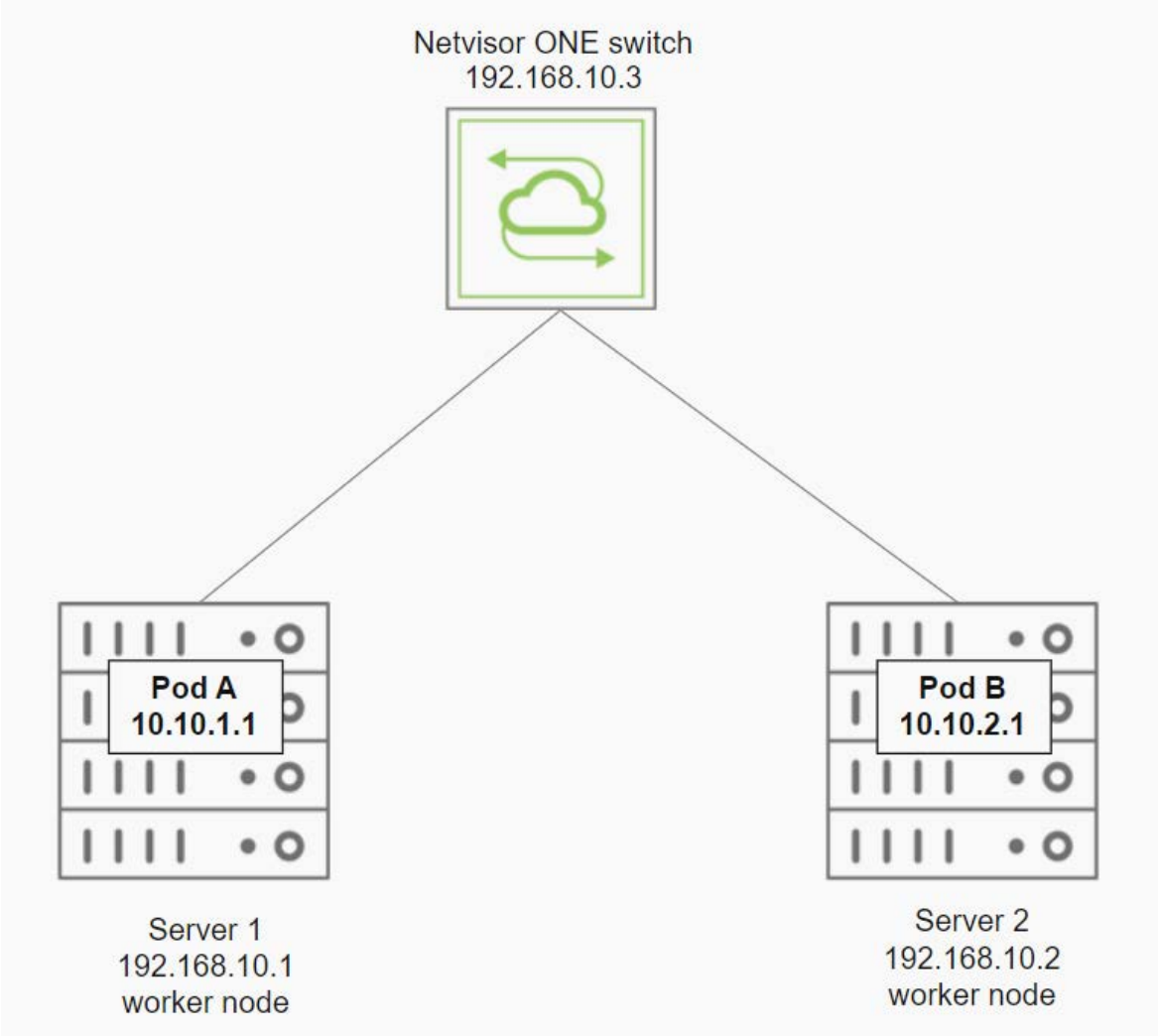


Figure 14-1: Pod to Pod Kubernetes Traffic Analytics

Configure the connection to the Kubernetes API server and enable vPort table creation by using the commands:

```
CLI (network-admin@switch) > k8s-connection-create cluster-name pluribus
kube-config /root/.kube/config enable

CLI (network-admin@switch) > k8s-connection-modify create-vport
```

The CNI information can be displayed by using the command:

```
CLI (network-admin@switch) > k8s-cni-show
```

`k8s-cni-show`	Display CNI information.
`cluster-name` *`cluster-name-string`*	The name of the Kubernetes cluster.
`start-time date/time: yyyy-mm-ddTHH:mm:ss`	The start time of statistics collection.
`end-time date/time: yyyy-mm-ddTHH:mm:ss`	The end time of statistics collection.
`duration duration: #d#h#m#s`	The duration of statistics collection.

`interval duration: #d#h#m#s`	The interval between statistic collection.
`since-start`	Statistics collected since start time.
`older-than duration: #d#h#m#s`	Display statistics older than the specified duration.
`within-last duration: #d#h#m#s`	Display statistics within the last specified duration.
`cni-plugin` *`cni-plugin-string`*	The CNI plugin name.
`backend` *`backend-string`*	The backend used by the CNI plugin.
`udp-port` *`udp-port-number`*	The VXLAN destination port.
`vxlan` *`0..16777215`*	The VXLAN ID for the vFlow.
`network-type` *`network-type-string`*	The network type implemented by the CNI plugin.

```
CLI (network-admin@switch) > k8s-cni-show
cluster-name  cni-plugin  backend  network-type
------------  ----------  -------  ------------
pluribus      flannel     host-gw  underlay
```

For the two Pods running on different worker nodes, the `vport-show` output is:

```
CLI (network-admin@switch) > vport-show svc-name pluribus
owner  mac               vlan ip        svc-name hostname entity power   portgroup       status
------ ----------------- ---- --------- -------- -------- ------ ------- --------------- ------------
switch 88:88:88:bf:5d:be 4090 10.10.2.1 pluribus server2   pod-A   Running app:hello-world host,pod,k8s
switch 88:88:88:bf:2e:ac 4090 10.10.1.1 pluribus server1   pod-B   Running app:hello-world host,pod,k8s
```

The vPort table is created statically and does not depend on the traffic that goes through the switch. Therefore the `mac` and `vlan` fields are populated by dummy values. The `portgroup` field displays the Pod label and `svc-name` field displays the cluster name. The `hostname` field represents the node where the Pod is running, `entity` represents the Pod name, and `power` is the Pod container's state.

Note: The vPort table is created locally on the switch where the Kubernetes connection is configured and will not be shared across other nodes in the fabric.

The `connection-show` command displays the connection information:

```
CLI (network-admin@switch) > connection-show

switch vlan src-ip    dst-ip    dst-port cur-state latency age
------ ---- -------- --------- -------- --------- ------- ---
switch 1    10.10.1.1 10.10.2.1 8080     syn-ack   4.01s   9s
```

In the case of a Kubernetes environment having Flannel as the CNI plugin with VXLAN backend, you can enable `overlay-vxlan-analytics` to view the connection statistics for inter-Pod traffic.

Guidelines and Limitations

- NetVisor OS supports only Flannel and Calico as CNI plugins for Kubernetes visibility through vPort and

connection tables.
- Overlay VXLAN anaytics with non-default VNI may not work for Calico CNI plugin.
- For the connection table to be populated, you must enable connection statistics on the switch:

```
CLI (network-admin@switch) > connection-stats-settings-modify enable
```

- If you change the CNI configuration, you must reboot the Kubernetes nodes. You must also restart the Kubernetes connection.
- The `k8s-connection-create` and `k8s-connection-modify` commands have `fabric` scope. You can configure up to four Kubernetes connections in a fabric.

Related Documentation

For further information on concepts mentioned in this section, refer to these sections in the Configuration Guide:

- Configuring and Using vFlows
- Configuring VXLAN
- Configuring and Admistering the Unified Cloud Fabric

Configuring Network Security

This chapter provides information about understanding and configuring network security features on a NetVisor OS switch using the NetVisor OS command line interface (CLI).

Understanding Network Security

In modern fast-changing data center environments *cyber-security* represents a crucial aspect that unfortunately is not always top-of-mind for IT professionals. Instead, it's oftentimes an afterthought.

As a consequence, cyber-security may be overlooked in the list of key investments to be budgeted to improve infrastructure and service resiliency.

The network infrastructure is often a prime target for malicious attacks because of the possibility of inflicting the most damage to as many devices as possible. Other reasons to target the network may be to attempt to redirect and snoop traffic to learn about clear-text information and find out possible other weaknesses that can lead to further malicious actions.

Both Internet- and Intranet-facing networks are vulnerable to malicious attacks, from internal and external sources. The news media provide ample documentary evidence about the nefarious effects of cyber-attacks on networks, servers and end-stations, oftentimes facilitated by well-known vulnerabilities left unpatched and exploitable. The total costs of such neglected vulnerabilities are in the millions of dollars on average, reaching in the billions in the case of the most widespread attacks.

As a general best practice, security should be considered an *indispensable day-one foundational component for any IT architecture*, meant to secure data and user access *end-to-end* and thus protect systems from tampering and unauthorized access.

Therefore, the *network plays a critical role* with regard to the hardening of the overall IT infrastructure, which is a complex system based on a multitude of interconnected sub-systems. Such complex system is only as robust as its *weakest element or link*, wherever that may be located. Therefore, contrary to popular belief, it's usually not sufficient to just secure any one individual component or even a limited portion of the network topology (for example, its perimeter) to achieve optimal security.

For instance, in case of enterprise networks, *firewalls* are very powerful security devices that are commonly deployed to harden the network's perimeter. However, many serious security breaches oftentimes come from within the network, after getting past the firewalls. Even the addition of sophisticated intrusion detection systems (IDS) may not completely avert those exploits that leverage the weakest components of the system.

In the following, we will describe a number of powerful networking technologies that can be employed to implement advanced end-to-end security measures so as to protect all nodes and interconnections from internal and external attacks.

About Network Security Threats

Before describing specific network security strategies, it can be convenient to go through a high-level classification of security threats, as listed below. Subsequently, by delving into the nature of a number of the most common attacks, it will be easier to understand how the respective countermeasures work and should be deployed.

The most common types of attacks tend to fall into a few main categories:

- Layer 2 Attacks: These attacks leverage vulnerabilities of the *data link layer*, also known as *Layer 2*, of the OSI networking model. Some Layer 2 protocols are not particularly robust (such as ARP) and hence are very frequently used as *attack vectors* (that is, means to perform higher-level attacks).

  For example, *ARP poisoning attacks* can be used to hijack a connection in one direction, or in both, to glean information from the traffic (hence they also fall into the category of *man-in-the-middle* (MitM) attacks). This kind of low-level attack vector can then be used to discover and exploit potential higher-level vulnerabilities.

  Other Layer 2 attacks are *DHCP spoofing attacks* (which use the DHCP protocol to impersonate a single host or even poison multiple hosts' address assignments), or *MAC flooding attacks* (which can be leveraged to cause potential traffic leaks and/or denial of service (DoS)).

  Layer 2 attacks typically occur only within the same broadcast domain/local subnet and hence can be contained by using appropriate fine-grained network segmentation techniques (e.g., granular VLANs). Outright prevention instead requires more specific technologies, as described in the following.

- Layer 3 and Higher Attacks: These attacks target vulnerabilities of the *OSI network layer* (also known as *Layer 3*) and higher layers (*Layer 4-7*). They typically leverage weaknesses in the IP protocol, ICMP protocol, or similar and at Layer 4 and higher they target the TCP or UDP ports and protocol behavior.

  The most well-known and simple attack is called *port scanning*, in which a worm or other malicious piece of software rapidly scans all open Layer 4 ports reachable in the network to find a vulnerable one that would allow it to spread to another device. In this category other serious attacks are: *TCP SYN floods*, *ICMP ping of death*, *IP address spoofing*, *DNS attacks*, etc.

  These attacks too can be somewhat contained by using appropriate segregation and filtering technologies, as well as by adopting protocol hardening schemes. A major help also comes from network analytics technologies, which allow the network admin to actively oversee the traffic, discover anomalies and promptly react to attacks.

- Denial of Service (DoS) and Distributed Denial of Service (DDoS) Attacks: These types of attacks seek to leverage any vulnerability of any network layer to produce a very severe impact on one or more services, to the point of degrading them so much that they would fail. DDoS attacks are typically performed by a large number of entities (for instance, a botnet of compromised hosts) to multiply the aggregate impact by a potentially huge (and often devastating) factor. They can sometimes be pure Layer 3 attacks (as in the case of *smurf attacks*, which leverage broadcast ICMP messages to trigger a bombardment of ICMP responses toward an attacked device) or they can be performed in conjunction with a Layer 3 attack such as *IP address spoofing*.

- **Unauthorized Access**: This type of sneaky vulnerability allows malicious (as well as non-malicious) users

to gain access to restricted resources through a valid but unchecked entry point or through a so-called backdoor.

Other types of attacks exist, and can be addressed with dedicated security services (like firewalling and intrusion prevention).This guide is not meant to be a comprehensive treatise on all of them.

Due to the *holistic* nature of network security, even the most advanced services have to be based on a *solid security foundation*, which is going to be described in the subsequent sections.

About Network Infrastructure Hardening

A network device typically comprises a general-purpose management CPU (or custom processor) and one or more forwarding engines. The former is in charge of the so-called *control plane*, which includes any network management function. The latter manage the high-speed switching and routing of packets, that is, the so-called *data plane* (also known as *forwarding plane*).

The management processor is in charge of communication exchanges with other networking devices as well as of interactions with a portion of the traffic coming from the rest of the network (that is, from the data plane). In general, all the traffic that is natively directed or purposefully redirected to such management processor is commonly referred to as *control plane*.

When the amount of any class or classes of traffic belonging to the control plane becomes abnormal (e.g., due to a DoS attempt), or the traffic itself is maliciously altered, then the network device needs to take some protective action, lest a portion (or even the entirety) of the network be severely affected.

In addition, a network device can apply security and quality of service (QoS) policies to the data plane traffic too in order to protect and/or optimize the traffic for any device in the network.

In other words, security and QoS data plane features can be considered as an additional tool for infrastructure hardening, with the goal of optimizing any traffic flows and blocking or limiting the malicious ones.

As a consequence, a full-blown network hardening strategy has to apply to both networking planes: control plane as well as data plane.

It should also be apparent how it is critically important to first start to harden the control plane of a networking device, and subsequently to leverage such hardened plane to securely apply traffic policies to the data plane too. Following this logical order, this chapter will start by describing the control plane hardening features first, to then move to cover any data plane ones.

In order to be able to take advantage of the latest security features and bug fixes, it is imperative to keep NetVisor OS up to date. Customers are encouraged to use the latest software release with the most recent security fixes. Arista Networks runs a vulnerability assessment on each NetVisor OS release and publishes security patches for any security issue. Software upgrades are signed and encrypted to prevent installation of rogue software. For more details on the upgrade procedure, refer to the *Upgrading the NetVisor OS Software* section.

About Basic Control Plane Security

Basic system security starts from system configuration and standard protocol best practices.

The main categories of best practices to start from are:

1. Limiting switch management access
2. Disabling unused services
3. Setting up a fabric password
4. Using secure versions of management and control protocols
5. Enabling system logging
6. Implementing user Authentication, Authorization and Accounting (AAA)

Limiting Switch Management Access

NetVisor OS is managed through a command line interface or a corresponding REST API. The CLI and API are equivalent in terms of functionality and both interfaces require a username and a password for user authentication.

Limiting switch access means setting up proper user credentials and privileges to implement strict access control policies based on essential/minimum access requirements.

Usernames and passwords may be managed locally on each network node, or globally on all nodes of the fabric. They can also be managed remotely using AAA servers. A user may have `read-write` or `read-only` access. See the *Configuring Users Accounts and Setting Credentials* for more details.

You can use the `user-create` command to create a user and assign him/her a password and an access policy. By default, the access policy is `read-write`. For users that do not need to configure the system, you can create a `read-only` role with the `role-create` command and then use it as part of the user creation command.

Furthermore, a user may have access to the entire system or may be constrained to managing only the configuration of a specific vNET.

vNET

A vNET allows the NetVisor OS administrator to delegate the management of a set of network resources to a tenant (*vNET administrator*). In the `vnet-create` command it is possible to specify the administrator for the vNET and his/her username.

Tenants have restricted access to NetVisor OS and can only see and manage resources assigned to their respective vNET thanks to the use of container technology (see the *Configuring Virtual Networks* section for more details).

SSH & SSHD

The NetVisor OS CLI may be accessed through the serial console or by using SSH. Non-secure protocols such as Telnet, rlogin, and FTP are not enabled.

In NetVisor OS version 6.1.1, a shell script is included in the */usr/bin* directory and ensures that the SSH and SSHD configurations in NetVisor OS are better compliant with Common Criteria requirements.

To setup a system for SSH conformance, the script available in */usr/bin/setup-cc-ssh-config.ksh* should be executed on the target switch. To run the script, as a root user, type:

```
root@switch:  /usr/bin/setup-cc-ssh-config.ksh
```

When you run this new script, the previously existing script, */etc/ssh/sshd_config* is saved to the */etc/ssh/sshd_config.save* directory.

Warning: Running this script removes any previously customized configuration for SSH and SSHD configurations.

Later, if desired, you can restore the earlier *sshd_config* by using the command as a root user:

/usr/bin/cp /etc/ssh/sshd_config.save /etc/ssh/sshd_config

Note: You must restart the SSH daemon service after restoring the earlier configuration.

When you upgrade to NetVisor OS version 6.1.1, the following changes are made to the configuration for a more secured SSH daemon. The changes are:

- Supported ciphers include:
  - AES256-CTR
  - AES256-CBC
- Supported set of Message Authentication Codes (MACs) include:
  - HMAC-SHA2-512
  - HMAC-SHA2-256
- The set of key exchange methods that are used to generate per-connection keys (KexAlgorithms) include:
  - ECDH-SHA2-NISTP521
  - ECDH-SHA2-NISTP384
  - ECDH-SHA2-NISTP256
- Supported set of HostKeyAlgorithms include:
  - ECDSA-SHA2-NISTP521
  - ECDSA-SHA2-NISTP384
  - ECDSA-SHA2-NISTP256
  - SSH-RSA
- SSH's RekeyLimit (the maximum amount of data that may be transmitted before the session key is renegotiated) is: 1 GbE for 3600 seconds
- LogLevel is set to VERBOSE
- The SSH variables KeyRegenerationInterval and ServerKeyBits are removed.

SFTP

Admins should only use the SFTP protocol to upload to or download from a NetVisor OS device. SFTP is

disabled by default, so it needs to be explicitly enabled and a password needs to be configured as shown in the *Enabling Administrative Services* section in the *Installing NetVisor OS and Initial Configuration* chapter.

REST API

REST APIs are accessed through a system's HTTP services, either globally for the entire switch or on a per-vNET basis. In particular, users should only use REST API *access over HTTPS* (*TLS*), which can be enabled with the `web-ssl` option as explained in the *Enabling Administrative Services* section.

TLS also requires you to configure a server certificate with the following steps:

1. Use the `web-cert-self-signed-create` command to create a private key and a self-signed certificate
2. Use the `web-cert-request-create` command to create a certificate signing request (CSR) using the public key from the self-signed certificate
3. Download the certificate signing request from the switch using SFTP
4. Get the certificate signing request signed by a certificate authority (CA)
5. Upload the signed certificate to the switch using SFTP
6. Use the `web-cert-import` command to import the signed certificate.

For more details refer to the *Managing NetVisor OS Certificates* and *Configuring REST API Access* sections in the *Installing NetVisor OS and Initial Configuration* chapter, and to the *Using OpenSSL TLS Certificates for OVSDB* section in the *Configuring Open Virtual Switch* chapter.

In addition, it's a best practice to include a warning message in a text banner to be displayed at login. Typically, this message includes some specific warning about monitoring and logging of user activities in compliance with local regulations. A banner can also be used to inform about access limits and other restrictions imposed on the users.

For example, the command `switch-setup-modify banner banner-string` can be used to configure a simple banner to warn unauthorized users, like so:

```
* Welcome to Arista Networks Inc. Netvisor(R). This is a monitored system.

*

* ACCESS RESTRICTED TO AUTHORIZED USERS ONLY *
```

In some legal jurisdictions it may not be possible to prosecute and may be illegal to monitor malicious users unless they have been notified that they are not permitted to use the system. One common method to provide such notification is to place that information into a properly crafted banner message (as shown above).

Legal notification requirements are complex, vary by jurisdiction and situation, and therefore should be discussed with a legal counsel. For example, a banner text can be put together to provide some or all of the following information:

- A notice that the system can be logged in to and used only by authorized personnel
- Carefully chosen information about who can authorize use of the system
- A warning that any unauthorized use of the system is unlawful and can be subject to civil and criminal penalties

- A notice that any use of the system can be logged and monitored without further notice and that the resulting logs can be used as evidence in court.

Plus, additional notices may be required by local laws.
However, a login banner should only contain legal text. It should not contain information that can be exploited by a malicious user. From a security point of view, a login banner should not contain specific information about the device such as name, model, software version, ownership, or any other information that may be abused by malicious users.

Disabling Unused Services

It is a common hardening best practice for any system to disable any service that is unused or that will remain unused for an extended period of time (it can always be enabled on demand when necessary).

To achieve this with basic NetVisor OS administrative services, on a local or fabric scope, you can use the `admin-service-modify` command to disable any unused services on the management interface and/or on the front panel ports.

In most cases the default settings are okay, but they are not always optimal. For example, if not needed, SNMP can be disabled on both management and front panel ports. Also, unencrypted HTTP services can be disabled (also on the management interface) after the `web-ssl` services have been set up and tested by the admin.

Setting up a Fabric Password

Within a fabric NetVisor OS switches communicate with each other using the TCP protocol to exchange status messages, perform fabric transactions, make remote procedure calls and proxy CLI commands between fabric nodes.

Fabric communication over TCP is secured using TLS with mutual certificate authentication, with an additional bi-directional challenge/response where each side proves it knows the fabric password by computing an HMAC SHA256 digest of a nonce from the other side along with the fabric password and the TLS session key. The certificates used are provisioned during manufacturing and do not need to be configured by the customer.

A fabric password can be set up by using the `fabric-create password` command, as explained in the *Creating an Initial Fabric* section. You can only create a password when you create a fabric for the first time. After you create a password, a password is required to add a new node to a fabric.

Using Secure Versions of Management and Control Protocols

As explained above, REST API access can be configured to use TLS encryption and authentication.

If SNMP is also required by the admin (for example, for interoperability with SNMP-based tools), version 3 (SNMPv3) can be enabled to secure the communication as described in RFC 3410, RFC 3411, RFC 3412, RFC 3413, RFC 3414 and RFC 3415. SNMPv3 provides secure access to devices by authenticating and optionally encrypting SNMP messages over the network.

SNMPv3's user creation process through the `snmp-user-create` command provides three

configuration options, of which only one provides full privacy:

- `no-auth` — This mode does not implement any authentication or encryption of SNMP packets
- `auth` — This mode enforces authentication of SNMP packets without encryption
- `priv` — This mode enforces both authentication and encryption (privacy) of SNMP packets.

By default, the community names, `public` and `private`, are not allowed. An authoritative engine ID must exist in order to use the SNMPv3 security mechanisms, authentication or authentication and encryption, to handle SNMP packets. By default, the engine ID is generated locally. The engine ID can be displayed with the `snmp-engineid-show` command. For more configuration details, refer to the *Understanding and Configuring SNMP* section.

Security of Routing Protocols

The Border Gateway Protocol (BGP) is the routing foundation of the Internet. As such, any organization with more than modest connectivity requirements often uses BGP, which can be targeted by attackers because of its ubiquity and the set-and-forget nature of BGP configurations in smaller organizations. However, there are various BGP-specific security features that can be leveraged to increase the security of a device configuration. They can be configured as options of the `vrouter-bgp-add` command.

The first recommended step is to enable mutual BGP peer authentication using an MD5 digest for each packet sent as part of a BGP session. Specifically, portions of the IP and TCP headers, of the TCP payload, and a secret key are used in order to generate the MD5 digest.

The second recommendation is to configure a maximum limit to the number of BGP prefixes.

Prefixes are stored by a router in memory. The more prefixes a router must hold, the more memory BGP must consume. In some configurations, a subset of all Internet prefixes should be stored, such as in configurations that leverage only a default route or routes for a provider's customer networks.

In order to prevent memory exhaustion, it is recommended to configure the maximum number of prefixes that is accepted on a per-peer basis. The `max-prefix` parameter can be used for that purpose.

Furthermore, it is recommended to filter BGP prefixes with prefix lists to implement stricter control over routes.

Prefix lists allow a network administrator to permit or deny specific prefixes that are sent or received via BGP. Prefix lists should be used wherever possible in order to ensure that network traffic is sent over the intended paths. They should be applied to each eBGP peer in both inbound and outbound directions.

Configured prefix lists limit the prefixes that are sent or received to those specifically permitted by the routing policy of a network. If this is not feasible due to the large number of prefixes received, a prefix list should be configured to specifically block known bad prefixes. These known bad prefixes include, for instance, unallocated IP address space and networks that are reserved for internal or testing purposes by RFC 3330. Outbound prefix lists should be configured to specifically permit only the prefixes that an organization intends to advertise. For more details, refer to the *Configuring BGP on a vRouter* section.

Similarly for OSPF, in the `vrouter-ospf-area-add` command, it is possible to specify inbound and outbound prefix lists to filter incoming and outgoing packets. For more details, refer to the *Configuring Open Shortest Path* First (OSPF) section.

Enabling System Logging

NetVisor OS maintains audit, system and event logs. All CLI commands and REST API calls that modify the system are logged in the audit log. Changes in system state and important alerts are logged in the system log. Details regarding the operation of various NetVisor OS features may optionally be logged to the event log for troubleshooting/monitoring purposes.

Log messages are accessible by using any of the following services: the CLI, the REST APIs, SFTP, or when syslog is enabled.

In general, users are advised to send logging information to a centralized syslog server. This makes it possible to correlate and audit network and security events across network devices more effectively.

To use the syslog service, the administrator has to specify which syslog servers the log messages should be forwarded to. The administrator can also select which messages should be transmitted, and how NetVisor OS's log parameters are translated to syslog parameters.

Messages can be sent to syslog servers using either TCP/TLS or UDP. TCP/TLS is recommended over UDP because with the former syslog messages are reliably delivered over a secure transport. TCP/TLS can be selected in the `admin-syslog-create` command by using the `tcp-tls` parameter. In addition, it is necessary to provision a client certificate with the `syslog-tls-cert-*` commands. For more information, refer to the *Understanding Logging* section.

In addition to the audit, system and event logs, NetVisor OS stores real-time data and history about TCP connections, endpoints accessing the network, routes, fabric communication, as well as flow, port, VLAN, VXLAN, tunnel, hardware, system, class of service and control statistics. This real-time data and history is extremely useful for operational and security purposes. In particular, the user can leverage the `connection-show` command to monitor TCP connections going through the device.

Starting with NetVisor OS version 6.1.1, the system log component of NetVisor uses OpenSSL version 1.1.1. Due to the upgrade to OpenSSL version 1.1.1, the following changes are available in NetVisor OS version 6.1.1's syslog client:

- SSL options are changed to support hardened approaches.
- Validation is now performed on Local Certificate chains during import.
- Validation is now performed on Local Certificate chains prior to connection.
- Validation is now performed on over-the-wire certificates from the peer.
- SSLv2, v3, and TLS version 1 and version 1.1 are deprecated and are no longer allowed.
- Select Ciphers are supported. For details, see the SSH & SSHD section.
- Audit messages are now produced for all failure cases.
- Certificate revocation list (CRLs) are now supported as in:
    - If no CRL can be loaded or an error is encountered loading the CRLs, then execution continues as normal and certificates are not verified against the CRL.
    - If a CRL can be loaded, the revocation status will be read from it and applied during verification if applicable.
    - CRLs should be periodically updated manually.

Prior to NetVisor OS version 6.1.1, importing syslog certificates by using the `syslog-tls-cert-import` command delivered a success message whether the certificates provided were valid for use or not.

However, with NetVisor OS version 6.1.1 onward, the certificates are validated at import and displays the

following message upon failure: “`syslog-tls-cert-import: Failed to verify syslog certificate`”.

The complete validation messages are available in *perror.log* file.

> **Note**: While configuring the syslog targets under TCP-TLS, the hostname of the target syslog server must match the hostname in the CN of the certificate. For example,

If your server is address is *server.example.com*, and the generated certificate CN states the hostname as server.example.com, then the hostname supplied to the syslog target creation must also be 'server.example.com'. Any mismatch here prevents the connection from being made.

For details on the log messages, see the *Log Messages* document.

Implementing user Authentication, Authorization and Accounting (AAA)

AAA is a set of features that are critical in order to secure interactive access to network devices. AAA provides a highly configurable environment that can be tailored based on the needs of the network admin. Arista NetVisor OS supports the TACACS+ standard for AAA.

TACACS+ is an authentication protocol that NetVisor OS switches can use for authentication of management users against a remote AAA server. These management users can access the switch using SSH or HTTPS. TACACS+ authentication, or more generally AAA authentication, provides the ability to use individual credentials for each network administrator. When you do not depend on a single shared password, the security of the network is improved and your accountability is strengthened.

TACACS+ can be used for any combination of Authentication, Authorization and Accounting (AAA), which means that it can be used to authorize session access as well as individual commands.

Command authorization with TACACS+ permits or denies each command that is entered by an administrative user. When the user enters a CLI command or an API client invokes an API call, NetVisor OS sends the command to the configured AAA server. The AAA server then looks up any configured policies in order to permit or deny the command for that particular user.

TACACS+ can be configured with the `aaa-create` command. In addition, for service continuity, it is recommended to use redundant AAA server. And for stronger security it is important to use a shared secret to secure TACACS+ traffic to/from the servers.

For more information, refer to the *Configuring TACACS+* section.

About Control Plane Traffic Protection (CPTP)

About the Importance of Hardening of Network Infrastructure

The network infrastructure is often a prime target for malicious attacks because of the possibility of inflicting the most amount of damage to as many devices as possible (in the worst-case scenario, to bring down the entire network and, with it, all the attached devices).

Other reasons to target the network may be to attempt to redirect (for example, through unintended flooding) and snoop traffic to learn about clear-text information and find out possible other weaknesses that can lead to further malicious actions.

Last but not least, network instability sometimes caused by mis-configurations or failures may result in an excessive amount of traffic that can put a strain on the network infrastructure and further exacerbate its stability problems.

Therefore, first and foremost, it is of foundational importance to apply robust control to the traffic that reaches the network devices and to implement appropriate protections against any potentially disruptive traffic.

In particular, this section focuses specifically on the hardware-based protection of the network *control plane*. Other complementary mechanisms, such as hardware-based security and QoS policies described in detail in other topics can be applied to the data plane too for comprehensive network hardening.

About Control Plane Traffic Protection

In any network device there exists a management entity (typically a CPU) that is in charge of communication exchanges with other networking devices as well as of interactions with a portion of the traffic coming from the rest of the network (the so-called *data plane*). In general, all the traffic that is natively directed or purposefully redirected to such management processor is commonly referred to as *control plane traffic*.

When the amount of any class or classes of traffic belonging to the control plane becomes abnormal--e.g., due to a Denial of Service (DoS) attack attempt--then the network device needs to take some containment action.

Hardware-based queuing and rate limiting are two common techniques employed to implement CPU protections, with *different* levels of granularity and control depending on the switch model's hardware capabilities.

About Port-based Control Plane Protection

Certain switch models use an internal Ethernet port to transport a portion of the control plane traffic to the management CPU. This special type of interface is referred to as a *rear-facing interface*. The list of models with a rear facing interface includes the following platforms:

- Edgecore: AS7712-32X, AS7312-54XS

- Freedom switches: F9532C-XL-R
- Ericsson: NSU, NRU01, NRU02, NRU03

For this case 8 queues are available for control plane traffic segregation and rate-limiting, which NetVisor OS leverages to protect the CPU from anomalous traffic.

By default, mission-critical control plane traffic is split across 7 weighted queues based on common network management requirements.

In addition, queue 0 is the default 'catch-all' queue that corresponds to all the control plane traffic not specifically segregated into one of the other seven queues.

Any of the eight CPU queues is configured with a default maximum transmission rate suitable to protect the control plane from overloading. If needed, the default rate values may be modified by the network administrator to match specific design requirements.

About Advanced Control Plane Protection

Arista Networks has implemented support for *Advanced Control Plane Traffic Protection (CPTP) with Auto-Quarantine*. This feature is supported on the CPU inband interface of the Freedom and Edgecore data center platforms, as well as of Dell's Open Networking Switches.

This very granular capability allows the control plane's processing path to be protected against both misbehaving and malicious devices (compromised end-points, rogue network nodes, etc.) that may start pumping an abnormal amount of control plane traffic. In Arista parlance, this is also referred to as *CPU hog protection*.

Advanced CPTP operates over 43 independent queues (from 0 to 42) in order to be able to provide separation and granular control over different types of control plane traffic classes. It can be enabled with the `system-settings-modify cpu-class-enable` command.

The Advanced Control Plane Traffic Protection with Auto-Quarantine is *not supported* on the following platforms:

- NSU
- NRU01
- NRU02
- NRU03

Note: In NetVisor OS releases prior to version 6.0.0, the default setting is `no-cpu-class-enable`. To change the setting to `cpu-class-enable` requires a subsequent system reboot for the setting to take effect. The default or the modified setting is preserved when upgrading to release 6.0.0 (or later). However, for new installations starting with release 6.0.0, the default setting becomes `cpu-class-enable`.

Note: CPTP on the inband interface performs CPU traffic classification and queuing in hardware, therefore there is no performance penalty in enabling this feature. CPTP queuing supports round-robin scheduling and rate limiting of individual CPU traffic classes, in addition to guaranteeing minimum buffer

space allocation to each class.

CPU resources are protected by segregating into separate queues the following types of traffic by default: various standard network control packets, cluster communication messages, fabric updates as well as regular flooded traffic, packets required for MAC learning and copy-to-cpu packets, analytics, etc.

In addition, custom traffic classes can be added by configuring user-defined CPU policies (for example, for troubleshooting purposes).

Note: Traffic flows that end up sharing a user-defined CPU queue will compete with each other for bandwidth. It is therefore recommended to configure queue-sharing only for traffic that does not constantly compete for CPU time under identical circumstances. Whenever possible, competing classes should be assigned to different queues.

About Auto-Quarantine

Advanced CPTP can be very granular with its innovative *auto-quarantine* (a.k.a. CPU hog protection) mechanism. As a user, you can enable the CPU hog protection capability (through the `cpu-class-modify` command) for the following protocols (in Arista' parlance also called *CPU classes*): OSPF, BGP, BFD, LACP, STP, ARP, VRRP, and LLDP.

When auto-quarantine is enabled, the NetVisor OS software monitors control plane packets arriving at the CPU on a per-source-device basis. Traffic from a source device that is deemed to be consuming too much bandwidth (as per user-configurable `rate-limit` value) is redirected to a dedicated per-protocol `quarantine queue` by installing a hardware policy entry.

At the same time a syslog alert is displayed and the offending source device's subsequent activity is monitored. Quarantine state is left automatically only when the traffic activity returns below acceptable limits for a pre-configured `timeout` time; then a corresponding syslog is displayed. You can view messages using the `log-event-show event-type system` command.

Only certain system-defined protocol queues support hog protection. For these CPU classes the user can choose to enable the CPU hog protection capability, or to select the `enable-and-drop` option. In the latter case, all traffic from the quarantined source with the assigned protocol is dropped during ingress.

Since hardware resources are limited, it is also possible to specify a threshold for the maximum number of acceptable CPU hog violators per port. When reached, such threshold causes that class's auto-quarantine hardware policy to become per switch-port (i.e., less granular) instead of being per-port per-offender.

About Data Plane Security Features

Data plane security features leverage the hardware and software forwarding capabilities of a switch to perform policing actions such as: forwarding or dropping certain packets, limiting certain resources below a set limit, constraining the forwarding of traffic, etc. In addition, controlling the bandwidth used by certain classes of traffic or certain devices can be used to guarantee optimal network performance. It can also be used to prevent certain attacks such as DoS (as explained in the previous section regarding the control plane).

The following are commonly used data plane security features that admins can employ to protect network resources and nodes:

- Port Isolation
- Limiting the number of MAC addresses on a per port basis (also known as Port Security)
- vPort-based Activity Tracking and Security
- Dynamic Host Configuration Protocol (DHCP) Snooping
- Router Advertisement (RA) Guard
- MAC Access Control Lists (ACLs)
- IP ACLs

Let's see how each one works and how it can be used as a countermeasure/protection against network attacks.

About Port Isolation

The Port Isolation functionality prevents local switching among ports on a NetVisor OS switch or on a pair of NetVisor OS switches configured as a cluster.

With Port Isolation NetVisor OS disables direct communication between hosts that are part of the same Layer 2 domain and are connected to isolated ports. That has the associated security benefit that hosts cannot see each other's MAC address and ARP requests (or replies), thus MAC or ARP spoofing attacks on other hosts are not possible (or become more difficult).

Communication between isolated hosts can still occur through a Layer 3 device, if needed, but Access Control Lists can be used to prevent that too when necessary.

Example of Usage

A network admin can use this feature to isolate bridged East-West traffic between hosts while allowing it through a firewall or router that can properly police it.

In the case of a router as shown in Figure 12-1 below, ports 1, 2, and 3 are configured as isolated ports so that the hosts attached to these ports cannot communicate at Layer 2 with each other directly, but only through the upstream router that is connected to the uplink vLAG 64.

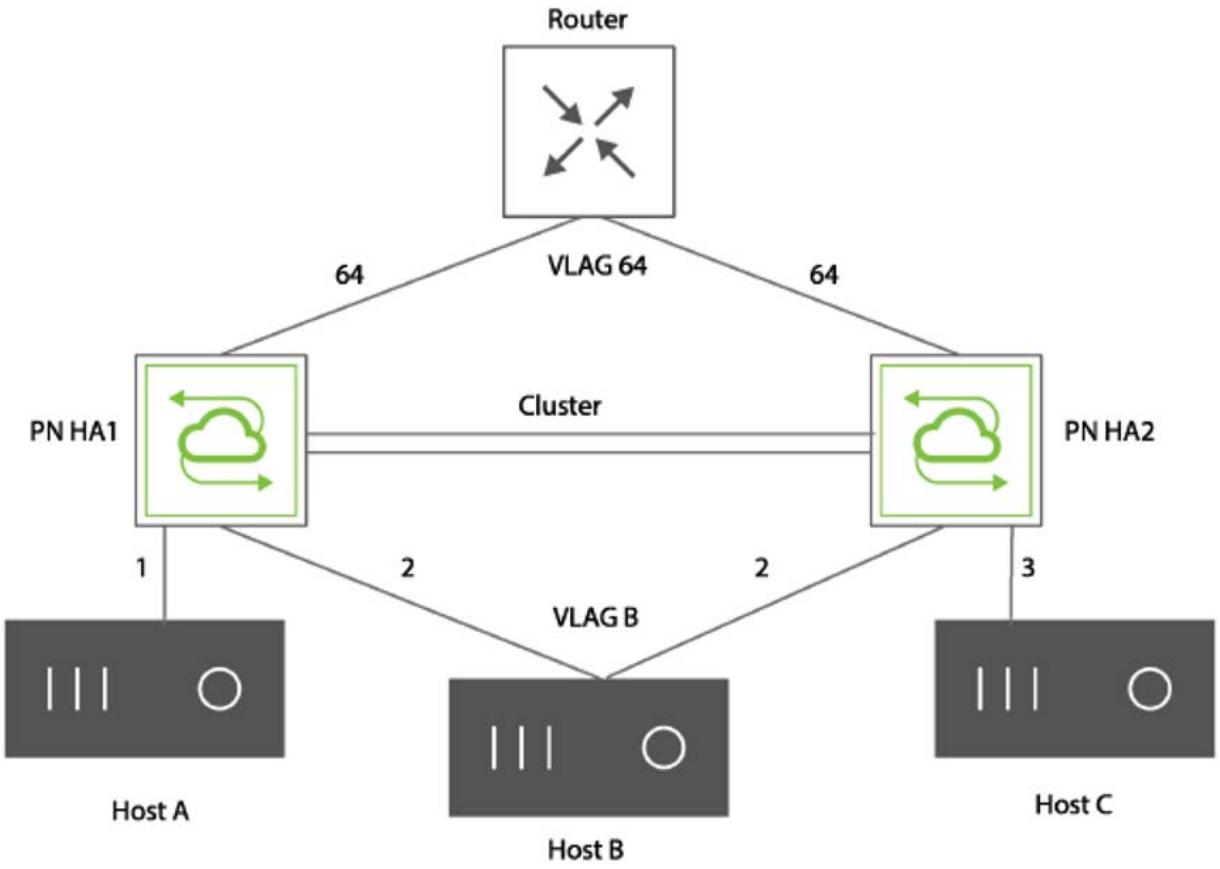


Figure 15-1 - Port Isolation Scenario

When using this feature on downlink ports within a cluster, especially when using downlink vLAGs (as with Host B above), you must configure port state association rules between the uplink ports and the downlink isolated ports.

This is needed because *isolated port traffic cannot cross the cluster link*.

In other words, a port state association (with the `port-association-create-name` command) is required in order to mirror the uplink state to the downlink state so that vLAG member link state remains *symmetrical in case of uplink failure*. That allows path redundancy to work with isolated vLAGs when an uplink vLAG member fails, given that the cluster link cannot be relied upon as fallback path.

For configuration information refer to the *Configuring Port Isolation* section below.

About MAC Address Limit

In order to prevent MAC flooding attacks, starting from NetVisor OS release 2.6.0, a new capability has been added to restrict the maximum number of MAC addresses that can be learned on a switch port. It's called *MAC address limit* (or sometimes *Port Security*). It uses the `mac-limit-modify` command to set a limit value and a corresponding action to perform when the limit is exceeded.

The configured policy action determines the remediation strategy that the control plane will undertake: just notify the user or disable the affected port.

For more details refer to the *Limiting the Number of MAC Addresses per Port* section below.

About DHCP Snooping

DHCP is a protocol that can be exploited by malicious agents to mess with the address assignment of the network. Therefore, starting from release 2.6.0, NetVisor OS has introduced support for *DHCP protection* (sometimes also referred to as *DHCP Snooping*). This requires that all DHCP packets be 'snooped' and sent to the CPU using a protocol-specific rate limiter to be inspected.

To prevent rogue agents from posing as DHCP servers, only switch ports that connect to known legitimate DHCP servers are defined as trusted. Therefore, typical DHCP message exchanges from server to client (such as DHCPOFFER and DHCPACK) are considered legitimate only on such trusted ports.

In a DHCP packet exchange there are various packet types:

- DHCPDISCOVER/DHCPREQUEST — Packets from the DHCP client to server (using UDP `dest-port = 67`)
- DHCPOFFER/DHCPACK — Packets from the DHCP Server to client (using UDP `dest-port = 68`)

NetVisor OS must snoop the DHCP packets in order to leverage this feature, and achieves this by installing a `copy-to-cpu` vFlow with the parameter, `bw-max`, to set packet rate limits.

- DHCP-client-vflow — Packets with UDP `dest-port=67, copy-to-cpu`
- DHCP-server-vflow — Packets with UDP `dest-port=68, copy-to-cpu`

A trusted port is a port receiving the DHCP server messages from a trusted DHCP server. Any DHCP server message, such as OFFER/ACKNOWLEDGE, received from trusted ports are valid. Ports not specifically configured as trusted are untrusted ports.

NetVisor OS drops any DHCP server message received from an untrusted port, and ensures that a rogue DHCP server cannot assign IP addresses to devices on your network.

In order for the users to control the trusted DHCP server port list, the CLI/APIs provide create/modify/show commands on the *dhcp-lease* object database with a choice of notify, drop or disable policy actions.

About Router Advertisement (RA) Guard

The IPv6 RA Guard feature allows a network administrator to block or reject unwanted or rogue RA guard messages by filtering them on switches before they reach their target(s).

Router advertisements are used by IPv6 devices to announce themselves on a link. The IPv6 RA Guard feature analyzes these RAs and filters them out when sent by unauthorized devices.

When a port is configured in host mode, all router advertisements and redirect messages are not allowed. The RA Guard feature compares configuration information on the switch with the information found in the received RA frame. Once the switch has validated the content of a router advertisement or of router redirect frame against the configuration, it forwards it to its unicast or multicast destination. If an RA frame content is not validated, the RA is dropped.

> **Note**: Internal ports and cluster ports are not blacklisted.

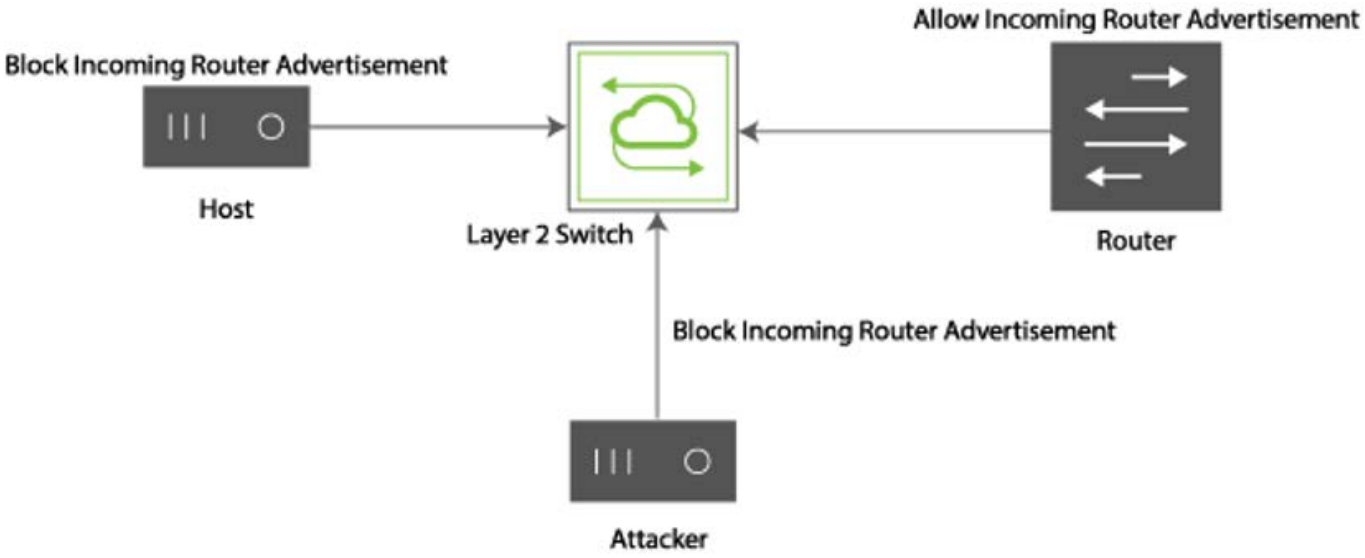


Figure 15-2 - Router Advertisement (RA) Configuration

In ***Figure 12-6*** above, the switch receives an RA from the router and floods the RA on the ports. The attacker, attempting to gain control over the network, sends a misleading RA with different prefixes, link-local or global IP addresses. Any receiving host would then assume the attacker to be the router, based on priority or arrival order.

By configuring RA Guard policies, the user can disallow any RAs sent from ports connected to untrusted devices. Instead, the RAs sent by a known good router, the source IP address and prefixes should be whitelisted by the RA Guard policies defined by the user. For more details and an example of configuration, refer to the *Configuring Router Advertisement Guard* section.

About Access Control Lists (ACLs)

Access Control Lists allow a network administrator to apply granular filtering to the traffic based on various packet parameters. Each packet is examined by the switch hardware at wire speed to determine if it is to be forwarded or dropped based on the criteria configured in the ACLs. NetVisor OS supports Layer 2 ACLs, based on MAC addresses, as well as IP-based ACLs, which can be based on source and destination addresses, and on protocol type. NetVisor OS supports UDP, TCP, IGMP, and ICMP protocol types.

For example, ACLs can allow one host to access part of your network and prevent another host from accessing the same area. You can also use ACLs to decide what types of traffic are forwarded or blocked.

For configuration examples on ACLs, see the following sections.

Moreover, for more granular security policies, which also support statistics collection, see the *Configuring and Using vFlows* chapter and in particular the *Using vFlows to Disable Communication for Security Monitoring* section.

IP Spoofing Protection

Using someone else's identity ("spoofing") is a very desirable characteristic in various types of attacks, for example for untraceability purposes and/or to amplify the effect of an attack. As a matter of fact, DDoS attacks would usually spoof random systems' source addresses.

In some cases, spoofed addresses are selected on purpose in a specific target network, so that when attacking one or more target devices those would respond with ICMP messages or other traffic, thereby unwittingly overloading the spoofed devices selected by the attackers.

As described in more detail in RFC 2827, sites can protect themselves by implementing proper filtering techniques to check the source addresses of the traffic. Firewalls, for example, can filter traffic coming from outside the DC and filter out spoofed IP sources (for example, sources that are known to be inside the DC, not outside, or that use invalid/reserved addresses such as the so-called "Martian addresses" which include any address within the 0.0.0.0/8, 10.0.0.0/8, 127.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16, 224.0.0.0/4, or 240.0.0.0/4 ranges).

However, when a spoofing attack is locally originated, DC network devices connected to the servers (typically the leaf switches) need to implement the source address filtering mechanism.

In high-performance environments a correspondingly high-performance solution needs to be deployed to be able to cope with IP spoofing issues with zero (or minimal) performance loss. On routers, this solution is usually known as Unicast Reverse Path Forwarding (uRPF, in short) technology, which in some cases may have performance implications.

NetVisor OS leverages its high-performance vFlow hardware technology to implement comprehensive uRPF validation that works with all types of traffic: bridged, routed, and VXLAN- encapsulated (whether pass-through or terminated).

vFlow can be used to completely prevent servers from sourcing IP traffic using an illegitimate address and to monitor attempted violations through dedicated statistics. A simple two-level source filter is described below in this example:

```
CLI (network-admin@switch) > vflow-create vlan 11 src-ip 10.1.11.0/27 name
amber-urpf-permit action none table System-VCAP-table-1-0

CLI (network-admin@switch) > vflow-create vlan 11 src-ip 0.0.0.0/0 name
amber-urpf-deny action drop table System-VCAP-table-1-0
```

Configuring Users Accounts and Setting Credentials

NetVisor OS provides a pre-created user account called *network-admin* with all access privileges set. However, other user accounts can be created by specifying only the required privileges. For these users, usernames and passwords can be managed locally on each network node, or globally on all nodes of the fabric.

For local authentication, the user attributes are defined and stored locally on each switch. In the case of fabric-wide authentication, the attributes are stored on every switch of the fabric. For example, the *network-admin* user (which is the default user) created by NetVisor OS is a fabric-scoped user and the attributes are stored on all switches in the fabric.

You can create new roles and new users with specified roles on a local switch or on all nodes of a fabric.

To create a new user and apply a new role on a local switch, use the command:

```
CLI (network-admin@switch) > user-create name name-string scope local
```

name name-string	Enter a name for the new user that you are configuring.
scope local|fabric	Specify if the scope is local or fabric.
Specify any of the following options:	
initial-role role name	Specify the initial role for the new user.
login-fail-count login-fail-count-number	This option enables you to customize a user with a pre-allowed limit for authorization failures. When the user exceeds the maximum allowed or configured number of authentication failures, then NetVisor automatically locks the user account. The default value for this option is zero (0), which means the account does not get locked out. This parameter is added in NetVisor OS version 6.1.1.
lock-account|no-lock-account	The lock-account option is used to manually lock any user account (that is purportedly misbehaving with the system privileges). This can be done only by the *network-admin* account. Being the super-user for NetVisor, the *network-admin* account cannot be locked out. Likewise, any user account that is in locked state can be un-locked only by *network-admin* using the no-lock-account option in the command. This parameter is added in NetVisor OS version 6.1.1.

`minimum-pw-length 6..65`	Specify a minimum password length for user accounts. The password length should be in the range of 6-65 characters. The default value is 6 characters. An error is displayed if the configured `minimum-pw-length` is not honored. This parameter is added in NetVisor OS version 6.1.0 as per Common Criteria compliance.

Note: The effectiveness of a selected `login-fail-count` value is dependent on the client and server configuration. For example, if you have *N* as NumberOfPasswordPrompts for ssh client and *N+1* as MaxAuthTries for ssh server, then setting `login-fail-count` of N will lock the user account when N failed attempts are made with non-empty passwords.

Note: If the default value of zero (0) is configured for `login-fail-count`, then the user account does not get locked even after multiple failed authentications as the failed authentication limit error check is invalid (scenario prior to NetVisor OS 6.1.0).

Additionally, you can also configure the `minimum-pw-length` parameter for SFTP service by using the `admin-sftp-modify` command. For example,

```
CLI (network-admin@switch) > admin-sftp-modify enable minimum-pw-length 10
sftp password:
confirm sftp password:
password length matched..!
CLI (network-admin@switch) >
```

To delete an existing user, use the command:

```
CLI (network-admin@switch) > user-delete name name-string
```

To modify an existing user, use the command:

```
CLI (network-admin@switch) > user-modify name name-string
```

`name name-string`	Enter the username.
Specify any of the following options:	
`password password-string`	Enter the plain text password. You must enter the password characters based on the limit set for the `minimum-pw-length` parameter.
`login-fail-count login-fail-count-number`	Specify the allowed limit for authorization failures before locking the user account. For more details, see the `user-create` table above.

`lock-account|no-lock-account`	Specify to lock a user account upon reaching maximum login errors. For more details, see the `user-create` table above.
`minimum-pw-length 6..65`	Specify a minimum password length for user accounts. The password length should be in the range of 6-65 characters. The default value is 6 characters. For more details, see the `user-create` table above.

Note: The details of all changes are added to the *session logs* file and you can view the details using the `log-session-show` command. A sample configuration output is provided below as a reference:

```
CLI (network-admin@ara01) > log-session-show
```

```
category     time                             name            code  level  user         client-addr   message
-------- -------------------------------- --------------- ----- ----- ------------- ------------ ------------------------------------
session  2021-05-10,02:30:19.718069-07:00 login           11099 info  network-admin 10.140.0.104 login

session  2021-05-10,04:12:01.918420-07:00 session_timeout 11542 note  network-admin              session timed out for user network-admin
session  2021-05-10,04:12:01.921920-07:00 logout          11100 info  network-admin 10.140.0.104 logout

session  2021-05-10,11:04:42.828059-07:00 login           11099 info  network-admin 10.140.1.14  login

session  2021-05-10,12:21:28.590472-07:00 session_timeout 11542 note  network-admin              session timed out for user network-admin
:
:
session  2021-06-22,01:47:41.583393-07:00 login_failed    11103 error network-admin              login failed: wrong password

session  2021-06-22,01:47:59.376142-07:00 logout          11100 info  network-admin 10.140.0.104 logout

session  2021-06-22,06:26:52.686519-07:00 login           11099 info  network-admin 10.140.0.104 login

session  2021-06-22,07:27:15.307374-07:00 session_timeout 11542 note  network-admin              session timed out for user network-admin
session  2021-06-22,07:27:15.310101-07:00 logout          11100 info  network-admin 10.140.0.104 logout
:
:
session  2021-06-22,07:55:14.931522-07:00 login           11099 info  network-admin 10.140.0.116 login

session  2021-06-22,07:59:44.813695-07:00 logout          11100 info  network-admin 10.140.0.116 logout

session  2021-06-22,12:27:53.769534-07:00 login           11099 info  network-admin 10.140.1.14  login
```

To view the configuration details, use the command:

```
CLI (network-admin@switch) > user-show name name-string
```

Specify any of the following options:	
`name name-string`	Enter the username for which you want to view the details.
`[ scope local|fabric ]`	Specify to view user account details on either a local switch or on all switches in the fabric.
`[ uid uid-number ]`	Enter the user ID of the account.
`[ type netvisor|unix |tacacs|web-`	Enter the user type to view any specific type of

`token|mfg ]`	account.
`[ server aaa-tacacs name ]`	Enter the TACACS server name to view only those details.
`login-fail-count login-fail-count-number`	Enter the allowed limit for authorization failures before locking the user account. For more details, see the `user-create` table above.
`lock-account|no-lock-account`	Specify to view the locked/unlocked account details. For more details, see the `user-create` table above.
`minimum-pw-length 6..65`	Specify to view the minimum password length for user accounts. For more details, see the `user-create` table above.

Note: As in all the show commands, use the *formatting options* to display the details as desired.

Further, here is an example configuration to explain the security aspects of different user accounts:

- Create a user account on a local switch using the command:

```
CLI (network-admin@switch) > user-create name example1 scope local
```

- Modify the account specifying:

```
CLI (network-admin@switch) > user-modify name example1 no-lock-account
```

- Verify the configuration:

```
CLI (network-admin@switch) > user-show

name          scope  uid   type     login-fail-count lock-account minimum-pw-length
------------- ------ ----- -------- ---------------- ------------ -----------------
network-admin fabric 39999 netvisor 0                false        6
example1      local  40000 netvisor 0                false        6
```

Note: The minimum-pw-length field is displayed only for network-admin accounts.

- Now, modify the configuration to lock the user account after 3 failed attempts and also set other parameters:

```
CLI (network-admin@switch) > user-modify name example1 login-fail-count 3
lock-account minimum-pw-length 9
```

This command prompts the user to enter a new password that meets `minimum-pw-length` specified.

```
CLI (network-admin@switch) > user-show

name          scope  uid   type     login-fail-count lock-account minimum-pw-length
------------- ------ ----- -------- ---------------- ------------ -----------------
network-admin fabric 39999 netvisor 0                false        6
example1      local  40000 netvisor 3                true         9
```

- Additionally, you can also configure `minimum-pw-length` after enabling SFTP:

```
CLI (network-admin@switch) > admin-sftp-modify minimum-pw-length 9 enable
sftp password:
confirm sftp password:

CLI (network-admin@switch) > admin-sftp-show

switch:             switch
sftp-user:          sftp
enable:             yes
```

General Guidelines

Following are the guidelines to keep in mind while configuring new user accounts:

- To setup a system to be ready for Common Criteria conformance test with regard to SSH, the scripts */usr/bin/setup-cc-ssh-config.ksh* should be executed on target switch. And the script */usr/bin/setup-cc-ssh-keygen.ksh* should be run from the client system.
- If a user account gets locked out by NetVisor OS due to authentication failure, then SSH sessions are disabled until a *network-admin* user (with full privileges) unlocks the account.
- If all accounts get locked out without access to SSH sessions, then only the *network-admin* account user can log in using local or management console (telnet) because the *network-admin* user is a pre-created account by NetVisor OS and cannot be locked out.
- You can use either the CLI or the RESTful API to configure the user authentication parameters.
- Other applications such as UNUM that uses NetVisor OS's authentication mechanism currently reflects the configuration changes.

Changing NTP Secondary Server During Switch Setup

From NetVisor OS version 6.0.0 and later release, you can undo or unset a configured NTP secondary server by using the `switch-setup-modify` command along with an empty string ("") as parameter. You can use either single-quotes ('') or double-quotes ("") to indicate the empty string.

In NetVisor OS version 6.1.0, an additional third NTP server is introduced as part of the Common Criteria compliance process.

First, view the switch setup details using the command:

```
CLI (network-admin@switch) > switch-setup-show
switch-name:                switch
mgmt-ip:                    10.13.0.55/23
mgmt-ip-assignment:         static
mgmt-ip6:                    fe80::e6f0:4ff:fec9:dcbb/64
mgmt-ip6-assignment:        autoconf
mgmt-link-state:            up
mgmt-link-speed:            1g
in-band-ip:                  192.168.66.3/24
in-band-ip6:                 fe80::640e:94ff:fe69:5fd8/64
in-band-ip6-assign:         autoconf
gateway-ip:                 10.13.0.1
dns-ip:                     10.135.2.13
dns-secondary-ip:           10.20.4.1
domain-name:                 pluribusnetworks.com
ntp-server:                  0.ubuntu.pool.ntp.org
ntp-secondary-server:        1.ubuntu.pool.ntp.org
timezone:                    America/Los_Angeles
date:                       2021-01-11,21:57:43
hostid:                     184558697
location-id:                2
enable-host-ports:          yes
```

To remove or unset the secondary NTP server, use the command with an empty string in single quotes or double quotes as below:

```
CLI (network-admin@switch) > switch-setup-modify ntp-secondary-server
Server ip or host-name / empty string to unset

CLI (network-admin@switch) > switch-setup-modify ntp-secondary-server ""
or
CLI (network-admin@switch) > switch-setup-modify ntp-secondary-server ''
```

Additionally, from NetVisor OS version 6.1.0 onward, you can add a tertiary NTP server using the `switch-setup-modify` command:

```
CLI (network-admin@switch) > switch-setup-modify ntp-tertiary-server
3.ubuntu.pool.ntp.org

CLI (network-admin@switch) > switch-setup-show
```

```
switch-name:                switch
mgmt-ip:                    10.13.0.55/23
mgmt-ip-assignment:         static
mgmt-ip6:                    fe80::e6f0:4ff:fec9:dcbb/64
mgmt-ip6-assignment:        autoconf
mgmt-link-state:            up
mgmt-link-speed:            1g
in-band-ip:                  192.168.66.3/24
in-band-ip6:                 fe80::640e:94ff:fe69:5fd8/64
in-band-ip6-assign:         autoconf
gateway-ip:                 10.13.0.1
dns-ip:                     10.135.2.13
dns-secondary-ip:           10.20.4.1
domain-name:                 pluribusnetworks.com
ntp-server:                  0.ubuntu.pool.ntp.org
ntp-secondary-server:
ntp-tertiary-server:        3.ubuntu.pool.ntp.org
timezone:                    America/Los_Angeles
date:                       2021-01-11,21:57:43
hostid:                     184558697
location-id:                2
enable-host-ports:          yes
```

Note: NetVisor OS logs the changes such as the current time, new time, and the origin of the change into the audit log file. For example:

```
CLI (network-admin@switch) > log-audit show

audit   2021-01-11,21:57:31.077766-08:00 user_command 11001 info   network-
admin 10.140.0.129 Command "switch-setup-modify ntp-tertiary-server 3.ubunt
u.pool.ntp.org" result success
audit   2021-01-11,22:00:35.036675-08:00 user_command 11001 info   network-
admin 10.140.0.129 Command "switch-setup-modify ntp-tertiary-server 7.ubunt
u.pool.ntp.org" result failure: "Could not resolve NTP server's name/addr"
```

Configuring Control Plane Traffic Protection (CPTP)

A network device's management processor entity needs to be protected from getting overloaded with potentially malicious (or even non malicious) excess traffic received from the network. In NetVisor OS, that can be achieved with the hardware-based Control Plane Traffic Protection feature.
As discussed in the *About Control Plane Traffic Protection* section above, CPTP is supported in two flavors depending on the switch model, the default configuration and the NetVisor OS release:

- Port-based Control Plane Traffic Protection
- Advanced Control Plane Traffic Protection with Auto-Quarantine

The following sections describe the configuration of both modes.

Configuring Port-based Control Plane Traffic Protection

Certain switch models make use of an internal rear-facing interface for CPU communication in addition to a special `control-port`. Other models use the `control-port` only.
For all these cases, by default 8 queues are available for control plane traffic segregation and rate-limiting on a per internal port basis. The corresponding eight default packet rates (pps) can be displayed with the following command:

```
CLI (network-admin@switch) > port-cos-rate-setting-show layout-vertical

switch:             switch
port:               control-port
ports:              0
cos0-rate(pps):     5000
cos1-rate(pps):     5000
cos2-rate(pps):     5000
cos3-rate(pps):     5000
cos4-rate(pps):     5000
cos5-rate(pps):     5000
cos6-rate(pps):     5000
cos7-rate(pps):     5000
```

Internal (rear facing) data and span ports can be present in a system to carry control plane traffic to the CPU, each using 8 separate queues and rates, as shown below in condensed form:

```
CLI (network-admin@switch) > port-cos-rate-setting-show

port         ports cos0-rate(pps) cos1-rate(pps) cos2-rate(pps) ... cos6-rate(pps) cos7-rate(pps)
------------ ----- -------------- -------------- -------------- ... -------------- --------------
control-port 0     100000         100000         100000         ... 100000         100000
data-port    117   100000         100000         100000         ... 100000         100000
span-ports   118   100000         100000         100000         ... 100000         100000
```

It is possible to modify the `default` rate settings in packets per second using the `port-cos-rate-setting-modify` command:

```
CLI (network-admin@switch) > port-cos-rate-setting-modify ?
```

`port-cos-rate-settings-modify`	Update the port cos rate limit
`port control-port|data-port|span-ports`	port
Specify at least one of the following options	
`cos0-rate unlimited|0..10000000`	cos0 rate limit (pps)
`cos1-rate unlimited|0..10000000`	cos1 rate limit (pps)
`cos2-rate unlimited|0..10000000`	cos2 rate limit (pps)
`cos3-rate unlimited|0..10000000`	cos3 rate limit (pps)

`cos4-rate`	`unlimited|0..10000000`	cos4 rate limit (pps)
`cos5-rate`	`unlimited|0..10000000`	cos5 rate limit (pps)
`cos6-rate`	`unlimited|0..10000000`	cos6 rate limit (pps)
`cos7-rate`	`unlimited|0..10000000`	cos7 rate limit (pps)

For example, the Class 0 rate for control traffic can be configured using the following command:

```
CLI (network-admin@switch) > port-cos-rate-setting-modify port control-port
cos0-rate <rate>
```

In addition, to show the per-queue *traffic* statistics you can issue the following command:

```
CLI (network-admin@switch) > port-cos-stats-show port 0 layout vertical

switch:        switch
time:               11:59:15
port:             0
cos0-out:        58.8M
cos0-drops:    180M
cos1-out:        58.8M
cos1-drops:    185M
cos2-out:        0
cos2-drops:    0
cos3-out:        0
cos3-drops:    0
cos4-out:        0
cos4-drops:    0
cos5-out:        0
cos5-drops:    0
cos6-out:        65.5M
cos6-drops:    1.06G
cos7-out:        483K
cos7-drops:    493M
```

To clear the queue statistics on the internal ports, use the `port-cos-stats-clear` command.

Configuring Advanced Control Plane Traffic Protection

To configure this feature, you must first enable it using the `system-settings-modify` command. The command syntax is:

```
CLI (network-admin@switch) > system-settings-modify cpu-class-enable|no-cpu-class-enable
```

After you enable Advanced Control Plane Traffic Protection (with the `cpu-class-enable` option), NetVisor OS prompts you to restart the switch with the following message:

```
Note: nvOSd must be restarted for this setting to take effect.
```

The same message is also printed when the feature is disabled (with the `no-cpu-class-enable` option).

> **Note**: The alternative 8-queue mode described in the previous section is applied to the main control plane communication channel when `system-settings-modify` is set to `no-cpu-class-enable`. Advanced Control Plane Traffic Protection support is hardware dependent and may not be available on all switch models.

To show the pre-configured Advanced Control Plane Traffic Protection classes, you can use the `cpu-class-show` command:

```
CLI (network-admin@switch) > cpu-class-show format all count-output
```

```
name                  scope rate-limit hog-protect hog-protect-support buffer-pool-ratio queue
------------------  ----- ---------- ----------- ------------------- ----------------- -----
class0              local 3000       disable     none                3                 0
dmac-miss           local 1000       disable     none                3                 1
smac-miss           local 1000       disable     none                3                 2
l3-miss             local 1000       disable     none                3                 3
l2mc-miss           local 3000       disable     none                3                 4
ttl1                local 1000       disable     none                3                 5
stp                 local 1000       disable     supported           3                 6
lacp                local 1000       disable     supported           3                 7
system-d            local 1000       disable     none                3                 8
igmp                local 1000       disable     supported           3                 9
bcast               local 1000       disable     none                3                 10
icmpv6              local 1000       disable     supported           3                 11
tcp-analytics       local 1000       disable     none                3                 12
kpalv               local 1000       disable     none                3                 13
ecp                 local 1000       disable     none                3                 14
arp                 local 3000       disable     supported           3                 15
lldp                local 1000       disable     supported           3                 16
dhcp                local 1000       disable     none                3                 17
pim                 local 1000       disable     supported           3                 18
local-subnet        local 1000       disable     supported           3                 19
bgp                 local 1000       disable     supported           3                 20
ospf                local 1000       disable     supported           3                 21
bfd                 local 1000       disable     supported           3                 22
vrrp                local 1000       disable     supported           3                 23
control             local 3000       disable     none                3                 24
dhcp-log-drop       local 1000       disable     none                3                 25
http-rest           local 3000       disable     none                3                 26
vport-messages      local 1000       disable     supported           3                 27
hog-arp             local 100        disable     none                1                 28
hog-ospf            local 100        disable     none                1                 29
hog-bgp             local 100        disable     none                1                 30
```

```
hog-bfd            local 100       disable     none              1               31
hog-lacp           local 100       disable     none              1               32
hog-stp            local 100       disable     none              1               33
hog-vrrp           local 100       disable     none              1               34
hog-lldp           local 100       disable     none              1               35
hog-local-subnet   local 100       disable     none              1               36
hog-igmp           local 100       disable     none              1               37
hog-pim            local 100       disable     none              1               38
hog-icmpv6         local 100       disable     none              1               39
hog-vport-messages local 100       disable     none              1               40
Count:  41
```

This command shows the different categories of control plane traffic that get protected by this feature (for example, `smac-miss` and `dmac-miss` for MAC address learning as part of the vPort database entry creation; or `stp`, `lacp`, and `lldp` for the Layer 2 protocol classes, etc.). It also shows the respective default `rate-limit` values (in packets per second), the queue numbers (0-42, where some queue numbers are *unused by default*) and also whether or not each class supports *auto-quarantine* (`hog-protect-support`).
Auto-quarantine queues are labeled with a special name *hog-<class name>*, such as: `hog-arp, hog-ospf, hog-bgp, hog-bfd, hog-lacp, hog-stp, hog-vrrp, hog-lldp, hog-local-subnet, hog-igmp, hog-pim, hog-icmpv6`.

Note: Starting with NetVisor OS version 6.0.0, on certain platforms only (due to hardware dependencies) the `l2mc-miss` class is available to control the rate of incoming unknown multicast packets when Multicast Fabric VRFs are used. Supported platforms are the Dell S4100 and S5200 Series.

Note: Starting from NetVisor OS release 5.1.0 two new queues, one for CPU-bound REST API traffic (TCP port 80 and 443) and another for vPort database-related messages (UDP port 23398), are added with the names: `http-rest` and `vport-messages`. The default `rate-limit` values are set to 3000 pps and 1000 pps respectively. An auto-quarantine queue is added for the latter: `hog-vport-messages`.

Furthermore, starting from NetVisor OS release 5.1.0 the default `rate-limit` values for `arp` and `control` have been conservatively lowered to 3000. When upgrading to this release, existing user configuration changes will be honored; however, in the absence of user modified values, the old default values will be replaced with the new more conservative ones.

Note: The total number of CPU classes available for CPTP is limited by the hardware. In case of conflict, system-created CPU classes are prioritized over user-defined ones at bootup. Given that, if all available classes are used up, some user-defined classes will not persist across an upgrade if more system classes are added in the new release. In such cases, users should account for any (potential) CPTP system class differences between releases while planning an upgrade.

Settings of pre-configured system classes (except the catch-all class 0) can be modified with the following command:

```
CLI  (network-admin@switch)  >  cpu-class-modify
```

`cpu-class-modify`	Modify a CPU class.
`name` *`name-string`*	Specify the name of the CPU class.
Specify one of more of the following options	

`rate-limit`	`rate-limit-number`	Specify the cap for the rate limit.
`hog-protect`	`disable|enable|enable-and-drop`	Specify if you want to enable, enable and drop packets, or disable hog protection.

Note: Starting from NetVisor OS release 5.1.0 the default cos0-rate value is set to 3000 pps automatically when Advanced CPTP is enabled.

Starting with NetVisor OS release 5.1.1, the `class0` rate can be configured by using the following command:

```
CLI (network-admin@switch) > cpu-class-modify name class0 rate-limit <rate>
```

Configuring User-Defined Traffic Classes

These commands are available to configure custom CPU classes that are added to the default list (shown in previous section) in order to address special user requirements:

To add a CPU class, use the command:

```
CLI (network-admin@switch) > cpu-class-create
```

`name`	Specify a name for the CPU class.
`scope local|fabric`	Specify the scope as local or fabric.
`rate-limit rate-limit-number`	Specify the cap for the rate limit.
`hog-protect disable|enable|enable-and-drop`	Specify if you want to enable, enable and drop packets, or disable hog protection.

The `cpu-class-create` command can be used to allocate new CPU traffic queues for special cases that are identified by user-configurable policies.

For instance, when the culprit of a high CPU utilization issue is being investigated in connection with an errant traffic flow (e.g., FTP), the Arista TAC team may recommend configuring a user-defined traffic class with an associated policy to protect the CPU.

In this example a new class `TAC-class` is created and associated to a rate of 100 pps and automatically allocated to free queue 40:

```
CLI (network-admin@switch) > cpu-class-create name TAC-class scope local
rate-limit 100

CLI (network-admin@switch) > cpu-class-show name TAC-class

name      scope rate-limit hog-protect  hog-protect-support queue
-------   ----- ---------- -----------  ------------------- -----
TAC-class  local 100        disable      none                40
```

TAC also requests to create/modify a vFlow policy to direct FTP packets to the new CPU class (`TAC-class`) queue. vFlow match criteria can be freely chosen to match the specific rate limiting need of the user provided they include a `to-cpu` or `copy-to-cpu` action for packets:

```
CLI (network-admin@switch) > vflow-create name myflow scope local in-port
10 proto ftp action to-cpu cpu-class TAC-class

CLI (network-admin@switch) > vflow-show

name   scope type  in-port burst-size precedence action enable cpu-class
------ ----- ----- ------- ---------- ---------- ------ ------ ---------
myflow local vflow  10     auto       default    to-cpu enable TAC-class
```

Note: CPTP's vFlow match criteria can be freely chosen to match the specific rate limiting need of the user provided they include a `to-cpu` or `copy-to-cpu action` for packets.

After issue resolution, the user can delete a previously created custom CPU class if no longer needed:

```
CLI (network-admin@switch) > cpu-class-delete
```

`name`	Specify the name of the CPU class to delete.

Or you can modify the CPU class after creating it:

```
CLI (network-admin@switch) > cpu-class-modify
```

`name`	Specify the name of the CPU class.
`rate-limit rate-limit-number`	Specify the cap for the rate limit.
`hog-protect disable|enable|enable-and-drop`	Specify if you want to enable, enable and drop packets, or disable hog protection.

Note: You cannot modify the scope of the CPU class. If you want to change the scope, you must delete the existing CPU class and create a new CPU class with the correct scope.

Configuring Other Advanced CPTP Settings

Various additional settings are available to tweak the behavior of Control Plane Traffic Protection to one's needs (e.g., the threshold for the maximum number of acceptable CPU hog violators per port).

It is possible to show the values of the settings with the following command:

```
CLI  (network-admin@switch)  >  cpu-class-settings-show

switch:                          switch
hog-checker-interval(ms):     100
hog-max-violators-per-port:   50
hog-warning-threshold:         5
hog-violator-timeout(s):       20
```

In this example, the timeout for a violator (which is no longer misbehaving) to be removed from quarantine is 20 seconds. The maximum number of violators per port is set to 50.
These settings can be modified with the following command:

```
CLI  (network-admin@switch)  >  cpu-class-settings-modify
```

`hog-checker-interval` *`hog-checker-interval-number`* `(ms)`	Specify the hog checking interval in milliseconds.
`hog-max-violators-per-port` *`hog-max-violators-per-port-number`*	Specify the maximum number of hog violators per port.
`hog-warning-threshold` *`hog-warning-threshold-number`*	Specify the number of warnings at which host becomes hog violator.
`hog-violator-timeout` *`hog-violator-timeout-number`* `(s)`	Specify the timeout before restoring the hog violator to normal queue after an idle state.

Showing and Clearing CPTP Statistics

You can view the CPTP statistics, including class 0s by using the following command:

```
CLI (network-admin@switch) > cpu-class-stats-show
```

`name`	Specify the name of the CPU class to clear statistics.
`cos` *`cos-number`*	Specify the CoS value for the CPU class.

Or to clear them:

```
CLI (network-admin@switch) > cpu-class-stats-clear
```

`name`	Specify the name of the CPU class to clear statistics.
`cos` *`cos-number`*	Clear the CoS value for the CPU class.
`hw-out-pkts` *`hw-out-pkts-number`*	Clear the hardware transmitted packet count.
`hw-drop-pkts` *`hw-drop-pkts-number`*	Clear the number of hardware dropped packets.
`sw-pkts` *`sw-pkts-number`*	Clear the number of packets processed in software.
`sw-drops-pkts` *`sw-drops-pkts-number`*	Clear the number of packets dropped in software because the queue is full.
`hog-violations` *`hog-violations-number`*	Clear the number of hog protection host violations and moved to separate queue.
`hog-warnings` *`hog-warnings-number`*	Clear the number of hog protection delegated bandwidth warnings.
`hog-hosts-in` *`hog-hosts-in-number`*	Clear the number of added hosts for hog protection.
`hog-hosts-out` *`hog-hosts-out-number`*	Clear the number of hosts removed from hog protection.
`hog-max-hosts-drops` *`hog-max-hosts-drops-number`*	Clear the number of dropped hosts with hog protection because the maximum number of hosts is reached.

A handy command to periodically check the CPU traffic statistics is the following:

```
CLI (network-admin@switch) > cpu-class-stats-show show-diff-interval 1
```

which refreshes the statistics every second on screen, and thus can be used to observe traffic spikes.

In case of an auto-quarantined traffic source, the violator's information and related statistics can be displayed with the following commands:

```
CLI (network-admin@switch) > hog-violator-show
```

mac *mac-address*	Displays the hog violator MAC address.
vlan *vlan-id*	Displays the hog violator VLAN ID.
vxlan *vxlan-id*	Displays the hog violator VXLAN ID.
port *port-number*	Displays the hog violator ingress port.
cpu-class *cpu-class-string*	Displays the hog violator original class.
hog-cpu-class *hog-cpu-class-string*	Displays the hog violator hog queue CPU class.
created date/time: *yyyy-mm-ddTHH:mm:ss*	Displays the time and date when hog violator is created.
vflow *vflow-string*	Displays the redirect vFlow.
vflow2 *vflow-string*	Displays the redirect vFlow 2.
vflow3 *vflow-string*	Displays the redirect vFlow 3.

```
CLI (network-admin@switch) > hog-violator-stats-show
```

time date/time: *yyyy-mm-ddTHH:mm:ss*	Displays the time and date to start statistics collection.
start-time date/time: *yyyy-mm-ddTHH:mm:ss*	Displays the start time of the statistics collection.
end-time date/time: *yyyy-mm-ddTHH:mm:ss*	Displays the end time of the statistics collection.
duration duration: *#d#h#m#s*	Displays the duration of statistics collection.
interval duration: *#d#h#m#s*	Displays the interval between statistics collection.
since-start	Displays the statistics collection since the start time.
older-than duration: *#d#h#m#s*	Displays the statistics collection older than the time.
within-last duration: *#d#h#m#s*	Displays the statistics collection within a specified time period.
name *vflow-name*	Displays the name of the vFlow.
vnet *vnet-name*	Displays the VNET name of the vFlow.
id	Displays the ID assigned to the vFlow.

About CPTP Changes in the Z9432F-ON Platforms

Starting from NetVisor OS release 7.0.1 the Z9432F-ON platforms are supported. They introduce a new generation of programmable ASICs, which has some differences in the CPTP support:

- The `cpu-class-enable` in `system-setting-show` is always `on` and cannot be disabled.
- CPTP with CPU hog protection does not work for BFD protocol packets.
- For CPTP classes a total of 48 queues are supported. Since one additional custom CPU class is reserved for internal use, only one user-defined CPTP class is supported with the `cpu-class-create` command.

For example:

```
CLI (network-admin@switch) > cpu-class-create name cpu-class1 scope local rate-limit 10
cpu class created

CLI (network-admin@switch) > cpu-class-create name cpu-class2 scope local rate-limit 10
cpu-class-create: all classes used up
```

Using this command, you can set the egress rate limiting for the CPU:

```
CLI (network-admin@switch) > sys-flow-setting-modify cpu-rx-rate-pps 4999
CLI (network-admin@switch) > sys-flow-setting-show
cpu-rx-rate-pps:               4999
```

The default value for the Z9432F-ON Platforms is 5000 pps:

```
CLI (network-admin@switch) > sys-flow-setting-show format cpu-rx-rate-pps
cpu-rx-rate-pps:  5000
```

Configuring Port Isolation

As explained in the *About Port Isolation* section above, a network admin can use this feature to isolate bridged East-West traffic between hosts while allowing it through an upstream firewall or router.

To configure isolated ports, use the `no-local-switching` parameter in the `port-config-modify` command like so:

```
CLI (network-admin@switch) > port-config-modify port 1,2 no-local-switching
```

So, referring to the example shown in ***Figure 12-1*** above, on the cluster pair the isolated port configuration can be entered as follows:

PN-HA1

```
CLI (network-admin@pn-ha1) > port-config-modify port 1 no-local-switching

CLI (network-admin@pn-ha1) > port-config-modify port 2 no-local switching
```

PN-HA2

```
CLI (network-admin@pn-ha2) > port-config-modify port 2 no-local-switching

CLI (network-admin@pn-ha2) > port-config-modify port 3 no-local-switching
```

Typically, if inter-host Layer 3 connectivity is needed, you would configure the upstream router or firewall to perform *local proxy* ARP and/or *NDP proxy* so as to respond to all ARP requests and/or neighbor solicitations coming from isolated hosts.

To avoid interfering with local proxy ARP and NDP proxy, you may want to disable ARP and ND optimization as follows:

```
CLI (network-admin@pn-ha1) > system-settings-modify no-optimize-arps
CLI (network-admin@pn-ha1) > system-settings-modify no-optimize-nd

CLI (network-admin@pn-ha2) > system-settings-modify no-optimize-arps
CLI (network-admin@pn-ha2) > system-settings-modify no-optimize-nd
```

Additionally, configure the port link state association between downlinks and uplinks for proper vLAG redundancy. A port association is required to match the link state of downlink isolated ports with the one of uplink ports.

When all uplink ports are down, downlink isolated ports are administratively disabled until one of the uplinks becomes operational again.

In this example, the port association names are PA1 and PA2. On PN-HA1, the uplink port number is 64, and isolated downlink ports' numbers are 1 and 2. On PN-HA2, the uplink port number is also 64 (uplink vLAG member), and isolated downlink ports' numbers are 2 and 3:

```
CLI (network-admin@pn-ha1) > port-association-create name PA1 master-ports 64 slave-ports 1,2
policy any-master
CLI (network-admin@pn-ha2) > port-association-create name PA2 master-ports 64 slave-ports 2,3
policy any-master
```

To view ports that are configured with the `no-local-switching` command parameter, use the `port-egress-show` command:

```
CLI (network-admin@switch) > port-egress-show

switch  port      egress  rx-only  active-active-vlags  loopback  mir-prevent-out  no-local-switching-out
------  --------  ------  -------  -------------------  --------  ---------------  ----------------------
1       0-72      none    none     none                 none      none             none
2       0-72      none    none     none                 none      none             none
3       0-72      none    none     none                 none      none             none
4       0-72      none    none     none                 none      none             none
5       0-4,11-72 none    none     none                 none      none             5-10
6       0-4,11-72 none    none     none                 none      none             5-10
7       0-4,11-72 none    none     none                 none      none             5-10
8       0-4,11-72 none    none     none                 none      none             5-10
```

The `no-local-switching` configuration option is available also for the `trunk-create`, `trunk-modify` and `trunk-show` commands, as follows:

```
CLI (network-admin@switch) > trunk-create
```

`trunk-create`	Create a trunk configuration for link aggregation
One or more of the following options:	
`local-switching|no-local-switching`	Specify local-switching or no-local-switching. A no-local-switching port cannot bridge traffic to another no-local-switching port

```
CLI (network-admin@Leaf1) > trunk-modify
```

`trunk-modify`	Modify a trunk configuration for link aggregation
One or more of the following options:	
`local-switching|no-local-switching`	Specify local-switching or no-local-switching. A no-local-switching port cannot bridge traffic to another no-local-switching port.

```
CLI (network-admin@Leaf1) > trunk-show
```

`trunk-show`	Display trunk configuration
One or more of the following options:	
`local-switching|no-local-switching`	Specify local-switching or no-local-switching. A

	no-local-switching port cannot bridge traffic to another no-local-switching port.

Limiting the Number of MAC Addresses per Port

In NetVisor OS you can limit the number of MAC addresses learned on a per port basis. You can configure this capability (sometimes referred to as *port security*) on ports or trunks, for example like so:

```
CLI (network-admin@switch) > mac-limit-modify port 45,49 mac-limit 5 mac-limit-action log
```

The configuration options for the command are:

```
CLI (network-admin@switch) > mac-limit-modify
```

`port port-list`	Specify the port list.
`mac-limit mac-limit-number`	Specify the number of MAC addresses to limit on the port.
`mac-limit-action log|disable`	Specify the action to take when the MAC address limit is exceeded. If you select log, an event is logged to the event log. If you specify disable, the event is logged and the port is disabled.

To display the MAC limits, you can use:

```
CLI (network-admin@switch) > mac-limit-show
```

`port port-list`	Displays the port list.
`mac-limit mac-limit-number`	Displays the number of MAC addresses to limit on the port.
`mac-limit-action log|disable`	Displays the action taken when the MAC address limit is exceeded.
`num-macs num-macs-number`	Displays the number of MAC addresses learned on the port.

```
CLI (network-admin@switch) > mac-limit-show

port  mac-limit  mac-limit-action  num-macs
----  ---------  ----------------  --------
5     0          log               0
5     0          log               0
```

Monitoring vPort-based Activity

In NetVisor OS *virtual ports*, in short *vPorts*, are software-based Layer 2 entries associated to any ports a switch performs MAC address learning on.

While a plain-vanilla hardware Layer 2 table is limited in its capacity by a switch's dedicated ASIC memory size, the NetVisor OS software runs in the control plane processor's much larger DRAM memory space and hence is capable of tracking a large list of Layer 2 entries. Such list can grow much larger than what can fit into the limited space of a hardware table. This logical software extension of the Layer 2 table is the *vPort database*.

vPort database entries are persistent and are synchronized across the fabric so that every fabric member is aware of every other Layer 2 table entry. This capability is particularly useful for instance to track (physical or virtual) mobile end-points as well as users so as to make sure that movements are legitimate. (See also the *Understanding Fabric Status Updates, vPorts and Keepalives* section.)

In a nutshell, the vPort database can be considered the authoritative *distributed endpoint directory and switching activity history book* of the entire unified cloud fabric. Its innate flexibility lends itself very well to security applications, for example in the case of ARP spoofing attacks.

Let's first see what a portion of the vPort datable may look like. The output of a `vport-show` command is displayed below with the two key columns in bold typeface (in this example, the inspected forwarding context is a VLAN, but can also be a VXLAN domain):

```
CLI network-admin@switch > vport-show format ip, mac, hostname, vlan, last-
active

ip          mac               vlan hostname last-active
----------   ----------------- ---- -------- -------------------
192.168.1.3 00:50:56:b2:73:e1     7 db-serv1 2014-08-07,12:25:11
192.168.1.6 12:5c:19:69:25:30   123 db-serv2 now
192.168.1.9 d6:f9:8a:29:25:44    42 db-serv1 2014-08-07,12:25:11
```

In case of ARP poisoning attacks, the *dynamically populated* IP-to-MAC-address-binding entries (shown above) may be corrupted by a malicious end-point pretending to be someone else and sending a (series of) forged ARP packet(s) to overwrite legitimate gleaned information.

A simple protection for this exploit is to proactively create (with the `vport-create` command) or reactively make (as shown below) a forge-proof static binding in the vPort database. An ARP spoofing detection syslog message is printed when someone attempts the now-blocked forgery, as exemplified below:

```
CLI (network-admin@switch) > vport-modify mac 00:50:56:b2:73:e1 vlan 7 ip
192.168.1.3 intf 11
[…]
ARP spoofing detection syslog message:
mac/ip access denied: mac=00:50:56:b6:f9:10 ip=192.168.1.3 caller=ARP
```

Configuring DHCP Snooping

NetVisor OS supports DHCP snooping as a security feature enabling the network admin to prevent denial-of-service (DoS) or Man-in-the-Middle (MiM) attacks from rogue DHCP agents.
You can define trusted ports to connect to the known good DHCP servers. DHCP snooping also maintains a mapping table for current assignments.

Enable DHCP snooping and specify the list of trusted server ports using the following command:

```
CLI (network-admin@switch) > dhcp—filter-create name name-string trusted-ports port-list
```

`name name-string`	Specify a name for the filter.
`trusted-ports port-list`	Specify a list of trusted ports.

The port list can then be modified or deleted with the following commands:

```
CLI (network-admin@switch) > dhcp-filter-modify name name-string trusted-ports port-list
```

`name name-string`	Specify the name for the filter to modify.
`trusted-ports port-list`	Specify a list of trusted ports.

To delete:

```
CLI (network-admin@switch) > dhcp-filter-delete name name-string
```

A DHCP filter can be shown with the command:

```
CLI (network-admin@switch) > dhcp-filter-show name name-string trusted-ports port-list vlan vlan-list
```

`name name-string`	Displays the name of the filter.
`trusted-ports port-list`	Specify a list of trusted ports.
`vlan vlan-list`	Displays a list of VLANs.

In order to drop packets from rogue DHCP agents connected through untrusted ports, NetVisor OS supports a specific system vFlow entry in hardware: DHCP-LOG-DROP.

The vFlow entry sends the packets to the CPU in order to track and log the untrusted DHCP messages, and then drops them. This entry is set to a higher precedence than the one used for trusted DHCP ports.

Ports that are not in the trusted list connect to hosts whose DHCP trust level is unknown or zero, therefore NetVisor OS ensures that the DHCP messages to be logged by the CPU are rate limited using a dedicated `dhcp` class so that its processing capacity is not exceeded. (See the `Configuring CPTP section` for more details.)

The output for the `dhcp-lease-show` command has two new parameters to display trusted and untrusted DHCP agents:

```
CLI (network-admin@Spine1) > dhcp-lease-show trusted-server|no-trusted-server

CLI (network-admin@Spine1) > dhcp-lease-show format ip, mac, port, vlan, db-state, server, server-ip, server-port, trusted-server, last-msg

ip        mac                port vlan db-state   server server-ip   server-port trusted-server last-msg
--------  -----------------  ---- ---- ---------  ------ ----------  ----------- -------------- --------
10.1.1.2 00:12:c0:80:1f:b8 9    1    unknown          10.1.1.100 65          no             offer
```

Log messages indicate the presence of an unknown or rogue DHCP agent:

```
DHCP server message received from untrusted port=<x> server-ip=<ip-addr>
```

Configuring Router Advertisement Guard

The IPv6 RA Guard feature requires the creation of a filter for addresses and prefixes in order to apply a security profile to RA messages.

To configure the RA Guard feature, follow these steps:

1. Create an access list using the command: `access-list-create`.
2. Create a prefix list using the command: `prefix-list-create`.
3. Create an IPv6 security profile using the command: `ipv6security-raguard-create`.

This configuration installs two vFlow entries specific for RA Guard:

- One vFlow entry drops RAs sent by devices with the role of host as assigned using the `ipv6security-raguard-create` command.
- The second vFlow entry sends RAs to the CPU on ports configured with the role of `router`.

The router advertisements are received and examined, then the necessary action is taken based on the access and prefix lists, or on port and VLAN policies. A permitted router advertisement is flooded to its destination ports.

These are the commands to configure RA Guard:

```
CLI (network-admin@switch) >  access-list-create
```

`name` *`name-string`*	Specify a name for the access list.
`scope` *`scope`*	Specify if the scope is local or fabric.

```
CLI (network-admin@switch) > access-list-delete name name-string
```

`name` *`name-string`*	Specify the name for the access list to delete.

```
CLI (network-admin@switch) > access-list-show

switch name scope
------ ---- -----
spine1 test local


CLI (network-admin@switch) > access-list-ip-add
```

`name` *`name-string`*	Specify a name for the access list.

ip *ip-address*	Specify the IP address for the access list.

```
CLI (network-admin@switch) > access-list-ip-delete name name-string ip ip-address

CLI (network-admin@switch) > access-list-ip-show

switch    name  ip
--------  ----  -------
spine1    test  1.1.1.4

CLI (network-admin@switch) > prefix-list-create
```

name *name-string*	Specify a name for the prefix list.
scope *scope*	Specify if the scope is local or fabric.

```
CLI (network-admin@switch) > prefix-list-delete name name-string

CLI (network-admin@switch) > prefix-list-show
```

name *name-string*	Displays the name for the prefix list.
scope *scope*	Displays if the scope is local or fabric.

```
CLI (network-admin@switch) > prefix-list-network-add
```

name *name-string*	Specify the name for the prefix network list.
network *ip-address*	Specify the IP address for the network.
netmask *netmask*	Specify the netmask.

```
CLI (network-admin@switch) > prefix-list-network-delete name name-string

CLI (network-admin@switch) > prefix-list-network-show
```

name *name-string*	Displays the name for the prefix network list.
network *ip-address*	Displays the IP address for the network.

netmask *netmask*	Displays the netmask.

CLI (network-admin@switch) > ipv6security-raguard-create

name *name-string*	Specify the RA policy name.
device host|router	Specify the type of device as host or router.
router-priority low|medium|high	Specify the router priority as low, medium, or high.
access-list *name-string*	Specify the access list name.
prefix-list *name-string*	Specify the prefix list name.

CLI (network-admin@switch) > ipv6security-raguard-delete

name *name-string*	Specify the RA policy name.

CLI (network-admin@switch) > ipv6security-raguard-modify

name *name-string*	Specify the RA policy name.
device host|router	Specify the type of device as host or router.
router-priority low|medium|high	Specify the router priority as low, medium, or high.
access-list *name-string*	Specify the access list name.
prefix-list *name-string*	Specify the prefix list name.

CLI (network-admin@switch) > ipv6security-raguard-show

name *name-string*	Displays the RA policy name.
device host|router	Displays the type of device as host or router.
router-priority low|medium|high	Displays the router priority as low, medium, or high.
access-list *name-string*	Displays the access list name.
prefix-list *name-string*	Displays the prefix list name.

CLI (network-admin@switch) > ipv6security-raguard-port-add

name *name-string*	Specify the name of the RA Guard policy to add ports.
ports *port-list*	Specify the list of ports to add to the policy.

```
CLI (network-admin@switch) > ipv6security-raguard-port-remove
```

`name` *`name-string`*	Specify the name of the RA Guard policy to remove ports.
`ports` *`port-list`*	Specify the list of ports to remove from the policy.

```
CLI (network-admin@switch) > ipv6security-raguard-port-show
```

`name` *`name-string`*	Displays the name of the RA Guard policy.
`ports` *`port-list`*	Displays the list of ports.

```
CLI (network-admin@switch) > ipv6security-raguard-vlan-add
```

`name` *`name-string`*	Specify the name of the RA Guard policy to add VLANs.
`vlans` *`vlan-id`*	Specify the VLANs to add to the policy.

```
CLI (network-admin@switch) > ipv6security-raguard-vlan-remove
```

`name` *`name-string`*	Specify the name of the RA Guard policy to remove VLANs.
`vlans` *`vlan-id`*	Specify the VLANs to remove from the policy.

```
CLI (network-admin@switch) > ipv6security-raguard-vlan-show
```

`name` *`name-string`*	Displays the name of the RA Guard policy to add VLANs.
`vlans` *`vlan-id`*	Displays the VLANs to add to the policy.

Configuring MAC-based ACLs

You can configure MAC-based ACL entries to permit or deny network traffic based on a number of parameters:

- Source MAC address
- Source MAC address mask
- Destination MAC address
- Destination MAC address mask
- EtherType value
- Name of the vNET
- Bridge domain name
- VLAN number
- Switch port number

These capabilities enable various common use cases as exemplified in the following sections.

Configuring a MAC-based ACL to Deny Network Traffic

Figure 12-2 below shows the example of a server MAC address and Ethertype (IPv4) that you want to block when the traffic is ingressing a switch:

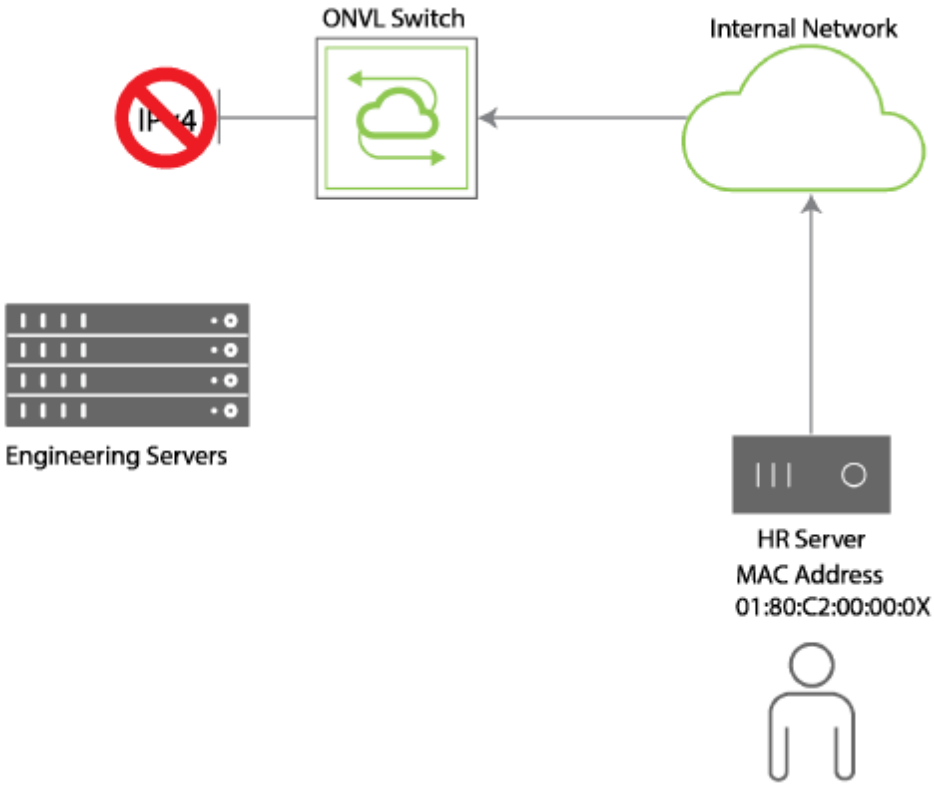


Figure 15-3 - MAC ACL Blocking Access

To deny IPv4 network traffic from MAC address, `01:80:c2:00:00:0X`, for the scope `fabric`, create the MAC ACL, `deny-MAC`, using the following syntax:

```
CLI (network-admin@switch) > acl-mac-create name deny-mac action deny src-
mac 01:80:c2:00:00:0X ether-type ipv4
 scope fabric
```

To review the configuration, use the `acl-mac-show` command:

```
CLI (network-admin@switch) >acl-mac-show name deny-mac layout vertical

name:                           deny-mac
id:                             b000015:12
action:                         deny
src-mac:                        01:80:c2:00:00:0X
dst-mac:                        00:00:00:00:00:00
dst-mac-mask:                   aa:aa:aa:aa:aa:aa
ether-type:                     ipv4
vlan:                           0
scope:                          fabric
port:                           0
```

Configuring a MAC-based ACL to Permit Network Traffic

In addition to the deny action, it is also possible to permit network traffic by using Layer 2 parameters, as shown in the example below:

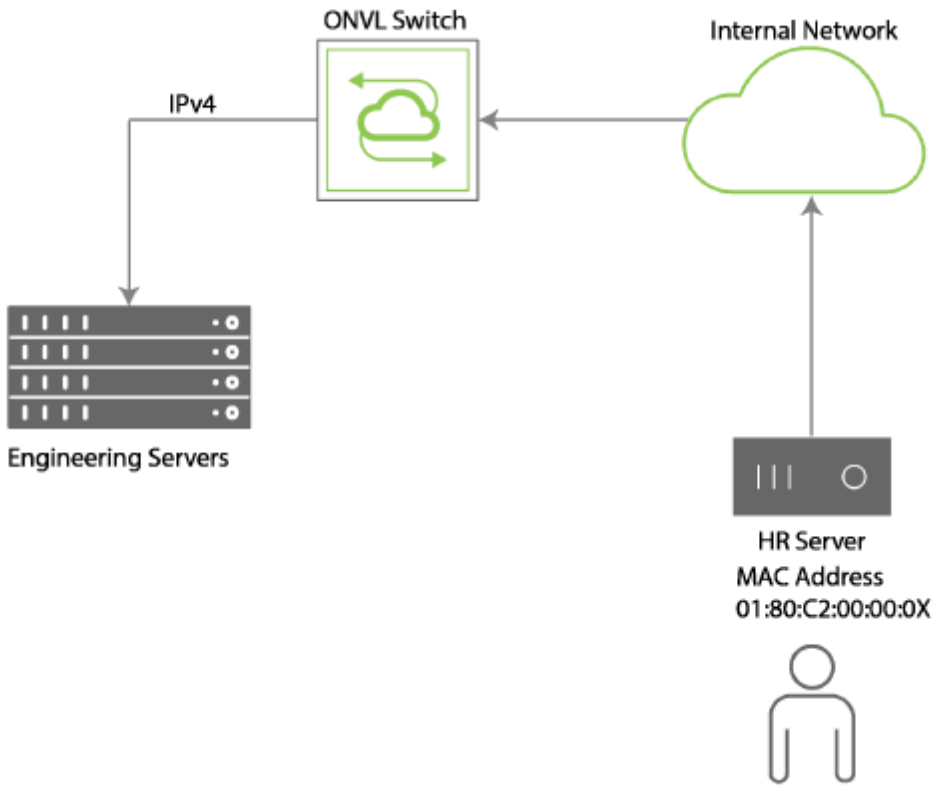


Figure 15-4 - MAC ACL Allowing Access

To permit IPv4 network traffic from MAC address `01:80:c2:00:00:0X`, create a MAC-based ACL using the following syntax:

```
CLI (network-admin@switch) >  acl-mac-create name allow-mac action permit
src-mac 01:80:c2:00:00:0X ether-type ipv4 scope fabric
```

To review the configuration, use the `acl-mac-show` command:

```
CLI (network-admin@switch) >  acl-mac-show name deny-mac layout vertical

name:                           deny-mac
id:                             b000015:12
action:                         deny
src-mac:                        01:80:c2:00:00:0X
dst-mac:                        00:00:00:00:00:00
dst-mac-mask:                   aa:aa:aa:aa:aa:aa
ether-type:                     ipv4
vlan:                           0
scope:                          fabric
port:                           0
```

To delete the ACL configuration, use the `acl-mac-delete` command.

To modify the ACL configuration, use the `acl-mac-modify` command.

Configuring IP-based ACLs

In IP-based ACLs the following parameters are available:

- Source IP address
- Source netmask
- Destination IP address
- Destination netmask
- IP Protocol (`tcp` | `udp` | `icmp` | `igmp` | `ip` | `icmpv6`)
- Source Layer 4 Port
- Destination Layer 4 Port
- vNET
- Bridge Domain
- VLAN Number
- Port Number

Using an IP-based ACL to Block IP Traffic

The network example shown in ***Figure 15-5 - IP-based ACL for Internal Servers*** below includes a Finance server in one part of the network and an Engineering server in another part.

For example, the network admin may want to block the Engineering server from communicating with the IP protocol with the Finance server in order to protect sensitive company information. For that purpose, you can use an IP-based ACL which can block a specific IP traffic source.

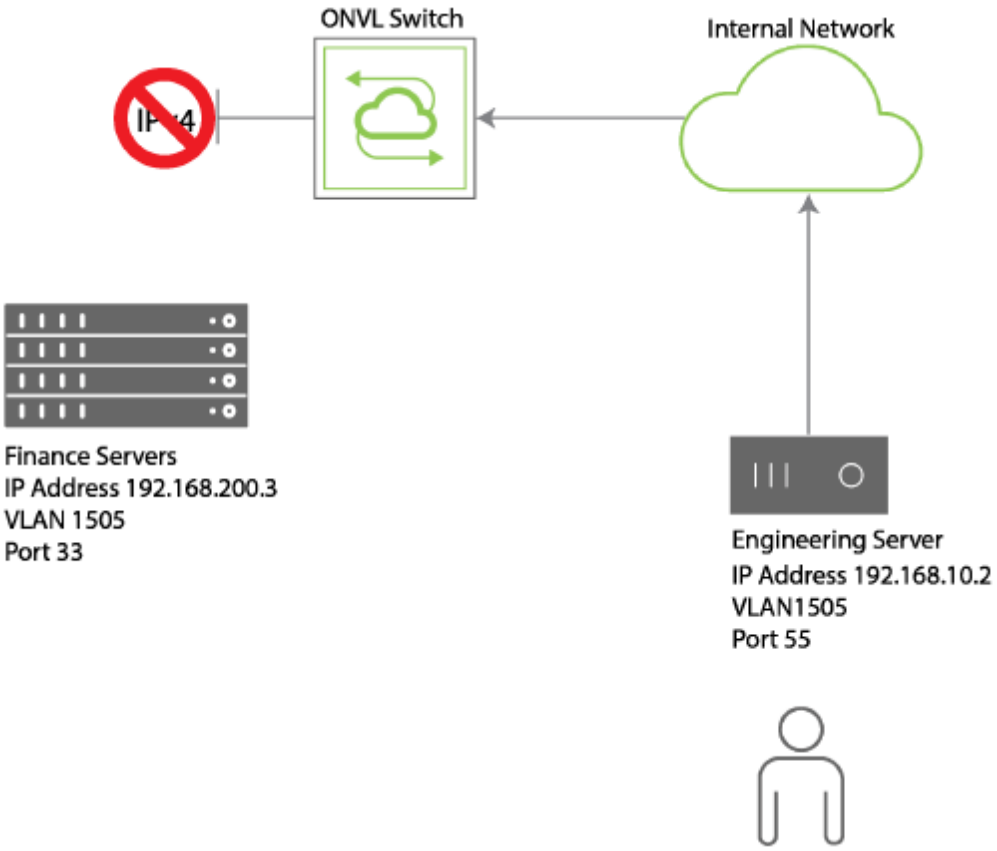


Figure 15-5 - Network Example - IP ACL for Internal Servers

You can use an IP-based ACL also to protect internal devices from external access as depicted in ***Figure 12-5 – IP-based ACL Blocking External Access*** below.

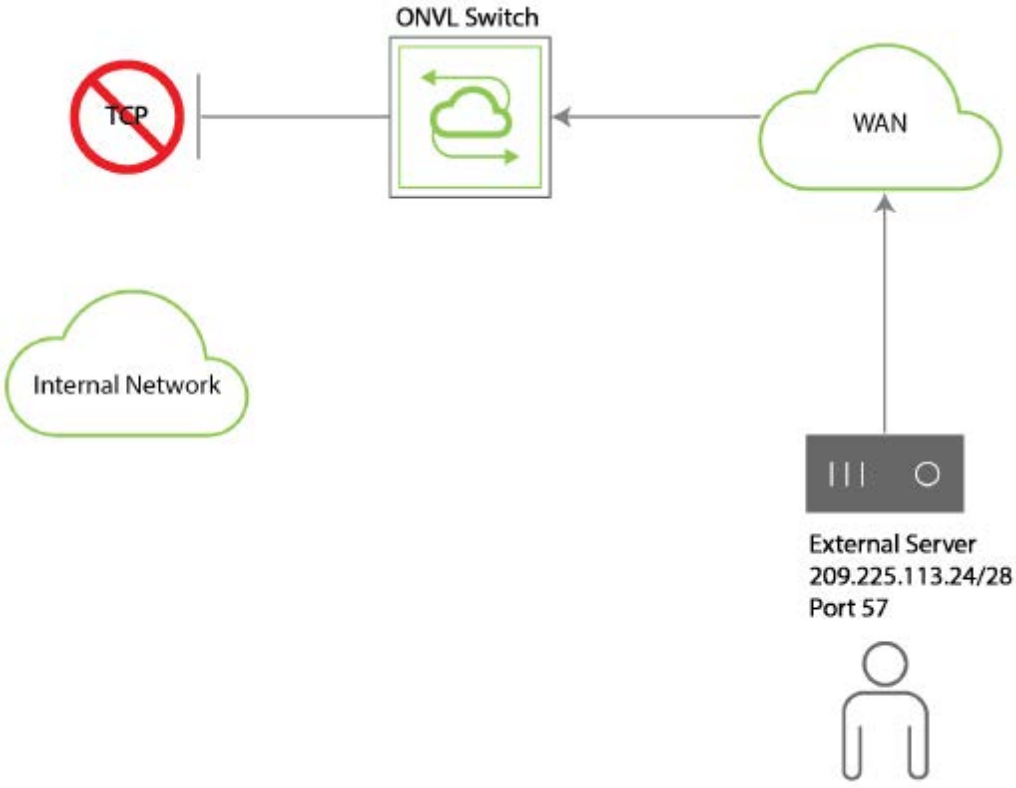


Figure 15-6 - IP ACL Blocking External Access

Configuring an IP-based ACL for Internal Servers

To configure an ACL for denying TCP traffic from the Engineering server to the Finance server you can use the following syntax:

```
CLI (network-admin@switch) > acl-ip-create name deny-finance action deny
scope local src-ip 192.168.10.2 src-ip-mask 24 dst-ip 192.168.200.3 dst-ip-
netmask 24 proto tcp src-port 55 dst-port 33 vlan 1505
```

To review the configuration, use the `acl-ip-show` command:

```
CLI (network-admin@switch) > acl-ip-show name deny-finance layout vertical
name:                   deny-finance
id:                     b00011:20
action:                 deny
proto:                  tcp
src-ip:                 192.168.10.2/24
src-port:               55
dst-ip:                  192.168.200.3/24
dst-port:               33
vlan:                   1505
scope:                  local
port:                   0
```

Configuring an IP-based ACL Blocking External Access

Referring to ***Figure 15-5 – IP-based ACL Blocking External Access*** above, a certain source IP address needs to be blocked.

To configure an ACL to deny traffic from the external server, use the `acl-ip-create` command to create an ACL named `deny-external` like so:

```
CLI (network-admin@switch) > acl-ip-create name deny-external scope fabric
src-ip  209.255.113.24/28
```

To review the configuration, use the `acl-ip-show` command:

```
CLI (network-admin@switch) > acl-ip-show name deny-external layout vertical

name:                   deny-external
id:                     b000022:20
action:                 deny
proto:                  ip
src-ip:                  209.225.113.24/28
src-port:               0
dst-ip:                 ::/0
dst-port:               0
```

```
vlan:               0
scope:              fabric
port:               0
```

Configuring an IP-based ACL to Permit Network Traffic

ACLs can be used to permit traffic too: for example, to allow HTTP traffic to the external subnet `209.225.113.24` with a netmask of `255.255.255.240` and a scope of `fabric`, you can create an IP-based ACL called `allow-http` using the following syntax:

```
CLI (network-admin@switch) > acl-ip-create name allow-http permit scope
fabric src-ip 0.0.0.0 src-ip-mask 255.255.255.255 dst-ip 209.225.113.24
dst-ip-mask 255.255.255.240 protocol tcp dst-port 57
```

To review the configuration, use the `acl-ip-show` command:

```
CLI (network-admin@switch) > acl-ip-show name allow-http layout vertical

name:               allow-http
id:                 b000025:20
action:             allow
proto:              tcp
src-ip:              0.0.0.0/255.255.255.255
src-port:           0
dst-ip:              209.225.113.24/28
dst-port:           57
vlan:               0
scope:              fabric
port:               0
```

To delete the ACL, use the `acl-ip-delete` command.

To modify the ACL configuration, use the `acl-ip-modify` command.

Configuring DSCP to CoS Mapping

NetVisor OS supports creating Quality of Service (QoS) maps that configure hardware based mapping of Differentiated Services Code Point (DSCP) value in a received IP header to a Cost of Service (CoS) priority. This helps in prioritizing traffic based on DSCP markings by using the appropriate egress CoS queues to send packets out.

NetVisor OS sets the DSCP value to the 6 upper bits in the 8-bit ToS field of an IP header. Details about the specific values and the proposed traffic disposition can be found in these RFCs :

- RFC 2474 (DS Fields Definitions)
- RFC 2475 (DiffServ architecture)
- RFC 2597 (AF PHB Group)
- RFC 2780 (IANA Allocation Guidelines)

A quick summary of DSCP in NetVisor OS:

- DSCP values range from 0 to 63, while packet priorities map to 8 CoS values or priority queues.
- Standards (IANA) include specific values in their guidelines. These values are used by different vendors to facilitate inter-connectivity.
- Class selector code points (CS0 through CS7, multiples of 8) are backwards compatible with IP ToS values. These values also serve as base selectors for other values.
- Assured Forwarding (AF) code points have 4 priority classes, each class has three code points indicating the drop precedence.
    - Class1: AF11/12/13 (DSCP 10, 12, 14)
    - Class2: AF21/22/23 (DSCP 18, 20, 22)
    - Class3: AF31/32/33 (DSCP 26, 28, 30)
    - Class4: AF41/42/43 (DSCP 34, 36, 38
- 0 is best effort (CoS 0, default)
- 46 is an Expedited Forwarding (EF) code point, indicating critical traffic.

There are new commands to support this feature:

```
CLI  (network-admin@switch)  >  dscp-map-create
```

dscp-map-create	Create a DSCP priority mapping table with default DSCP to priority mappings.
name name-string	Create a name for the DSCP map

To delete, use the command:

```
CLI  (network-admin@switch)  >  dscp-map-delete
```

To view the configuration details, use the command:

```
CLI  (network-admin@switch)  >  dscp-map-show
```

This command displays output only if there are maps configured.

```
CLI (network-admin@switch) > dscp-map-pri-map-modify
```

dscp-map-pri-map-modify	Update priority mappings in tables.
dscp-map selector: name name-string	Specify the name for the DSCP map to modify.
Specify the following pri-map arguments:	
pri number	Specify a CoS priority from 0 to 7.
dsmap number-list	Specify a DSCP value(s)as a single value, comma separated list, or a number range.

To view the details, use the command:

```
CLI (network-admin@switch) > dscp-map-pri-map-show
```

The `dscp-map-pri-map-show` displays output only if there are maps configured.

The default values are listed in the following `dscp-map-pri-map-show` output:

```
CLI (network-admin@switch) > dscp-map-pri-map-show name dscp-map1

switch  name pri dsmap
------- ---- --- -----------
switch  ds2  0   none
switch  ds2  1   8,10,12,14
switch  ds2  2   16,18,20,22
switch  ds2  3   24,26,28,30
switch  ds2  4   32,34,36,38
switch  ds2  5   40
switch  ds2  6   48
switch  ds2  7   56
```

The command, `port-config-modify`, has a new parameter, `dscp-map map-name | none` to support this feature.

Using the option `none` deletes or cancels a DSCP map previously configured on the port.

Applying CoS Queue Mapping based on Re-Marked DSCP in vFlow

Currently, NetVisor OS allows a vFlow to mark or re-mark matched packets with a DSCP value on egress. NetVisor OS does not prioritize this traffic in terms of the egress port CoS queue selected for transmit. Another feature, *Enabling DSCP to Priority and CoS Mappings* introduces the ability to create DSCP QoS maps and apply to ports, but the maps apply to ingress packets. This feature introduces the ability prioritize traffic based on the remarked DSCP value in a vFlow

NetVisor OS enables you to create named DSCP maps as independent objects and applies the maps to ingress ports for prioritization of packets based on the DSCP markings. In this feature, you can apply the same maps in a vFlow. QoS maps can be applied to ports, but not to Flow Processor entries corresponding to vFlows. This implementation does the prioritization explicitly, since flows can be configured with CoS values.

The implementation has the following features:

- Verify the DSCP map named in the vFlow exists.
- Determine the priority and CoS for the DSCP value assigned to the vFlow.
- Apply this CoS value to the Flow Processor entry in hardware.
- Reconfigure CoS in the flow when the vFlow DSCP setting changes.
- Prevent deleting a DSCP map in use by a vFlow.
- Update the CoS setting of vFlows using the DSCP map when the DSCP map priority settings are updated.

You can specify the name of a DSCP map in the `vflow-create` command:

*`dscp-map dscp-map name	none`*	Specify the DSCP map to apply on the flow. Please reapply if map priorities are updated

Supported Releases

Command/Parameter	NetVisor OS Version
`mac-limit-modify, mac-limit-show`	Commands introduced in version 2.6.0
`drop option removed from mac-limit-action`	Option deprecated in version 5.1.0
`acl-ip-create, acl-ip-delete, acl-ip-modify, acl-ip-show`	Commands introduced in version 1.2.1
`igmp`	Parameter added to IP-based ACLs in version 2.4
`vnet`	Parameter added to IP-based ACLs in version 2.4.1
`acl-mac-create, acl-mac-delete, acl-mac-modify, acl-mac-show`	Commands introduced in version 1.2.1
`ipv6security-raguard-create, ipv6security-raguard-delete, ipv6security-raguard-modify, ipv6security-raguard-remove, ipv6security-raguard-show, ipv6security-raguard-vlan-add, ipv6security-raguard-vlan-remove, ipv6security-raguard-vlan-show`	Commands introduced in version 3.0.0
`dhcp—filter-create, dhcp—filter-modify, dhcp—filter-delete, dhcp—filter-show`	Commands introduced in version 2.6.0
`login-fail-count, lock-account|no-lock-account, minimum-pw-length`	Parameters added to user-create command in version 6.1.0

Please also refer to the Arista NetVisor OS Command Reference document.

Related Documentation

For further information on concepts mentioned in this section (such as Layer 2, Layer 3, etc.), refer to these sections of the Configuration Guide:

- Configuring Switch Ports
- Configuring Layer 2 Features
- Configuring Layer 3 Features
- Configuring and Administering the Unified Cloud Fabric
- Configuring and Using vFlows

Configuring and Using Network Telemetry

This chapter provides information about configuring and using the Network Telemetry (Analytics) features on a NetVisor OS switch.

About sFlow

As businesses rely on network services for mission critical applications, small changes in network usage can impact network performance and reliability. These changes can impact a business' ability to conduct important business functions, which can increase the cost of maintaining network services.

sFlow is a technology for monitoring traffic in data networks as defined by the Internet Engineering Task Force (IETF) in RFC 3176 and later superseded by version 5 in sflow_version_5.

The sFlow monitoring system consists of an sFlow Agent, embedded in a switch or a router, and a central sFlow Collector. The architecture and sampling techniques used in the sFlow monitoring system allows continuous monitoring of high speed traffic in data networks.

The sFlow system provides the data required to effectively control and manage network usage and supports application-level traffic flows at wire-speed on all physical interfaces. You can use this information for troubleshooting a network, performing diagnostics, and analyzing the data. This capability ensures that network services provide a competitive edge to the businesses.

In NetVisor, the sFlow monitoring system has two main components: the sFlow Collector and sFlow Agent. As displayed in ***Figure 16-1***, the sFlow Agent runs on Pluribus switches, samples the packets, and sends the packets to the sFlow Collector for further processing.

sFlow Collector: An sFlow Collector is a network device that receives sFlow packets from one or more sFlow Agents.

sFlow Agent: The sFlow Agent is a thread that runs on Pluribus switches and receives the sFlow packets from the hardware, modifies by adding the header and sends the packets to the sFlow Collector.

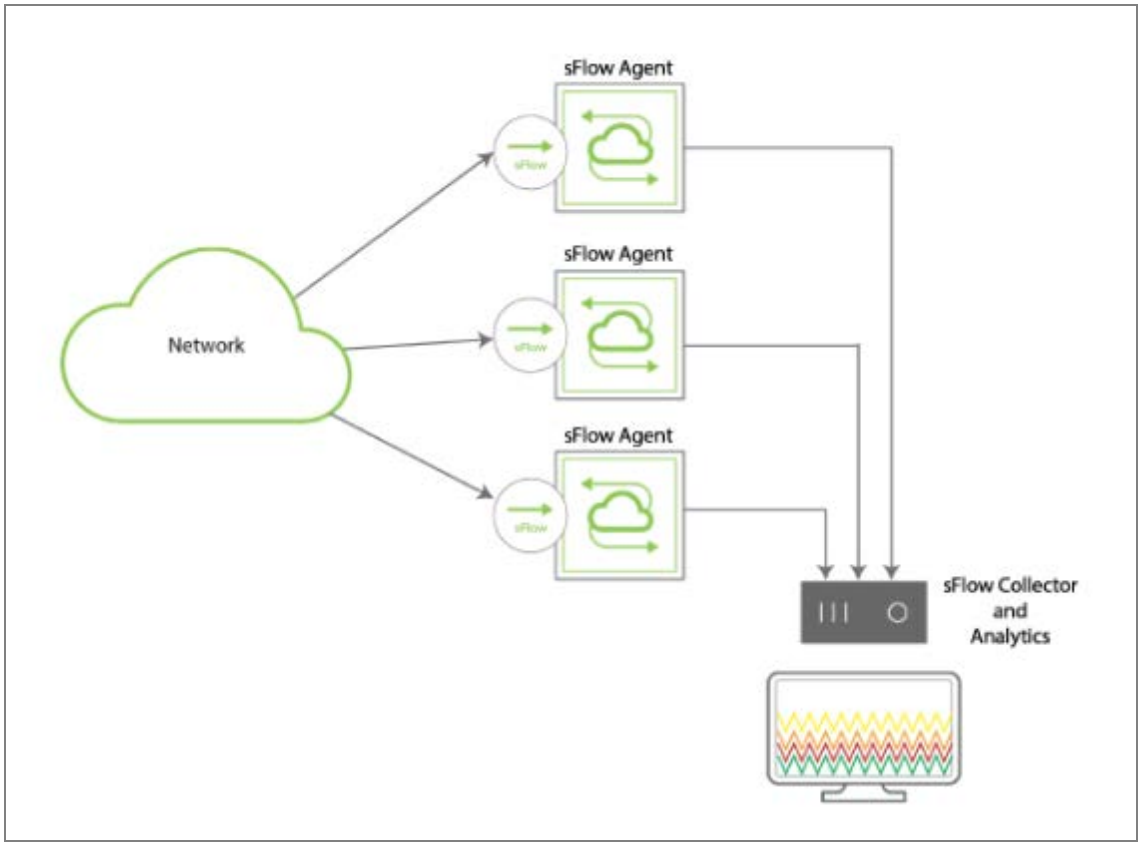


Figure 16-1 - Sample Topology of sFlow Monitoring System

Packet Sampling: Packet Flow Sampling refers to the statistical selection of a fraction of the Packet Flows observed at a Data Source. If the sFlow Agent is configured on Pluribus switches, then, NetVisor performs two sampling mechanisms:

- **Sampling Rate**: The packets are first sampled by the hardware and is passed onto the software where the sFlow thread adds sample header and sends the same to the Collector. You can configure the number of packets to sample from the total packets using the CLI option, `sample-rate`.
- **Counter Polling Interval**: You can configure a timer using the CLI option, `counter-polling-interval`. On expiry of the timer, NetVisor collects the statistics from the hardware and construct a sample with sFlow header and then sends the same to the Collector via the UDP socket.

You can sample different types of packets such as:
- Frames sent to the CPU or interfaces of the switch
- IP Options and MTU violations
- Flooded packets
- Multicast packets

However, the following packet types are not sampled by sFlow:
- LACP frames
- PAUSE frames
- PIM hello packets
- CRC error frames
- Packets dropped by ACLs or due to VLAN violations

The Pluribus switches support sFlow at the port level and the sFlow monitoring system supports two types of samplers: (i) Ingress sFlow sampler and (ii) Egress sFlow sampler. You can either configure one of the sampler types, or both the sampler types on the Pluribus switches to enable sampling of IN and OUT packets simultaneously.

If the configured sFlow Collector is unreachable due to any connectivity issue, then the sFlow Agent retries to send the packets every 60 seconds. During this time , the packets are recorded as *sFlow drop packets*. The sFlow packets could get dropped (not sent to sFlow Collector) when:

- The sFlow port is invalid
- The sFlow Agent fails to match the sample port
- There is a network connectivity issue
- The sFlow packets are malformed or are IPv6 packets
- There are malformed VLAN packets or oversized packets
- There are internal Queue drops

You can use the `sflow-show` command to display the datagram drop counts.
To enable sFlow on a switch, you should configure the following functions (described in subsequent sections):

1. Configuring the sFlow Collector
2. Add Exporter license (see the sFlow sections in the Pluribus Installation and User Guide for details on adding the license)

- Configuring sFlow Agents on the Network

Configuring the sFlow Collector

You must configure sFlow Collector before configuring the sFlow Agents. The sFlow Collector receives sFlow Datagrams or packets from the sFlow Agents.

To configure an sFlow Collector's IP address and port, use the command:

```
CLI (network-admin@switch) > sflow-collector-create collector-ip ip-address
collector-port collector-port-number name name-string scope local|fabric
```

`collector-ip` *`ip-address`*	Specify the IP address of the sFlow Collector.
`collector-port` *`collector-port-number`*	Specify the port number assigned to the sFlow Collector.
`name` *`name-string`*	Specify a name for the sFlow.
`scope local|fabric`	Specify if the scope is local or fabric. **Note**: If the scope is fabric, then the new switches that join the fabric receive the sFlow Collector configuration. If the scope is local, then the sFlow Collector is configured only on the local switch.

Below is an example configuration of an sFlow Collector with IP address 10.10.10.12, on port 2055. The Collector name is sf_test, and the scope is fabric.

```
CLI (network-admin@switch) > sflow-collector-create collector-ip
10.10.10.12 collector-port 2055 name sf_test scope fabric
```

Note: You can add as many Collectors as needed while configuring the sFlow monitoring system in a datacenter.

To delete an already configured sFlow Collector, use the command:

```
CLI (network-admin@switch) > sflow-collector-delete name name-string
```

Note: While deleting the sFlow Collectors, note that the last available Collector cannot be deleted if an sFlow is still configured and available. That is, if you want to delete all sFlow Collectors, you must first delete all the sFlows before deleting the last available Collector.

To modify an already created sFlow Collector, use the command:

```
CLI (network-admin@switch) > sflow-collector-modify name name-string
```

`name` *`name-string`*	Specify the name of the sFlow you want to modify.
Specify one or more of the following options:	
`collector-ip` *`ip-address`*	Specify the change in IP address for the sFlow Collector.
`collector-port` *`collector-port-number`*	Specify the change in port number for the sFlow Collector.

Note: You cannot modify the *scope* parameter once the sFlow Collector is configured.

To view the details of the configured sFlow Collectors, use the `sflow-collector-show` command specifying one or more of the following options:

```
CLI (network-admin@switch) > sflow-collector-show
```

Specify one of more of the following options:	
`collector-ip` *`ip-address`*	Specify to display the details of the sFlow Collector.
`collector-port` *`collector-port-number`*	Specify to display the details.
`name` *`name-string`*	Specify to display the details of the specified sFlow.
`scope local|fabric`	Specify to display the details of either the local scoped or fabric scoped sFlow Collectors.

For example, to view the details of the previously configured sFlow collector, use the command:

```
CLI (network-admin@switch) > sflow-collector-show

collector-ip collector-port name          scope
------------ -------------- ------------- ------
10.10.10.12  6343           sf_test       fabric
10.13.26.75  6343           sflow_collect local
```

Configuring sFlow Agents on the Network

You must first configure the sFlow Collector before configuring the sFlow Agent using the `sflow-create` command. You must configure and enable sFlow Agent on each switch that you want to be part of the sFlow monitoring system to monitor the network traffic.

To configure and enable an sFlow Agent on a switch, use the `sflow-create` command:

```
CLI (network-admin@switch) >  sflow-create name name-string
```

`name` *`name-string`*	Specify a name for the sFlow.
`type ingress|egress`	Specify whether the sFlow sampler type is for ingress or egress traffic.
`sample-type raw|cooked`	Specify the sFlow sample type, the default sample type is `raw`.
`ports` *`port-list`*	Specify the sFlow ports.
`sample-rate` *`256-16000`*	Specify the sFlow sampling rate. The value ranges from 256-16000 and the default value is 4096. Enter zero (0) if you want to disable sFlow sampling.
`counter-polling-interval 0..120`	Specify the sFlow interface statistics for counter polling interval. This is the periodic sampling or polling counters associated with a data source.
`trunc-length` *`trunc-length-number`*	Specify the truncated length of the sFlow sample (sample packet size).
`agent-id` *`ip-address`*	Specify the local IP address.

Note: While creating the sFlow, do not configure both the `sample-rate` and `counter-polling-interval` to zero (0). If you do so, an error message: `sflow-create: Sample-rate/counter-polling-interval both can't be 0` is displayed. However, later, you can change both values to zero (0) using the `sflow-modify` command if desired.

Note: When you configure the `sample-rate` and the `counter-polling-interval` to zero (0) using the `sflow-modify` command, then no packets are sent out for sFlow monitoring.

Below is an example configuration on a switch to enable the sFlow Agent, *net-monitor*, on the ingress ports 57-59, with sample-type raw, sample-rate *4096*, counter-polling-interval of 5 seconds (i.e., 1 sample in every 5 seconds), and trunc-length of *160* bytes:

```
CLI (network-admin@switch) >  sflow-create name net-monitor type ingress
sample-type raw ports 57-59 sample-rate 4096 counter-polling-interval 5
trunc-length 160
```

For deleting an existing sFlow Agent, use the command:

```
CLI (network-admin@switch) >  sflow-delete name name-string
```

To view the sFlow Agent details, use the command:

```
CLI (network-admin@switch) >  sflow-show
```

Note: To view the binding between the sFlows and the `nv-message-queues`, use the `sflow-thread-binding-show` command. This command helps during debugging process and should be run with the support from Pluribus TAC team. See the Example: *A Sample Configuration below*.

To modify the configuration details for an sFlow Agent, use the command:

```
CLI (network-admin@switch) >  sflow-modify name name-string
```

`name` *`name-string`*	Specify the sFlow name that you want to modify.
Modify one or more of the following options:	
`type ingress|egress`	Specify to change the sFlow type to ingress or egress traffic.
`sample-type raw|cooked`	Specify to change the sample type to raw or cooked. Default value is `raw`.
`ports` *`port-list`*	Specify to change the sFlow ports.
`sample-rate` *`256-16000`*	Modify to change the sFlow sampling rate. The default value is 4096.
`counter-polling-interval 0..120`	Modify and change the sFlow interface stats-counter polling interval.
`trunc-length` *`trunc-length-number`*	Modify the truncated length of the sFlow sample.
`agent-id` *`ip-address`*	Modify the local IP address.

For example, to modify and view the `counter-polling-interval` of the sFlow, *net-monitor*, use the command:

```
CLI (network-admin@switch*) >  sflow-modify name net-monitor counter-
polling-interval 120

CLI (network-admin@switch*) > sflow-show format all layout vertical
name:                   net-monitor
type:                   ingress
sample-type:            raw
ports:                  1,3,53,57-59
sample-rate:            4096
counter-polling-interval:120
sample-interval:        5
trunc-length:           160
agent id:               10.1.1.200
```

```
sample-pkt-cnt:          2845
sample-drops:            87
```

Based on the the configuration following details are displayed for the `sflow-show` command:

```
CLI (network-admin@switch*) > sflow-show
```

Specify one or more of the options:	
`name` *`name-string`*	Displays the name for the sFlow.
`type ingress|egress`	Displays whether the sFlow type is for ingress or egress traffic.
`sample-type raw|cooked`	Displays the sFlow sample type.
`ports` *`port-list`*	Displays the sFlow ports.
`sample-rate 256-16000`	Displays the sFlow sample rate.
`counter-polling-interval` *`0..120`*	Displays the sFlow interface stats-counter polling interval.
`trunc-length` *`trunc-length-number`*	Displays the truncated length of the sFlow sample.
`agent-id` *`ip-address`*	Displays the local IP address.
`sample-pkt-cnt sample-pkt-cnt-number`	Displays the sFlow sample packet count.
`sample-drops sample-drops-number`	Displays the sFlow sample dropped packets.
`malformed-packet-drops malformed-packet-drops-number`	Displays the sFlow malformed packet drops.
`malformed-vlan-packet-drops malformed-vlan-packet-drops-number`	Displays the sFlow malformed vlan packet drops.
`malformed-ip-packet-drops malformed-ip-packet-drops-number`	Displays the sFlow malformed ip packet drops.
`internal-q-drops internal-q-drops-number`	Displays the sFlow internal Q drops.
`oversize-packet-drops oversize-packet-drops-number`	Displays the sFlow over size packet drops.
`internal-nq-drops internal-nq-drops-number`	Displays the sFlow internal nQ drops

If all the command parameters are configured for an sFlow Agent, then the `sflow-show` command displays an output as given here:

```
CLI (network-admin@switch*) > sflow-show format all layout vertical

switch:                      switch
name:                        sf25
```

```
type:                          ingress
sample-type:                   raw
ports:                         25
sample-rate:                   4096
trunc-length:                  160
agent-id:                      10.14.22.100
sample-pkt-cnt:                66798
sample-drops:                  0
malformed-packet-drops:             0
malformed-vlan-packet-drops:        0
malformed-ip-packet-drops:     0
internal-q-drops:              0
oversize-packet-drops:         0
internal-nq-drops:             0
```

Example: A Sample Configuration

Below is a set of sample configuration details for the `sflows` created in Pluribus lab for further understanding.

```
CLI  (network-admin@switch*)  >  sflow-show
```

name	type	sample-type	ports	sample-rate	counter-polling-interval	trunc-length	agent-id	sample-pkt-cnt	sample-drops
s1	ingress	raw	1	4096	2	160	10.13.27.218	2198	89
s2	egress	raw	5	4096	2	160	10.13.27.218	2200	0
s3	ingress	raw	9	4096	2	160	10.13.27.218	4035	92
s4	egress	raw	13	4096	2	160	10.13.27.218	1131	0
s5	ingress	raw	25	4096	2	160	10.13.27.218	3916	88
s6	egress	raw	37	4096	2	160	10.13.27.218	5350	0
s7	ingress	raw	41	4096	2	160	10.13.27.218	2313384	86
s8	egress	raw	49	4096	2	160	10.13.27.218	1051	0
s9	ingress	raw	57	4096	2	160	10.13.27.218	2419	114
s10	egress	raw	65	4096	2	160	10.13.27.218	678	0
s11	ingress	raw	69	4096	2	160	10.13.27.218	71304	64
s12	egress	raw	73	4096	2	160	10.13.27.218	364897	0
s13	ingress	raw	77	4096	2	160	10.13.27.218	817011	112
s14	egress	raw	105	4096	2	160	10.13.27.218	672	0
s15	ingress	raw	109	4096	2	160	10.13.27.218	2698	159

The `sflow-show` command displays 15 sFlows Instances (s1-s15) configured using the `sflow-create` command. In NetVisor OS, when you configure the sFlows, every six sFlows are bound to one thread (sFlow Agent). To view the details, use the command:

```
CLI  (network-admin@switch*)  >  sflow-thread-binding-show

name  thread-id          nv-msg-queue-name  nv-cache-queue-name
----  ---------------    -----------------  -------------------
s1     140323777259264   sflow-packet-q-9   sflow-cache-q-9
s2     140323777259264   sflow-packet-q-9   sflow-cache-q-9
s3     140323777259264   sflow-packet-q-9   sflow-cache-q-9
s4     140323777259264   sflow-packet-q-9   sflow-cache-q-9
s5     140323777259264   sflow-packet-q-9   sflow-cache-q-9
s6     140323777259264   sflow-packet-q-9   sflow-cache-q-9
s7     140322640197376   sflow-packet-q-10  sflow-cache-q-10
s8     140322640197376   sflow-packet-q-10  sflow-cache-q-10
```

```
s9     140322640197376  sflow-packet-q-10  sflow-cache-q-10
s10    140322640197376  sflow-packet-q-10  sflow-cache-q-10
s11    140322640197376  sflow-packet-q-10  sflow-cache-q-10
s12    140322640197376  sflow-packet-q-10  sflow-cache-q-10
s13    140322665375488  sflow-packet-q-11  sflow-cache-q-11
s14    140322665375488  sflow-packet-q-11  sflow-cache-q-11
s15    140322665375488  sflow-packet-q-11  sflow-cache-q-11
```

From the above output, you can see that the first six sFlows Instances (s1-s6) have the same `thread-id`, `nv-msg-queue-name`, and `nv-cache-queue-name` parameters. Similarly, the next six sFlows Instances (s7-s12) have a new set of parameters and so on. This helps in understanding the bindings between the sFlows and the parameters.

In the above output:

- `thread-id`: is the unique identifier of the CPU resources associated to sFlow Instances (in groups of 6)
- `nv-msg-queue-name`: is the actual software queue used for sFlow data
- `nv-cache-queue-name`: is a pool of buffers. NetVisor OS allocates one memory block from a `cache_queue` whenever a sample is received from the hardware, which is then queued to the `nv_queue`.

To view the nv_queue details, use the command:

```
CLI (network-admin@switch*) > nv-queue-stats-show | grep sflow
```

name	q-high	q-in	q-out	q-delay-high	q-delay-avg	q-overflow	q-underflow
sflow-packet-q-9	2	18571	18571	13.8us	10.1us	0	0
sflow-cache-q-9	100	18571	18571	0.00ns	0.00ns	0	0
sflow-packet-q-10	4	2757320	2757320	141us	13.1us	0	0
sflow-cache-q-10	100	2757320	2757320	0.00ns	0.00ns	0	0
sflow-packet-q-11	2	821132	821132	50.7us	10.8us	0	0
sflow-cache-q-11	100	821132	821132	0.00ns	0.00ns	0	0

Guidelines to Remember While Configuring sFlow

There are some guidelines to consider while configuring the sFlow monitoring feature:

- The sFlow monitoring system is a local scoped feature.
- You can configure the sFlow sample rate between the range 256-16000, by default the sample rate is set to 4096. To disable the sFlow sampling, configure the sample rate to zero (0) .
- The sFlow Agent ID is typically a switch management IP address and indicates the originator or sample frames provider.
- The sFlow monitoring system supports only IPv4 sampling.
- The sFlow sample type can be either `raw` or `cooked mode`, and the default mode is `raw`.
- By default, the `counter-polling-interval` is 2 seconds and the available range is 1-120 seconds. However, to disable the counter sampling, you can set the interval to zero (0).
- The CLI parameter, `sample-pkt-cnt` is the actual sampled packets from the sFlow Agent.
- The sFlow Collector configuration requires the Collector IP address and Collector port list.
- The sFlow Collector configuration can be `local` or `fabric` scoped feature.
- You can configure multiple collector configurations, however, the first or the latest available Collector is selected for sending the sampled packets.
- The egress `sflow` samples do not make modification to the packets in the egress pipeline(for example, VLAN translation, VXLAN header encap).

Adding or Removing Additional Ports to sFlow

Based on the specific sampling requirements (see the configuration in the previous section (Configuring sFlow Agents on the Network), you can add additional ports to the sFlow Agent on each switch by modifying the desired sFlow Instances with the following command:

```
CLI (network-admin@switch) > sflow-port-add sflow-name name-string ports port-list
```

Specify the sflow selector:	
`sflow-name` *`name-string`*	Specify the sFlow name to which you want to add additional ports.
`ports` *`port-list`*	Specify the ports or range of ports.

For example, to add the ports, **`61-62,`** to the sFlow configuration, use the following command on each switch:

```
CLI (network-admin@switch) >  sflow-port-add sflow-name net-monitor ports 61-62
```

To remove the additional ports from the sFlow configuration, use the `sflow-port-remove` command. For example,

```
CLI (network-admin@switch) >  sflow-port-remove sflow-name net-monitor ports 61-62
```

Understanding gRPC-based Telemetry

gRPC (Google Remote Procedure Call) is a communication technology based on a modern open source high performance software framework that has been adopted in numerous applications.
There are several benefits when using gRPC: a secure HTTP/2 transport with TLS authentication and encryption support, an underlying message interchange format (protobuf), performance and scalability, flexible authentication, streaming capabilities, push/pull data exchange, etc.

gRPC-based telemetry leverages gRPC's benefits to remotely collect data from devices at high speed and with great scalability. Streaming of telemetry data over gRPC can help accelerate network troubleshooting, automation management, traffic analysis and optimization.

Unlike other network management protocols such as SNMP, gRPC telemetry uses the 'push' model to stream device data (such as statistics) from the network devices to the configured data collector(s). Collectors can then store, filter and analyze the collected data from the network devices.

Starting from NetVisor OS release 7.0.2, a switch can act as gRPC client and the data collector as gRPC server. The network device initiates a gRPC connection to the collector and pushes subscribed data to it. This configuration is known as *dial-out* mode.

Note: The other telemetry mode, called dial-in mode, in which the switch acts as the server and the collector acts as the client, is not supported in NetVisor OS release 7.0.2.

Note: By default, the gRPC telemetry service streams fabric-wide data.

In NetVisor OS release 7.0.2 only `connection-show` data is supported for telemetry export: the gRPC client running on a network device fetches the `connection-show` data from all nodes in the fabric (or from a specified node) and streams it to the gRPC collectors encoded as protocol buffer (protocolbuf) messages.

Protocol buffers are used in gRPC as an Interface Definition Language (IDL) for describing both the service interface and the structure of the payload of the messages.

Below is an example of protocol buffer definition for telemetry data:

```
syntax = "proto3";         // Current Protobuf version
package  telemetry; //Give a package name
service  NetvisorTelemetry  {
     //Define remote methods to stream data
     rpc  PushConnections  (stream  Connection)  returns  (Result)  {}
}

// Define the message
message Connection {   // Message Connection
   string  switch  =  1  [json_name  =  "api.switch-name"];
   uint32  sum_by_count  =  2  [json_name  =  "count"];
   string vnet = 3;
   string  br_dom  =  4  [json_name  =  "bd"];
   uint32 vlan = 5;
   string  egress_vnet  =  6  [json_name  =  "egress-vnet"];
```

```
        .
        .
        .
    }
```

For more information on the protocol buffers technology, refer to the developer documentation: https://developers.google.com/protocol-buffers.

Configuring gRPC-based Telemetry

You can configure gRPC-based telemetry with the following steps.

Step 1: Create a configuration object called destination group (to add telemetry collector devices to it) like so:

```
CLI (network-admin@switch) > telemetry-dst-group-create name <group-name>
```

which you can show and delete with the following commands:

```
CLI (network-admin@switch) > telemetry-dst-group-show [name <group-name>]

CLI (network-admin@switch) > telemetry-dst-group-delete name <group-name>
```

Then add one or more collectors to the destination group using the following command:

```
CLI (network-admin@switch) > telemetry-dst-group-collector-add name <group-
name> collector-ip <collector-ip-address> port <collector-port-number>
```

You can show and remove collectors from the group using the following commands:

```
CLI (network-admin@switch) > telemetry-dst-group-collector-show name
<group-name> [collector-ip <collector-ip-address>] [port <collector-port-
number>]

CLI (network-admin@switch) > telemetry-dst-group-collector-remove name
<group-name> collector-ip <collector-ip-address> port <collector-port-
number>
```

Step 2: Configure the data that you need to stream as part of a configuration object called sensor group.

Step 2.1: Create a sensor group like so:

```
CLI (network-admin@switch) > telemetry-sensor-group-create name <sensor-
group-name>
```

which you can show and delete with the following commands:

```
CLI (network-admin@switch) > telemetry-sensor-group-show [name <sensor-
group-name>]

CLI (network-admin@switch) > telemetry-sensor-group-delete name <sensor-
group-name>
```

Step 2.2: Add sensor paths to the sensor group: a sensor path describes a valid show command supported for telemetry. In NetVisor OS release 7.0.2 the only supported sensor path is the connection-show command. You can also specify a path with filters, for example like so: `connection-show src-ip 30.1.1.1.`

You can add a sensor path like so:

```
CLI (network-admin@switch) > telemetry-sensor-group-path-add name <sensor-
group-name> path <path-string>
```

which you can show and remove with the following commands:

```
CLI (network-admin@switch) > telemetry-sensor-group-path-show name <sensor-
group-name> [path <path-string>]

CLI (network-admin@switch) > telemetry-sensor-group-path-remove name
<sensor-group-name> path <path-string>
```

> **Note:** Sensor paths used by other vendors (such as Cisco and Juniper) usually indicate a YANG path or a subset of data definitions in a YANG model. NetVisor OS release 7.0.2 does not support YANG models.

Step 3: Create a subscription object to bind a sensor group to a destination group, so that the device can start exporting data from the specified sensors paths to the specified collectors.

You can create a telemetry subscription (a binding) like so:

```
CLI (network-admin@switch) > telemetry-subscription-create name
<subscription-name> dst-group <destination-group-name> sensor-group
<sensor-group-name>
```

and you can specify one or more of the following options:

```
sample-interval <duration>: #d#h#m#s
switch-name <fabric-node-name>
switch-group <switch-group-name>
```

With these options, you can configure how often the telemetry data is sampled and exported. Also, you can select which switch or switch group the sampling needs to be performed on.

> **Note:** The default sampling interval is 60 seconds. The minimum configurable sampling interval is 15 seconds.

You can show, modify or delete a subscription with the following commands:

```
CLI (network-admin@switch) > telemetry-subscription-show [name
<subscription-name>] [dst-group <destination-group-name>] [sensor-group
<sensor-group-name>] [switch-name <fabric-node-name>] [switch-group
<switch-group-name>] [sample-interval <duration>]

CLI (network-admin@switch) > telemetry-subscription-delete name
<subscription-name>

CLI (network-admin@switch) > telemetry-subscription-modify name
<subscription-name>
```

and you can specify one or more of the following options:

```
sample-interval <duration>: #d#h#m#s
```

```
dst-group  <destination-group-name>
sensor-group  <sensor-group-name>
switch-name  <fabric-node-name>
switch-group  <switch-group-name>
```

Note: NetVisor OS release 7.0.2 supports a maximum of 10 subscriptions.

Step 4: Enable (or disable) the gRPC service and configure any parameters such as TLS secure transport:

```
CLI (network-admin@switch) > telemetry-grpc-settings-modify
```

you can specify one or more of the following options:

```
enable-service|  disable-service
tls-enable|no-tls-enable
no-tls-enable  corresponds  to  insecure  communication.
```

You can then display the settings:

```
CLI  (network-admin@switch)  >  telemetry-grpc-settings-show
```

Note: By default, the gRPC telemetry service uses TLS credentials and looks for a client certificate at this location: `/var/nvos/cert/grpc/ca_cert.pem`. So in NetVisor OS release 7.0.2 certificate management is manual.

Note: All gRPC telemetry commands are locally scoped. It is sufficient to enable telemetry services on one fabric node.

Example of Configuration:

1. Configure destination groups and collectors:

```
CLI  (network-admin@switch)  >  telemetry-dst-group-create  name  dg1
CLI  (network-admin@switch)  >  telemetry-dst-group-collector-add  name  dg1
collector-ip  192.168.99.20  port  50051

CLI  (network-admin@switch)  >  telemetry-dst-group-show
switch  name
------  ----
switch  dg1

CLI  (network-admin@switch)  >  telemetry-dst-group-collector-show
switch  name  collector-ip   port
------  ----  ------------   -----
switch  dg1   192.168.99.20  50051
```

2. Configure sensor groups and sensor paths:

```
CLI  (network-admin@switch)  >  telemetry-sensor-group-create  name  sg1
CLI  (network-admin@switch)  >  telemetry-sensor-group-path-add  name  sg1  path
"connection-show"
```

```
CLI (network-admin@switch) > telemetry-sensor-group-show
switch name
------ ----
switch sg1

CLI (network-admin@switch) > telemetry-sensor-group-path-show
switch name path
------ ---- ---------------
switch sg1   connection-show
```

Other examples for the sensor path string can be:

```
connection-show src-ip x.x.x.x
connection-show within-last 15m
```

> **Note:** In the sensor path string you cannot specify any format or switch argument such as in this invalid examples: connection-show format field1,field2 or switch switch1 connection-show .

3. Configure subscriptions:

```
CLI (network-admin@switch) > telemetry-subscription-create name sub1 dst-
group dg1 sensor-group sg1 sample-interval 1m

CLI (network-admin@switch) > telemetry-subscription-show
switch name dst-group sensor-group sample-interval
------ ---- --------- ------------ ---------------
switch sub1  dg1       sg1          1m
```

To stream telemetry data of a specific switch only (instead of all fabric nodes), specify the switch name in the configuration, as shown below:

```
CLI (network-admin@switch) > telemetry-subscription-create name sub1 dst-
group dg1 sensor-group sg1 switch-name switch1

CLI (network-admin@switch) > telemetry-subscription-show
switch name dst-group sensor-group switch-name sample-interval
------ ---- --------- ------------ ----------- ---------------
switch sub2  dg1       sg1          switch1     60s
```

To stream telemetry data from a group of switches, you can specify a switch group as shown below:

```
CLI (network-admin@switch) > switch-group-create name sw_grp_1 description
"telemetry switch group"

CLI (network-admin@switch) > switch-group-member-add name sw_grp_1 member
switch1,switch2

CLI (network-admin@switch) > switch-group-member-show
switch name      member
------ -------- -------
switch sw_grp_1 switch1
```

```
switch sw_grp_1 switch2

CLI (network-admin@switch) > telemetry-subscription-modify name sub1 dst-
group dg1 switch-group sw_grp_1

CLI (network-admin@switch) > telemetry-subscription-show
switch name dst-group sensor-group switch-group sample-interval
------ ---- --------- ------------ ------------ ---------------
switch sub1  dg1       sg1          sw_grp_1     60s
```

4. Enable the gRPC telemetry service:

```
CLI (network-admin@switch) > telemetry-grpc-settings-modify enable-service

CLI (network-admin@switch) > telemetry-grpc-settings-show
enable-service: yes
state:          enabled
```

Developing a Collector Application

The gRPC telemetry data from a NetVisor OS switch can be sent to a collector that can be a tool developed in house or a commercial data monitoring application.

To create your own tool, obtain the `nvos.proto` file for NetVisor OS so that the collector can receive and resolve the data obtained from the network device.

Utilize any of the gRPC-supported languages (e.g., C++, Java, python, Go, etc.) to develop the collector application. The example below uses the Go language.

Generate the server code from the `nvos.proto` file using the protocol buffer compiler `protoc`:

```
protoc --go_out=. --go_opt=paths=source_relative --go-grpc_out=. \   --go-
grpc_opt=paths=source_relative   nvos.proto
```

Import the generated package:

```
import (
pb "path the nvos.proto generate directory"
)
```

Define the struct to implement the generated `nvos_proto.TelemetryServer` interface:

```
type server struct {
pb.UnimplementedTelemetryServer
}
```

Implement the interface method to read the streamed connections and to return a response:

```
func (s *server) PushConnections(stream pb.Telemetry_PushConnectionsServer)
```

```
error {
     for {
          conn, err := stream.Recv()
          if err == io.EOF {
               // Close the connection and return the response to the client
               return  stream.SendAndClose(&pb.Response{
                    Status:  pb.Response_SUCCESS,
                    Message: "",
               })
          }
          if err != nil {
               //Handle any possible errors
          }
          log.Printf("Connection Data received: %v", conn)
          //The data received here can be forwarded to any database or
monitoring tool
     }
}
```

Then start the server:

```
//specify the address to listen to
lis, err := net.Listen("tcp", ":<port no>")
if err != nil {
log.Fatalf("failed to listen: %v", err)
}
//Load the credentials.
creds, err := credentials.NewServerTLSFromFile("server_cert.pem",
"server_key.pem")
// create grpc server
s = grpc.NewServer(grpc.Creds(creds))
pb.RegisterTelemetryServer(s,  &server{})
log.Printf("server listening at %v", lis.Addr())
// and start
if err := s.Serve(lis); err != nil {
     log.Fatalf("failed to serve: %v", err)
}
```

Using Wireshark to Analyze Packets in Real Time

To use Wireshark to interactively analyze packets in real time, you need to capture a packet traffic flow, either on a specific switch or across the entire fabric using the scope option. For example:

```
CLI (network-admin@switch) > vflow-snoop scope fabric src-ip
112.168.3.105 action copy-to-cpu
```

Next, create a fifo on the host running Wireshark.

```
mkfifo /tmp/pcap
```

Start Wireshark, and select **Options** from the **Capture** menu.

Enter the fifo path that you created in the Interface field: `/tmp/pcap`

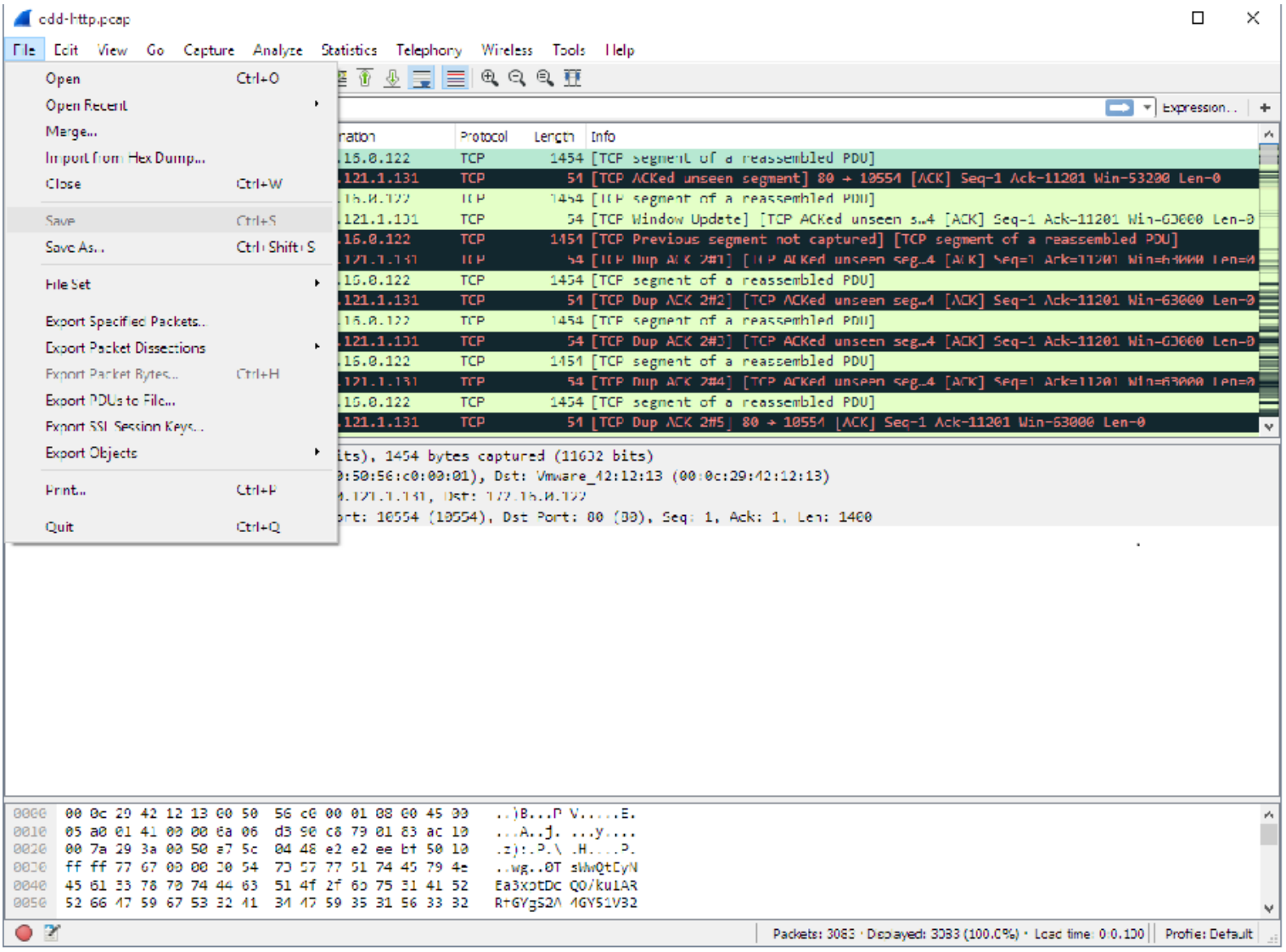


Figure 16-2 - Wireshark Interface

Wireshark Capture Options

Use tail to copy the pcap file to the FIFO:

```
tail +0f \
/net/ServerSw_Name//global/flow/Flow_Name/switch/Switch_Name/pca
p/tmp/pcap
```

You need to substitute ServerSw_Name, Flow_Name and Switch_Name to match your environment.

Live capture continues until the packet capture file is rotated. By default, the maximum packet capture file size is 10MB but it is configurable with the `packet-log-max` option of the vflow-create and `vflow-modify` commands.

> **Note**: The `mkfifo` command used in this task is a standard feature of Linux-like operating systems, including MacOS. For Windows platforms, you may need to install the

GNU CoreUtils package available at
http://gnuwin32.sourceforge.net/packages/coreutils.htm.

Analyzing Live Traffic Using Wireshark

Wireshark is a well known network protocol analyzer and one of many applications used for network protocol analysis.

Wireshark can interactively browse packet data from a live network or from a previously saved pcap file.

> **Note:** You can download Wireshark from https://www.wireshark.org/

To use Wireshark to decode a previously saved packet flow capture file, export the file from the switch and analyze it with Wireshark.

> **Note:** The path to a NetVisor OS switch pcap file is: `/net/<ServerSw_Name>/ONVL/global/flow/<Flow_Name>/<Switch_Name>/pcap`

Configuring FlowTracker for non-TCP Analytics

Analytics is a process where the data is collected and analyzed for storing statistics, improving performance and network security in case of any diagnostic network problems.

Starting from NetVisor OS 7.0.1 version, support for non-TCP analytics is available with the introduction of FlowTracker feature.

The main objective of the FlowTracker feature is to extend Arista Analytics solution beyond TCP to ensure visibility into DNS/ DHCP, generic UDP and ICMP echo traffic. This is done by collecting information about the non-TCP flows and creating statistical data using the Exact Match sub-system.

FlowTracker creates 5-tuples (source IP address, destination IP address, source port number, destination port number and the protocol in use) flow entries in Exact-match sub-system in the hardware. Packet and bytes statistics are also maintained for the flow in hardware. The stats data is periodically fetched from software and hence data shown in the `connection-show` is not real-time.

The FlowTracker feature for non-TCP analytics is available on the following platforms in NetVisor OS version 7.0.1:

- Dell: S5232F-ON, S5248F-ON
- Ericsson: NRU03, NRU-S0301
- Edgecore: AS7726-32X, AS7326-56X, AS5835-54X, AS5835-54T
- Freedom: F9432-C, F9480-V, F9460-X, F9460-T

> **Note**: By default, the FlowTracker feature is disabled. You must reboot the switch after enabling or disabling the FlowTracker feature.

While showing DHCP or a DNS, the show command will display protocol as UDP whereas for ICMP, the protocol will be displayed as ICMP.

System vFlow entries are created to identify the broad family of protocols of interest (DHCP, DNS and other generic UDP flows) for which analytics is needed.
The first packet of the flow is copied to the CPU using the `copy-to-cpu` vFlow action and from the vFlow callback, a signature for the flow is extracted from the packet which consists of source IP address, destination IP address, destination (dest) port, source (src) port, protocol, ICMP type-code (in case of ICMP).

The subsequent packets for the flow are not copied to the CPU by the vFlow.

If limits are reached in the table when creating a new entry, the oldest flow is purged and that slot is reclaimed. There will be no explicit timer to age out the entries. An upper limit would be set on how many packets are copied to the CPU for each category of flows per second. Statistics from the hash table entries is collected in a periodic manner.

> **Note:** Each entry is visited once in 8 seconds. In one second, stats are collected for a maximum of 4000 entries.

> **Note:** Support for scale number is available for tracking a maximum of 32k Non-TCP flows.

To start the analysis for a protocol, perform the following:

- Enable the FlowTracker feature.
- Add the protocols.

Enabling and Configuring FlowTracker:

To enable the FlowTracker feature, use the command :

```
CLI (network-admin@switch) > system-settings-modify flow-tracker-enable
```

Enabling FlowTracker results in resizing of Unified Field Tables (UFT) which reduces space for L2 table.

You should reboot the system for this change to take effect.

> **Note:** To enable the FlowTracker feature, the analytics needs to be enabled. By default, the FlowTracker feature is disabled.

To disable the FlowTracker feature, use the command:

```
CLI (network-admin@switch) > system-settings-modify no-flow-tracker-enable
```

To see if the FlowTracker is enabled, use the command:

```
CLI (network-admin@switch) > system-settings-show
optimize-arps:                          off
lldp:                                   on
policy-based-routing:                   on
optimize-nd:                            off
reactivate-mac:                         on
reactivate-vxlan-tunnel-mac:            on
manage-unknown-unicast:                 off
manage-broadcast:                       off
manage-wake-on-lan:                     off
auto-trunk:                             off
auto-host-bundle:                       off
cluster-active-active-routing:          on
routing-over-vlags:                     off
source-mac-miss:                        copy-to-cpu
optimize-datapath:                      all
cpu-class-enable:                       on
usb-port:                               on
igmp-snoop:                             use-l3
vle-tracking-timeout:                   3
pfc-buffer-limit:                       40%
cosq-weight-auto:                       off
lossless-mode:                          off
snoop-query-stagger:                    no-stagger-queries
host-refresh:                           off
proxy-conn-retry:                       on
proxy-conn-max-retry:                   3
proxy-conn-retry-interval:              500
optimize-rxlos:                         off
xcvr-link-debug:                        disable
```

```
fastpath-bfd:                           on
linkscan-interval:                      150000
linkscan-mode:                          software
single-pass-l2-known-multicast:         on
single-pass-flood:                      on
single-pass-riot:                       on
hybrid-mode:                            off
xcvr-oir-debug-port:                    none
batch-move-mac-hw-group-for-vlan-only: on
memory-tracker:                         on
bcm-ctr-thread-freq:                    1
symmetric-hash:                         off
hash-suppress-unidir-fields:            off
bcm-interrupt-timeout:                  2000
prioritize-rx-reasons:                  off
hash-inner-ip-over-gre:                 on
multiple-tunnel-next-hops:              on
l3-use-host-table:                      off
offline-notification-new-thread:        off
uplink-oversubscription-action:         default
flow-tracker-enable:                    on
```

Note: By default, the FlowTracker feature is disabled. You must reboot the switch after enabling or disabling FlowTracker.

When analytics is enabled, it is enabled only for TCP by default. To enable it for non-TCP protocols, Flow-Tracker needs to be enabled as well. After the feature is enabled, to start the analysis for a particular protocol, it needs to be added separately using the CLI flow-tracker-add-protocol protocol <protocol>.

You can monitor the following protocols using the FlowTracker:

- DNS
- NTP
- DHCP
- ping4
- UDP
- ping6
- DHCPv6

To enable the analysis for any of the protocols, you need to first add the protocol by using the command:

```
CLI (network-admin@switch) > flow-tracker-add-protocol protocol <protocol> server-
port <port>
```

To stop the analysis for a particular protocol and remove it from the analytics list, use the command.

```
CLI (network-admin@switch) > flow-tracker-remove-protocol protocol <protocol>
server-port <port>
```

Note: The server-port is needed only in case of protocol choice of **UDP**. For other protocols, the server-port is not valid.

Note: In case of any UDP-based protocols, both client side and server side flows are captured for analysis.

Note: For ping4 and ping6, both echo request and reply are captured.

To see which protocols are enabled for analysis, use the command:

```
CLI (network-admin@switch) > flow-tracker-show
switch protocol server-port
------ -------- -----------
switch ping4
switch udp      20010
```

To view the flows that are being monitored after adding an ICMP protocol, use the command:

```
CLI (network-admin@switch) > connection-show
switch vlan src-ip         dst-ip         proto icmp_type obytes total-bytes total-
pkts age
------ ---- -------------- -------------- ----- --------- ------ -----------
---------- ---
switch 1    200.220.220.20 200.220.220.10 icmp  reply     204    204         2
    3s
switch 1    200.220.220.10 200.220.220.20 icmp  echo      204    204         2
    3s
switch 1    200.220.220.10 200.220.220.20 icmp  reply     204    204         2
    11s
switch 1    200.220.220.20 200.220.220.10 icmp  echo      204    204         2
    11s
```

To view the flows that are being monitored after adding a UDP protocol, use the command:

```
CLI (network-admin@switch) > connection-show
switch vlan src-ip         dst-ip         src-port dst-port proto obytes total-bytes
total-pkts age
------ ---- -------------- -------------- -------- -------- ----- ------ -----------
---------- ---
switch 1    200.220.220.10 200.220.220.20 42155    20010    udp   64     64
 1          5s
switch 1    200.220.220.10 200.220.220.20 41882    20010    udp
            11s
```

Note: Connections that are created in the non-TCP FlowTracker feature are seen using the same commands as in TCP.

While adding a DHCP, the protocol is displayed as UDP. Only for ICMP, the protocol will be displayed as

ICMP whereas for DHCP or DNS, the `connection-show` will display protocol as UDP. The destination (dst) port helps to identify the application.

> **Note:** For details regarding the protocol in use, refer the destination (dst) port.

To clear the entries from the table while the protocol is still enabled for analysis, use the command:

```
CLI (network-admin@switch) > connection-clear
```

Limitations and Guidelines

The non-TCP flows are first matched using vFlows and hence have the following limitations:

- The first packet of the flow is copied to CPU, which causes CPU load and bandwidth utilization of vFlow queues.
- Exact Match does not work with VXLAN encapsulated packets and hence the solution can work only on leaf nodes.

Supported Releases

Command/Parameter	NetVisor OS Version
`sflow-thread-binding-show`	Command introduced in version 6.1.0
`counter-polling-interval 0..120`	Parameter introduced in version 6.1.0
`malformed-packet-drops malformed-packet-drops-number`	Parameter introduced in version 6.1.0
`malformed-vlan-packet-drops malformed-vlan-packet-drops-number`	Parameter introduced in version 6.1.0
`malformed-ip-packet-drops malformed-ip-packet-drops-number`	Parameter introduced in version 6.1.0
`internal-q-drops internal-q-drops-number`	Parameter introduced in version 6.1.0
`oversize-packet-drops oversize-packet-drops-number`	Parameter introduced in version 6.1.0
`internal-nq-drops internal-nq-drops-number`	Parameter introduced in version 6.1.0
`layout json`	Parameter introduced in 7.0.2
`telemetry-dst-group-create, telemetry-dst-group-show, telemetry-dst-group-delete`	Commands introduced in 7.0.2
`telemetry-dst-group-collector-add, telemetry-dst-group-collector-show, telemetry-dst-group-collector-remove`	Commands introduced in 7.0.2
`telemetry-sensor-group-create, telemetry-sensor-group-show, telemetry-sensor-group-delete`	Commands introduced in 7.0.2
`telemetry-sensor-group-path-add, telemetry-sensor-group-path-show, telemetry-sensor-group-path-remove`	Commands introduced in 7.0.2
`telemetry-subscription-create, telemetry-subscription-show, telemetry-subscription-delete, telemetry-subscription-modify`	Commands introduced in 7.0.2
`telemetry-grpc-settings-modify, telemetry-grpc-settings-show`	Commands introduced in 7.0.2

Please also refer to the Arista NetVisor OS Command Reference document.

Related Documentation

For further information on concepts mentioned in this section (such as Layer 2, Layer 3, etc.), refer to these sections of the Configuration Guide:

- Configuring Switch Ports
- Configuring Layer 2 Features
- Configuring Layer 3 Features
- Configuring and Using vFlows

Configuring TACACS+

This chapter provides information about configuring the Terminal Access Controller Access Control System Plus (TACACS+) on a NetVisor OS switch using the NetVisor OS command line interface.

Understanding TACACS+

Terminal Access Controller Access Control System (TACACS+) is an Authentication, Authorization, and Accounting (AAA) protocol that is used for authenticating access to network devices, providing central authorization of operations and auditing of these operations. This protocol uses a client-server approach by which the client queries a server and the server replies with a pass or fail result for authentication. The communication between the client and server uses TCP as the transport protocol and requires a secret key.

You must configure the TACACS+ server before you can avail the TACACS+ functionalities on your network device. NetVisor OS allows you to configure external TACACS+ servers for authentication, authorization, and accounting of sessions. You can configure any number of TACACS+ servers, and each server may be configured to handle any combination of authentication, session authorization, command authorization, session accounting, and command accounting.

Authentication, authorization, and accounting in NetVisor OS have the attributes described below:

Authentication: Verifies the identity of the user through different authentication protocols with varying levels of security.

The authentication protocols supported by TACACS+ are:

- Password Authentication Protocol (PAP): The client must authenticate by providing the username and password during the establishment of a TACACS+ session. The password is sent to the TACACS+ server in cleartext.
- Challenge-Handshake Authentication Protocol (CHAP): A more secure authentication protocol that verifies the identity of the client periodically through a three-way handshake and uses a hashing algorithm for the password. NetVisor OS uses CHAP authentication by default.
- Microsoft Challenge-Handshake Authentication Protocol (MS-CHAP): Microsoft's version of CHAP that provides authenticator-controlled password change and authentication retry mechanisms.

Authorization: Gives fine-grained control over a TACACS+ session by stipulating the capabilities of the user. Authorization gives users a customized access to a device where they can be permitted or restricted in terms of using specific services.

Accounting: Keeps record of all the operations that a user performs while in an authorized TACACS+ session. TACACS+ supports three types of accounting messages based on the lifecycle of a service. These include messages indicating that a service is about to begin, a service is being performed, and a service is terminated.

Figure 17-1 illustrates a simple TACACS+ implementation.

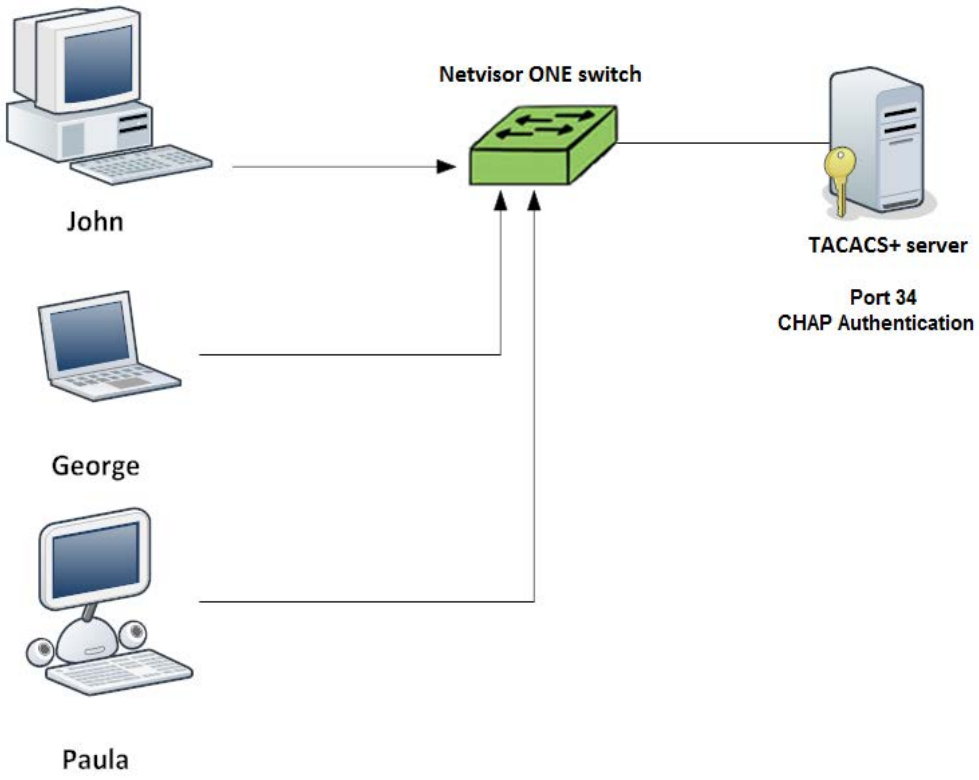


Figure 17-1: TACACS+ Implementation

Configuring TACACS+

You can configure TACACS+ services on a switch by using the `aaa-tacacs-create` command.

```
CLI (network-admin@switch) > aaa-tacacs-create
```

`name` *`name-string`*	Specify the name for TACACS+ configuration.
`scope local|fabric`	Specify the scope of TACACS+ configuration.
`server` *`server-string`*	Specify the TACACS+ server name.
Specify one of more of the following options:	
`port` *`port-number`*	Specify the TACACS+ communication port.
`secret secret-string`	Specify the shared secret for TACACS+.
`timeout timeout-number`	Specify the number of seconds before communication times out.
`priority priority-number`	Specify the priority for TACACs+.
`authen|no-authen`	Specify to enable or disable authentication.
`[authen-local|no-authen-local]`	Specify if the authentication overrides local users. The `no-authen-local` parameter overrides local users and gives them access while `authen-local` prevents local users from logging in.
`[authen-method pap|chap|ms-chap]]`	Specify one among the authentication methods: PAP, MS-CHAP or CHAP (default).
`[sess-acct|no-sess-acct]`	Specify to enable or disable session accounting.
`[cmd-acct|no-cmd-acct]`	Specify to enable or disable command accounting.
`[acct-local|no-acct-local]`	Specify to enable or disable accounting for local users.
`[sess-author|no-sess-author]`	Specify to enable or disable session authorization.
`[cmd-author|no-cmd-author]`	Specify to enable or disable command authorization.
`[author-local|no-author-local]`	Specify to enable or disable authorization for local users.
`[service service-string]`	Specify the service name used for TACACS+ requests sent from NetVisor OS to the TACACS+ server for commands run at the NetVisor CLI and REST APIs. The default service is `shell`.
`[service-shell service-shell-string]`	Specify the TACACS+ service name string for shell commands.
`[service-vtysh service-vtysh-`	Specify the TACACS+ service name string for

string]	vtysh commands.

The parameter `authen-local` controls if a TACACS+ server is used to authenticate local accounts or not. To allow an account to authenticate locally without relying on the TACACS+ server, you must configure all active TACACS+ instances with `no-authen-local` option. The local accounts include the accounts configured by using the commands `user-create` and `user-modify`, and the accounts such as `admin` and `pluribus`. Logins for both these types of accounts are disabled if a TACACS+ server is configured with the `authen-local` option.

NetVisor OS tracks errors between itself and the TACACS+ server and if a communication error occurs, local authentication is allowed until communication with the TACACS+ server is re-estalished. Local authentication is allowed after three failed attempts to reach the TACACS+ server. This enables the recovery of the system if TACACS+ is unreachable.

To create a TACACS+ account named `tac` having `local` scope with no local authentication privilege, use the command:

```
CLI (network-admin@switch) > aaa-tacacs-create name tac scope local server
appliance.pluribusnetworks.com  authen-local
```

To create a secret key, use the command:

```
CLI (network-admin@switch) > aaa-tacacs-modify secret name tac
shared secret:
confirm shared secret:
CLI (network-admin@switch) >
```

To modify the configuration and to allow local authentication, use the command:

```
CLI (network-admin@switch) > aaa-tacacs-modify name tac no-authen-local
```

Use the parameters `author-local` and `acct-local` to configure for locally authenticated accounts the sending of authorization and accounting messages to the TACACS+ server. For example:

```
CLI (network-admin@switch) > aaa-tacacs-modify name tac author-local acct-
local
```

The `vtysh` command is used to display FRR information from the CLI. To use the `vtysh` command, you must first enter the shell prompt.

To specify the service for authorization and accounting messages for `shell` and `vtysh` commands, use the commands:

```
CLI (network-admin@switch) > aaa-tacacs-modify name tac service-shell unix-
shell

CLI (network-admin@switch) > aaa-tacacs-modify name tac service-vtysh
vtysh-shell
```

If `service-shell` or `service-vtysh` is not specified, then the value specified under the `service` option is used.

For example, if a `service` or `service-shell` is not specified, the default `service`, that is, `shell` will be used for authorization and accounting of shell and vtysh commands.

To view the configuration, use the command:

```
CLI (network-admin@switch) > aaa-tacacs-show
name scope server                        port timeout priority authen authen-local authen-method sess-acct cmd-acct acct-local sess-author cmd-author
author-local service service-shell service-vtysh
---- ----- ------------------------------ ---- ------- -------- ------ ------------ ------------- --------- -------- ---------- ----------- ----------
------------ ------- ------------ ------------
tac  local appliance.pluribusnetworks.com 49   10      1        on     off          chap          on        off      off        on          off
on           shell   unix-shell    vtysh-shell
```

To delete a specific TACACS+ configuration, use the `aaa-tacacs-delete` command:

```
CLI (network-admin@switch) > aaa-tacacs-delete name <name-string>
```

To display the status of the TACACS+ server, use the `aaa-tacacs-status` command:

```
CLI (network-admin@switch) > aaa-tacacs-status name tac
name server                          port priority status
---- ------------------------------ ---- -------- ------
tac  appliance.pluribusnetworks.com 49   1        up
```

Note :

- The default `network-admin` account is exempt from all TACACS+ configuration as a fail-safe account for network designs without a TACACS+ server and also to allow access to Pluribus Networks facilities if TACACS+ is unavailable or unreachable.
- The `pluribus` account password is the same as the `network-admin` password and when the `network-admin` password is changed, the `pluribus` account password also changes. The login shell for the `pluribus` account is `nvauditsh`.
- The `admin` account requires a code from Arista (previously Pluribus) Networks Customer Advocacy to login. Because you can access the shell through the CLI, the `admin` account is rarely needed. The login shell for the `admin` account is `adminsh`.
- The `adminsh` shell uses `nvauditsh` to enable auditing for commands run from the `admin` account.

When the TACACS+ server is configured with the `authen-local` option and is working correctly:

- Local users such as `admin` or `pluribus` are not allowed to login.
- Local users configured by using the `user-create` or `user-modify` commands are not allowed to login.
- Authorization and Accounting for shell commands are enabled.
- Authorization and Accounting for FRR commands are enabled.
- You can switch to the shell prompt from the CLI if the user role has the permission to do so.

You can exempt the desired CLI, shell, or vtysh commands from authorization and accounting by using the `log-audit-exception-create` command. For more information, see *Configuring Audit Logging* section of the *Configuring Network Management and Monitoring* chapter.

Creating Users and Roles

In NetVisor OS, the default roles include the `network-admin` role with full access to a switch and the `read-only-network-admin` role with limited access allowing you to execute show commands only. In addition, you can configure custom roles with different permissions.

You can create users on the switch and assign roles to them (the default roles of `network-admin` and `read-only-network-admin` or roles created by using the `role-create` command). To create a new user role, use the command:

```
CLI (network-admin@switch) > role-create
```

`role-create`	Create a user role.
`name` *`name-string`*	Specify the name of the user role.
`scope local|fabric`	Specify if the scope is `local` or `fabric`.
Specify any of the following options:	
`access read-only|read-write`	Specify the type of access. The default is `read-write`.
`running-config|no-running-config`	Specify to allow or deny displaying of running configuration of switch.
`shell|no-shell`	Specify to allow or deny shell access.
`sudo|no-sudo`	Specify to permit or deny `sudo` command from the shell prompt.

For example, to create a user role named `role1` having `local` scope with access to the shell prompt, use the command:

```
CLI (network-admin@switch) > role-create name role1 scope local shell sudo
```

To display the role configuration, use the command:

```
CLI (network-admin@switch) > role-show
name                    scope vnet-access access     running-config shell  sudo
----------------------- ----- ----------- ---------- -------------- ------ ------
network-admin           local all         read-write permit         deny   deny
read-only-network-admin local all         read-only  deny           deny   deny
role1                   local all         read-write deny           permit permit
```

To delete a role, use the command:

```
CLI (network-admin@switch) > role-delete name <name-string>
```

To modify a role, use the `role-modify` command:

`role-modify`	Modify a user role.
`name` *`name-string`*	Specify the name of the user role.
Specify any of the following options:	
`access read-only|read-write`	Specify the type of access. The default is `read-`

	`write`.
`running-config|no-running-config`	Specify to allow or deny displaying of running configuration of switch.
`shell|no-shell`	Specify to allow or deny shell access.
`sudo|no-sudo`	Specify to allow or deny `sudo` command from shell.
`delete-from-users`	Delete the role from the users.

For example, to modify a role in order to deny access to the shell, use the command:

```
CLI (network-admin@switch) > role-modify name <name-string> no-shell
```

You can create a user and assign an initial role to the user by using the `user-create` command. Thereby, you can impart all the privileges associated with the role to the user.

`user-create`	Create a user and apply a role.
Specify any of the following options:	
`name name-string`	Specify the name of the user.
`scope local|fabric`	Specify the scope of the user.
`initial-role` *`role-name`*	Specify the initial role for the user
`login-fail-count` *`login-fail-count-number`*	Specify the number of allowed authentication failures before locking the user.
`lock-account|no-lock-account`	Specify if you want to lock the user after the stipulated number of authentication failures.
`minimum-pw-length` *`6..65`*	Specify the minimum password length for user account. The default value is 6. The number of allowed characters is between 6 and 65.

For example, to create a user `user1` with `local` scope and initial role `role1`, use the command:

```
CLI (network-admin@switch) > user-create name user1 scope local initial-
role role1
password:
confirm password:
CLI (network-admin@switch) >
```

The CLI command `shell` enables you to start an interactive UNIX shell. In order to be able to run the `shell` command successfully, the authenticated user's role must have the permission to enter the shell. By default, all user roles including `network-admin` do not have shell access. For example:

```
CLI (network-admin@switch) > shell
shell: shell access denied by role
```

The above error message indicates that the `network-admin` role does not have shell access configured. Hence, to enter the shell prompt, you must login from a user account with a role that has shell access. For example, you can login to the switch as `user1` which has an assigned role of `role1` with shell access privileges (configured previously):

```
PS C:\Users\user1>:~$ ssh user1@switch
```

```
* Welcome to Arista Networks Inc. Netvisor(R). This is a monitored system.
  *
*                ACCESS RESTRICTED TO AUTHORIZED USERS ONLY
    *
* By using the Netvisor(R) CLI,you agree to the terms of the Arista
Networks  *
* End User License Agreement (EULA). The EULA can be accessed via
    *
* http://www.arista.com/eula or by using the command "eula-show"      *
user1@switch's password:
Netvisor OS Command Line Interface 7.0
Connected to Switch switch; nvOS Identifier:0x900104c; Ver: 7.0.0-
7000019011
CLI (user1@switch) > shell
root@switch:~$
```

To delete a user, use the command below (add the parameter `forcefully` to delete a active users):

```
CLI (network-admin@switch) > user-delete name <name-string> forcefully|no-
forcefully
```

Use the `user-modify` command to modify a user:

```
CLI (network-admin@switch) > user-modify
```

`user-modify`	Create a user and apply a role.
Specify any of the following options:	
`password` *`password-string`*	Specify a plaintext password.
`login-fail-count` *`login-fail-count-number`*	Specify the numbers of allowed authentication failures before locking the user.
`lock-account|no-lock-account`	Specify if you want to lock the user after the stipulated number of authentication failures.
`minimum-pw-length` *`6..65`*	Specify the minimum password length for user account.

To set a user password or change the current password, use the command:

```
CLI (network-admin@switch) > user-password-set
```

`user-password-set`	Set a user password.
`name` *`<name-string>`*	Specify the username.
Specify any of the following options:	
`scope local|fabric`	Specify if the scope is local or fabric
`uid` *`uid-number`*	Specify the user ID.
`server` *`aaa-tacacs-name`*	Specify the TACACS+ server.
`initial-role` *`role-name`*	Specify the initial role for the user
`login-fail-count` *`login-fail-count-`*	Specify the numbers of allowed authentication

number	failures before locking the user.
`lock-account|no-lock-account`	Specify if you want to lock the user after the stipulated number of authentication failures.
`minimum-pw-length` *6..65*	Specify the minimum password length for user account.

For example, to change the current password for `user1` and to change the minimum password length, use the command:

```
CLI (network-admin@switch) > user-password-set name user1 scope local
minimum-pw-length 7
password:
confirm password:
CLI (network-admin@switch) >
```

Configuring Virtual Networks

This chapter provides information about configuring Virtual Networks on a NetVisor OS switch using the NetVisor OS command line interface.

Understanding Virtual Networks (vNETs)

Virtual Networks Overview

Arista Networks supports various network virtualization capabilities as part of a highly flexible approach to software-defined networking (SDN), called *Unified Cloud Fabric* (previously known as Adaptive Cloud Fabric).

Arista's fabric addresses all the most common network design requirements, including scalability, redundancy, predictable growth capability, fast convergence in case of a failure event, etc. Such requirements also include *multi-tenancy* support.

Therefore, in the data center, by leveraging NetVisor OS's virtualization features, network designers can implement a variety of multi-tenancy models such as Infrastructure as a Service (IaaS) or Network as a Service (NaaS).

To support multiple tenants, the fabric's data plane segmentation technologies include standard features (such as VLANs) as well as advanced virtualization features such as VXLAN and distributed VRFs, to be able to deploy an open, interoperable, high-capacity and high-scale multi-tenant network. (Refer to the *Configuring VXLAN* chapter for more details on VXLAN and distributed VRFs).

In addition to data plane segmentation, fabric virtualization also comprises the capability of separating tenants into isolated management domains. In general network parlance, this capability is handled by a virtual POD (or vPOD), which is a logical construct in a shared infrastructure where a tenant can only see and use resources that are allocated to that particular tenant. Arista calls this capability *Virtual Networks* (*vNETs*).

Low-end platforms that have limited memory cannot host multiple vNETs. In such cases, you can host the vNETs on a separate server (such as ESXi servers) to provide multi-tenancy and vNET based management by using *virtual NetVisor*.

Virtual NetVisor or vNV is a virtual machine running NetVisor OS to provide vNET capabilities to a unified cloud fabric, even with low end platforms which are not able to run multiple vNET Managers on it. For details, see the *Configuring vCenter Features* chapter of this guide or the *Virtual NetVisor Deployment Guide*.

By joining the Unified Cloud Fabric that is composed of ONVL switches, vNV provides more compute resources to run vNET Manager containers, in standalone mode or in HA pair, and offers multitenancy capability in an existing fabric.

Two vNV instances can share the same vNET Manager interface to provide a fully redundant management plane for each tenant created. A pair of vNV instances can support up to 32 vNETs in HA mode and 64 vNETs in standalone mode.

The vNV workflow includes creating a virtual machine (VM) from a template on a ESXi host, provisioning network adaptors (vNICs) and port-groups on the vSwitches, and provisioning fabric and data network on NetVisor OS. Each such tenant can view and manage its network slice by logging into the vNET manager zone (described in later sections).

About vNETs

A vNET is an abstract control plane resource that is implemented globally across the fabric to identify a tenant's domain. By using vNETs, you can segment a physical fabric into many logical domains, each with separate resources, network services, and Quality of Service (QoS) guarantees. vNETs therefore allow the network administrator to completely separate and manage the provisioning of multiple tenants within the network.

In simple terms, vNETs are separate resource management spaces and can operate in the dataplane as well as in the control plane. That is, a physical network can be divided into multiple virtual networks to manage the resources of multiple customers/organizations (multi-tenancy).

Each vNET has a single point of management. The fabric/network administrator can create vNETs and assign ownership of each vNET to the vNET administrators with responsibility for managing those resources (for example, manage the VLANs, vLEs, vPGs, vFlows associated with that vNET). For details about these resources, refer to the respective sections in the *Configuration Guide* (see *Related Documentation* below).

About vNET Architecture

By creating a vNET in the Unified Cloud Fabric, the network resources can be partitioned and dedicated to a single tenant with fully delegated management capabilities. This partition of the fabric can be configured either on one switch, a cluster of two switches, or the entire fabric by leveraging the "scope" parameter in the CLI command, depending on where you want the vNET to deliver a reserved service to a tenant.

The ***figure 18-1*** shows how a multi-tenant leaf-spine design can be achieved with a Arista's Unified Cloud Fabric, the details of the components are explained below.

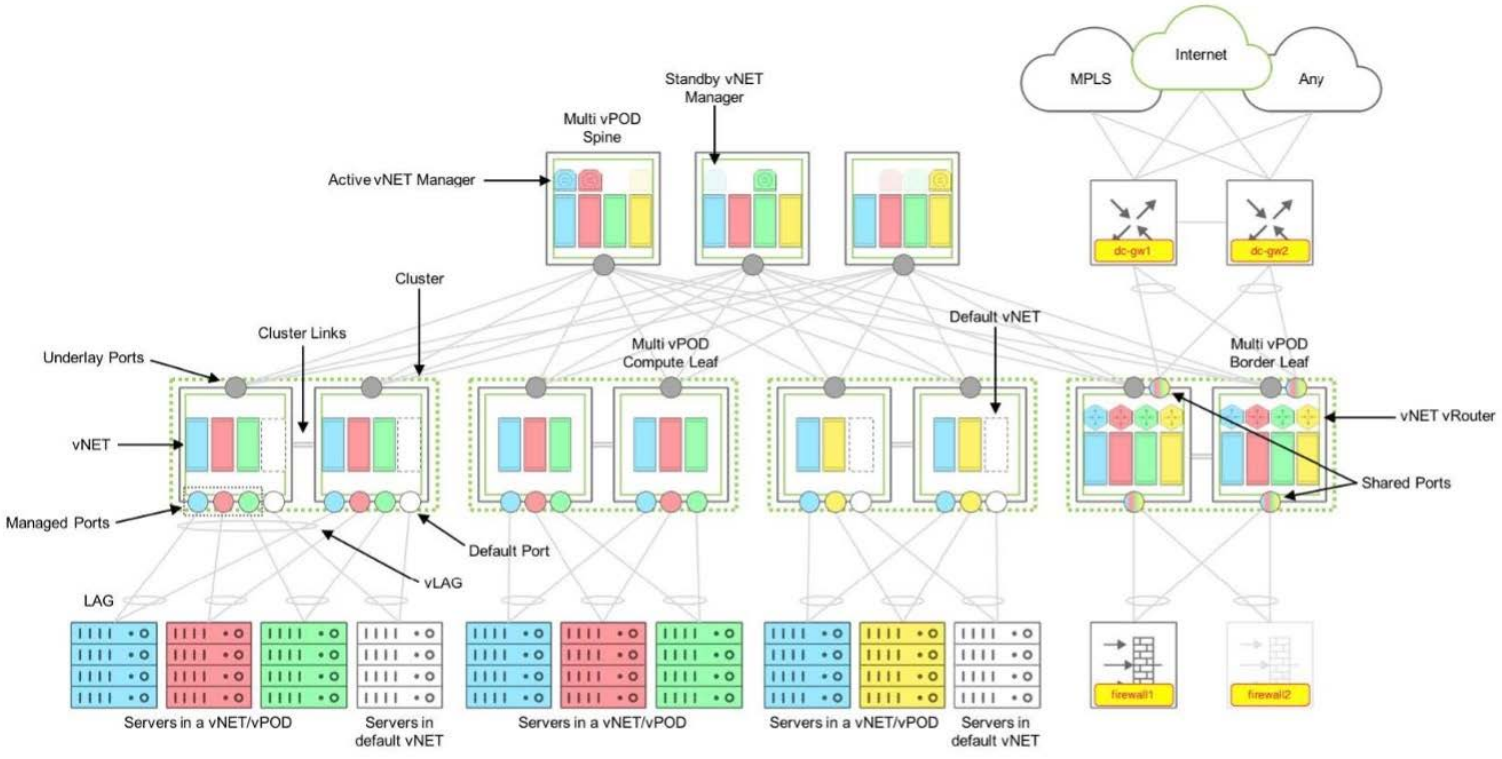


Fig 18-1: Multi-tenant Fabric Design Components

Further, vNETs are very flexible and can be used to create complex network architectures. In another example, an Arista Networks switch, or a fabric of switches, are used to create multiple tenant environments (***Fig 18-2***). In this diagram, there are three vNETs (yellow, blue, and red), each one with a management interface and a data interface. Each vNET is assigned an IP address pool used for DHCP assignment of IP addresses to each node, server, or OS component.

With vNETs, you can have segmentation (logical network slices) of a physical infrastructure at data plane, control plane, and management plane with a dedicated management interface (see ***Fig 18-2***).

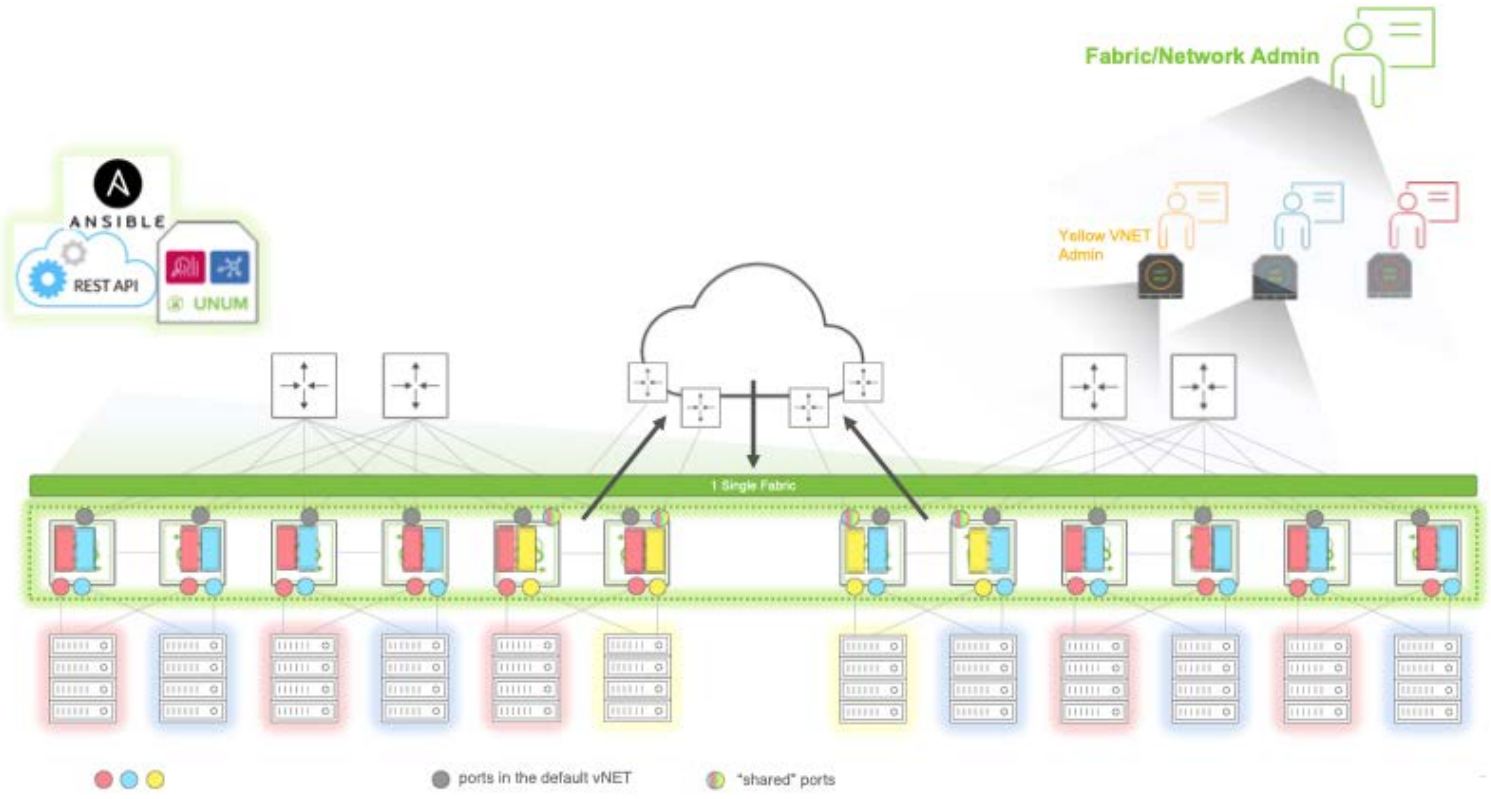


Fig 18-2: A vNET Infrastructure with Multi-site Fabrics

Underlying each vNET is the vNET manager (see below for the description of a *vNET Manager*). Each vNET manager runs in a zone. When services are created for a vNET, they occupy the same zone on the switch. This is called a shared service and it is the default when creating services. However, each zone can only support a single instance of a service. If a second service instance is needed for a vNET, then it needs to occupy a separate zone. This is called a dedicated service. In most cases, you can create services as shared resources unless you specifically want to create a dedicated service.

For details of each component, see the sections below.

Default vNET (Global vNET)

The Default or Global vNET is the default partition on a switch that is used when the switch has not been

configured with additional vNETs. That is, when a fabric is created, a vNET is also automatically created and is named `fabric-name-global`. This default partition *owns* all the resources on a NetVisor switch such as:

- Physical ports
- Default VLAN space
- Default Routing table/domain (if enabled on the switch)
- vRouter functions
- VTEP function
- VLE function
- VPG function
- vFlows
- Any L2/L3 entries learned through static and dynamic protocols
- Any additional containers created

Later, when the user specific tenant vNETs are created, the network administrator can allocate resources from the default vNET to those tenant vNETs.

The default vNET is not configured by the fabric/network administrator, it is a partition created when the switch becomes part of a fabric. Also, there is no vNET Manager container associated to the *default* vNET (see below for the description of a *vNET Manager*).

Note: The default vNET uses the naming convention: *<fabric-name>-global*.

Tenant vNET (referred as just vNET)

A Tenant vNET is a partition that carves out the following software and the hardware resources from the switch. This ensures the tenant to use and manage their reserved resources.

- VLAN IDs
- MAC addresses
- VRouter or VRFs
- Remote VTEPs (Virtual Tunnel Endpoints)
- Physical ports

You can create a vNET with different scopes such as:

- Scope `local`: The vNET is locally created on the switch and cannot be used by any other switch in the same fabric.
- Scope `cluster`: The vNET is created on a cluster pair that will share the same Layer 2 domain.
- Scope `fabric`: The vNET partition is created on all switches in the same fabric.

After a vNET is created, the network or fabric administrator can allocate resources to it when new objects (such as VLANs, vRouters, etc.) are created in the fabric.

Although vNETs could be compared to Virtual Routing and Forwarding (VRFs), the vNETs have more capabilities compared to the VRF capabilities. The vNETs not only provide an isolated Layer 3 domain, but also offers the capability to have VLAN and MAC overlapping between vNETs. In other words, vNET is a complete isolation of data plane and control plane along with the management plane if there is a vNET

Manager associated to it (see below for the description of a *vNET Manager*).

Each vNET has a single point of management. As the network/fabric administrator, you can create vNETs and assign ownership of each vNET to individuals with responsibility for managing those resources and with separate usernames and passwords for each vNET manager.

NetVisor OS support two types of vNETs: Public vNET and Private vNET.

Public vNET

A Public vNET is the normal vNET, which uses the default 4K VLAN space available on a switch. In this case, the default vNET and any other vNET created shares the range of *0-4095* VLAN IDs available on the switch with no option to have overlapping VLAN IDs across vNETs.

By default, a vNET is created as a Public vNET, although with overlay networks and multitenancy requirements, you can use Private vNETs almost exclusively.

Private vNET

A Private vNET is using an independent 4K VLAN space, where the vNET administrator can create any VLAN ID he prefers regardless of whether that ID is used in the default vNET or in another tenant vNET. That is, private vNET supports VLAN re-use. NetVisor OS supports this functionality by providing an internal VLAN mapping in the hardware tables.

As the hardware architecture on Open Networking platforms does not offer unlimited resources, the hardware tables between the default vNET and other created vNETs have to be shared. Arista provides a way to limit the number of VLANs that can be created in a Private vNET to avoid a situation where a tenant creating too many VLANs prevents other tenants from creating their VLANs as there are no hardware resources available on the switches.

To create an independent 4K VLAN ID space for each tenant, a mapping between the two types of VLANs (Public VLANs and Private VLANs) is required and is called vNET VLANs.

> **Note**: Private vNETs are extensively used on Ericsson's *NRU-xx platforms* as they support multiple containers.

vNET VLANs

A vNET VLAN consists of:

- **Public VLAN** - A Public VLAN is a VLAN that uses the default VLAN ID space on a switch. When a network/fabric administrator creates a VLAN on any NetVisor switch, which has only the default vNET (no other vNET is created), it takes the VLAN IDs in the Public VLAN space (0-4095 range).
- **Private VLAN** - A Private VLAN is a user-defined VLAN ID in a private space, that has nothing to do with the actual VLAN ID (in the public space) used on the switch to learn Layer 2 entries for this bridge domain. When the switch receives a frame with an 802.1q tag using the Private VLAN ID, it will translate this VLAN ID to an internal VLAN ID in the 4K Public VLAN space.
  For a Private VLAN to be transported over a VXLAN infrastructure, it requires a VNI to be associated to it

so that it can be identified as unique in the shared (or public) Layer 2 domain.
For details, see the *Configuring VXLAN* and *Configuring Advanced Layer 2 Transport Services* chapters of this *Configuration Guide*.

The following diagram (***Figure 18-3***) depicts the relationship between Public and Private VLANs in a Private vNET implementation:

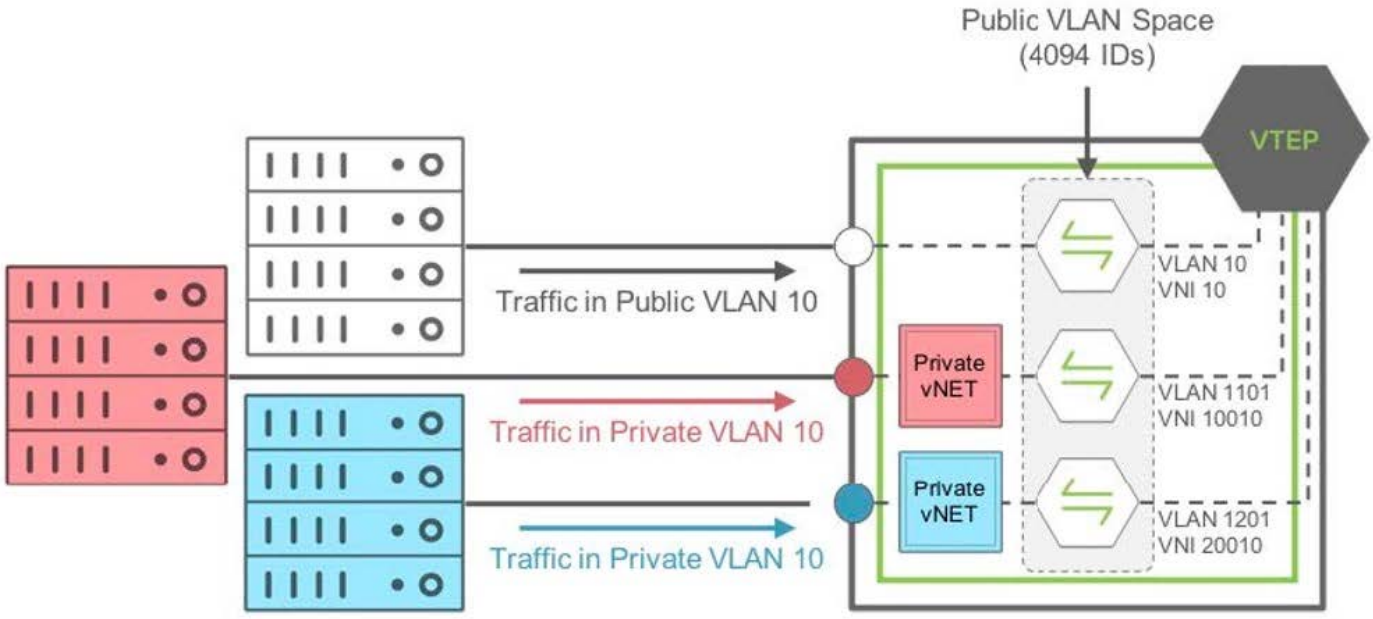


Figure 18-3 Private VLAN to Public VLAN Mapping in Private vNETs

In this example (***Figure 18-3***), the traffic coming on a managed port from the *red* private vNET, using private *VLAN ID 10* is mapped internally to the public *VLAN ID 1101*. Similarly, traffic coming on a managed port in the *blue* private vNET, using private *VLAN ID 10* as well is mapped internally to the public *VLAN ID 1201*.

The mapping is done automatically in NetVisor OS by defining a pool of usable public *VLAN IDs* for private vNETs.

> **Note**: Traffic from the *red* private VLAN 10 and *blue* private VLAN 10 are NOT bridged as they both are not in the same bridge domain internally. Also, traffic coming on a port on the default vNET using VLAN 10 (here, public VLAN 10), is mapped internally to the same public VLAN 10.

VNET Ports

Apart from the VLAN and MAC table isolation mentioned above, a complete isolation for each tenant at the physical port level is required. To achieve this isolation at the access ports level, NetVisor OS introduces different types of ports (explained below) to address different scenarios in a shared network.

Managed Port

A managed port is an access port that is associated with only one tenant (see ***Fig-18-1***). That is, this port is

dedicated to a vNET and cannot be shared with other vNETs configured on the same switch. The managed port is typically used to connect and isolate resources dedicated to a particular tenant such as servers or any other resource that is not an active Layer 2 node.

> **Note**: As an access port, a managed port does not support spanning tree (STP). In this case, the Layer 2 loop prevention mechanism is implemented through another feature called *Block-Loops*.

A managed port can be configured as an "*access port*" or a "*Dot1q Trunk port*" when the VLANs carried on that port are part of the same private vNET. It transports only private VLANs belonging to the same vNET. It also supports *link aggregation* (IEEE standard 802.3ad) on a single switch or with two different switches that are part of the same cluster, called virtual LAG (vLAG) with Arista switches.

Shared Port

As the name suggests, a shared port (***Fig-18-1***) is an access port that can serve different tenants on the same port where a shared resource is connected. That is, this port is not dedicated to a vNET and can be shared with other vNETs configured on the same switch. It is typically used to connect shared resources like *gateways*, *firewalls*, or even *hypervisors hosting several VM servers* from different tenants.

A Shared port is always configured as a "*Dot1q Trunk port*" transporting only public VLANs from the same vNET or different vNETs on the same port. It also supports *link aggregation* (IEEE standard 802.3ad) on a single switch or with two different switches that are part of the same cluster, called Virtual LAG (vLAG) with Arista switches.

Underlay Port

In a vNET design, the underlay network topology is shared by all vNETs configured in the fabric. To avoid dedicating one underlay port (per tenant) for connecting leaf switches to spine switches, NetVisor OS cloud fabric has abstracted the underlay ports from the vNET so as to leave their management to the network administrator.

Underlay ports are the ports that interconnect spine and leaf switches or two switches that are part of the same cluster (see ***Fig-18-1***). These ports are not configurable for the private vNETs, but are configured automatically based on the underlay design defined by the network administrator.

vNET Manager

A vNET Manager is a container or portal that is created on a fabric node and is associated to a vNET to provide a dedicated management portal for the specified vNET. A vNET Manager is a completely stateless object, but it is deployed in high availability (HA) mode to ensure the tenant never loses management access to the vNET.

A vNET Manager in high availability mode have two states:

- **Active vNET Manager**: preferred container to manage resources allocated to a vNET. This container is used by the vNET Administrator to configure resources using the CLI or API allocated by the network administrator to this vNET.
- **Standby vNET Manager**: container in standby mode, which shares the same IP address with the Active

vNET Manager container to offer high availability when the Active container fails or becomes inaccessible.

Guidelines for vNET Manager

Consider the following guidelines while creating a vNET Manager:

- A vNET Manager is not required for creating a vNET, however, the vNET manager is created only to provide a dedicated management portal to the tenant.
- A vNET Manager container is an object local to the switch where it is created. In other words when the network Administrator creates a vNET with a fabric scope, the vNET Manager associated to this vNET is created only on the switch where the command is instantiated.
- A vNET Manager can be created on any switch in a fabric only if local resources such as compute, memory, and storage are available.

vNET Router

A vNET vRouter is a network service container that provides Layer 3 services with dynamic routing control plane to a vNET. A vNET can be created in one of the following ways:

- Without a vRouter container (Layer 2-only vNET)
- With one single vRouter (Layer 3 vNET).
- With more than one vRouter (Multi-VRF model in a vNET)

The first vRouter associated to a vNET builds the vNET (default) routing table. For some designs, a vNET can have more than one vRouter to support multiple routing tables. This can be achieved by creating additional vRouter containers associated to the same vNET. Unlike in a traditional multi-VRF router, multiple vRouter containers ensures complete isolation between routing instances within the same vNET, by maintaining isolation of resources between routing domains. Later, each vRouter can be extended to support multi-VRF to help maximize the scale of VRFs without depending on resource-consuming vRouter containers.

> **Note**: A vRouter in any vNET supports the same feature set that a vRouter created in the default vNET supports.

vNET Administrator

A vNET administrator (referred to as *vNET-admin*) account is created when a vNET Manager container is created. This account has full read/write access to its associated vNET. Typically, when a VNET is created, an associated VNET manager zone is created (if you specify the `create-vnet-mgr` parameter) to host all vNET related services as well as to provide administrative access to the vNET to manage these resources. Location of the container zone is typically on the same switch on which the VNET is created and consumes space on the device.

A vNET-admin account can access only its own vNET Manager and can see resources belonging to its vNET only. The vNET-admin is responsible for the creation/deletion of the following resources in its vNET:

- vNET VLANs/VNIs

- LAGs/vLAGs for Managed ports associated to the vNET
- vNET vRouters
- VXLAN Tunnels for the vNET
- L3 routing protocols for the vNET

Using the vNET administration credentials, the vNET admin can use Secure Shell (SSH) to connect to the vNET manager and access a subset of the NetVisor OS CLI commands to manage a specific vNET. This way, multiple tenants can share a fabric with each managing a vNET with security, traffic, and resource protection from other vNETs.

Staring with NetVisor OS version 7.0.0, the vNET admin can use REST API to access, configure, and manage the vNET resources using the Arista NetVisor UNUM Management Platform.

Network Administrator (Fabric Administrator)

The Network administrator (referred to as *network-admin*) is the *root account equivalent* on a Linux distribution. It has all the privileges to create, modify, or delete any resources created on any fabric node. Besides having the same privileges that a vNET-admin has, the network-admin can also manage the following fabric attributes:

- Fabric nodes
- Clusters
- vNETs
- All Ports
- VTEPs
- vFlows (Security/QoS)
- Software upgrades

Note: The network-admin can create separate *user names* and passwords for each vNET manager.

Creating a Virtual Network (vNET)

To separate resources, including switch ports, IP addresses, VLANs, and VXLAN IDs into separate management spaces, create a vNET(s) and associate the desired resources to it (see below for an example of configuration).

You can create a vNET using the `vnet-create` command followed by a list of required parameters. Subsequently, you can configure a separate vNET administrator to manage each newly created management domain.

```
CLI (network-admin@switch) > vnet-create
```

`name` *`name-string`*	Defines the virtual network (vNET) name. Enter a name for identification of this vNET.
`scope local|cluster|fabric`	Defines the scope of this vNET. Use `fabric` scope if this vNET should be seen by all switches. If this vNET should not be seen by other switches, use `cluster` or `local` scope.
Specify one or more of the following options:	
`vrg` *`vrg-name`*	Defines the name of the virtual resource group (vRG) to be associated with this vNET.
`vlan-type public|private`	Defines the type of VLAN used in this vNET, which in turn defines the type of vNET (private or public).
`num-vlans` *`1..4094`*	Defines the number of global VLANs you want to assign to this vNET. Using this parameter allows you to assign a group of VLANs rather than specific VLANs.
`vlans` *`vlan-list`*	Defines the VLANs that can be assigned to the public VLAN vNET. You can specify a list or range of VLANs that this vNET can assign to vNET interfaces.
`public-vlans vlan-list`	Defines the VLAN IDs that can be used to map a Public VLAN to a Private VLAN. This is useful when resources in a vNET have to communicate with a shared resource (such as a shared server, DC GW, firewall, etc.) hosted on the global vNET (default vNET) connected to Shared Ports. You can use the same VLAN ID for both Private vNET and global vNET.
`num-private-vlans` *`1..4094`*	Defines the number of private VLANs that can be configured in this vNET. You can create up to 4094 VLANs.

`num-bridge-domains 0..4094`	Defines the number of bridge domains allowed to be created in this vNET.
`vxlans` *`0..16777215`*	Defines the range of VNIs that can be used by this vNET administrator. The VXLAN ID is a combination of the tenant ID and VLAN ID.
`vxlan-end` *`0..16777215`*	The last VNI or VXLAN ID assigned to this vNET.
`managed-ports` *`port-list`*	The exclusive port(s) associated with this vNET.
`shared-ports` *`port-list`*	List of ports that are shared with other vNETs.
`shared-port-vlans` *`vlan-list`*	The shared port VLANs associated with this vNET.
`config-admin|no-config-admin`	The `config-admin` option creates an administrator for the vNET. This is an optional parameter, and the default value is `no-config-admin`.
`admin` *`user-name`*	Add a user name for the admin role.
`create-vnet-mgr|no-create-vnet-mgr`	Choose one of the options to create or not create a dedicated vNET manager. If `no-create-vnet-mgr` option is selected, then the network/fabric admin manages this vNET.
`vnet-mgr-name` *`vnet-mgr-name-string`*	Specify the name of the vNET manager, if you had selected `create-vnet-mgr` above. If you don't specify a name, one is automatically configured.
`vnet-mgr-storage-pool` *`storage-pool-name`*	Add the storage pool for this vNET.

Note: You cannot create a vNET inside a vNET.

To delete an existing vNET, use the vnet-delete command:

```
CLI (network-admin@switch) > vnet-delete name <vnet-name>
```

To modify the configuration of an existing vNET, use the `vnet-modify` command.

```
CLI (network-admin@switch) > vnet-modify
```

`name` *`name-string`*	Enter the name of the virtual network (vNET) you want to modify.
Specify one or more of the following options that needs to be modified:	
`vlans` *`vlan-list`*	The list of VLANs assigned to this vNET. You can specify a list or range of VLANs that the VNET assigns to VNET interfaces.
`managed-ports` *`port-list`*	Modify the list of managed (exclusive) ports associated for this vNET.

`num-private-vlans` *`1..4094`*	Modify the number of private VLANs that the vNET administrator can create for this VNET. You can configure up to 4094 VLANs.
`public-vlans vlan-list`	Modify the public VLANs assigned to this private VLAN vNET.
`shared-ports` *`port-list`*	Modify the shared ports for this vNET.
`vxlans` *`0..16777215`*	Modify the VXLAN IDs assigned to this vNET.
`vxlan-end` *`0..16777215`*	Modify the last VXLAN ID assigned to the vNET.
`shared-port-vlans` *`vlan-list`*	Modify the vNET shared port VLANs.
`num-bridge-domains` *`0..4094`*	Modify the number of bridge domains allowed to be created in this private vNET.

Administering vNET Specific Commands

The purpose of vNET objects is to provide independent network domains whose administrators can manage a set of dedicated resources without having to involve the network/fabric administrator, but within the constraints that the network/fabric had administrators defined their roles.

Access control is performed based on the scope of a vNET, which includes several dedicated ports on which it is possible to apply certain commands or use certain dedicated resources.

The applicable dedicated resources include three categories of commands with vNET *admin* scope, as below:

- Layer 1 commands (i.e., port-related commands):
  - `port-show` (shows vNET ports but not internal/cluster ports)
  - `port-phy-show` (shows managed and shared ports)
  - `port-config-modify/show` (shows managed ports)
  - `bezel-portmap-show` (shows managed ports)
  - `port-vnet-show` (shows managed ports)

- Layer 2 commands (i.e., VLAN/LACP/STP/vLAG commands allowed on managed ports only):
  - `port-lacp-modify/show`
  - `vlag-create/modify/delete`
  - `trunk-create/modify/delete`
  - `stp-port-modify/show/stp-portevent-show`
  - A vNET admin is not allowed to change the native VLAN (untagged VLAN) on a shared port with the `port-vlan-add` command.

- Layer 3 commands (i.e., vRouter commands, where the scope is `local` for the vNET):
  - `static-ecmp-group-show/static-ecmp-group-nh-show`
  - `vrouter-ping/traceroute`
  - `bridge-domain-*`

> **Note**: If any of the above commands are run on a port not in the scope of a vNET, a `No permission for port 'port = %d'` message is displayed, for example like so:

```
CLI (network-admin@switch) > vlag-create name vl1 port 99 peer-port 99
vlag-create: No permission for port 'port = 99'
```

Let us consider an example of vNET creation and see how the above commands behave within it:

```
CLI (network-admin@switch) > vnet-create name vn1 scope fabric vlan-type
private public-vlans 2000-2099 num-private-vlans 10 num-bridge-domains 3
vxlans 10000100-10000109 managed-ports 9,17 shared-ports 18 shared-port-
vlans 105-109
Creating vn1-mgr zone, please wait...
```

With this command the network administrator creates a vNET as a dedicated domain comprising managed and shared ports as well as private and public VLANs and VXLAN IDs. In other words, the fabric admin is partitioning the resources to provide a dedicated and restricted view on the network for the vNET admin.

The following examples show how a vNET's view gets constrained for each command:

```
CLI  (network-admin@switch)  >  port-show

switch  port  bezel-port  status      config
------  ----  ----------  ----------  ------
switch  9     9           up,vlan-up  fd,10g
switch  17    17          disabled    fd,10g
switch  18    18          disabled    fd,10g
```

In this example the vNET admin can only view the managed and shared ports chosen as part of the vNET creation process, out of all the front panel ports.

Note: When you run the `port-show` command for a private vNET, only vNET managed ports are displayed. But for a public vNET, all ports except internal ports and cluster ports are displayed.

Starting with NetVisor OS version 7.0.0, a new CLI command, `port-vnet-show` is available to view the ports that are assigned to vNETs and the respective VLANs of a vNET. This command displays the details that you had configured after you had added port(s) using the `vnet-port-add` command:

```
CLI  (network-admin@switch)  >  port-vnet-show
```

`name` *`name-string`*	The name of the virtual network (vNET).
`managed-ports` *`port-list`*	The list of vNET exclusive ports.
`num-vlans` *`1..4094`*	The number of global VLANs assigned to this vNET. This is a number between 1 and 4094.
`vlans` *`vlan-list`*	The VLANs assigned to public VLAN vNET.

For example, if you had configured ports 9, 17, 18 and VLANs 3 and 5 on vNET vn1-global, then the output looks like:

```
CLI  (network-admin@switch)  >  port-vnet-show
name       managed-ports num-vlans vlans
---------- ------------- --------- ---------
vn1-global 9,17          10        2000-2099
```

To view the PHY details, use the command:

```
CLI  (network-admin@switch)  >  port-phy-show

port bezel-port state speed eth-mode    max-frame def-vlan
---- ---------- ----- ----- ---------- --------- --------
9    9          up    10000 10Gbase-cr 1540      0
17   17         down  10000 10Gbase-cr 1540      1
18   18         down  10000 10Gbase-cr 1540      0
```

Also, the front panel and PHY information is constrained to the ports selected as part of the vNET creation

process. Port configuration gets constrained too, for example:

```
CLI (network-admin@switch) > port-config-show format port,enable,

port enable
---- ------
9    on
17   off
```

Regarding the Layer 2 configuration, these commands also get a constrained view:

```
CLI (network-admin@switch) > port-vlan-add port 9 untagged-vlan 45
port-vlan-add: No permission to modify untagged-vlan field
```

As displayed, untagged VLANs are prevented from being changed.

The vLAGs can be created only using accessible ports (as port 9 in the example below):

```
CLI (network-admin@switch) > vlag-create name vl1 port 9 peer-port 9
CLI (network-admin@switch) > port-lacp-show layout vertical

switch:   switch
port:     9
name:     v11
port-type:    vlag
mode:     passive
timeout:  slow
system-id:    66:0e:94:b6:ab:01
lacp-key: 36285
system-priority:   32768
port-priority:32768
aggregatable: yes
sync:     yes
coll:     yes
dist:     no
defaulted:    yes
expired:  no
port-state:   0x5c
```

whereas inaccessible ports are blocked in the configuration:

```
CLI (network-admin@aquarius00) > vlag-create name vl1 port 99 peer-port 99
vlag-create: No permission for port 'port = 99'
```

Similarly, a VLAN trunk can be created (and then deleted) using accessible ports 9 and 10 like so:

```
CLI (network-admin@switch) > trunk-create name t ports 9-10
trunk 273 defer-bringup set to 1 based on first port 9
Created trunk t, id 273

CLI (network-admin@switch) > trunk-show format name,trunk-id,ports

name                trunk-id ports
```

```
------------------- -------- -----
t                   274      9-10
vxlan-loopback-trunk 397
```

```
CLI (network-admin@switch) > trunk-delete name t
```

Furthermore, spanning tree (STP) commands are limited to accessible ports only:

```
CLI (network-admin@switch) > stp-port-show port 10

port block filter edge bpdu-guard root-guard priority cost
---- ----- ------ ---- ---------- ---------- -------- ----
10   on    off    no   no         no         128      2000

CLI (network-admin@switch) > stp-port-modify port 53 cost 10000
stp-port-modify: No permission over ports 53

CLI (network-admin@switch) > stp-port-event-show

switch   time      port vlan    instance count initial-state other-state final-state
-------  --------  ---- ------  -------- -----  -------------  ----------  -----------
switch   01:13:52  17   1,4094 0        3      Disabled       Disabled    Forwarding
switch   01:15:42  9    1      0        1      Disabled       Disabled    Discarding
switch   01:16:02  9    1      0        1      Discarding     Disabled    Learning
switch   01:16:12  9    1      0        1      Learning       Disabled    Forwarding
switch   01:17:53  17   1      0        1      Forwarding     Disabled    Disabled
switch   01:17:53  9    1      0        1      Forwarding     Disabled    Disabled
switch   01:29:00  17   4094   0        1      Disabled       Disabled    Forwarding
```

Further, for Layer 3 configuration, these commands display a constrained view:

The vRouter commands get constrained too (to accessible VLANs and interfaces) like so:

```
CLI (network-admin@switch) > vrouter-create name vr1 vnet vn1 router-type
hardware
Creating vr1 zone, please wait...
vrouter created

CLI (network-admin@switch) > vrouter-interface-add vrouter-name vr1 ip
192.168.99.13/24 vlan 100
Added interface eth0.100 with ifIndex 159

CLI (network-admin@switch) > vrouter-interface-show format vrouter-
name,nic,ip,vnet,vlan,vlan-type,nic-state,mtu

vrouter-name nic       ip                vnet vlan vlan-type nic-state mtu
------------ --------  ----------------  ---- ---- ---------  ---------  ----
vr1          eth0.100 192.168.99.13/24 vn1  100  private    up         1500
```

The `static-ecmp-group-*` commands are allowed only within the vNET scope ¬(*local*) and duplicate entries from other switches are blocked when logged in as vnet-admin. For example:

```
CLI (network-admin@switch) > vnet-create name VNET2 scope fabric
Creating VNET2-mgr zone, please wait...
Vnet created. Vlans assigned: 5
CLI (network-admin@switch) > vrouter-create name vr_fabric vnet VNET2 router-type hardware
Creating vr_fabric zone, please wait...
```

```
vrouter created

CLI (network-admin@switch) > static-ecmp-group-create group-name G1 scope fabric vrouter-name
vr_fabric hash-type non-resilient

CLI (network-admin@switch) > static-ecmp-group-nh-add group-name G1 ip 1.1.1.1

CLI (network-admin@switch) > vnet-create name vnet2-data scope fabric
Creating vnet2-data-mgr zone, please wait...
Vnet created. Vlans assigned: 6

root@switch:~# lxc-attach -n vnet2-data-mgr
root@vnet2-data-mgr:~# nvOS_cli --quiet static-ecmp-group-show
root@vnet2-data-mgr:~# exit

root@switch:~# lxc-attach -n VNET2-mgr

root@VNET2-mgr:~# nvOS_cli --quiet static-ecmp-group-show

group-name scope  vrouter-name hw-ecmp-id hash-type
---------- ------ ------------ ---------- -------------
G1         fabric vr_fabric    200000     non-resilient

root@VNET2-mgr:~# nvOS_cli --quiet static-ecmp-group-nh-show

group-name ip      vlan egress-id
---------- ------- ---- ---------
G1         1.1.1.1 0    -1

root@VNET2-mgr:~# exit

CLI (network-admin@switch) > vnet-create name VNET1 scope cluster
Creating VNET1-mgr zone, please wait...
Vnet created. Vlans assigned: 7

CLI (network-admin@switch) > vrouter-create name vr_cluster vnet VNET1
Creating vr_cluster zone, please wait...
vrouter created

CLI (network-admin@switch) > static-ecmp-group-create group-name G2 scope cluster vrouter-name
vr_cluster hash-type non-resilient

CLI (network-admin@switch) > static-ecmp-group-nh-add group-name G2 ip 2.1.1.1
```

On the Global vNET, you can view all details, for example:

```
CLI (network-admin@switch) > static-ecmp-group-show

switch   group-name  scope    vrouter-name  vrid  hw-ecmp-id  hash-type
-------  ----------  -------  ------------  ----  ----------  -------------
switch1 G1          fabric                -1    -1          non-resilient
switch2 G1           fabric   vr_fabric           200000       non-resilient
switch1 G2          cluster               -1    -1          non-resilient
switch2 G2           cluster  vr_cluster     1     200001      non-resilient

root@switch:~# lxc-attach -n vnet2-data-mgr
root@vnet2-data-mgr:~# nvOS_cli --quiet static-ecmp-group-nh-show
root@vnet2-data-mgr:~# exit
root@switch:~# lxc-attach -n VNET2-mgr
```

```
root@VNET2-mgr:~# nvOS_cli --quiet static-ecmp-group-nh-show

switch group-name ip      vlan egress-id
------ ---------- ------- ---- ---------
switch  G1        1.1.1.1  0    -1

root@VNET2-mgr:~# nvOS_cli --quiet static-ecmp-group-show

switch  group-name  scope   vrouter-name  hw-ecmp-id  hash-type
------  ----------  ------  ------------  ----------  -------------
switch G1           fabric vr_fabric     200000      non-resilient

root@VNET2-mgr:~# exit
root@switch:~# lxc-attach -n VNET1-mgr

root@VNET1-mgr:~# nvOS_cli --quiet static-ecmp-group-show

switch  group-name  scope    vrouter-name  vrid  hw-ecmp-id  hash-type
------  ----------  -------  ------------  ----  ----------  -------------
switch G2           cluster vr_cluster    1     200001      non-resilient
root@VNET1-mgr:~# exit
```

Displays a constrained view of the `vrouter-ping/traceroute` commands as well:

```
CLI (network-admin@switch) > vrouter-ping vrouter-name vr1 host-ip
192.168.99.13

PING 192.168.99.13 (192.168.99.13) 56(84) bytes of data.
64 bytes from 192.168.99.13: icmp_seq=1 ttl=64 time=0.066 ms
64 bytes from 192.168.99.13: icmp_seq=2 ttl=64 time=0.071 ms
64 bytes from 192.168.99.13: icmp_seq=3 ttl=64 time=0.063 ms
^C
--- 192.168.99.13 ping statistics ---
```

Specifying the Type of vNET Interface

The `mgmt`, `data`, and `span` keywords used in different commands specify the path used to connect to the network service. For example, to specify an out-of-band connection to a management interface of a vNET, the interface is specified using the `mgmt` keyword. If in-band access to that management interface of the vNET is required, then the `data` or `span` keywords are used in the specific command.

The keywords, `data` and `span`, are essentially equivalent but apply to two separate paths. To maximize throughput between the server and the switch components, it is recommended to use both. The `data` keyword applies to port 65, and the `span` keyword applies to port 66.

Each vNET can have one or more isolating zones and network services are applied to each zone. Network services have their own zone or share the zone with the vNET manager which is the zone that the vNET manager logs into to manage the vNET. In shared zones, the network interfaces are available to all network services in the shared zones, regardless of the service that created the network interface.

> **Note:** This is an important concept as you can use service commands such as `vnet-manager-interface-add` to add interfaces to a vNET. If you want the service to be specific to a vNET as a dedicated service, then add the interfaces using the `service-interface-add` commands.

Configuring vNET High Availability (HA)

A vNET HA provides high availability for switch access through a vNET manager. As described in the *Understanding vNET Manager* section, the vNET manager is a zone with one IP interface that allows a vNET administrator to log into and administer a vNET using the CLI or REST API commands.

NetVisor OS enables access to the vNET managers when you add either standard or VRRP VIP interfaces to the vNET managers.

Without vNET HA, the vNET administrators have only a single point of access where the vNET manager zone runs on a particular switch. If that switch fails, the vNET administrator cannot log into the fabric and administer the vNET resources.

You can create or delete a vNET manager zone by using the `vnet-manager-create` and `vnet-manager-delete` commands. The `vnet-manager-interface-add` command accepts VRRP interfaces, which allows you to create VRRP interfaces on a vNET manager zone.

Note: While creating the vNET, the `vnet-create` command provides an option to create or not create a vNET manager zone.

Note: Copy and paste SSH host keys used on different vNET managers. This is needed when using a VRRP VIP to access the vNET managers to avoid SSH host key violations if the VIP fails over to the standby vNET manager. Install these keys on the client machine used to connect to the vNET managers.

Creating a vNET Manager Zone

To create additional vNET manager zones, use the `vnet-manager-create` command.

```
CLI (network-admin@switch) > vnet-manager-create name name-string vnet vnet-name [disable|enable][location fabric-node name][storage-pool storage-pool name]
```

`vnet-manager-create`	Creates a vNET manager
`name name-string`	The name of service configuration.
`vnet vnet-name`	The vNET assigned to the service.
Specify any of the following options:	
`disable|enable`	Enables or disables the service.
`location fabric-node name`	The location of the service.
`storage-pool storage-pool name`	The storage pool assigned to the service.

Deleting a vNET Manager Zone

To delete a vNET manager zone, use the `vnet-manager-delete` command.

```
CLI (network-admin@switch) > vnet-manager-delete name name-string
```

Specify the name of service configuration.

Copying and Pasting SSH Keys

To output SSH host keys to copy and paste to `~/.ssh/known_hosts` file of the client host, use the `vnet-manager-ssh-host-key-show` command.

This allows you to SSH to any vNET manager zone and avoid issues with invalid key hosts.

```
CLI (network-admin@switch) > vnet-manager-ssh-host-key-show [vnet vnet name]
```

`vnet-manager-ssh-host-key-show`	Displays the vNET Manager host keys to copy and past to: `~/.ssh/known_hosts`.
`name name-string`	Displays the name of service configuration.

vNET Manager Command Options

This feature uses a new option `[no-]create-vnet-mgr` which controls whether to create a vNET manager.

The default behavior is creating a vNET manager as this is the current behavior of creating a vNET manager as part of `vnet-create`.

```
CLI (network-admin@Leaf1) > no-create-vnet-mgr
```

`vnet-create`	Creates a virtual network (vNET)
`name` *`name-string`*	The vNET name.
`scope local|cluster|fabric`	The vNET scope as local, cluster, or fabric.
`create-vnet-mgr|no-create-vnet-mgr`	Create or not create a vNET manager service.

VRRP Interfaces Option

This feature now accepts options for VRRP interfaces. This allows you to create VRRP interfaces on vNET manager zones.

```
CLI (network-admin@Leaf1) > vnet-manager-interface-add vnet-manager-name name-string [vrrp-id 0,,255][vrrp-primary vrrp-primary-string][vrrp-priority 0..254][vrrp-adv-int 300..40950]
```

`vnet-manager-interface-add`	Adds an interface to a vNET manager.
`vnet-manager-name` *`name-string`*	The name of service configuration.

`vrrp-id 0..255`	The ID assigned to VRRP.
`vrrp-primary vrrp-primary-string`	The VRRP primary interface.
`vrrp-priority 0..254`	The VRRP priority for the interface.
`vrrp-adv-int 300..40950`	The VRRP Advertisement Interval in milliseconds. The minimum interval is 300ms and the maximum interval is 40950ms. The default interval is 1000ms.

Enabling Web-API Access for vNET and vNET Manager

Starting with NetVisor OS version 7.0.0, you can access and configure vNET features by using REST API commands. This web-API support enables the Arista NetVisor UNUM customers to access vNET, vNET Manager, and vNET resources by using the UNUM interface.

To enable web-API access through UNUNM interface, the vNET-admin must login using the *mgmt IP address* of the switch. Prior to version 7.0.0, NetVisor allowed the vNET-admin to login by using only the interface IP address (or vNET IP address). The vNET-admin and the users created with vNET-admin role can also access the CLI shell by logging in using the mgmt IP address of the switch.

Prior to NetVisor OS version 7.0.0, to access vNET and it's resources, you must login into the vNET container, which is present only on the switch where vNET is created (even in the case of *fabric* scoped vNET). However, starting with version 7.0.0, you can access the vNET context from any of the nodes regardless of where the vNET is created.

To access vNET and vNET Manager through UNUM interface, you must enable web API on all devices by using the `admin-service-modify if mgmt web` command and then use the vNET credentials in the curl request.

You can access a vNET and manage the vNET resources by using the `vnet-admin` user and corresponding password just as similar to `network-admin` user. Following is an example format to log into the switch by using vnet-admin user:

Example of a GET request:

```
curl -s -u vn1-admin:vn1-admin -X GET http://switch-test1/vRest/port-configs
```

or

```
curl -s http://vn1-admin:vn1-admin@switch-test1/vRest/ports-phys | python -m json.tool
```

To access using RESTful API,

```
root@switch-test1:~# curl -s -u vnet-1-admin:vnet-1-admin -X GET http://192.168.22.82/vRest/vnets | python -m json.tool
{
    "data": [
    {
        "admin": 40000,
            "global": false,
            "id": "c0000bf:1",
            "managed-ports": "",
            "name": "vnet-1",
            "num-private-vlans": 0,
            "num-vlans": 1,
            "public-vlans": "",
            "scope": "fabric",
            "shared-port-vlans": "",
            "shared-ports": "",
            "vlan-type": "public",
```

```
                "vlans": "5",
                "vnet-mgr-name": "vnet-1-mgr",
                "vrg-id": "c0000bf:0",
                "vxlan-end": 0,
                "vxlans": 0
        }
        ],
        "result": {
             "result": [
             {
                  "api.switch-name": "local",
                  "code": 0,
                  "message": "",
                  "scope": "local",
                  "status": "Success"
             }
             ],
             "status": "Success"
        }
}
```

You can also access vNET and vNET Manager on NetVisor OS by enabling SSH logging into the switch using one of the following methods:

- `ssh vnet-admin@<switch-name>`
- `ssh vnet-admin@<switch mgmt IP address>`

Below is an example of how to login and access using the switch name and to verify the configurations:

```
ssh vn1-admin@switch-test1
* Welcome to Arista Networks Inc. Netvisor(R). This is a monitored system.
  *
*                ACCESS RESTRICTED TO AUTHORIZED USERS ONLY
    *
* By using the Netvisor(R) CLI,you agree to the terms of the Arista
Networks *
* End User License Agreement (EULA). The EULA can be accessed via
    *
* http://www.arista.com/eula or by using the command "eula-show"      *
vn1-admin@switch1's password:
Last login: Thu Jan  6 23:10:42 2021 from 10.140.0.48
Netvisor OS Command Line Interface 7.0
Connected to Switch switch-test1; nvOS Identifier:0xc0000bf; Ver: 7.0.0-
7000018494

CLI (vn1-admin@switch-test1) > vnet-show

name scope vlan-type vlans public-vlans  vxlans           managed-ports shared-ports shared-
port-vlans admin
---- ----- --------- ----- ------------  ----------------- ------------ ------------
----------------- ---------
vn1  local private   none  2000-2099     10000110-10000119 9,17          18           none
          vn1-admin
```

```
CLI (vn1-admin@switch-test1) > port-phy-show

port  state  speed  eth-mode  max-frame  def-vlan
----  -----  -----  --------  ---------  --------
9     down   10000  xfi       1540       1
17    down   10000  xfi       1540       1
18    down   10000  xfi       1540       1
```

Below is an example of how to login and access using the *in-band IP address* and to verify the configurations:

```
root@switch-test2:~#  cli
Netvisor OS Command Line Interface 7.0
Connected to Switch switch-test2; nvOS Identifier:0xc00021d; Ver: 7.0.0-
7000018494

CLI (network-admin@switch-test2) > fabric-node-show

name         fab-name mgmt-ip        in-band-ip       in-band-vlan-type fab-tid out-port
version          state  device-state
------------ -------- -------------- ---------------- ----------------- ------- --------
---------------- ------ ------------
switch-test2 fab-1    10.14.22.84/23 192.168.22.84/24 public            3
7.0.0-7000018494 online ok
switch-test1 fab-1    10.14.22.82/23 192.168.22.82/24 public            3       49
7.0.0-7000018494 online ok

root@switch-test1:~#  ssh  vnet-1-admin@192.168.22.82
The authenticity of host '192.168.22.82 (192.168.22.82)' can't be
established.
ECDSA key fingerprint is
SHA256:XXosFnaL8bQ/BUFbJAXP3bTXpgpwGvhrSuYvSgjmSv0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.22.82' (ECDSA) to the list of known
hosts.
* Welcome to Arista Networks Inc. Netvisor(R). This is a monitored system.
  *
*                ACCESS RESTRICTED TO AUTHORIZED USERS ONLY
    *
* By using the Netvisor(R) CLI,you agree to the terms of the Arista
Networks  *
* End User License Agreement (EULA). The EULA can be accessed via
    *
* http://www.arista.com/eula or by using the command "eula-show"      *
vnet-1-admin@192.168.22.82's  password:
Netvisor OS Command Line Interface 7.0
Connected to Switch switch-test1; nvOS Identifier:0xc0000bf; Ver: 7.0.0-
7000018494

CLI (vnet-1-admin@switch-test1) > vnet-show

switch        name    scope  vlan-type vlans public-vlans num-private-vlans vxlans managed-ports
```

```
shared-ports  shared-port-vlans  admin
------------ ------ ------ --------- ----- ----------- ---------------- ------ ------------
------------  ----------------  ------------
switch-test1 vnet-1 fabric public    5     none         0                0      none
 none         none              vnet-1-admin
switch-test2 vnet-1 fabric public    5     none         0                0      none
 none         none              vnet-1-admin
```

To view the details of users, use the command. For example:

```
CLI  (network-admin@switch-test1)  >  user-show
switch         name            scope  uid    type      login-fail-count lock-
account  minimum-pw-length
-------------  -------------  ------  -----  --------  ----------------
------------   -----------------
network-admin fabric 39999 netvisor 0                 false          0
switch-test1  vn2-admin      local  20001 netvisor 0                  false
     6
switch-test1  vn1-admin      local  20000 netvisor 0                  false
     6
switch-test1  user-1         local  20002 netvisor 0                  false
     6
switch-test1  user-2         local  20003 netvisor 0                  false
     6
```

To view the user roles, use the command:

```
CLI  (network-admin@switch-test1)  >  user-role-show

switch         user-name      role
-------------  -------------  -------------
network-admin  network-admin
switch-test1 vn2-admin      vn2-admin
switch-test1 vn1-admin      vn1-admin
switch-test1 user-1         vn1-admin
switch-test1 user-2         vn2-admin
```

Limitations and Restrictions

In cases, where the vNET container is not created and if you do not want access to vNET resources, then use the no-config-admin option along with no-create-vnet-mgr parameter while creating the vNET. For example,

To create a vNET with `no-vnet-managers`, `no-config-admin`:

```
CLI (network-admin@switch-test1) > vnet-create name vnet-test   scope local
no-create-vnet-mgr  no-config-admin
```

The above command creates a vNET without an *admin* and no users are listed with vNET admin role in `user-show` output as below:

```
CLI  (network-admin@switch-test1)  >  vnet-show  name  vnet-test
```

```
switch         name        scope vlan-type vlans public-vlans num-private-vlans vxlans managed-ports shared-
ports  shared-port-vlans
------------ --------- ----- --------- ----- ------------ ----------------- ------ -------------
------------   -----------------
switch-test1 vnet-test local public    5     none         0                 0      none          none
  none
```

```
CLI (network-admin@switch-test1) > user-show

name          scope  uid   type     login-fail-count lock-account minimum-pw-length
------------- ------ ----- -------- ---------------- ------------ -----------------
network-admin fabric 39999 netvisor 0                false        0
```

The above configuration restricts the access of vNET with *mgmt-ip/ inband-ip* as there are no users on the vNET that is created.

Related Documentation

For further information on concepts mentioned in this section (such as Layer 2, Layer 3, etc.), refer to these sections of the Configuration Guides:

- Configuring Switch Ports
- Configuring Layer 2 Features
- Configuring Layer 3 Features
- Configuring and Administering the Unified Cloud Fabric
- Configuring High Availability
- Configuring VXLAN
- Configuring and Using Network Management and Monitoring

Overview vCenter

This chapter provides information for understanding and using the Arista Networks NetVisor OS command line interface (CLI) on a NetVisor OS switch to configure vCenter features.

Understanding vCenter Connection Service

Note: Though you can run vCenter Connection Service (VCCS) on any platform, running VCCS on a virtual NetVisor Switch is recommended. For details on configuring the virtual NetVisor switch, see the Deployment Guide for Virtual Networks with Virtual NetVisor.

Connectivity to vCenter enables a network administrator to get visibility into the physical and/or virtual VMware entities behind each port of the Arista's Unified Cloud fabric, which helps in for operational simplification. You need to configure only one vcenter-connection for a fabric and should enable VCCS only on one of the nodes of fabric (applies to cluster pair also).

NetVisor OS categorizes the properties of the entities into three classes:

- Server, hypervisor, and one or more physical NICs.
- Virtual Machine including guest OS, virtual NIC properties such as MAC address, IP address, and VLAN.
- VM Kernel ports and the associated services like vMotion and vSAN.

The vCenter Connection Service enables you to connect to the VMware vCenter and to perform the following tasks:

- Identify the switch port used for each Virtual Machine (VM).
- Track the movement of VMs from one host (ESXi) to another host.
- Identify the VLAN requirements of each VM on the network.
- Track network statistics at a granular level for troubleshooting purposes.
- Track VM configuration changes such as additions, deletions, or modifications of VLANs, and configure VLANs on NetVisor OS switch ports.
- Track the additions or deletions of VMs and hosts, and configure VLANs on NetVisor OS switches.
- Track the state of VMs and dynamically provision VLANs on the servers facing physical ports.

The vCenter Connection Service synchronizes with VMware vCenter to obtain the following information:

- The host where the VMs exist.
- The NetVisor OS switch ports connected to the VM.
- The virtual network interface card (vNIC) that connects the VM to a virtual switch.
- The Power status of the VM as on or off.
- VLAN information of port groups.
- The port groups connected to the VM.

This information is associated with MAC addresses and VLANs on the network.

In NetVisor OS, the vCenter Connection Service:

- Supports up to 4 vCenter connections from a switch
- Supports ESXi and vCenter Server versions 5.5, 6.0, 6.5, and 7.0

Note: There are some restrictions depending upon the supported version of dswitch, in order to use the 5.5, 6.0, 6.5 and 7.0 versions through distributed switches.

You can check the status of the connection by using `vcenter-connection-show` command.

Note: If the connection status shows an error message with the state as **enabled**, then you should first disable the connection and then enable it to restart the connection service.

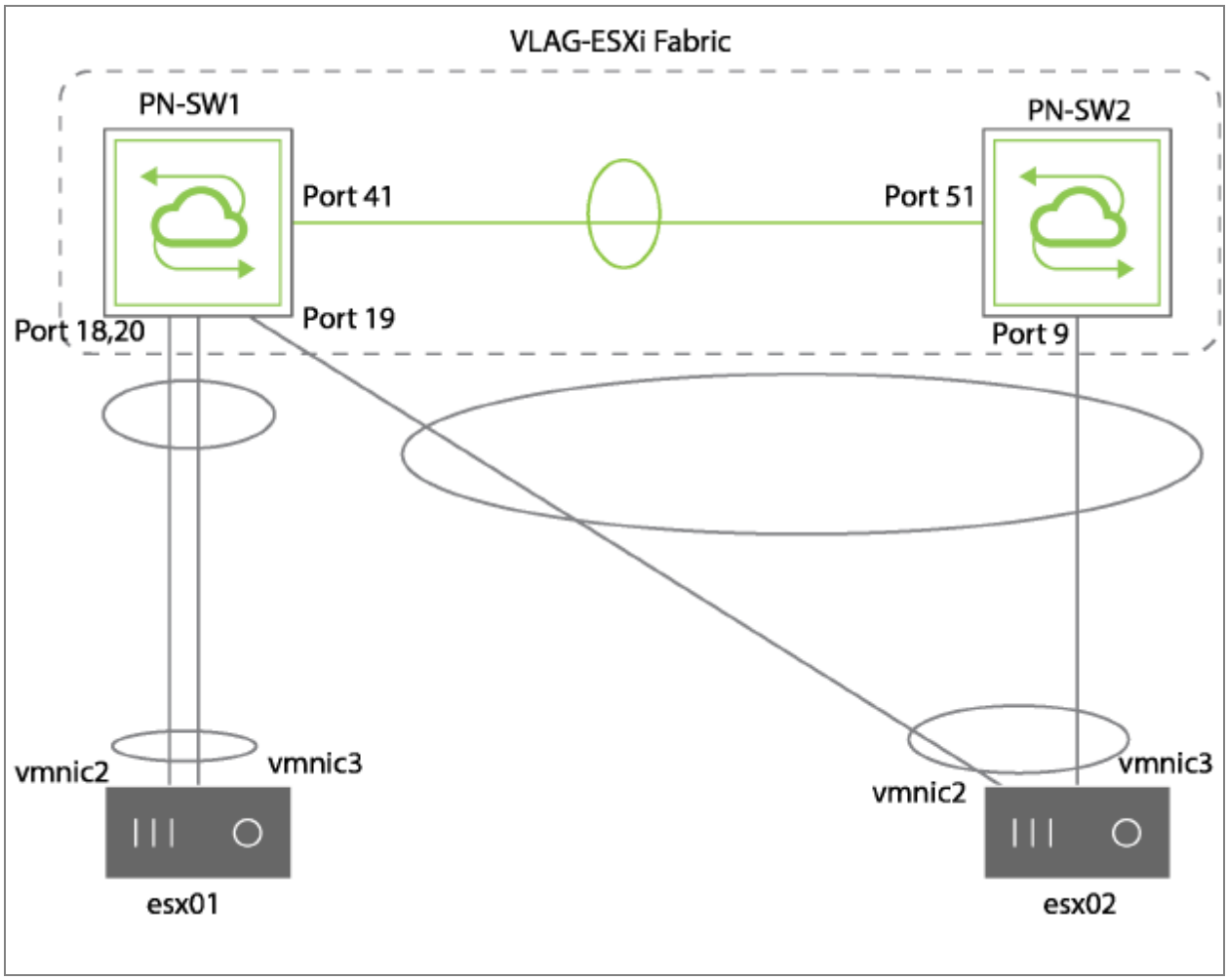


Figure 19-1 - Example LAG and VLAG Topology for vCenter

Example output with a list of VMs running on the two VMware ESXi servers along with visibility into the associated VLANs, attached port group, vswitches is given for reference:

```
CLI (network-admin@switch) > vport-show mac 00:50:56:xx:xx:xx layout vertical
owner:            ghleaf02-new
mac:              00:50:56:xx:xx:xx
vxlan:            12700000
ip:               200.200.200.2
ports:            3
svc-name:         vcenter-server
hostname:          server-test-69.pluribusnetworks.com
entity:           st-69_linux_vm1
power:            powered-on
os:               Ubuntu Linux (64-bit)
portgroup:         DPortGroup-vlan-trunk
pg-vlans:         1-4094
vswitch:          DvSwitch_SC1
vs-type:          distributed-vs
vnic-type:        trunked
status:           host,vm
```

Note: Some fields like 'os' need VMware tools to be installed on the VMs.

Example output with vPort statistics is given for reference:

```
CLI (network-admin@switch) > vport-stats-show format time,ip,entity,type,vnic-type

 time           ip            entity    status    vnic-type    ibytes ipkts obytes opkts
 -------------- ------------- --------  --------- ----------   ------ ----- ------ -----
 08-26,13:52:16 10.9.8.13     MyTestVM  vm        tagged       31.7K  507   112K   1.86K
 08-28,10:20:54 192.168.30.1  vmk1      vm-kernel vMotion,VSAN
 08-25,23:16:27 10.7.17.8     vmk2      vm-kernel VSAN
```

Guidelines and Limitations

Please keep in mind the following guidelines and limitations while configuring VCCS:

- Only one vcenter-connection is needed for a fabric. That is, VCCS should be enabled only on one of the nodes of fabric (includes cluster pair also).
- High Availability (HA) support is not available on VCCS. If an associated switch goes down, vcenter connection service is disrupted until the switch is back online or re-provisioned on a different node of the fabric.
- The network data associated with VMs collected by VCCS in vPort and connection table can be viewed on the UNUM dashboard. For details, see the UNUM documentation.

Configuring a vCenter Service

To create a vCenter service, use the `vcenter-connection-create` command:

```
CLI (network-admin@switch) > vcenter-connection-create
```

`name` *`name-string`*	Specify a name for the vCenter connection.
`host` *`host-string`*	Specify the host name of the vCenter to connect with.
`user` *`user-string`*	Specify the authorized vCenter user name.
`password` *`password-string`*	Choose one of the options to enable or disable.
`enable|disable`	Specify to enable to disable the vCenter connection.
`vlans` *`vlan-list`*	Specify the vCenter plugin VLANs for provisioning. Note: For vCCS, NetVisor OS supports the following VLAN provisioning limits: For all platforms except Z9264F-ON - supports up to 512 VLANs For Z9264F-ON - Supports up to 1536 VLANs (starting with 512 VLANs and then incrementing with additional 64 VLANs up to a maximum of 1536 VLANs)
`network-provisioning none|l2-underlay|l3-underlay`	Specify if the network provisioning should be none, L2 underlay or L3 underlay.

For example, to configure a vCenter service named *vcenter2*, on VLANs *190-198*, use the below command and enter the password when prompted:

```
CLI (network-admin@switch) > vcenter-connection-create name vcenter2 host
vcenter-server user administrator@lab.pluribus vlans 190-198 network-
provisioning l2-underlay
vCenter user password: <=== password is not visible while typing
vCenter connection service vcenter2 started.
```

To modify a vCenter service, use the command:

```
CLI (network-admin@switch) > vcenter-connection-modify
```

`host` *`host-string`*	The host name of the vCenter you want to modify.
`user` *`user-string`*	Modify the authorized vCenter user name.
`password` *`password-string`*	Specify a valid password for the user on the VMware vCenter server.
`enable|disable`	Modify to enable or disable the vCenter connection.

`vlans` *`vlan-list`*	Modify the vCenter plugin VLANs for provisioning.
`network-provisioning none|l2-underlay|l3-underlay`	Modify the network provisioning parameters.

Note: Provisioning of VLANs is supported only when `network-provisioning` is l2-underlay or l3-underlay. For None, no VLANs can be provisioned and hence no ports are added to the provisioned VLAN.

To delete a vCenter service, use the following syntax:

```
CLI (network-admin@switch) > vcenter-connection-delete name name-string
```

To display information about a vCenter service, use the command:

```
CLI (network-admin@switch) > vcenter-connection-show

CLI (network-admin@dorado-leaf3) > vcenter-connection-show layout vertical
switch:                switch
name:                  vcenter-server
host:                   vcenter-server.pluribusnetworks.com
user:                   administrator@lab.pluribus
enable:                yes
state:                 ok
connected-time:         connected at 2022-04-29 01:28:22
vlans:                 2100-2111
network-provisioning:   l3-underlay
```

To display information about VLANs before network-provisioning, use the command:

```
CLI (network-admin@switch-1) > vlan-show
switch   id   type   vxlan   vxlan-type scope  description      active stats ports
      untagged-ports   active-edge-ports
-------- ---- ------ ------- ---------- ------ ---------------- ------ -----
---------------------  ---------------  -----------------
switch-1 1    public                    local  default-1        yes    yes   0-11,14-29,32-72
      0-11,14-29,32-72 0,49
switch-2 1    public                    local  default-1        yes    yes   0-11,14-29,31-72
      0-11,14-29,31-72 0,44
switch-1 4091 public                    local  vlan-4091       yes    yes   0,13
      13               0,13
switch-2 4091 public                    local  vlan-4091       yes    yes   0,12
      12               0,12
switch-1 4092 public                    local  vlan-4092       yes    yes   0,12
      12               0,12
switch-2 4092 public                    local  vlan-4092       yes    yes   0,13
      13               0,13
switch-1 4093 public                    local  vlan-4093       yes    yes   30,253
      30,253           none
switch-2 4093 public                    local  vlan-4093       yes    yes   30,253
      30,253           none
```

To identify VTEP, use the command:

```
CLI (network-admin@switch-1) > vtep-show
scope  name   location       vrouter-name ip        virtual-ip
------ ------ -------------- ------------ --------- ----------
fabric vtep-1 switch-1 vr1          102.1.1.1 ::
fabric vtep-2 switch-2 vr2          103.1.1.1 ::
```

To display information about VLANs after network-provisioning, use the command:

```
CLI (network-admin@switch-1) > vlan-show
switch   id   type   vxlan   vxlan-type scope  description      active stats ports
untagged-ports    active-edge-ports
-------- ---- ------ ------- ---------- ------ ---------------  ------ ----- ----------------------
---------------  -----------------
switch-1 1    public                   local  default-1        yes    yes   0-11,14-29,32-72       0-11,14-
29,32-72 0,49
switch-2 1    public                   local  default-1        yes    yes   0-11,14-29,31-72       0-11,14-
29,31-72 0,44
switch-1 140  public 1400000           local  vCenter vlan 140 yes    yes   0,49                   none
         49
switch-2 140  public 1400000           local  vCenter vlan 140 yes    yes   0,44                   none
         none
switch-1 141  public 1410000           local  vCenter vlan 140 yes    yes   0,49                   none
         49
switch-2 141  public 1410000           local  vCenter vlan 140 yes    yes   0,44                   none
         none
switch-1 142  public 1420000           local  vCenter vlan 140 yes    yes   0,49                   none
         49
switch-2 142  public 1420000           local  vCenter vlan 140 yes    yes   0,44                   none
         none
switch-1 150  public 1500000           local  vCenter vlan 140 yes    yes   0,49                   none
         49
switch-2 150  public 1500000           local  vCenter vlan 140 yes    yes   0,44                   none
         none
switch-1 4091 public                   local  vlan-4091        yes    yes   0,13                   13
         0,13
switch-2 4091 public                   local  vlan-4091        yes    yes   0,12                   12
         0,12
switch-1 4092 public                   local  vlan-4092        yes    yes   0,12                   12
         0,12
switch-2 4092 public                   local  vlan-4092        yes    yes   0,13                   13
         0,13
switch-1 4093 public                   local  vlan-4093        yes    yes   30,253                 30,253
         none
switch-2 4093 public                   local  vlan-4093        yes    yes   30,253                 30,253
         none
```

```
CLI (network-admin@switch-1) > vtep-vxlan-show
name   vxlan
------ -------
vtep-1 1400000
vtep-1 1410000
vtep-1 1420000
vtep-1 1500000
vtep-2 1400000
vtep-2 1410000
vtep-2 1420000
vtep-2 1500000
```

Setting Automatic Link Aggregation on ESXi-facing Ports for vCenter

Currently, automatic Link Aggregation (LAG) is only supported between Arista Networks switches. With this feature, automatic Link Aggregation (LAG) includes ports between Arista Networks switches and ESXi hosts.

NetVisor OS implements LLDP and LACP to bundle the ports. Since there are no custom type-length-value (TLV) probes, NetVisor OS implements standard LLDP TLVs to uniquely identify ESXi hosts. The system description TLV is used to identify ESXi hosts and the system name is used to uniquely identify a specific ESXi host.

LACP mode is enabled and set to active on `auto-lag` with a fallback-option of `individual` to ensure that there is extra robustness in bundling and ensure that ports are bundled only if ESXi host is running LACP to avoid any data path issues.

You can enable or disable trunking towards hosts (default value is `off`).

> **Note**: Starting from NetVisor OS release 3.1.0, the `auto-host-bundle` configuration option is no longer required for trunking with ESXi host.

Example output from `trunk-show`, `port-show` and `lldp-show`.

```
CLI (network-admin@Leaf1) > trunk-show format trunk-
id,switch,name,ports,lacp-mode,lacp-fallback,lacp-individual,status,

trunk-id name     ports lacp-mode lacp-fallback  lacp-individual status
-------- -------- ----- --------- -------------  --------------- -----------
129      auto-129 42,44  active    individual    none            up,PN-other


CLI (network-admin@Leaf1) > port-show port 42,44 format all

port  bezel-port  status                                              rswitch  rem-ip  rem-mac               lport   c
trunk
----   ----------   ---------------------------------------   -------   ------   ----------------   ------
------ --------
42       42             up,PN-other,LLDP,trunk,LACP-PDUs,vlan-up  2987       ::         00:00:00:00:00:00  42        f
auto-129
44       44             up,PN-other,LLDP,trunk,LACP-PDUs,vlan-up  2987       ::         00:00:00:00:00:00  44        f
auto-129

CLI (network-admin@Leaf1) > lldp-show

local-port  bezel-port  chassis-id  port-id               port-desc                                    sys-
name
----------  ----------  ----------  ----------------  ---------------------------------
------------------------------
42            42            vmnic2      00:50:56:98:07:56 port 25 on dvSwitch DEV-CN-Tests-1 esx-
dev01.pluribusnetworks.com
44            44            vmnic3      00:50:56:98:07:57 port 24 on dvSwitch DEV-CN-Tests-1 esx-
dev01.pluribusnetworks.com
```

Configuring Open vSwitch Database (OVSDB)

This chapter provides information about configuring the Open vSwitch Database (OVSDB) on a NetVisor OS switch using the NetVisor OS Command Line Interface (CLI).

Understanding Open vSwitch Database (OVSDB)

Note: The OVSDB features are supported only on Ericsson platforms: NSU, NRU01, NRU02, NRU03, and NRU-S0301.

Open vSwitch (OVS) is a multilayer virtual switch that is designed to function in Virtual Machine (VM) environments. The typical use of an OVS is to forward traffic between VMs on the same physical host and between VMs and the physical network. OVS opens forwarding functions to programmatic extension and control through OpenFlow and Open vSwitch Database (OVSDB) management protocol. OVS also supports standard management protocols and interfaces like NetFlow, CLI and LACP. OVS is designed to support distribution across multiple physical servers.

The OVS implementation comprises a vSwitch daemon (ovs-vswitchd) and a database server (ovsdb-server). Arista makes use of the OVSDB database server to enable communication between an SDN controller and NetVisor OS to facilitate overlay network deployment. The OVSDB server provides a common database which both the SDN controller and the NetVisor OS OS accesses and modifies to stay synchronized. The information in OVSDB is stored as standardized database schema which enables heterogeneous entities such as baremetal servers and virtualized nodes to inter-operate and interact with the SDN controller. The hardware VTEP schema in OVSDB facilitates the provisioning of VXLAN network between hardware and software VTEPs, thereby allowing them to become a part of the network managed by SDN controller.

The OVSDB schema enables physical devices and the SDN controllers to exchange MAC address information. Arista devices capture MAC routes to entities in the physical network and updates table in the schema so that SDN controllers connecting to the devices can access the MAC tables. Conversely, SDN controllers capture MAC routes to entities in the virtualized environment and updates the schema. Having MAC addresses of both the physical and virtualized workloads in the database enables NetVisor OS to configure VLANs with VXLANs, assign ports to VLANs, and configure VXLAN tunnels between hardware and software VTEPs.

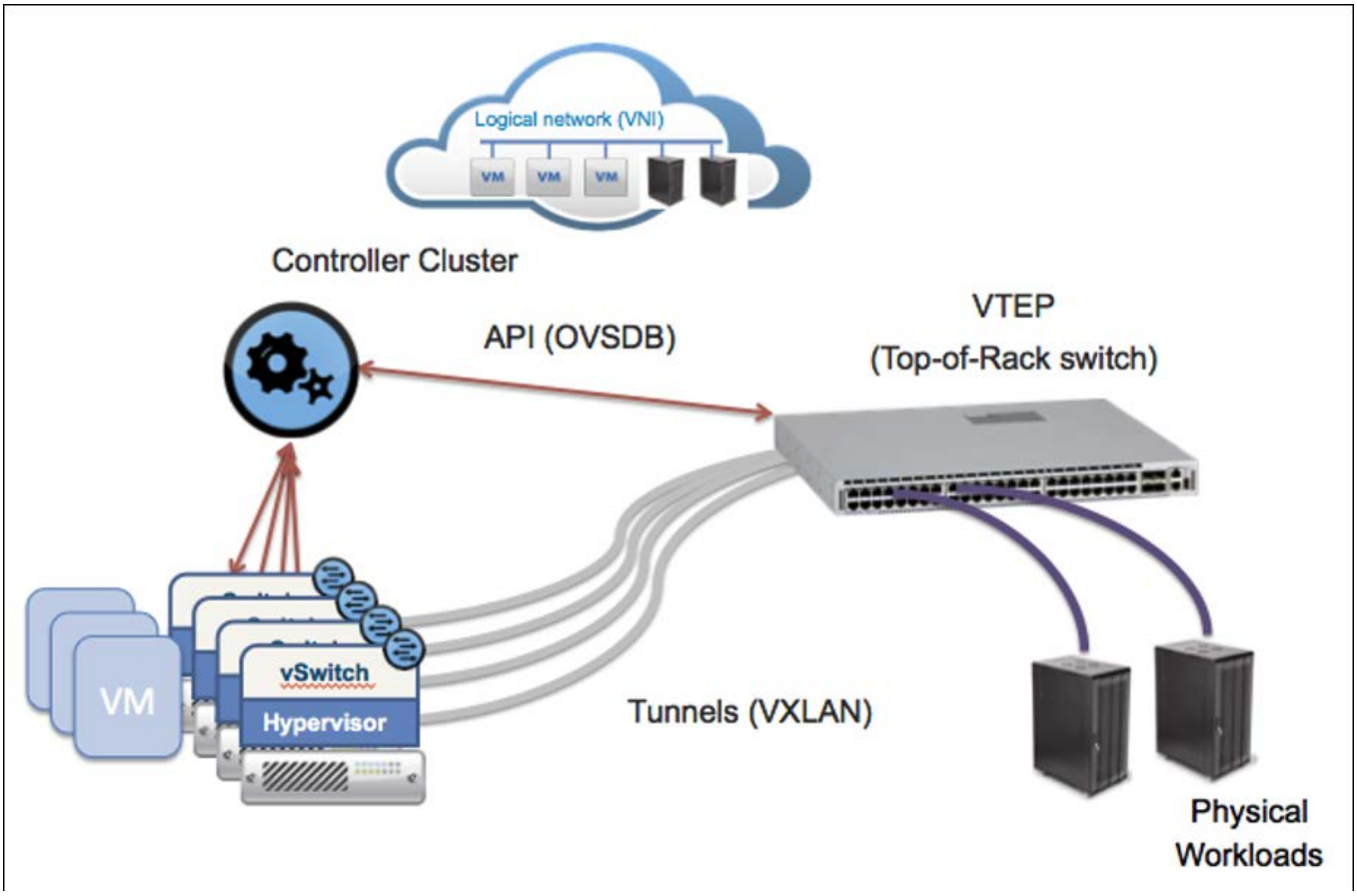


Figure 20-1 - Connecting physical and virtualized workloads using OVSDB

Configuring OVSDB with NetVisor OS

The NetVisor OS implementation of OVSDB provides a means of communication between SDN controllers and Arista switches. By leveraging OVSDB, Arista devices exchange control and network information with SDN controllers, thereby enabling VM traffic from the entities in a virtualized network to be forwarded to entities in a physical network and vice versa.

The OVSDB service runs in a container in NetVisor OS. This service maintains a database schema that stores information regarding Arista devices and SDN controllers in various tables. The OVSDB schema also includes the MAC address information of the physical servers and virtual hosts. SDN Controllers such as Ericsson ODL (Open-Day Light) controller communicates with OVSDB for provisioning and configuration of the overlay network. The overlay network configuration involves creating tunnels between local and remote hardware and software VTEPs.

Any configuration change in NetVisor OS updates the OVSDB schema which makes the SDN controller aware of the current configuration. A VTEP agent in NetVisor OS closely tracks the configuration updates pushed into OVSDB by the SDN controller as well as the changes in NetVisor OS. The VTEP agent then derives the required actions and executes the CLI commands needed to create VLANs and tunnels between local and remote VTEPs.

Follow the steps below to configure OVSDB using NetVisor OS.

- Configure the vNET with private VLANs, VXLANs, and managed ports:

```
CLI (network-admin@switch) > vnet-create name vpod1 scope fabric vlan-type
private num-private-vlans 100 managed-ports 12 vxlans 10000-10099
```

- Specify the VLANs reserved for the vNET by using the `vnet-public-vlans-modify` command. If you add managed ports to the vNET configuration, VLAN IDs from this range is assigned as public VLANs for the private VLANs.

```
CLI (network-admin@switch) > vnet-public-vlans-modify vlans 200-210
```

You can also define shared ports in a vNET configuration. Ideally, you can configure the border leaf ports that connect to the DC gateway as shared ports. If you add shared ports to the vNET configuration, specify a public VLAN range for the ports by using the command:

```
CLI (network-admin@switch) > vnet-modify name vpod1 public-vlans 300-310
```

For shared ports, private VLAN IDs should have the same range as that of public VLAN IDs configured in the vNET.

- Configure the underlay network:

```
CLI (network-admin@switch) > vrouter-create name vpod1-vr1 vnet vpod1
router-type hardware


CLI (network-admin@switch) > vrouter-interface-add vrouter-name vpod1-vr1
ip 192.168.50.2/24 vlan 50
```

- Create the SSL/TLS certificate for OVSDB if you want to create an SSL connection with the controller.

```
CLI (network-admin@switch) > cert-create country US state California city
PA organization "Pluribus Networks Inc" organizational-unit Engineering
common-name CN1 name cert1
Successfully generated self-signed certificate.
```

For detailed steps on how to configure and deploy TLS certificates for OVSDB, see *Using OpenSSL TLS certificates for OVSDB and other Services.*

- Create Open vSwitch configuration:

Use the `openvswitch-create` command to configure Open vSwitch. This command creates the OVS container and services.

`openvswitch-create`	Create an OVS instance.
`name name-string`	Specify a name for the OVS instance.
`vnet vnet-name`	Specify the name of the vNET for OVS.
`tunnel-ip` *`ip-address`*	Specify the IP address for the tunnel.
Specify any of the following options:	
`dedicated-vnet-service|shared-vnet-service`	Specify if Open vSwitch is a dedicated or shared vNET service.
`shared-vnet-mgr vnet-`*`manager name`*	Specify the vNET manager to share with if this is a shared service.
`location fabric-node-name`	Specify the location of the service.
`storage-pool` *`storage-pool-name`*	Specify a storage pool to apply to the vNET.
`gateway` *`ip-address`*	Specify gateway IP address for service.
`cert-name` *`cert-name-string`*	Specify the certificate name for SSL connections
`ca-cert-name` *`ca-cert-name-string`*	Specify the CA Certificate name for SSL connections
`cert-location none|global|container`	Specify the Certificate location - global or within container.
`global-vtep|local-vtep`	Specify the hardware VTEP associated with Open vSwitch.
`bfd|no-bfd`	Specify if you want to enable BFD for OVSDB created tunnels.

For example:

```
CLI (network-admin@switch) > openvswitch-create name ovs-1 vnet vpod1
tunnel-ip 192.168.0.10 cert-name cert1 ca-cert-name ca-cert1
```

- Create an OVSDB interface using the `openvswitch-interface-add` command.

`openvswitch-interface-add`	Add an interface to OVS.
`ovs-name` *`name-string`*	This parameter is not configurable. It assumes the name of the OVS service.
Specify one or more of the following options:	
`ip` *`ip-address`*	Specify the IP address for the interface.
`netmask netmask`	Specify the netmask.
`assignment none|static|dhcp|dhcpv6|autov6`	Specify the method of IP address assignment.
`linklocal` *`ip-address`*	Specify the IPv6 Link Local address.
`vnet` *`vnet-name`*	Specify interface VLAN vNET.
`bd` *`bridge-domain name`*	Specify the bridge domain name.
`vlan` *`vlan-id`*	Specify the VLAN assigned to the interface.
`vlan-type public|private`	Specify the type of VLAN for the interface.
`if mgmt|data|span|span2|span3`	Specify the interface type.
`alias-on` *`alias-on-string`*	Specify an alias if desired.
`exclusive|no-exclusive`	Specify if the interface is exclusive or not.
`nic-enable|nic-disable`	Specify to enable or disable the NIC.
`vrrp-id` *`id`*	Specify the ID assigned by VRRP.
`vrrp-primary` *`vrrp-primary-string`*	Specify the primary interface for VRRP.
`vrrp-priority` *`0..254`*	Specify the VRRP priority for the interface.
`vrrp-adv-int` *`milliseconds`*	Specify the VRRP advertisement interval in milliseconds. The range is 10 to 40950 with a default value of 1000.
`vrrp-preempt-mode disable|enable`	Used to allow/prevent high priority VRRP backup from becoming VRRP primary.
`secondary-macs secondary-macs-string`	Specify a secondary MAC address for the interface.
`if-nat-realm internal|external`	Specify the NAT interface realm.
`priority-tag|no-priority-tag`	Specify to add priority tag on forwarded traffic.

For example:

```
CLI (network-admin@switch) > openvswitch-interface-add ovs-name ovs-1 ip
192.168.31.148 netmask 24 vlan 3001 vlan-type public
```

Configuring the interface as `data` or `mgmt` depends on if the SDN controller is located on the data network

or the management network.

If the controller is on a Layer 3 network several hops away, use `openvswitch-modify` command to configure a gateway IP address.

```
CLI (network-admin@switch) > openvswitch-modify name ovs-1 gateway
192.168.31.1
```

- Add the hardware VTEP manager:

Use the `openvswitch-hwvtep-manager-add` command to configure the connection between OVSDB and the SDN controller.

`openvswitch-hwvtep-manager-add`	Create a VTEP manager for OVSDB.
`name` *`name-string`*	This parameter is not configurable. The parameter assumes the name of the OVS service.
Specify the following options:	
`manager-type odl|nsx`	Specify the type of SDN controller. **Note:** NetVisor OS only supports ODL controller currently.
`connection-method unix-socket|tcp|| unix-socketlisten|tcp-listen|ssl| default`	Specify the connection method.
`ip` *`ip-address`*	Specify the IP address of the SDN controller.
`username` *`username-string`*	Specify the username.
`password` *`password-string`*	Specify the password.
`port` *`port-number`*	Specify the port number of the database.

```
CLI (network-admin@switch) > opensvswitch-hwvtep-manager-add name ovs-
1 manager-type odl connection-method ssl ip 10.10.10.1
```

When you execute the command above, the connection to the ODL controller is established. You can now define the tunnel networks for software and hardware VTEPs by using the following commands:

```
CLI (network-admin@switch) > vnet-tunnel-network-add name vpod1 network
192.168.1.0/24 description SWVTEPS
CLI (network-admin@switch) > vnet-tunnel-network-add name vpod1 network
192.168.0.0/24 description HWVTEPS
```

You can establish a connection between the ODL controller and OVSDB by using SSL, TCP, or unix-socket options.

NetVisor OS features an error reporting mechanism to make the SDN controller aware of any error that occurs while orchestrating tunnels. If an error occurs while the ODL controller provisions tunnels between VTEPs, NetVisor OS updates the OVSDB VTEP schema with the error, thereby notifying the ODL controller.

Using OpenSSL TLS Certificates for OVSDB and other Services

NetVisor OS supports Transport Layer Socket (TLS) certificates that you can use for services such as OVSDB or web service. An SSL connection to any NetVisor OS service mandates TLS certificates. In the case of OVSDB, TLS is required to create a secure SSL connection to an SDN controller such as ODL.

You can create a common certificate for all NetVisor OS services or create multiple named certificates for distinct services. Each service can use a different certificate identified by name or container name or zone. The certificate facility in NetVisor OS keeps track of certificates used by using various applications. This facility notifies the applications when a certificate is updated and it also prevents a certificate from being deleted if an application is using it.

To enable SSL for OVSDB communication with the SDN controller, you must generate either of the two certificate types below:

- Self-signed certificate
- Certificate signed by a Certificate Authority (CA)

To create a self-signed server certificate, use the `cert-create` command:

```
CLI (network-admin@switch) > cert-create
```

`cert-create`	Creates a server certificate and self-sign.
`country` *`country-string`*	Specify a country name (two letter code).
`state` *`state-string`*	Specify a state or province name.
`city` *`city-string`*	Specify a city name.
`organization` *`organization-string`*	Specify an organization name.
`organizational-unit` *`organizational-unit-string`*	Specify an organizational unit name.
`common-name` *`common-name-string`*	Specify a common name.
`name` *`name-string`*	Specify a certificate name.
any of the following options:	
`container` *`zone-name`*	Specify a certificate zone name.
`status` *`status-string`*	The expiration status of the certificate.

For example:

```
CLI (network-admin@switch) > cert-create country US state California city
PA organization "Pluribus Networks Inc" organizational-unit Engineering
common-name CN1 name cert1
Successfully generated self-signed certificate.
```

Use the `cert-show` command to view the certificate:

```
CLI (network-admin@switch) > cert-show
switch:              switch
```

```
name:                 cert1
container:
country:              US
state:                California
city:                 PA
organization:          Pluribus Networks Inc
organizational-unit:   Engineering
common-name:          CN1
cert-type:            server
subject:               /C=US/ST=California/L=PA/O=Pluribus Networks
Inc/OU=Engineering/CN=CN1
issuer:                /C=US/ST=California/L=PA/O=Pluribus Networks
Inc/OU=Engineering/CN=CN1
serial-number:        1
valid-from:            Apr 24 09:11:57 2021 GMT
valid-to:              Apr 24 09:11:57 2022 GMT
```

If you want to get the certificate signed by a CA, follow the steps below:

- Create a Certificate Signing Request (CSR) by using the command:

`cert-request-create`	Create a certificate signing request.
`name` *`name-string`*	Specify the certificate name.
`container` *`zone name`*	Specify the container name or zone.

```
CLI (network-admin@switch) > cert-request-create name cert1
Certificate signing request successfully generated at /sftp/export/cert1-
cert.csr.
```

- Copy the CSR to your CA server, and get it signed by the CA.
- Copy the signed server certificate, CA root certificate, and intermediate certificate (if signed by an intermediate server) to the `/sftp/import` directory on the switch.
- Import the certificates onto the switch by using the command below:

```
CLI (network-admin@switch1) > cert-import
```

`cert-import`	Import certificates from /sftp/import directory.
`file-ca` *`file-ca-string`*	Specify the name of the CA certificate file.
`file-server` *`file-server-string`*	Specify the name of server certificate file (signed by CA).
`container` *`zone name`*	Specify the container name or zone.
`file-inter` *`file-inter-string`*	Specify the name of intermediate CA certificate file.
`status status-string`	The expiration status of the certificate.

For example:

```
CLI (network-admin@switch) > cert-import file-ca PN-cacert.pem file-server
cert1-cert.pem
Successfully imported certificates.
```

View the server-signed certificate by using the command:

```
CLI (network-admin@switch) > cert-show
switch:              switch
name:                cert1
container:
country:             US
state:               California
city:                PA
organization:        Pluribus Networks Inc
organizational-unit:  Engineering
common-name:         CN1
cert-type:           server
subject:              /C=US/ST=California/L=PA/O=Pluribus Networks Inc/OU=Engineering/CN=CN1
issuer:              /C=US/ST=California/L=Palo Alto/O=Pluribus Networks
Inc/OU=Engineering/CN=Pluribus Networks Test CA 2k-sha-256/emailAddress=abc@example.com
serial-number:       3
valid-from:          Apr 24 09:26:06 2021 GMT
valid-to:            Apr 24 09:26:06 2022 GMT
```

You can now configure the OVS service by specifying the name of the server certificate and the CA root certificate.

For example:

```
CLI (network-admin@switch) > openvswitch-create name ovs-tls-1 vnet vpod1
tunnel-ip 192.168.0.10 dedicated-vnet-service storage-pool rpool gateway
192.168.14.1 cert-name cert1 ca-cert-name ca-cert1 global-vtep
```

Related Commands

- To delete a certificate, use the `cert-delete` command:

```
CLI (network-admin@switch) > cert-delete
```

`cert-delete`	Deletes a certificate.
`name` *`name-string`*	Specify the name of the certificate.
`container` *`zone name`*	Specify container name or zone.

For example:

```
CLI (network-admin@switch) > cert-delete
Successfully deleted all certificate files.
```

- To display a certificate signing request, use the `cert-request-show` command:

```
CLI (network-admin@switch) > cert-request-show
```

`cert-request-show`	Displays the certificate signing request.

`cert-request`	*`cert-request-string`*	Specify the name of the CSR.

For example:

```
CLI (network-admin@switch) > cert-request-show
---------------------------------------------------------------
-----BEGIN CERTIFICATE REQUEST-----
MIICnDCCAYQCAQEwVzELMAkGA1UEBhMCdXMxCzAJBgNVBAgMAmNhMQswCQYDVQQH
DAJtcDELMAkGA1UECgwCcGwxDTALBgNVBAsMBGVuZ2cxEjAQBgNVBAMMCXBsdXJp
YnVzMTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAMrE6Jowg0VKUw2M
NlL8vp1N8dYE/UL5pvu8FKYWgwG7tC2fjHunZCI0XmssFtZysQul/r9nk+edA5tt
0zIWRmqTB60wnWmzl6uGymeAsC9OSm0ZHFc9zZfUxKjRM/n1dOri3Pw/rODbCjM9
qwO5hsvZc/c1o3ajYFrj1yMlKDIiPW1td1VTpc5TL6wCwnDM697Yb9oQ0cbLKTDl
w5AjQSgJK29rLUl8ptAZXIUkeendpE4MCYrl6Hd+ziOJHXncj65MJyfANTZMrtGD
IJD3m+JsKZt882vMw3AZ3C9WEuE0OZrbabGBHqVKARik2qFhu2bGjlbuj/M6TOf5
Jj1WROUCAwEAAaAAMA0GCSqGSIb3DQEBBQUAA4IBAQCh1YhXRNwkwmw3FVH4H0Xi
rczy0FkyHkdSbIUIf+6n3qroRpBpcEdrx8fREyiw8hLUks9OcUlT+nSshsWIitI7
R5dcFlyo5HUVjqQQVMlSq3j4fM9XE8y8KRMZ3mfLXRTmuFPxbBuE3ZGjlBSLnBgK
ODqHF1gVa4u7l9mO3TRXczLQiAPaw38/kxEwkh4erJp4jjXf8K0h9JMGvYONYWeI
1PbiZpjIWDLNbg6sKqqrPAxEAjzGNMgNPIMXRepmEmnC/BaLVA04noZran8LRLNp
Id41o3TnlXiAodF/Mc7H5fI1hYf0YzWDSfz3PNufn6Dusu5M2ma7jtWlEdBW8huH
-----END CERTIFICATE REQUEST-----
```

Configuring OVSDB High Availability

NetVisor OS supports the OVSDB high availability feature where OVSDB instances are configured on both the switches in a cluster to ensure redundancy. The ODL controller pushes vNET-specific configuration to the OVSDB schema in both the cluster nodes while the VTEP agent in the master node provisions the overlay as stipulated by the tables in the schema. The VTEP agent in the slave node does not act upon the configuration updates from the SDN controller since the VTEP agent in the slave node is passive. In the occasion of a failover, the slave node becomes the master node and the VTEP agent in the master node provisions the network.

Note: When the cluster goes offline, both the nodes operate in standalone mode which may result in divergent configurations.

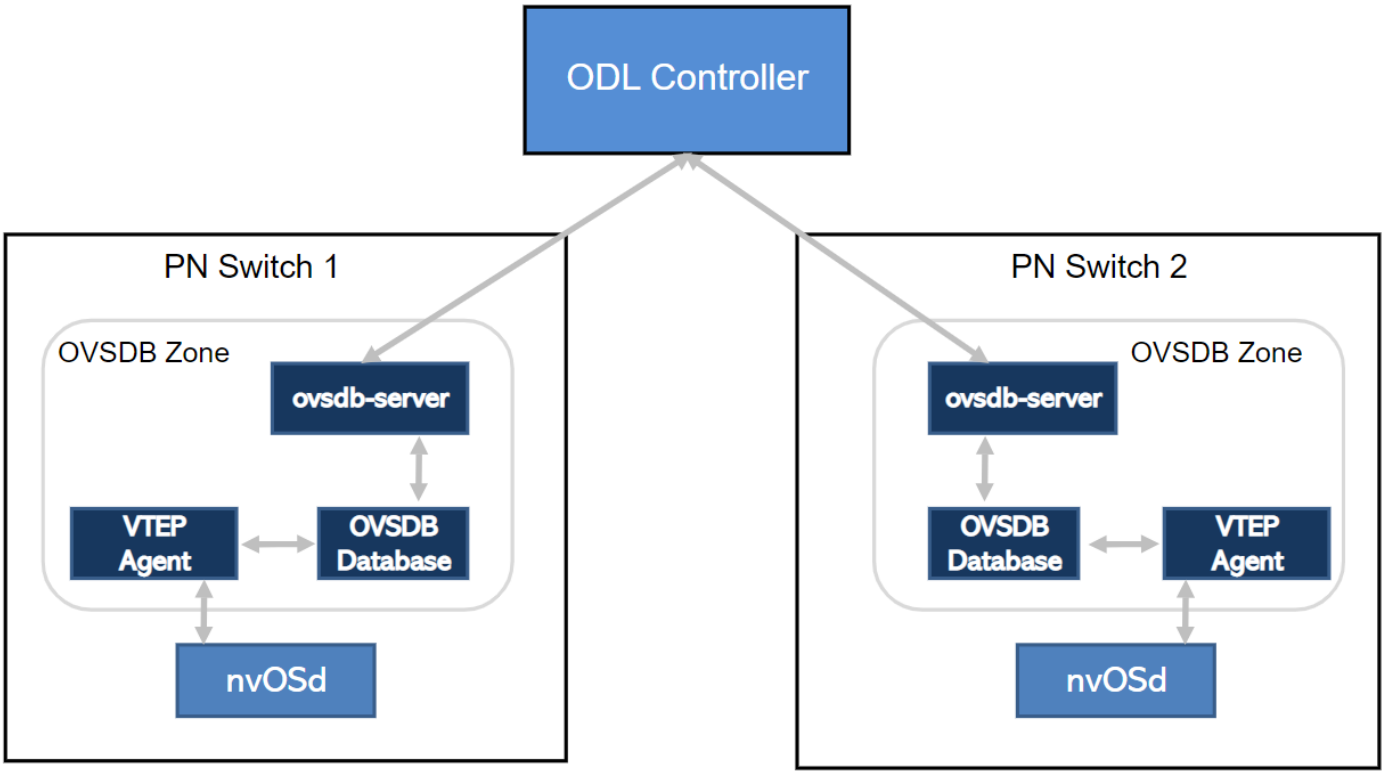


Figure 20-2 - OVSDB High availability

To configure High availability for OVSDB, follow the steps below:

- Configure the vNET for the cluster with private VLANs, VXLANs, and managed ports:

```
CLI (network-admin@switch1) > vnet-create name vpod1 scope fabric vlan-type
private num-private-vlans 100 managed-ports 12 vxlans 10000-10099
```

- Create a VLAN with `cluster` scope for OVSDB HA configuration:

```
CLI (network-admin@switch1) > vlan-create id 60 scope cluster description
ovsdb-ha ports 272
```

- Configure vRouter underlay on switch1 and create VRRP interfaces:

```
CLI (network-admin@switch1) > vrouter-create name vpod1-vr1 vnet vpod1
router-type hardware hw-vrrp-id 10


CLI (network-admin@switch1) > vrouter-interface-add vrouter-name vpod1-vr1
ip 10.10.10.1/29 vlan 60 vlan-type public


CLI (network-admin@switch1) > vrouter-interface-add vrouter-name vpod1-vr1
ip 10.10.10.3/29 vlan 60 vlan-type public vrrp-id 10 vrrp-primary eth1.60
vrrp-priority 250
```

- Configure vRouter underlay on switch2 and create VRRP interfaces:

```
CLI (network-admin@switch2) > vrouter-create name vpod1-vr2 vnet vpod1
router-type hardware hw-vrrp-id 10


CLI (network-admin@switch2) > vrouter-interface-add vrouter-name vpod1-vr2
ip 10.10.10.2/29 vlan 60 vlan-type public


CLI (network-admin@switch2) > vrouter-interface-add vrouter-name vpod1-vr2
ip 10.10.10.3/29 vlan 60 vlan-type public vrrp-id 10 vrrp-primary eth3.60
vrrp-priority 200
```

Note: The VRRP interfaces on switch1 and switch2 share the same virtual IP address.

- Configure and deploy TLS certificates for both the cluster nodes if you want to establish an SSL connection between OVSDB instances and the ODL controller. For configuration steps, see *Using OpenSSL TLS certificates for OVSDB and other Services* .

- Configure Open vSwitch on switch1 and create OVSDB interface:

```
CLI (network-admin@switch1) > openvswitch-create name ovs-ha-1 vnet vpod1
tunnel-ip 192.168.0.1 gateway 10.10.10.3 cert-name cert1 ca-cert-name ca-
cert1


CLI (network-admin@switch1) > openvswitch-interface-add ovs-name ovs-ha-1
ip 10.10.10.100 netmask 24 vlan 3001 vlan-type public
```

- Configure Open vSwitch on switch2 and create OVSDB interface:

```
CLI (network-admin@switch2) > openvswitch-create name ovs-ha-2 vnet vpod1
tunnel-ip 192.168.0.2 gateway 10.10.10.3 cert-name cert1 ca-cert-name ca-
cert1


CLI (network-admin@switch2) > openvswitch-interface-add ovs-name ovs-ha-2
ip 10.10.10.103 netmask 24 vlan 60 vlan-type public
```

- Configure the connection to the ODL controller on switch1:

```
CLI (network-admin@switch1) > opensvswitch-hwvtep-manager-add name ovs-ha-
1 manager-type odl connection-method ssl ip 20.20.20.1
```

- Configure the connection to the ODL controller on switch2:

```
CLI (network-admin@switch2) > opensvswitch-hwvtep-manager-add name ovs-ha-
2 manager-type odl connection-method ssl ip 20.20.20.1
```

OpenStack ML2 Plugin

The Arista Networks OpenStack ML2 Plugin is available from NetVisor OS 5.2.0 release onward. This feature is available on all whitebox platforms.

To understand and configure the Arista Networks OpenStack ML2 plugin, see the NetVisor® OS Openstack ML2 Plugin Deployment Guide from the Pluribus Networks Website.